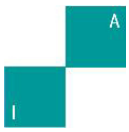
Inteligencia Artificial, 22(63) (2019), 1-15
doi: 10.4114/intartif.vol22iss63pp1-15



INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

Effects of Dynamic Variable - Value Ordering Heuristics on the Search Space of Sudoku Modeled as a Constraint Satisfaction Problem

James L. Cox[1], Stephen Lucci[2], Tayfun Pay[3]
[1]Brooklyn College of New York
2900 Bedford Avenue
Brooklyn, NY 11210
[2]The City College of New York
160 Convent Avenue
New York, NY 10031
[1,3]Graduate Center of New York
365 5^{th} Avenue
New York, NY 10016
[1]cox@sci.brooklyn.cuny.edu
[2]lucci@cs.ccny.cuny.edu
[3]tpay@gradcenter.cuny.edu

Abstract We carry out a detailed analysis of the effects of different dynamic variable and value ordering heuristics on the search space of Sudoku when the encoding method and the filtering algorithm are fixed. Our study starts by examining lexicographical variable and value ordering and evaluates different combinations of dynamic variable and value ordering heuristics. We eventually build up to a dynamic variable ordering heuristic that has two rounds of tie-breakers, where the second tie-breaker is a dynamic value ordering heuristic. We show that our method that uses this interlinked heuristic outperforms the previously studied ones with the same experimental setup. Overall, we conclude that constructing insightful dynamic variable ordering heuristics that also utilize a dynamic value ordering heuristic in their decision making process could improve the search effort for some NP-Complete problems.

Keywords: Backtracking-Search, Constraint Satisfaction Problems, Dynamic Variable Ordering Heuristics, NP-Completeness, Phase-Transition, Sudoku

1 Introduction

The manner in which the search space of a problem that has been modeled as a CSP (Constraint Satisfaction Problem) is explored depends on the various techniques that are built into the given CSP solver. These techniques might include filtering algorithms such as arc-consistency and path-consistency, random-restarts, back-jumping as well as dynamic variable ordering and dynamic value ordering heuristics.[0] All of these methods are important in their own right, but dynamic variable ordering and dynamic value ordering heuristics are especially important since they guide the backtracking search. In other words, combination of these heuristics could choose a variable and value pair that direct the search along a path

ISSN: 1137-3601 (print), 1988-3064 (on-line)

where it is hard to recognize that no solution exists. Therefore, it is very important that these heuristics choose a variable and value pair that is more likely to either, direct the search along a path that is easy to determine that no solution exists, or direct the search along a path where a solution exists.

The study of dynamic variable ordering heuristics began with the dom heuristic of [3] that picks the next variable with the smallest domain size. A tie-breaker was introduced for the dom heuristic in [4] that breaks ties by picking the next variable with the highest initial degree, the number of unassigned neighbors. A more sophisticated version of this heuristic was already in use for graph coloring [5]. This heuristic breaks ties for the dom heuristic by picking the next variable with the current highest degree.[1] A variant of this heuristic was proposed in [6], where the next variable with the minimum value of current domain size over current degree is selected. This heuristic works well for instances where the variables have wide range of degree in combination with small variance in their domain sizes, since dom/deg gives equal importance to domain size and degree. On the other hand, there has not been much attention given to dynamic value ordering. The most prominent work on this front was the lvo heuristics of [4] with four different ranking functions that employ forward-checking.

In this paper, we investigate various dynamic variable and value ordering heuristics with respect to Sudoku modeled as a CSP. Sudoku was first studied as a CSP in [7] and has been been getting a lot of attention in this field since then [8, 9, 10, 11, 12, 13], because it is a highly constrained challenging combinatorial search problem. We generated numerous Sudoku puzzles of size 16 by 16 (order-4), 25 by 25 (order-5) and 36 by 36 (order-6) using the setup outlined in [9]. We encoded the Sudoku puzzles as a CSP using the modeling technique from [11, 12] and also employed their arc-consistency algorithm. We observed the mean depth, mean number of explored variables and mean number of instantiations within a fixed time as well as the percentage of solved puzzles and mean time. We did this for seven different heuristics, including dom [3], an interpretation of dom+deg [5] and sd-lrv-mfv [11]. We refrained from investigating dom/deg [6]heuristic since it was already shown to not perform well for multiple permutation problems such as Latin Squares in [14] and Sudoku in [11].

We showed that lexicographically choosing the next variable performs better than randomly choosing the next variable. We demonstrated that using a dynamic variable ordering heuristic performs better than either randomly or lexicographically selecting the next variable. We also showed that the width of the search tree decreased when the dynamic variable ordering heuristic was used, which agrees with the results in [3, 15] and [16]. We demonstrated that adding a dynamic value ordering heuristic on top of a dynamic variable ordering heuristic does not necessarily improve the search effort, but instead adding a tie-breaker for the dynamic variable ordering heuristic does in fact improve the search effort. We showed that combining a dynamic value ordering heuristic with a dynamic variable ordering heuristic, that already utilizes a tie-breaker, might actually hinder the performance. We demonstrated that adding a dynamic value ordering heuristic as a second round of tie-breaker for the dynamic variable ordering heuristic enhances the search effort. We also detected the easy-hard-easy "phase-transition" as was previously observed with Sudoku puzzles in [9, 11, 12] and [13]. This is a phenomenon with all NP-Complete problems.

As far as we know, there has been no dynamic variable ordering heuristic that utilizes a dynamic value ordering heuristic as a tie-breaker, of course other than the ones introduced in [11]. Our experiments showed that even if we use a well-known dynamic variable ordering heuristic that employs a tie-breaker, such as dom+deg, that there could still be more ties and employing a dynamic value ordering heuristic as a second round of tie-breaker has a positive affect in the outcome of the search effort. Overall, we demonstrated that coming up with insightful dynamic variable ordering heuristics that also use dynamic value ordering heuristic in their decision making process, can help to guide the search effort for some NP-Complete problems. In fact, the performance achieved at the critical point by our most sophisticated heuristic is far better than the performance of the methods utilized in [9] and [13] with the same experimental setup.

[0] An excellent resource for most of these CSP techniques and many more is [1]. An excellent introduction written in Spanish is [2].

[1] This heuristic was initially called the Brelaz heuristic, then dom+futdeg heuristic and finally dom+deg heuristic. We should also note that in earlier papers the heuristic in [4] was called the dom+deg heuristic and the heuristic in [5] was called the dom+futdeg heuristic.

2 Methods

2.1 Encoding

We encode the Sudoku puzzles as a constraint satisfaction problem by using the natural combined model from [11, 12].

2.1.1 Natural Combined Model

The Sudoku puzzle is formulated on a 3D $\mathcal{S}(k*k)*(k*k)*(k*k)$ Boolean matrix, where $n=(k*k)$. We say that a Sudoku puzzle of order k is an n by n Sudoku puzzle.

The primal variable $x_{i,j}$ is represented by the slice $\{\mathcal{S}(i,j,v)|1 \leq v \leq n\}$. The initial domain of $x_{i,j}$ is denoted $D(x_{i,j})$ and is $\{v|\mathcal{S}(i,j,v)=True, 1 \leq v \leq n\}$.

The box dual variable $b_{v,p,q}$ where $\lceil i/k \rceil = p, \lceil j/k \rceil = q$ is represented by the slice $\{\mathcal{S}(i,j,v)|((p-1)*k+1) \leq i \leq p*k, ((q-1)*k+1) \leq j \leq q*k\}$. The initial domain of $b_{v,p,q}$ is denoted $D(b_{v,p,q})$ and, is $\{i,j|\mathcal{S}(i,j,v)=True, ((p-1)*k+1) \leq i \leq p*k, ((q-1)*k+1) \leq j \leq q*k\}$.

The column dual variable $c_{v,j}$ is represented by the slice $\{\mathcal{S}(i,j,v)|1 \leq i \leq n\}$. The initial domain of $c_{v,j}$ is denoted $D(c_{v,j})$ and is $\{i|\mathcal{S}(i,j,v)=True, 1 \leq i \leq n\}$.

The row dual variable $r_{v,i}$ is represented by the slice $\{\mathcal{S}(i,j,v)|1 \leq j \leq n\}$. The initial domain of $r_{v,j}$ is denoted $D(r_{v,j})$ and is $\{j|\mathcal{S}(i,j,v)=True, 1 \leq j \leq n\}$.

Then the $\neq$ constraints are enforced as follows: 1) on pairs of primal variables within each box, column and row. 2) on pairs of box dual variables of each value. 3) on pairs of column dual variables of each value. 4) on pairs of row dual variables of each value.

2.2 Filtering Algorithms

The following filtering algorithms were introduced in [11, 12] and work together with the natural combined model. Although the backtracking-search is being conducted on the primal variables, consistency checks on the other three view-points of the problem are performed whenever deemed necessary.

2.2.1 Redundantly Modeled Forward Checking Algorithm (RFC)

The RFC algorithm verifies whether or not assigning a value to a variable is consistent with respect to it's constraints. It also modifies the domains of the constrained variables accordingly. These checks and modifications are performed with respect to primal variables as well as the box, column and row dual variables.

Algorithm 1 Redundantly Modeled Forward Checking

Input: $(S,n,i,j,v)^2$
Output: Is $\{S(i,j,v) == \text{True}\}$ consistent?

```
S_temp ← S
if (REMOVE-ALL-EXCEPT-V == True) then
    if (REMOVE-ALL-OTHER-V == True) then
        S ← S_temp
        return True
    end if
end if
return False

function REMOVE-ALL-EXCEPT-V
    for k = 1...n do
        if (S_temp(i,j,k) == True ∧ k ≠ v) then
            if (EXISTS-IN-BOX(i,j,k) == False) then
                return False
            end if
```

```
15:             if (Exists-In-Col(i, j, k) == False) then
16:                 return False
17:             end if
18:             if (Exists-In-Row(i, j, k) == False) then
19:                 return False
20:             end if
21:             S_temp(i, j, k) ← False
22:         end if
23:     end for
24:     return True
25: end function

26: function Exists-In-Row(p, q, r)
27:     for k = 1...n do
28:         if (S_temp(p, k, r) == True ∧ k ≠ q) then
29:             return True
30:         end if
31:     end for
32:     return False
33: end function[3]

34: function Remove-All-Other-V
35:     if (Remove-V-From-Box == False) then
36:         return False
37:     end if
38:     if (Remove-V-From-Col == False) then
39:         return False
40:     end if
41:     if (Remove-V-From-Row == False) then
42:         return False
43:     end if
44:     return True
45: end function

46: function Remove-V-From-Row
47:     for k = 1...n do
48:         if (S_temp(i, k, v) == True ∧ k ≠ j) then
49:             t ← False
50:             for l = 1...n do
51:                 if (S_temp(i, k, l) == True ∧ l ≠ v) then
52:                     t ← True
53:                 end if
54:             end for
55:             if t == False then
56:                 return False
57:             end if
58:             if (Exists-In-Box(i, k, v) == False) then
59:                 return False
60:             end if
61:             if (Exists-In-Col(i, k, v) == False) then
62:                 return False
63:             end if
64:             S_temp(i, k, v)← False
65:         end if
```

66: **end for**
67: **return** True
68: **end function**[4]

2.2.2 Redundantly Modeled Arc-Consistency Algorithm One (RAC-1)

The RAC-1 algorithm takes the RFC algorithm one step further. Instead of just examining the constrained primal and dual variables of the instantiated variable, it checks all of the primal as well as all of the dual variables. It then verifies that they are consistent with the given assignment. Furthermore, it performs instantiations whenever the domain of some primal or dual variable becomes singleton, which forces it to repeat the whole process again. If the RAC-1 algorithm successfully executes, that is without finding an inconsistency, then the given puzzle is arc-consistent with respect to the primal and the box, column and row dual variables.

Algorithm 2 Redundantly Modeled Arc-Consistency One

Input: (S, n) [5]
Output: Is S arc-consistent?
1: $S_{temp} \leftarrow S$
2: $t \leftarrow 1$
3: **while** $t \neq 0$ **do**
4: $t \leftarrow 0$
5: **if** (Check-Primal-Variables == False) **then**
6: **return** False
7: **end if**
8: **if** (Check-Box-Variables == False) **then**
9: **return** False
10: **end if**
11: **if** (Check-Col-Variables == False) **then**
12: **return** False
13: **end if**
14: **if** (Check-Row-Variables == False) **then**
15: **return** False
16: **end if**
17: **end while**
18: $S \leftarrow S_{temp}$
19: **return** True

20: **function** Check-Primal-Variables
21: **for** $i = 1...n$ **do**
22: **for** $j = 1...n$ **do**
23: **if** $S_{temp}(i, j, 0) == 0$ **then**[6]
24: $c \leftarrow 0$
25: $v \leftarrow 0$
26: **for** $k = 1...n$ **do**
27: **if** $S_{temp}(i, j, k) ==$ True **then**
28: $c \leftarrow c + 1$
29: $v \leftarrow k$
30: **end if**
31: **end for**
32: **if** c==0 **then**

[2] S, n, i, j, v and S_{temp} are global variables and accessible by all functions.
[3] Exists-In-Box(p, q, r) and Exists-In-Col(p, q, r) functions behave similarly.
[4] Remove-V-From-Box and Remove-V-From-Col functions behave similarly.

```
33:                 return False
34:             else if c==1 then
35:                 if RFC(S_temp, n, i, j, v) == True then
36:                     t ← 1
37:                 else
38:                     return False
39:                 end if
40:             end if
41:         end if
42:     end for
43: end for
44: return True
45: end function[7]
```

2.3 Search Heuristics

The following heuristics are utilized with respect to primal variables.

2.3.1 Heuristic-1 (RAN&LEX)

Randomly select the unassigned variable; and lexicographically select the values in its domain.

2.3.2 Heuristic-2 (LEX&LEX)

Lexicographically select the unassigned variable; and lexicographically select the values in its domain.

2.3.3 Heuristic-3 (SD&LEX)

Lexicographically select the unassigned variable with the smallest domain size; and lexicographically select the values in its domain.

2.3.4 Heuristic-4 (SD&MFV)

Lexicographically select the unassigned variable with the smallest domain size; and order the values in it's domain that occur from most frequently to least frequently in the domains of all of the assigned variables.

2.3.5 Heuristic-5 (SD-LRV&LEX)

Among the variables with the smallest domain size, lexicographically select the variable that has the least number of fixed variables among those with which it shares a constraint; and lexicographically select the values in its domain.

2.3.6 Heuristic-6 (SD-LRV&MFV)

Among the variables with the smallest domain size, lexicographically select the variable that has the least number of fixed variables among those with which it shares a constraint; order the values in its domain that occur from most frequently to least frequently in the domains of all of the assigned variables.

[5] S, n and S_{temp} are global variables and accessible by all functions.
[7] Check-Box-Variables, Check-Col-Variables and Check-Row-Variables functions behave similarly.
[6] We assume that the integer puzzle is located at $v = 0$ of $S(i, j, v)$ for simplicity.

2.3.7 Heuristic-7 (SD-LRV-MFV&MFV)

Among the variables with the smallest domain size that also have the least number of fixed variables among those with which it shares a constraint, lexicographically select the one that contains a value in its domain that occurs the most frequent times in the domains of all of the assigned variables; order the values in its domain that occur from most frequently to least frequently in the domains of all of the assigned variables.

Heuristic-1 serves the purpose of understanding how the backtracking search performs when the next variable to be assigned is chosen uniformly at random. Heuristic-3 is clearly the dom heuristic of [3]. Heuristic-4 serves the purpose of understanding how a dynamic variable ordering heuristic works with no tie-breaker, but with a value ordering heuristic. Heuristic-5 can actually be viewed as an interpretation of the Brelaz (dom+deg) heuristic of [5]. Heuristic-6 serves the purpose of examining how a dynamic variable ordering heuristic that has a tie-breaker works with a value ordering heuristic.

We should also note that we did experiment with the opposite tie-breaker, as was done in [11], the most number of constrained variables (MRV) and the opposite dynamic value ordering heuristic, order the values in it's domain that occur from least frequently to most frequently in the domains of all of the assigned variables (LFV). However, none of the three possible combinations of these tie-breakers (LRV and MRV) and dynamic value ordering heuristics (MFV and LFV) worked as well as Heuristic-7.

3 Experimental Setup

We implemented the modeling method, the filtering algorithm and the search heuristics in C++. Then we generated the test instances by following the methodology that was outlined in [9] for Sudoku puzzles. We first randomly filled 5%-25% percent of the empty puzzle board without invalidating the puzzle and then randomly chose one of our algorithms to complete the puzzle. We generated 100 fully solved Sudoku puzzles of order 4, 5 and 6, which correspond to 16 by 16, 25 by 25 and 36 by 36 size Sudoku puzzles, respectively.

Then we removed a cell from a given fully solved puzzle with probability p, where $p = 0$ implies a fully solved puzzle and $p = 1$ implies an empty one. We did this for all of the 100 fully solved puzzles of order 4 and 5 for each probability p, from $p = 0.05$ through $p = 0.95$ with 0.05 increments. We used the Mersenne Twister Pseudo-Random generator mt19937 in C++ for generating our random numbers, that were uniformly distributed between 0 and 1. A total of 1900 puzzles were generated for Sudoku puzzles of order 4 and 5. We only generated 100 Sudoku puzzles of order 6 at the hard region so that we can verify our observations carry over to larger puzzles.

All of the puzzles we generated are guaranteed to have a solution, but they can have more than one solution. It would take a tremendous amount of time to verify that each puzzle has a unique solution, because we would have to check the possibility of a solution at every single path. Furthermore, the puzzles become so sparse at higher probabilities that it is impossible to guarantee that they have a unique solution.

We set the time limit to 30 seconds for Sudoku puzzles of order 4 and to 360 seconds for Sudokdu puzzles of order 5 since these were the time limits used in [9] and [13]. We set the time limit to 720 seconds for Sudoku puzzles of order 6.

We kept track of six different parameters at each probability p: 1) Percentage of puzzles solved 2) Mean time 3) Mean depth of the search tree 5) Mean number of explored variables 6) Mean number of instantiations. If no solution was found within the allocated time then the given time limit and the calculated values for each parameter up to that time limit were used in the calculations. We used a computer that possesses an Intel Dual Core (Four Threads) I3-3227U CPU at 1.90GHZ x64 based processor with 4GB of RAM to run our simulations.

4 Empirical Analysis

4.1 Order 4 Puzzles

The performance of our seven heuristics when solving order 4 puzzles at different probabilities are presented in tables 1 2 and 3 as well as in the corresponding graphs. The results of the methods in [9] and [13] are also included as graphs that show the percentage of solved puzzles at each probability versus mean time in seconds.

We observe that Heuristic-1 performed the worst with respect to being able to solve all of the instances at each probability within the allocated time. Heuristic-1 is followed by Heuristic-2 and Heuristic-3 where they also had an overall worse performance than the methods in [9] and [13]. However, when they were able to solve the given instances, they did so much faster than the methods in [9] and [13]. This can be seen in the corresponding graphs that show the mean time in seconds. On the other hand, Heuristic-5 and Heuristic-7 were able to solve all of the instances at each probability within the allocated time. And they did so in a very fast manner as can be seen by the corresponding graphs that show the mean time in seconds.

When a heuristic at some probability has the same or almost the same average depth, average number of explored nodes and average number of instantiations, it means that the search backtracked few times or it did not backtrack at all. This is the case at all of the probabilities for Heuristic-5 and Heuristic-7 and most of the probabilities for Heuristic-4 and Heuristic-6. The reason for this occurrence cannot really be attributed to the choice of a dynamic variable and value ordering heuristics, but it is due to the modeling choice as well as the constraint propagation algorithm that is being employed. In fact, the reason that the first three heuristics actually performed this well is due to this. In other words, the constraint propagation algorithm along with the encoding method overpowered the variable ordering aspect of our solver when solving order 4 puzzles.

Another observation is that the average number of instantiations that are roughly double the average number of explored nodes starting from Heuristic-3 and on. This is due to using the smallest domain heuristic in variable selection that allows the width of the search tree to be narrower and consequently results in a deeper search-tree. This observation concurs with the results in [3, 15] and [16] that the width of the search-tree decreases when the smallest domain heuristic is utilized. This difference will be more striking with order 5 puzzles.

4.2 Order 5 Puzzles

The performance of our seven heuristics when solving order 5 puzzles at different probabilities are presented in tables 4 5 and 6 as well as in the corresponding graphs. The results of the methods in [9] and [13] are also included as graphs that show the percentage of solved puzzles at each probability versus mean time in seconds.

Heuristic-1 cannot solve any instances at probabilities 0.55, 0.6, 0.65 and 0.8 within the allocated time. Heuristic-2 performs some what better than Heuristic-1, but it still performs worse than the methods in [9] and [13]. On the other hand, the difference in performance between Heuristic-2 and Heuristic-3 is quite striking at each probability as can be observed from the corresponding graphs and tables. Heuristic-3 allowed the search effort to go further down in the search tree, which as a consequence resulted in more nodes being explored in the allocated time if no solution was found; or a solution was found with less number of explored nodes. This was also true with respect to the instantiations at each probability. As a result, the search was not stuck at a certain depth and more puzzles were solved at each probability within the given time limit. As we also observed with order-4 puzzles, the width of the search tree was minimized when Heuristic-3 was employed. This can be recognized with respect to the ratio between the average number of instantiations over the average number of explored nodes at each probability, which was approximately three with Heuristic-2 and two with Heuristic-3. This observation, once again, concurs with the results in [3, 15] and [16] that the width of the search-tree decreases when the smallest domain heuristic is utilized.

We also witness the easy-hard-easy "phase-transition" that is a phenomenon with all NP-Complete problems. The hard region or the critical point of Sudoku is around $p = 0.55$. It is around this region where the most difficult puzzles reside.

We observe from the corresponding graphs that Heuristic-5 significantly improved the search effort at the critical point of $p = 0.55$ compared to Heuristic-4 which performed similar to Heuristic-3. In fact, the difference of performance between Heuristic-3 and Heuristic-4 at the hard region was not statistically significant. This experiment distinguishes the importance of using a tie-breaker for the smallest domain heuristic over just coupling the smallest domain heuristic with a value ordering heuristic.

We also notice from the corresponding graphs that although Heuristic-6 performed a little better at the critical point of $p = 0.55$ compared to Heuristic-5, it actually performed worse at the subsequent probabilities of the heavy-tail region. This experiment demonstrates that adding value ordering heuristic without really gathering insight from the variable ordering heuristic does not necessarily yield better performance and it may actually hinder it.

As can be seen from the corresponding graphs, Heuristic-7 achieved a significant improvement over both Heuristic-5 and Heuristic-6. The performance at the critical point of $p = 0.55$ is the most profound compared to the previous six heuristics. Heuristic-7, on average, was able to go further down in the search tree, traversed less number of nodes and made less number of instantiations. This demonstrates that a second round of tie-breaker for the smallest domain heuristic could be very useful. Indeed, the value ordering heuristic could be used for this purpose and it executes well for this intent. In other words, considering the values in the domains of the variables when breaking ties is relevant.

We also observe from the corresponding graphs that the methods in [9] and [13] did as well as our Heuristic-4 at the critical point of $p = 0.55$ whereas our Heuristic-7 clearly out performed them at this significant point.

4.3 Order 6 puzzles

The performance of our seven heuristics when solving order 6 puzzles at the hard region are presented in table 7. We detect similar patterns with order 6 puzzles that we also observed with order 4 and 5 puzzles with respect to average depth, average nodes and average instantiations. However, this time the differences among the heuristics are more profound, because there are more unassigned variables with domain sizes that are greater than two and as a result, Heuristic-1 and Heuristic-2 perform even worse. Although Heuristic-3 and Heuristic-4 cannot solve any instances within the allocated time, they still perform better than Heuristic-1 and Heuristic-2 since they were able to go further down the search-tree and thus explored more nodes on average. Heuristic-7 solves the most instances and it is followed by Heuristic-6 and Heuristic-5. The performance of Heuristic-7 when solving order 6 puzzles at the hard region is further evidence that considering the values in the domains of the variables when selecting the next variable to instantiate does in fact make a difference in guiding the search effort.

	0.50	0.55	0.60	0.65	0.70	0.75	0.80	0.85	0.90	0.95
Heuristic-1	5	9	16	27	39	53	70	83	98	118
Heuristic-2	4	9	17	27	44	61	77	95	113	137
Heuristic-3	5	11	22	41	63	83	103	123	138	156
Heuristic-4	6	12	22	44	69	87	109	131	147	164
Heuristic-5	5	11	20	39	60	84	104	126	140	156
Heuristic-6	6	13	22	44	70	89	112	132	148	165
Heuristic-7	6	13	24	45	70	90	113	134	149	167

Table 1: Average depth of the search tree at each probability for order 4 puzzles

	0.50	0.55	0.60	0.65	0.70	0.75	0.80	0.85	0.90	0.95
Heuristic-1	28	160	2,273	4,269	5,609	7,020	11,222	8,493	5,897	5,328
Heuristic-2	6	87	442	2,339	6,825	3,267	5,471	9,774	4,065	1,482
Heuristic-3	6	27	94	3,338	3,295	3,897	3,403	124	138	156
Heuristic-4	9	50	158	97	1,997	175	129	145	148	164
Heuristic-5	5	18	25	52	62	98	105	126	140	156
Heuristic-6	9	16	30	64	3,436	96	112	137	4,431	165
Heuristic-7	9	16	49	50	76	102	114	135	149	167

Table 2: Average number of explored nodes at each probability for order 4 puzzles

	0.50	0.55	0.60	0.65	0.70	0.75	0.80	0.85	0.90	0.95
Heuristic-1	90	384	8,024	15,574	20,969	25,509	40,781	29,485	21,714	19,225
Heuristic-2	9	206	1,061	7,663	20,936	8,934	15,501	27,426	11,076	3,267
Heuristic-3	9	46	170	6,640	6,532	7,741	6,715	125	139	157
Heuristic-4	13	91	297	154	3,928	266	150	161	149	165
Heuristic-5	7	28	34	67	65	114	108	127	142	157
Heuristic-6	12	20	41	86	6844	104	113	143	8771	165
Heuristic-7	14	20	76	57	83	117	116	136	149	167

Table 3: Average number of instantiations at each probability for order 4 puzzles

	0.50	0.55	0.60	0.65	0.70	0.75	0.80	0.85	0.90	0.95
Heuristic-1	22	49	69	101	138	170	210	248	288	326
Heuristic-2	20	48	70	102	141	169	216	260	280	320
Heuristic-3	26	69	102	154	203	250	300	346	387	427
Heuristic-4	25	70	107	162	218	263	321	361	404	446
Heuristic-5	25	69	108	153	206	253	301	348	391	429
Heuristic-6	27	72	118	163	219	263	313	362	398	442
Heuristic-7	26	74	116	162	216	266	319	362	404	443

Table 4: Average depth of the search tree at each probability for order 5 puzzles

	0.50	0.55	0.60	0.65	0.70	0.75	0.80	0.85	0.90	0.95
Heuristic-1	47,044	133,714	135,957	142,250	140,868	142,004	156,206	134,533	113,051	73,159
Heuristic-2	40,706	135,905	134,490	106,990	90,329	101,750	97,151	107,546	204,505	138,951
Heuristic-3	38,130	158,853	150,619	70,106	18,159	41,850	8,509	7,726	579	457
Heuristic-4	35,990	166,221	133,956	45,065	44,181	51,999	39,672	22,142	1,436	8,958
Heuristic-5	19,949	113,120	81,071	60,149	37,510	51,910	8,603	389	14,176	473
Heuristic-6	26,080	112,179	107,804	77,751	43,837	23,859	31,291	7,038	20,272	6,979
Heuristic-7	27,539	101,227	73,391	37,723	19,038	14,087	12,826	16,465	514	662

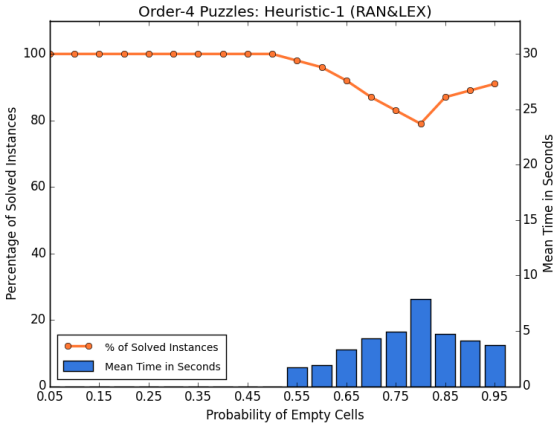
Table 5: Average number of explored nodes at each probability for order 5 puzzles

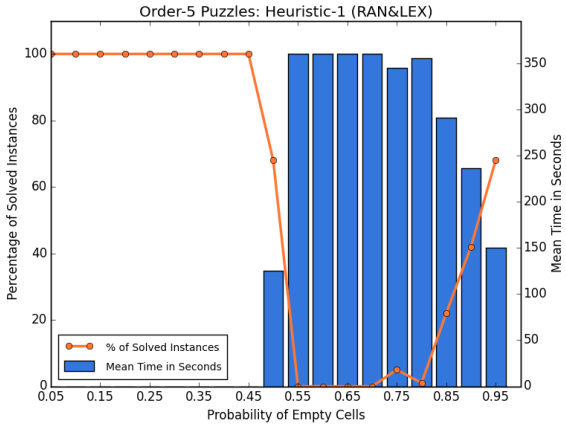
	0.50	0.55	0.60	0.65	0.70	0.75	0.80	0.85	0.90	0.95
Heuristic-1	176,119	515,880	535,476	566,626	561,713	561,895	611,545	520,519	433,597	278,044
Heuristic-2	112,075	391,026	395,833	300,883	258,723	290,502	286,054	306,336	597,076	466,662
Heuristic-3	76,250	317,693	301,200	140,112	36209	85714	16741	15164	778	494
Heuristic-4	71,970	332,428	267,866	90,186	88,936	105,301	79,123	44,225	2,664	17,513
Heuristic-5	39,886	226,222	162,081	120,190	74,894	103,666	16,927	438	28,022	536
Heuristic-6	52,148	224,313	215,561	155,410	87,599	47,529	63,880	13,741	42,252	13,553
Heuristic-7	55,610	202,314	146,721	75,322	37,920	24,852	26,190	32,610	626	883

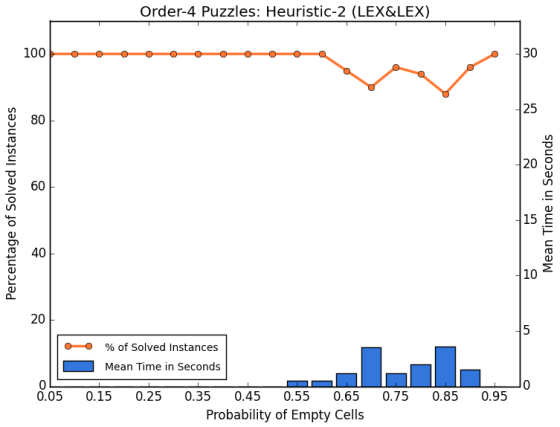
Table 6: Average number of instantiations at each probability for order 5 puzzles

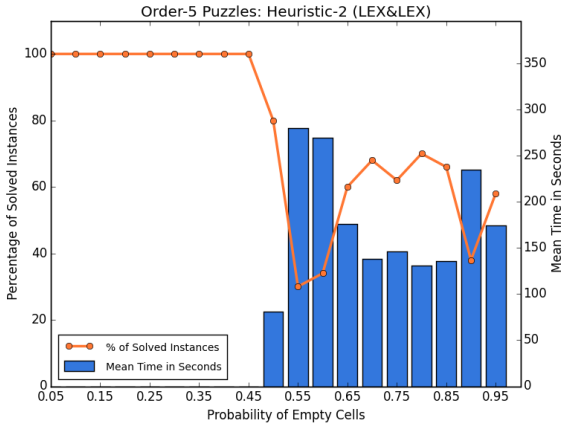
	Solved	Avg. Time	Avg. Depth	Avg. Nodes	Avg. Inst.
Heuristic-1	0%	720 secs	302	123,294	580,268
Heuristic-2	0%	720 secs	207	144,089	472,677
Heuristic-3	0%	720 secs	433	205,125	410,570
Heuristic-4	0%	720 secs	464	217,037	446,432
Heuristic-5	2%	691 secs	459	190,939	390,949
Heuristic-6	4%	677 secs	487	181,660	363,736
Heuristic-7	11%	623 secs	481	165,015	333,751

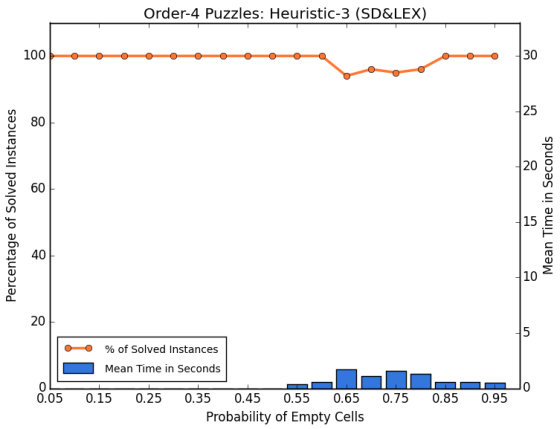
Table 7: 100 order 6 puzzles at the hard region with a time out of 720 seconds

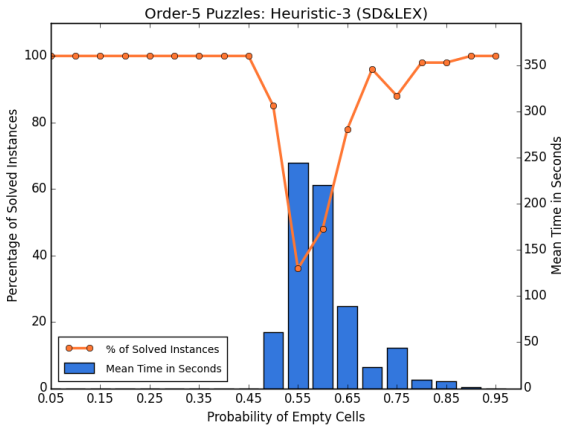

Order-4 Puzzles: Heuristic-1 (RAN&LEX)
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds

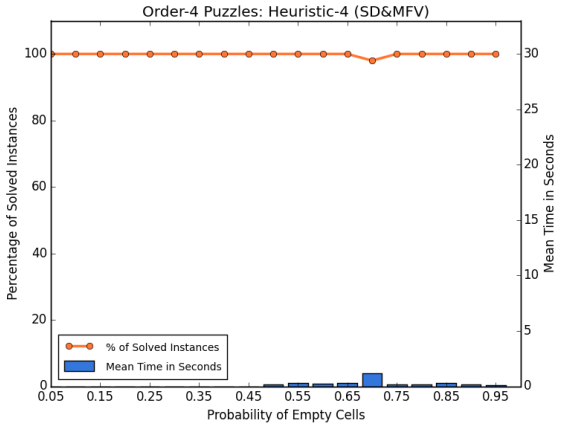

Order-5 Puzzles: Heuristic-1 (RAN&LEX)
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds

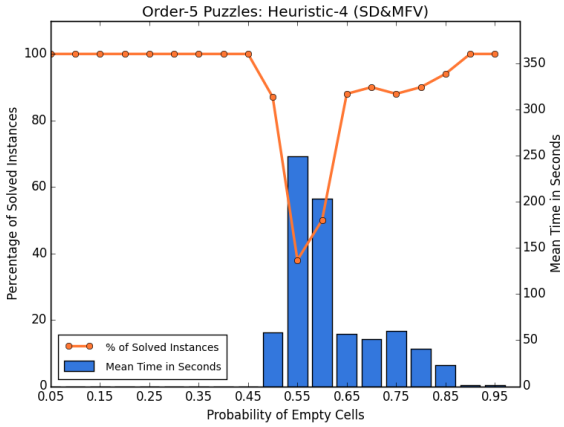

Order-4 Puzzles: Heuristic-2 (LEX&LEX)
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds

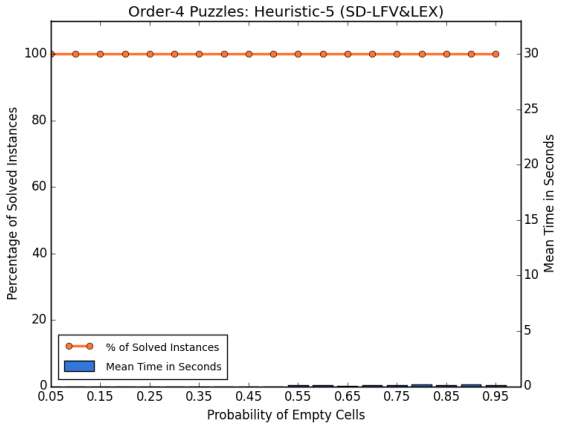

Order-5 Puzzles: Heuristic-2 (LEX&LEX)
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds

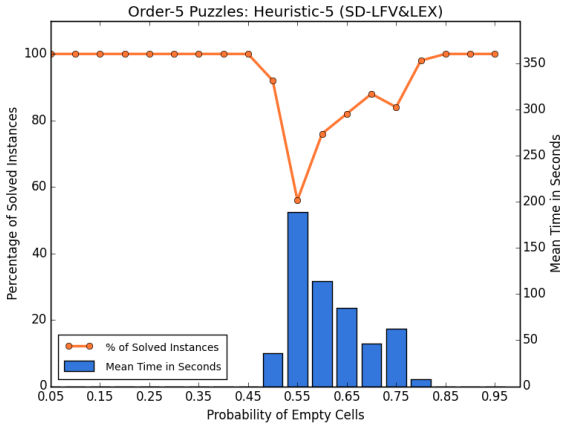

Order-4 Puzzles: Heuristic-3 (SD&LEX)
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds

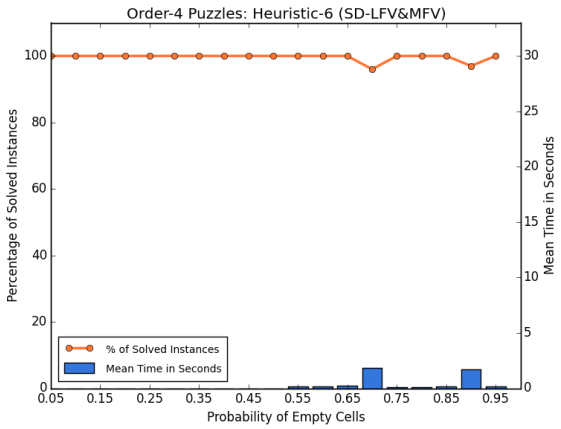

Order-5 Puzzles: Heuristic-3 (SD&LEX)
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds

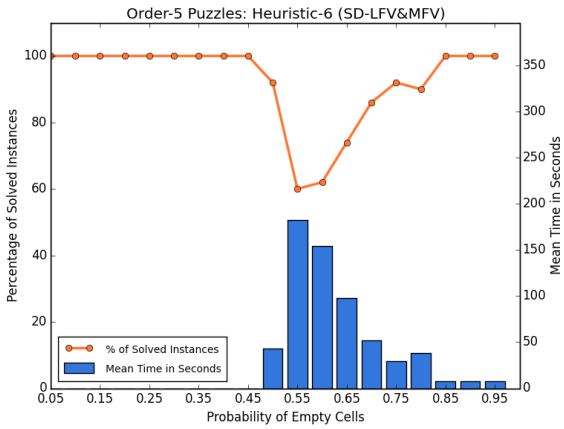

Order-4 Puzzles: Heuristic-4 (SD&MFV)
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds

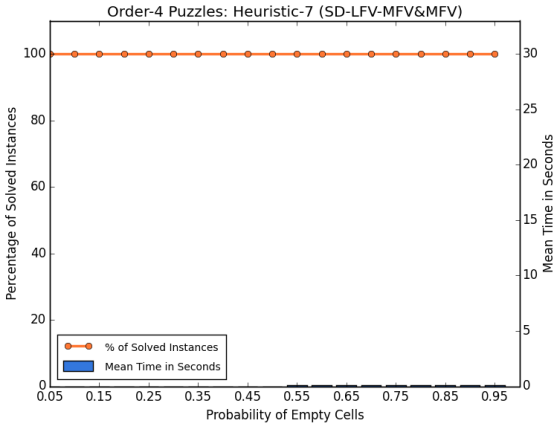

Order-5 Puzzles: Heuristic-4 (SD&MFV)
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds

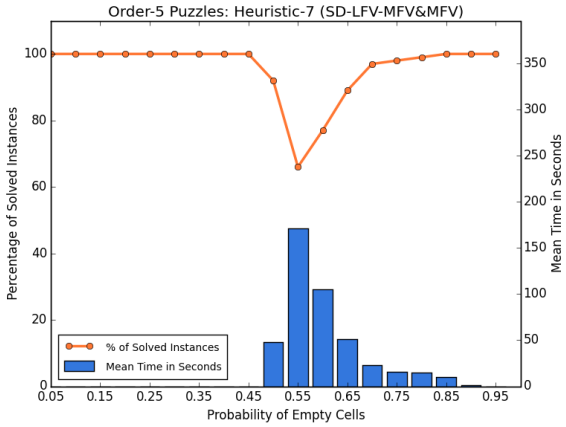

Order-4 Puzzles: Heuristic-5 (SD-LFV&LEX)
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds

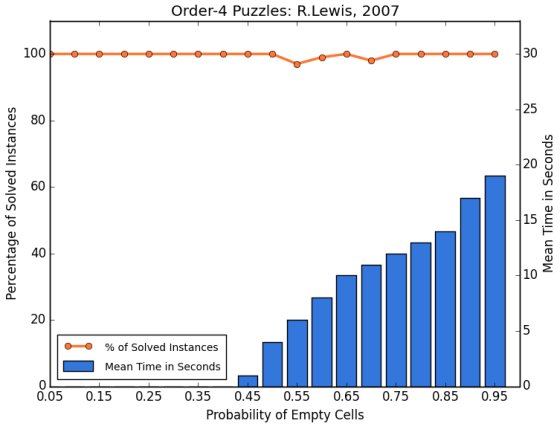

Order-5 Puzzles: Heuristic-5 (SD-LFV&LEX)
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds

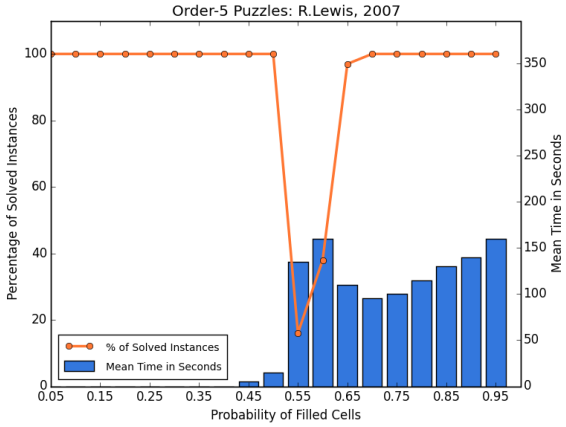

Order-4 Puzzles: Heuristic-6 (SD-LFV&MFV)
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds

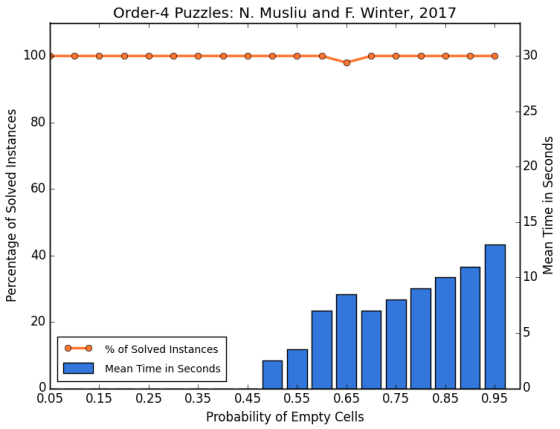

Order-5 Puzzles: Heuristic-6 (SD-LFV&MFV)
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds

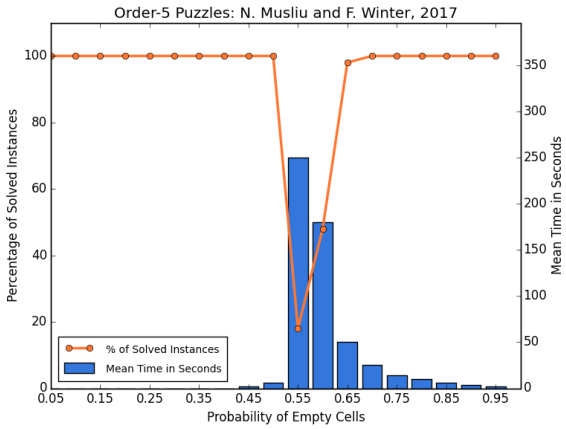

Order-4 Puzzles: Heuristic-7 (SD-LFV-MFV&MFV)
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds


Order-5 Puzzles: Heuristic-7 (SD-LFV-MFV&MFV)
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds


Order-4 Puzzles: R.Lewis, 2007
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds


Order-5 Puzzles: R.Lewis, 2007
Percentage of Solved Instances
Mean Time in Seconds
Probability of Filled Cells
% of Solved Instances
Mean Time in Seconds


Order-4 Puzzles: N. Musliu and F. Winter, 2017
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds


Order-5 Puzzles: N. Musliu and F. Winter, 2017
Percentage of Solved Instances
Mean Time in Seconds
Probability of Empty Cells
% of Solved Instances
Mean Time in Seconds

5 Conclusion

We conducted a detailed analysis of how dynamic variable and value ordering heuristics affect the search effort for Sudoku when the encoding method and the filtering algorithm are fixed. One of the most striking insights we gained from this experiment is the importance of incorporating a dynamic value ordering heuristic into the decision making process of a dynamic variable ordering heuristic. We observed that as the search space got smaller, there were still many more ties even after the first tie-breaker. If at this point the dynamic variable ordering heuristic makes a decision without considering the values in those variables domain, then it might very well go down a path where it is hard to recognize that no solution exists. However, by also considering the values in that variables domain, it can gain more insight on which path to guide the search. For instance, if there are some values that can only be fixed to few variables then it is better to guide the search with variables that have those values in their domains. So that we can either reach a solution faster or backtrack faster if it fails. Of course, this is especially true when the number of values in the domains of all the variables come from a discrete set of values, which is the case for Sudoku. However, we believe that the insights gained from this study can be carried over to other NP-Complete problems that are modeled as constraint satisfaction problems. This is because there are still many more ties left after applying a dynamic variable ordering heuristics and traditionally they are broken lexicographically without consulting any heuristic. We hope to further study the effects of dynamic variable and value ordering heuristics with more sophisticated constraint propagation algorithms.

References

[1] F. Rossi, P. van Beek, and T. Walsh. Handbook of constraint programming. *Elsevier*, 2006.

[2] F. Barber and M. A. Salido. Introduction to constraint programming. *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial*, 20:13–30, 2003.

[3] R. M. Haralick and G. L. Elliot. Increasing tree search efficiency for constraint satisfaction problems. *Artificial Intelligence.*, 14:263–313, 1980.

[4] D. Frost and Dechter R. Look-ahead value ordering for constraint satisfaction problems. *IJCAI95*, pages 572–578, 1995.

[5] D. Brelaz. New methods to color the vertices of a graph. *Communications of the ACM.*, pages 251–256, 1979.

[6] C. Bessiere and J.C. Regin. Mac and combined heuristics: Two reasons to forsake fc (and cbj?) on hard problems. *CP96*, pages 61–75, 1996.

[7] H. Simonis. Sudoku as a constraint problem. *Workshop on Modelling and Reformulating Constraint Satisfaction Problems.*, pages 13–27, 2005.

[8] T. Cazenave. A search based sudoku solver. *Labo IA Dept. Informatique Universite Paris*, 2006.

[9] R. Lewis. Metaheuristics can solve sudoku puzzles. *Journal of Heuristics.*, 13:387–401, 2007.

[10] L. Chaimowicz and M.C. Machado. Combining metaheuristics and csp algorithms to solve sudoku. *Games and Digital Entertainment (SBGAMES), 2011 Brazilian Symposium on*, pages 124–131, 2011.

[11] T. Pay. Some enhancement methods for backtracking-search in solving multiple permutation problems. *PhD Dissertation, Graduate Center of New York, CUNY.*, 2015.

[12] T. Pay and J. L. Cox. Encodings, consistency algorithms and dynamic variable-value ordering heuristics for multiple permutation problems. *International Journal of Artificial Intelligence*, 15(1):33–54, 2017.

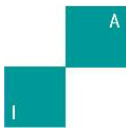
[13] N. Musliu and F. Winter. A hybrid approach for the sudoku problem: Using constraint programming in iterated local search. *IEEE Intelligent Systems*, 32(2):52–62, 2017.

[14] I. Dotu, A. Del Val, and M. Cebrian. Redundant modeling for the quasigroup completion problem. *The International Conference on Principles and Practice of Constraint Programming -03.*, pages 288–302, 2003.

[15] B. M. Smith and S. A. Grant. Trying harder to fail first. *Research Report Series-University of Leeds School of Computer Studies Lu Scs Rr*, 1997.

[16] J. C. Beck and R. J. Prosser, P.and Wallace. Trying again to fail-first. In *International Workshop on Constraint Solving and Constraint Logic Programming*, pages 41–55. Springer, 2004.

Inteligencia Artificial, 22(63) (2019), 16-38
doi: 10.4114/intartif.vol22iss63pp16-38



INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

Stereo Matching through Squeeze Deep Neural Networks

Gabriel D. Caffaratti[1,2,A], Martin G. Marchetta[1,B], and Raymundo Q. Forradellas[1,C]

[1]Laboratorio de Sistemas Inteligentes (LABSIN), Facultad de Ingeniería - Universidad Nacional de Cuyo, Centro Universitario, Mendoza, Argentina
[2]CONICET

[A]gabriel.caffaratti@ingenieria.uncuyo.edu.ar
[B]martin.marchetta@ingenieria.uncuyo.edu.ar
[C]kike@uncu.edu.ar

Abstract. Visual depth recognition through Stereo Matching is an active field of research due to the numerous applications in robotics, autonomous driving, user interfaces, etc. Multiple techniques have been developed in the last two decades to achieve accurate disparity maps in short time. With the arrival of Deep Leaning architectures, different fields of Artificial Vision, but mainly on image recognition, have achieved a great progress due to their easier training capabilities and reduction of parameters. This type of networks brought the attention of the Stereo Matching researchers who successfully applied the same concept to generate disparity maps. Even though multiple approaches have been taken towards the minimization of the execution time and errors in the results, most of the time the number of parameters of the networks is neither taken into consideration nor optimized. Inspired on the Squeeze-Nets developed for image recognition, we developed a Stereo Matching Squeeze neural network architecture capable of providing disparity maps with a highly reduced network size without a significant impact on quality and execution time compared with state of the art architectures. In addition, with the purpose of improving the quality of the solution and get solutions closer to real time, an extra refinement module is proposed and several tests are performed using different input size reductions.

Keywords: Stereo Matching, Deep Learning, Squeeze Nets, Artificial Intelligence, Artificial Vision, Disparity Maps.

1 Introduction

Stereo Matching is a research field inspired in human capabilities, in particular the stereopsis which is the ability of gathering depth information from the pair of images retrieved by the human binocular vision. Getting this information is essential to make decisions in different applications which interact with the world, like robotic object manipulation, unmanned vehicles navigation, security systems, user interfaces, etc. Since Hannah[1] and Marr et al.[2] proposed the matching of two images to extract stereo information, a number of techniques were developed to achieve this goal. As these Stereo Matching techniques started showing similarities, Scharstein et al.[3] developed a taxonomy that precisely defined common steps on them, specified their goals and instructed the process to measure them. Since then, most of the efforts have been focused on measuring how accurate were the disparity maps obtained in the matching cost calculation and post-processing steps, and how fast they were built.

ISSN: 1137-3601 (print), 1988-3064 (on-line)

Artificial Intelligence, and in particular Machine Learning, offer different techniques to solve complex problems through the training of different models. During training, these models learn to recognize different patterns in the data to perform a classification of the input provided. This represents an advantage over standard none-trainable programmatic techniques which can only recognize static defined patterns to perform classification. In other words, the analysis of the input data is performed automatically during training based on the expected results of the training set, rather than having to recognize them manually with the problem of the misinterpretation of the data or the omission of unseen patterns. Within this field, a popular technique was the MultiLayer Perceptron (MLP) developed by Rumelhart et al.[4], a trainable artificial neural network capable of learning models through the adjustment of the weights connected the different layers of neurons through the Back-Propagation algorithm. Even though these networks were widely applied after their appearance, one of their main problems was their lack of scalability as a product of the exponential growth of the number of weights when there is a big number of inputs and outputs. A different type of networks called Convolutional Neural Networks (CNN), a specific technique of Deep Learning, was also trained through the back-propagation algorithm [5]. However, CNN started to be widely adopted only when Hinton, G. [6] shown how to train Deep Network layers independently by tuning the back-propagation algorithm. Thanks to these improvements, CNN technique offered a much scalable version of a trainable neural network.

Multiple CNN architectures have been proposed due to their simple and flexible training mechanism. In addition, frameworks like Torch[7], Tensorflow[8] or Theano[9] simplified their construction, training and test. In particular, one of the benefits of using CNN appeared when they were applied in image recognition problems, outperforming all the state of the art techniques [10], and currently surpassing the human performance[11]. Due to their success in image classification problems, CNN were also applied recently by Zbontar et al.[12] for the disparity cost calculation, bringing CNN architectures to the Stereo Matching field for the first time.

Different challenges have been presented for image classification[13] and disparity map generation[14] aiming at reducing the error rate and execution time. The size of the network in terms of number of parameters is very important, because it affects the computational cost for training and execution, and also because several applications require remote updates of the trained architectures, presenting in some cases connectivity restrictions and making the size of the network an important feature to optimize. Iandola et al.[15] proposed a CNN architecture called Squeeze Nets for image recognition which decreases the number of parameters to train and store, thus reducing the size of the network significantly. Inspired on that work, we developed a squeeze-network-based model for the generation of disparity maps for Stereo Matching, which reduces the network size in storage, while also maintaining the runtime memory, execution time and quality of the results.

Due to the importance of high quality solutions generated close to real time, different additions were proposed. First, an extra refinement module based on well known morphological filters which helps to improve the solution quality depending on the system configuration with low execution time cost. Second, multiple experiments are performed using different input reductions, decreasing the execution time and making the systems suitable for real time applications with low impact on the quality. Both additions makes our solution an alternative to be used in embedded devices with less memory and resources. In addition with the usage of a Squeeze Net architecture, the reduction of the parameters of the network decreases the communication time in remote systems were the network weights needs to be replaced due to a better trained model or any other reason.

This paper is composed by a review of the state of the art techniques presented in Stereo Matching (Section 2), an explanation of our proposed architecture and extra refinement module (Section 3), experiments performed based on a case of study (Section 4) and the conclusions and future work proposals (Section 5).

2 Literature Review

The construction of disparity maps consist in calculating the distance between various points or sections in a scene according to the position of the cameras. This means the disparity map defines a sort of depth image where the shape of different objects can be presented in different colors or gray-scale according to how close are these to the cameras. This task has several complexities due to the nature of the

variable characteristics of the pictures and the ambiguous information they contain. Researchers have dealt with these ambiguities by making different assumptions of the images or data contained on it. The first pair of assumptions taken were uniqueness and continuity. Uniqueness establishes that each item of each image must be assigned with at most one disparity value. This condition relies on the assumption that an item correspond to something that has a unique physical position. Continuity states that disparity varies smoothly except on object boundaries where there are depth discontinuities[2]. Another important assumption is the epipolar rectification of the images which reduces the matching process to one dimension, or in other words a matching calculate over an horizontal line. Based on these assumptions stereo matching techniques were able to proceed with the matching of objects. However, we are far away from resolving all the different problems in the topic. Other important problems in stereo matching are the occlusions, textureless or repetitive texture surfaces, shape of the objects, differences in the intensities or noise in the images among others.

At side of the problems presented above, researchers found a number of mechanisms to retrieve dense disparity maps which can be divided in two groups: the ones detached of CNN and the ones based on these type of architectures.

2.1 CNN detached Stereo Matching techniques

Since the taxonomy proposed in [3] multiple techniques were proposed for the retrieval of dept information. These techniques can be categorized in two main groups based on the way the disparity map is calculated. The solutions that calculates the matching cost comparing a windows of neighbor pixels to gradually build disparity maps are considered local methods. Opposite to this, the solutions that retrieve a complete map and iteratively optimize it are considered global methods.

An extensive survey of local and global stereo matching algorithms where different comparisons and measures are made can be found in Hamzah et al.[16] work.

2.2 CNN based Stereo Matching techniques

CNN architectures marked a huge improvement in Artificial Vision areas. Particularly, this kind of artificial networks brought the attention of researchers when the Alexnet created by Krizhevsky et al.[10] reduced more than 10% the error rate on image classification problems reaching a 15.3%. Such achievement caused a revolution in this research field, having multiple CNN architectures proposed in the last five years. Zeiler et al.[17] was able to tune the hyperparameters of AlexNet creating the ZFNet and obtaining a 14.8% error rate. Later on GoogLeNet architecture by Russakovsky et al.[18] introduced the concept of inception modules enlarging the number of layers of the net with very small convolutional layers obtaining a 6.61% error and a considerable reduction of the parameters. An impressive improvement of the accuracy was achieved by He et al.[11] proposing the addition of residual links between layers in the ResNet achieving an error rate of 3.58%. A parameter reduction approach presented by Iandola et al.[15] decreased the network parameters size by 50x obtaining the same error rate as AlexNet.

All the previously mentioned networks were applied to the recognition of images with great success, however they had potential for other Artificial Vision problems like Stereo Matching. Zbontar et al.[12] worked on this idea inspired by the popularity of CNN and Mei et al.[19] work. He proposed a CNN based cost matching architecture with a series of post-processing steps described in Mei et al. paper obtaining an error rate of 2.63% on their accurate network version. It worth to mention a better implemented architecture designed by Shaked et al.[20] combining Zbontar et al.[12] suggested network with the residual networks proposed by He et al.[11]. In addition, Yang et al.[21] used CNN to perform not only the cost calculation but also to learn the smoothness constraint. Our work is highly inspired on [11, 12, 19] papers combined with the proposed Squeeze nets proposed in [15]. Since Zbontar et al.[12] network provided reproducible results in our tests and due to a closer similarity of their architecture with our solution, compared with other solutions in the field of stereo matching, our results are later compared with this work in Section 4.

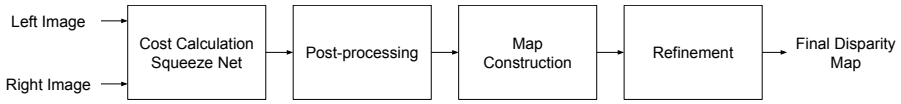


Figure 1: Four step proposed Stereo Matching system

2.3 Post-processing algorithms

In many occasions, the disparity maps obtained from the matching cost calculation process can be inaccurate, have occluded areas or unmatched sections. In order to improve their quality, different post-processing algorithms have been developed. An adaptive window algorithm based on the color similarity is proposed as cross-based cost aggregation by Ke Zhang et al.[22]. The method suggested by Hirschmüller[23] called semi-global matching (SGM) improves the smoothness term in the energy functions performing fast approximations of the neighbor pixels using Mutual Information in multiple directions. Another interesting post-processing method is proposed by Yao et al.[24] where the left and right based disparity maps are interpolated to get a better approximation of the mismatching areas. It worth to mention Williem et al.[25] who used a CNN network to increase the accuracy under the cost aggregation module. In addition, the disparity map results can be improved performing a quadratic interpolation of neighbor pixels as suggested by Miclea et al.[26].

3 Proposed System

Our model is composed of four steps as depicted the figure 1, following the common stereo matching taxonomy suggested by Scharstein et al.[3]. The system performs a cost calculation through a CNN extracting features of the left and right images individually and checking their similarity on a final layer, then a well known post-processing algorithm is applied in a cost aggregation module, then a disparity map is constructed based on the matching costs from the previous steps, and finally different known interpolation and image refinement algorithms are executed to obtain an optimized dense disparity map. Our main contribution can be found in the cost calculation module where a modified Squeeze Net architecture[15] is used to build a raw disparity map.

3.1 Cost Calculation module

The cost calculation of a common stereo matching algorithm is basically the comparison of pixels from different images. We implemented this functionality combining a CNN architecture and refinement algorithms as described below.

3.1.1 Network Architecture

The proposed deep network architecture for cost calculation first processes the pair of images separately, executing two passes on the same layers, and then joins the results in a final layer, as depicted in the figure 2. The cost calculation network provides a raw disparity map, in the form of a 3D matrix where each element (i, j, k) is the matching cost of pixel (i, j) for disparity k.

Each image passes through a set of layers that generate feature maps. The feature maps obtained from each image are then fed into a similarity calculation layer at the end. The weights of the feature extraction layers are shared at the time of processing both the left and right images. The last layer performs a dot product between the feature maps of the left and right images, which were previously normalized. The normalization and dot product steps are equivalent to the cosine similarity measure which is used to retrieve a cost cube of a raw disparity map based on the similarity of the feature maps.

The first three layers are a Convolutional layer, a ReLU activation layer and a Pooling layer. The parameters $< K, S, P >$ shown in figure 2 are the *kernel size*, *stride* and *padding* of the module. FM represents the number of features to be generated by the convolutional layer. The fire modules are

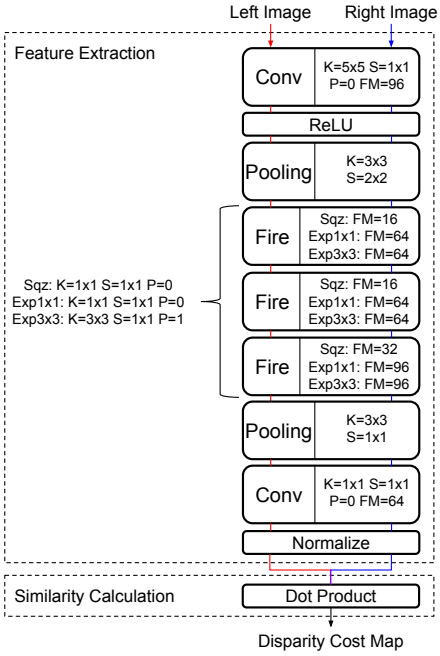


Figure 2: Cost Calculation network architecture. The left and right images are passed through the same feature extraction layers, and both results are then processed in the similarity calculation layer

composed of a set of layers. Each fire module shares the same $< K, S, P >$ parameters in the internal convolutional layers represented with Sqz, $Exp1x1$ and $Exp3x3$, but they differ in terms of feature maps.

3.1.2 Fire Module

The Fire module, depicted in the figure 3, is composed of two sequential steps with the purpose of squeezing the number of features received, and then expanding them in the next step, inspired by the Squeeze Net presented by Iandola et al.[15].

This model offers two advantages in terms of reduction of parameters. First, it reduces the number of input parameters to the convolutional layers. For example, an initial convolutional layer with 1x1 kernels (CL1x1) reduces by a factor of 9 the number of parameters of the layer when compared to a convolutional layer with 3x3 kernels (CL3x3), helping to decrease the size of the network. In addition, the number of features produced by the squeeze step is significantly reduced in contrast with other networks, generating a minimized input for the next CL3x3 of the expand module.

Second, in order to increase the number of features produced by the Fire module, a parallel CL1x1 is added, thus generating features maps that should otherwise be produced by the CL3x3. Consequently, the input required by both convolutional modules in the expand component to provide variety of feature maps can be reduced and so the number of parameters to be stored. It worths to mention that the CL1x1 of the expand module has padding equal to 1 in order to maintain the output feature map size equivalent to the CL3x3 one.

3.1.3 Cost Calculation

Since we assume that the images are rectified, their epipolar lines are completely horizontal and vertically coincident on both images. Unless they are occluded, the corresponding pixels can then be found in the same row, but differing by a certain disparity. Conversely, having the disparity corresponding to a pixel of one image, we can get the position of the matching pixel on the other image.

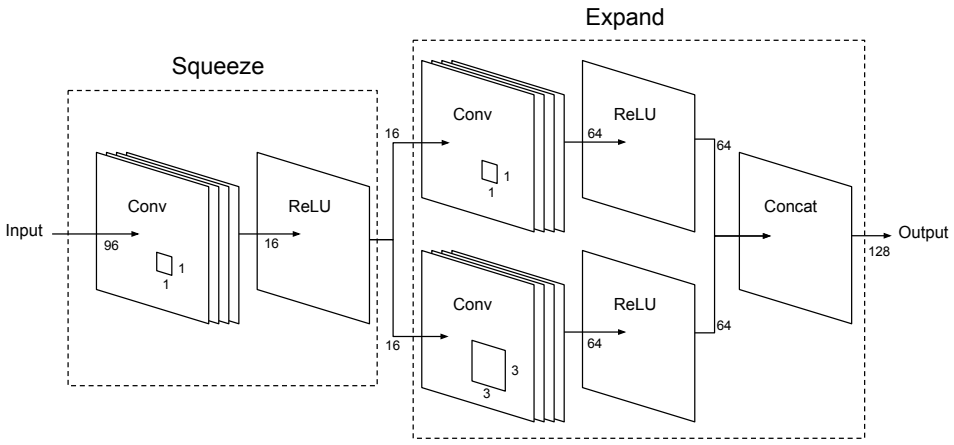


Figure 3: Fire Module layers composition. The number of feature maps in the output of the layers is illustrative and changes in each one of the fire modules of the architecture.

To train the proposed Squeeze Net we used the KITTI 2012 [27] or KITTI 2015 datasets[28, 29] which offers both left and right images rectified and a group of ground truth disparities. Taking these latter ones along with one of the images, we can create training patches with examples of correct and incorrect matches as suggested by Zbontar et al.[12]. Each patch is a matrix containing pixel intensities. The negative examples are obtained by adding an offset to the disparity provided by the ground truth, and getting the patch from the new position. Also we augmented the training samples by performing a series of transformations on the patches like rotation, brightness and saturation level modifications, among others. In this way we can train the network with positive and negative matching examples for each position and disparity.

Our cost function is described in the equation 1, where P^L and P^R are the left and right patches, $\mathbf{p}$ is the position of the center of the matrix representing the patch, d is the horizontal displacement and f is the network output, that measures the similarity of the patches, being zero when they match exactly.

$$C_{\text{Squeeze-Net}}(\mathbf{p}, d) = f\big(P^L(\mathbf{p}), P^R(\mathbf{p} - d)\big) \tag{1}$$

Because the output size we want to produce is fixed, we need to calculate the patch size ws_k to produce it. The patch size is calculated only once, and is defined by equation 2, being k the number of the last convolutional or pooling layer, $<S,\ P,\ K>$ the module's *stride*, *padding* and *kernel size* respectively, and ws_{i-1} the output size of the module. Notice that ws_k is calculated iteratively, where variable i is initialized to 1 and it is increased step by step up to the number of convolutional and pooling layers, accumulating its values. In other words, to obtain the network minimum input size, equivalent to the patch size, we need to go backwards from the last convolutional or pooling layer to the first one.

$$ws_i = \begin{cases} 1 & i = 1 \\ \big((ws_{i-1} - 1) * S_{k-i+1}\big) - (2 * P_{k-i+1}) + K_{k-i+1} & 1 < i \leq k \end{cases} \tag{2}$$

3.1.4 Network Training

During training, the network analyzes pairs of left/right patches, and the result is a measure of their similarity, as stated in equation 1. This means the lost function used performs a binary classification of the pair of patches as matching or not matching, converting the cost calculation step in a classification problem. The loss function to be minimized to adjust the networks parameters applies the hinge-loss to pairs of positive and negative matching samples, as defined by equation 3

$$L = max(0, margin + m^- - m^+) \tag{3}$$

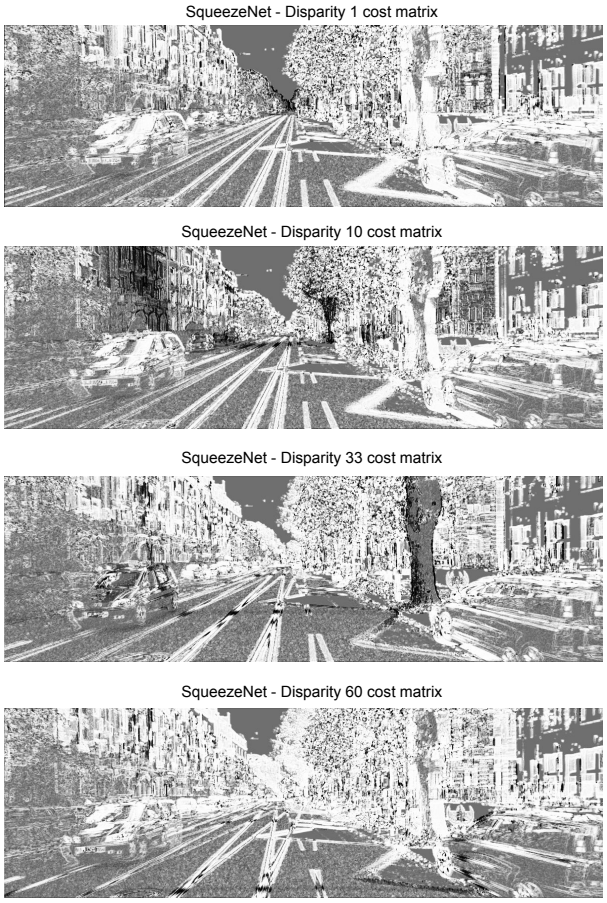


Figure 4: Different disparity matrices of the cost cube representing the cost calculated for each pixel.

where m^- and m^+ are the results of the negative and positive matching examples, and $margin$ is the tolerance margin. When the similarity of the positive example is greater than the similarity of the negative one by a certain margin the loss will be zero. In our experiments the $margin$ used was 0.2.

3.1.5 Results Produced

The Cost Calculation module produces a cost cube composed of a series of matrices along the cube's depth, where each of these represents the cost of each pixel under a certain disparity. In case each matrix of the cost cube is graphed a distorted image is obtained where the pixels matching the matrix disparity appear in black as depicted in the figure 4.

In case the only module executed is the Cost Calculation module, the Disparity Map Generation process (explained in Section 3.3) needs to be executed in order to obtain the disparity map from the cost cube. This procedure and result is depicted in the figure 5.

3.2 Post-processing module

The cost calculation network in combination with a disparity map construction process (explained in Section 3.3) provides a representation of a raw disparity map, in the form of a 3D matrix where each element (i, j, k) is the matching cost of pixel (i, j) for disparity k. The quality of this map can be improved through a set of post-processing steps. In our work we only perform a Cross-based Cost Aggregation proposed by Zhang et al.[22] and Semi-Global Matching inspired by Hirschmüller et al.[23].

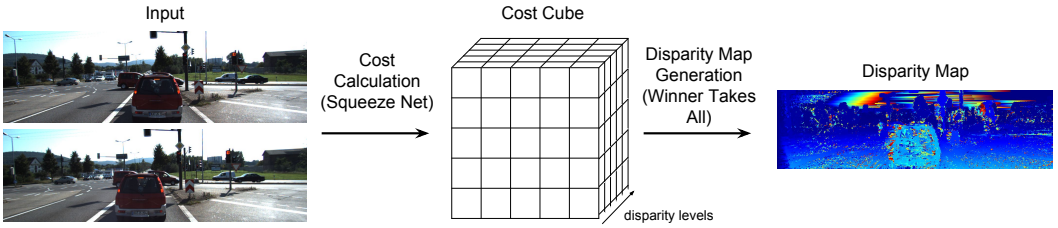


Figure 5: Steps, subproducts and result obtained executing Cost Calculation module and Disparity Map Construction process only

3.2.1 Cross-based Cost Aggregation (CBCA)

This method generates an adaptive window starting from the pixel position. The window is generated by retrieving the pixels with similar intensity hoping they belong to the same object. First, for each pixel, a cross is built by checking if the difference between the intensities of that pixel and the up, down, left and right arms is under a certain threshold. The length of these arms is extended as long as the threshold is not exceeded or up to a certain arm length limit. Then, for each pixel of the vertical arm, a horizontal arm is added to the window following the same intensity and distance verification. This procedure is performed also for the matching image and then both windows are joined to construct a final window. The equation 4 depicts the criteria used to construct each up and down arm (vertical arm) and the equation 5 shows the same condition used to construct each left and right arms (horizontal arms) for each pixel of the vertical arm created in the previous equation.

$$Arm_V(\mathbf{p}) = \begin{cases} I(\mathbf{p}_{U|D}) & |I(\mathbf{p}) - I(\mathbf{p}_{U|D})| < cbca\_intensity \wedge ||\mathbf{p} - \mathbf{p}_{U|D}|| < cbca\_distance \\ None & otherwise \end{cases} \tag{4}$$

$$Arm_H(\mathbf{p}_{V_i}) = \begin{cases} I(\mathbf{p}_{L|R)_{V_i}} & |I(\mathbf{p}) - I(\mathbf{p}_{L|R_{V_i}})| < cbca\_intensity \wedge ||\mathbf{p} - \mathbf{p}_{L|R_{V_i}}|| < cbca\_distance \\ None & otherwise \end{cases} \tag{5}$$

Where $\mathbf{p}$ is the position selected pixel to create region, $\mathbf{p}_{U|D}$ is the position of the pixel in the same vertical line of $\mathbf{p}$, $\mathbf{p}_{V_i}$ is the position of one of the pixels in the vertical arm Arm_V, $\mathbf{p}_{L|R_{V_i}}$ is the position of the pixel in the same horizontal line of the vertical arm pixel $\mathbf{p}_{V_i}$, $cbca\_intensity$ and $cbca\_distance$ are the intensity and distance thresholds respectively.

The support region of the selected pixel is built by joining all the horizontal arms of the vertical arm as depicted in the equation 6.

$$R^L(\mathbf{p}) = \bigcup_{\mathbf{p}_{V_i}}^{\mathbf{p}_{V_i} \in Arm_V} Arm_H(\mathbf{p}_{V_i}) \tag{6}$$

Then a combined support region is created by joining the left and right images' regions as shown in the equation 7.

$$U_d(\mathbf{p}) = \{\mathbf{q} | \mathbf{q} \in R^L(\mathbf{p}), \mathbf{q} - d \in R^R(\mathbf{p} - d)\} \tag{7}$$

were R^L and R^R are the left and right images regions for the pixel position $\mathbf{p}$. Finally, an average of the cost of all the pixels over the combined region is obtained and assigned to the pixel position used at the beginning of the process. This can be seen in the equation 8.

$$C_{CBCA}(\mathbf{p}) = \frac{1}{|U_d(\mathbf{p})|} \sum_{\mathbf{q} \in U_d(\mathbf{p})} C_r(\mathbf{q}, d) \tag{8}$$

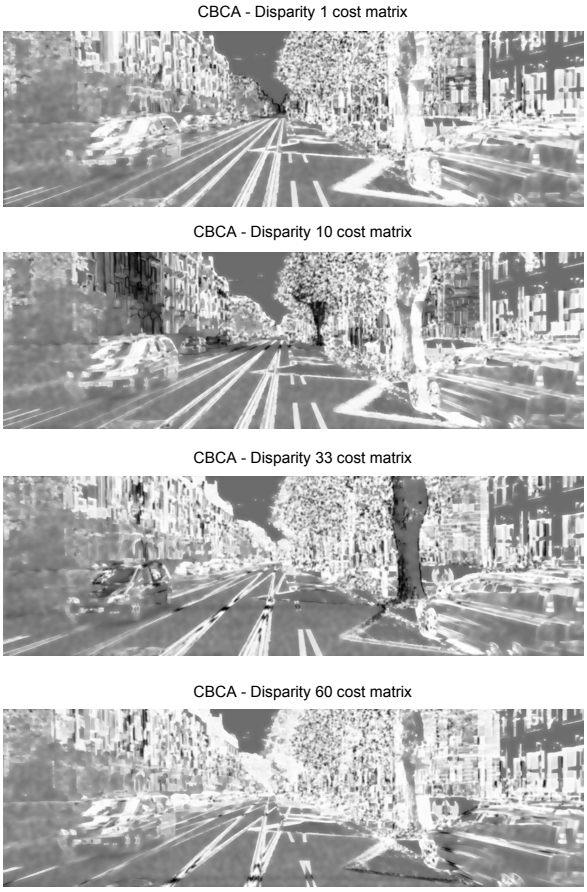


Figure 6: Different matrices of the cost cube representing the cost aggregation for each disparity after the CBCA procedure.

This process can be iterated different number of times since the support regions overlap between them producing different results on each iterations. It worth to mention the input of this procedure is each matrix of the cost cube produced by the Cost Calculation module or any previous procedure of the post-processing module. Then the final product of the CBCA is an improved cost cube where the matching pixels of each disparity are better defined with less noise as depicted in the figure 6.

3.2.2 Semi-global Matching (SGM)

An important refinement of the disparity map is related to smoothness. Since objects usually have similar disparities, intuitively we can say that differences in the disparity of neighboring pixels should be penalized. In general, the penalty applied increases according to how strong the difference between disparities is. We defined the local neighborhood of a pixel by moving 1 pixel away of if in the up, down, left and right directions, following the suggestion in [12]. If we call $\mathbf{r}$ the vector of the directions where the pixels $\mathbf{q}$ are found, a set of adjustments $C_{\mathbf{r}}(\mathbf{p}, d)$ for the matching cost $C_{\mathbf{r}(\mathbf{p},d)}$ computed by the CNN are calculated through equation 9 (and later averaged, as described below).

$$C_{\mathbf{r}}(\mathbf{p}, d) = C_D(\mathbf{p}, d) - \min_i C_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, i) + \min \Big(C_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, d), \\ C_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, d-1) + P_1, C_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, d+1) + P_1, \min_i C_{\mathbf{r}}(\mathbf{p} - \mathbf{r}, i) + P_2 \Big) \quad (9)$$

Here C_D is the matching cost calculated by the CNN, P_1 is the penalty when the difference between the disparity $D_{\mathbf{p}}$ and $D_{\mathbf{q}}$ of pixels $\mathbf{p}$ and $\mathbf{q}$ in the local neighborhood is 1. P_2 is the penalty when that difference is higher than 1, and i are the valid disparities. The second term of the equation compensates for cases where the values of the third term of the equation grow too large, for smoothing special cases (e.g. occluded or mismatching pixels).

The penalty values P_1 and P_2 varies according to disparity of the pixel compared with neighbors in edges of objects. Therefore, penalties are lower when pixels are detected in borders. The parameters are defined as follows:

$$D_1 = |I^L(\mathbf{p}) - I^L(\mathbf{p}-\mathbf{r})| \qquad D_2 = |I^R(\mathbf{p}-\mathbf{d}) - I^R(\mathbf{p}-\mathbf{d}-\mathbf{r})|$$

$$\begin{array}{lll} P_1 = P_1, & P_2 = P_2, & \text{if } D_1 < D_{sgm} \wedge D_2 < D_{sgm}; \\ P_1 = P_1/Q_2, & P_2 = P_2/Q_2, & \text{if } D_1 \geq D_{sgm} \wedge D_2 \geq D_{sgm}; \\ P_1 = P_1/Q_1, & P_2 = P_2/Q_1, & \text{otherwise} \end{array} \tag{10}$$

where I is the pixel intensity. In case the disparity shows a big discontinuity in the disparity map, i.e. when D_1 and D_2 are equal or bigger than a certain D_{sgm}, the penalty is reduced by a large factor Q_2 as it is assumed that it is a steep border. In case just D_1 or D_2 meets this condition, the border is not that steep so the penalty is reduced by a smaller factor Q_1. Otherwise the case is not a border. In case of vertical directions a different factor V is used to reduce P_1 as the disparities changes shown in ground truth images are more frequently vertical. After computing all these values for each direction, the final smoothed cost is an average of the results obtained (equation 11).

$$C_{\text{SGM}}(\mathbf{p}, d) = \frac{1}{4}\sum_{\mathbf{r}} C_{\mathbf{r}}(\mathbf{p}, d) \tag{11}$$

Similar to the CBCA process described before, the input of the SGM process are the different matrices of the cost cube produced by the Cost Calculation module or any previous post-processing procedure. Then, the product obtained after executing the SGM process is a refined cost matrix for each disparity of the cost cube with better defined matching pixels as depicted in the figure 7.

3.2.3 Results Produced

Once the different post-processing procedures are executed the result is an improved cost cube where the matching cost for each disparity is better defined. Then, this cube can be provided to the Map Generation module (explained in Section 3.3) to produce a disparity map with probably less errors than another one where no post-processing module was executed. The subproducts and complete process are depicted in the figure 8.

3.3 Map Construction

The disparity map is constructed by gathering the minimum cost from the cube of matching costs for each pixel position. This strategy is called winner-takes-all and is defined in the equation 12.

$$D(\mathbf{p}) = \arg\min_{d} C(\mathbf{p}, d) \tag{12}$$

3.4 Refinement module

Once the disparity map is constructed, problems like mismatching or occluded pixels can reduce its accuracy. This can be improved through different interpolation and refinement steps. These steps were inspired by Mei et al.[19] work. The execution order is depicted in figure 9.

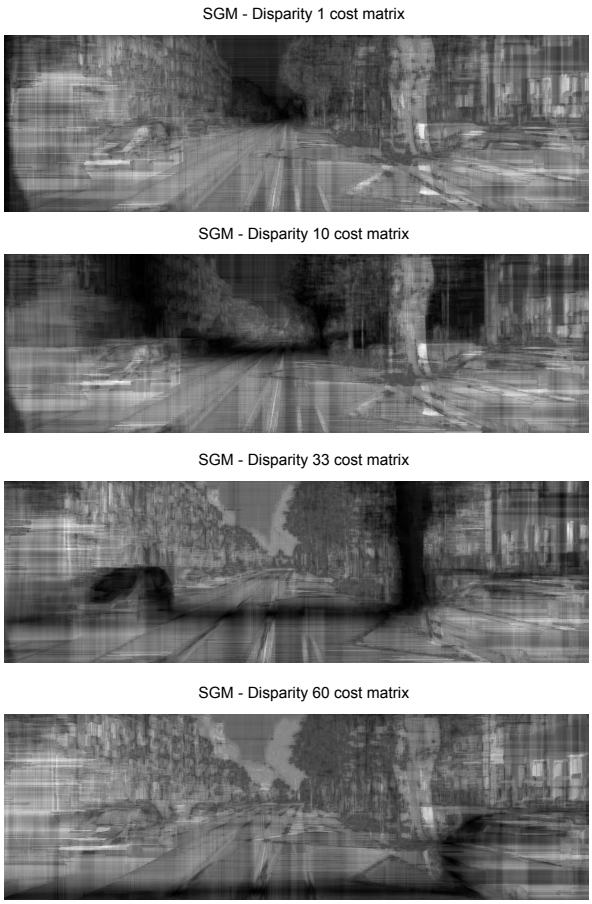


Figure 7: Different matrices of the cost cube representing the cost aggregation for each disparity after the SGM procedure.

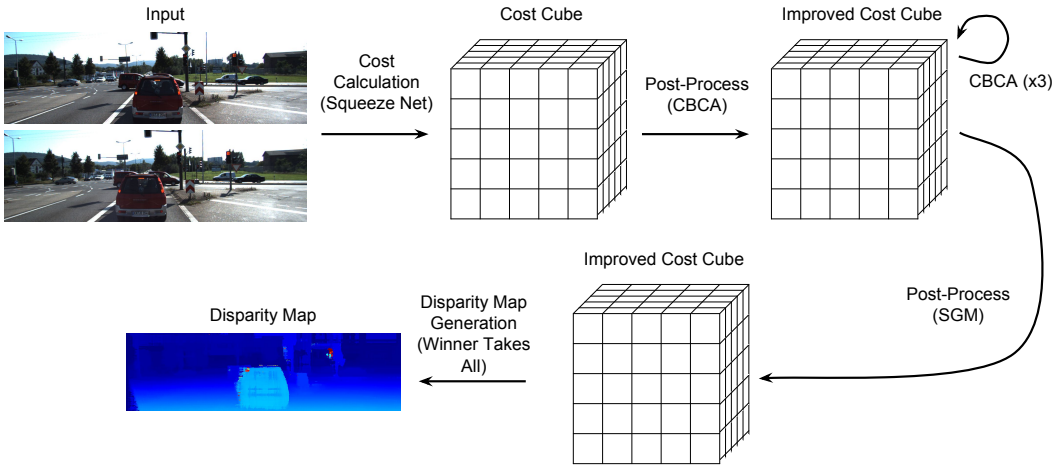


Figure 8: Steps, subproducts and result obtained executing Cost Calculation module followed by four iterations of CBCA, then the SGM process was executed and finalizing with a Disparity Map Generation process

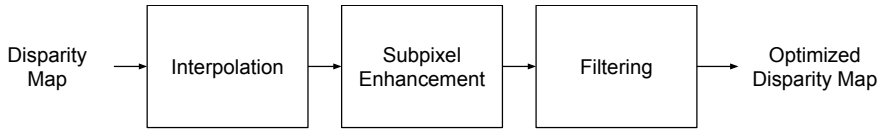


Figure 9: Refinement steps performed after the map construction module

3.4.1 Interpolation

The disparity map can be calculated either using the left image as reference or the right image. Changing the reference image produces different maps since the occluded pixels are different on each one. Having both disparity maps can help us to determine which are those occluded object by comparing the disparities of a matching pixel. If the disparities match (the absolute difference is less than one), we can consider them as correct. If the disparity of the pixel in one map matches the disparity of a pixel in the other map other than the corresponding one, we consider it as incorrect. If the disparity does not match any other disparity in the other map, it is an occluded pixel.

The occluded pixels disparity is obtained by looking at the nearest correct pixel at the left. For mismatching pixels we look for a disparity value as the median of sixteen directions pixels around them.

3.4.2 Sub-pixel Enhancement

In this step we use the SGM cost of pixel p and disparity d and its closest "disparity" neighbors to get a smoothing subtraction term, to improve the result, as shown in equation 13.

$$D(\mathbf{p},d) = D(\mathbf{p},d) - \frac{C_{SGM}(\mathbf{p},d+1) - C_{SGM}(\mathbf{p},d-1)}{2\Big(C_{SGM}(\mathbf{p},d+1) - 2C_{SGM}(\mathbf{p},d) + C_{SGM}(\mathbf{p},d-1)\Big)} \tag{13}$$

3.4.3 Filtering

This module applies a median filter with a 5x5 kernel followed by a bilateral filter, for the purpose of smoothing disparity changes without affecting the edges.

3.4.4 Results Produced

After the disparity map is constructed and the different refinement steps are executed the result is a filtered and smoothed disparity map as depicted in the figure 10.

3.5 Erode-Dilate module

With the purpose of improving the accuracy of the system, an extra module is proposed conformed by a binary mask generator and two morphological filters which can be applied to a disparity map. These morphological filters are the well known erosion and dilation processes which are basically conformed by a kernel of a certain size which is passed through all the disparity map mask. This module is depicted in the figure 11.

Once both filters are applied, the isolated pixels over a certain threshold surrounded by other pixels under the threshold are marked with the lowest disparity, as background, removing the disparity noise in early steps of the system. Taking into consideration that the disparities of the background of an image are usually smooth, this module tends to improve the accuracy of the generated disparity map by removing isolated foreground disparity values.

3.5.1 Mask Generation

A simple mask is generated by checking each value of the disparity map against a certain threshold t as shown in the equation 14.

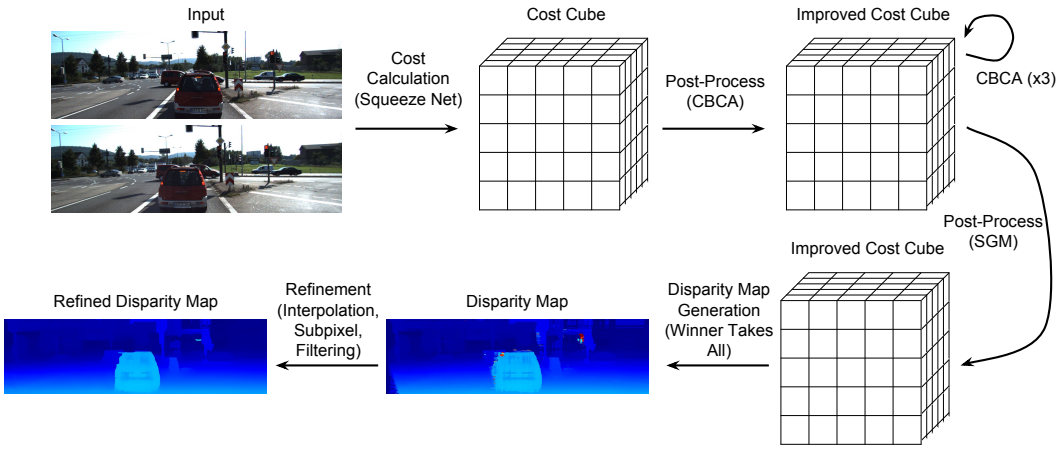


Figure 10: Steps, subproducts and result obtained executing Cost Calculation module followed by four iterations of CBCA, then the SGM process was executed and finalizing with a Disparity Map Construction and a full Refinement process

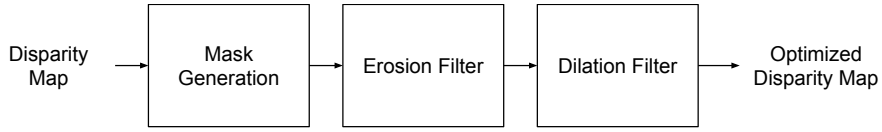


Figure 11: Steps performed in the Erode-Dilate module after the map construction or refinement module

$$D_{mask}(\mathbf{p}, d) = \begin{cases} d & d \geq t \\ 0 & otherwise \end{cases} \tag{14}$$

Every value under the threshold is considered as background, otherwise as foreground. In this way the background disparities are set to zero, generating a binary map composed by background (zero disparity) and foreground (any disparity over the threshold) values.

3.5.2 Erosion Filter

This step is composed by a binary erosion of the image performed by a binary kernel passed over each position of the disparity map obtained from the previous step. The kernel is defined with some of its values set as one. Taking the disparity map mask as binary image, the kernel is compared on each location of the binary image to verify if the kernel values matches the region being compared. In case region doesn't match, the center position of the region is set to zero, otherwise this value is kept unchanged as denoted in the equation 15.

$$D_{eroded}(\mathbf{p}, d) = D_{mask}(\mathbf{p}, d) \ominus K = \{d \in D_{mask} \mid K_{\mathbf{p}} \subseteq D_{mask}(\mathbf{p}, d)\} \tag{15}$$

Here K is the kernel, $K_{\mathbf{p}}$ is the kernel at the position $\mathbf{p}$ and d is the disparity at the center of that position.

The erosion process has the effect of removing the isolated disparities of the disparity map, but also makes the shapes in the conformed binary image thinner. This undesired effect can be mitigated applying a dilation filter over the result.

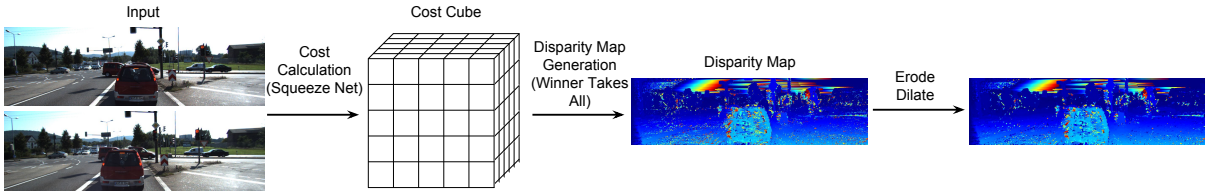


Figure 12: Steps, subproducts and result obtained executing Cost Calculation module followed by a Disparity Map Construction and the Erode-Dilate refinement process

3.5.3 Dilation Filter

As mentioned in the previous step, the execution of a dilation filter is required to mitigate some undesired effects of the erosion filter. This process has the same behavior of the erosion filter except that performs an union with the region being compared if any of the of the values set in the kernel matches. This can be seen in the equation 16.

$$D_{dilated}(\mathbf{p}, d) = D_{eroded}(\mathbf{p}, d) \oplus K = \{d \in D_{mask} \mid K_{\mathbf{p}} \cup D_{eroded}(\mathbf{p}, d) \wedge (K_{\mathbf{p}} \cap D_{eroded}(\mathbf{p}, d)) \neq \varnothing\} \quad (16)$$

As in in the equation 15, here K is the kernel, $K_{\mathbf{p}}$ is the kernel at the position $\mathbf{p}$ and d is the disparity at the center of that position.

It's important to notice that the disparities added by the dilation process should be retrieved from D_{mask} in order to obtain the thickening effect of the dilation process on the disparity map shapes but respecting the original disparities removed by the erosion process.

3.5.4 Results Produced

Similar to the Refinement module, the Erode-Dilate module is executed after the Disparity Map Construction module. The result is a less noisy disparity map where the isolated errors in the foreground were removed as depicted in the figure 12.

4 Case Study

In this section we present details of the environment where the different algorithms were executed, the training specifications and the system setup. The results obtained are shown in terms of parameters reduction, error rate, execution time and memory consumption. The experiments were performed on an AMD Ryzen 1700 CPU with 32 GB DDR–4 2400 MHz ram and a NVidia Titan Xp GPU. The source code of the project were all the experiments were performed can be found in a git repository[1].

4.0.1 Training set

We used the 2012 KITTI dataset [27] and 2015 KITTI dataset [28, 29] composed by 194 and 200 training pairs of images of 1240 x 376 pixels respectively. These images have a maximum of 228 disparity levels. The dataset provides around 30% of the image disparities measured with laser scanners. Leaving 40 images for test, the remaining 154 and 160 images made a training set of around 19 million positions with measured disparity on each dataset. Since we get positive and negatives examples as mentioned in Section 3.1.3, the number of training examples (sampled patches) is more than 38 million. Each sample is subject to several image transformations in order to provide different samples on each epoch. As a product of the CNN layers' kernels and padding the window size of our patches is 9x9.

[1] https://github.com/labsin-uncuyo/gdc-mc-cnn

4.0.2 Learning parameters

These training samples were provided in batches of 128 samples during 15 epochs. We used a learning rate of 0.05 which is decreased after epoch 11 by a factor of 10. The parameters used for image post-processing are the same used in Zbontar et al.[12] fast architecture.

4.0.3 Post-processing module configurations

In order to perform a fair comparison between Squeeze Net and Zbontar et al.[12] fast architecture solutions, the post-processing module we used executes only the SGM process, mentioned in Section 3.2 when using Kitti 2012 dataset. However, we included the CBCA process in our tests using Kitti 2015 to see how much affected are the memory, execution time and accuracy under this configuration. Also, the Zbontar et al.[12] accurate architecture uses both CBCA and SGM processes for post-processing. The different configurations are depicted in the table 1.

Table 1: Post-processing module configurations used over the different architectures and datasets

Configuration	CBCA (before SGM)	SGM	CBCA (after SGM)
ZBontar Fast (Kitti 2012)		•	
ZBontar Fast (Kitti 2015)	•	•	•
ZBontar Accurate	•	•	•
Squeeze Net (Kitti 2012)		•	
Squeeze Net (Kitti 2015)	•	•	•

4.0.4 Cost calculation parameters

The proposed Squeeze Deep Neural network considerably decreased the number of parameters in relation with the other two networks. This result is the product of the reduction of features in the squeeze layer of the fire module that are fed into the expansion CL3x3 module and the expansion of features through a parallel CL1x1. A comparison of the network size, the parameters involved and the reduction rate is shown in the table 2.

Table 2: Cost Calculation Deep Neural network sizes and parameters

Network	Size (KB)	# Parameters	Reduction
Zbontar Accurate	2,534	648,592	87.81%
Zbontar Fast	440	112,564	29.77%
Squeeze Net	**309**	**79,040**	-

4.0.5 System accuracy and execution time

The system was executed in two instances: with post-processing and refinement, and without post-processing. Table 3 shows the different results obtained as an average over the 40 testing images.

Table 3: System accuracy and execution time using Kitti 2012 dataset

	Error		Execution time	
Network	CNN + REF	Full method	CNN + REF	Full method
Zbontar Accurate	6.03%	2.54%	33.24 sec	33.84 sec
Zbontar Fast	8.39%	2.93%	0.33 sec	0.65 sec
Squeeze Net	11.87%	3.69%	0.58 sec	0.89 sec

Even through the error is 6% higher in case the system is executed without post-processing, the degradation of the quality including this module is below 1.5% even compared with the accurate architecture.

The execution time of the network is only comparable with the fast architecture and is over 38 times faster than the accurate architecture. The resulting disparity maps of each model, including post-processing and refinement steps, are shown in figure 13.

Figure 13: Full method generated disparity maps

4.0.6 GPU Memory consumption

There is a reduction of memory consumption by our solution, as compared to the fast architecture, making it feasible to use in smaller devices with less GPU resources as presented in table 4.

Table 4: Architecture GPU memory consumption comparison

Network	Zbontar Accurate	Zbontar Fast	Squeeze Net
GPU Mem	∼4200 MB	∼2000 MB	∼2000 MB

4.0.7 Reduction of the input images

In order to decrement the execution time of the stereo matching system and get closer to real time processing, different executions were performed using a reduced versions of the input images. At the end of system process, the output has a equivalent size of the input images. Since the ground truth is in the original size, a fair comparison of the results is performed by resizing the output to the original size, and then verifying all the measured points of the ground truth with the final disparity map obtained. The experiments were performed with different grades of input reductions over two of the previously described systems. The first one is the ZBontar et al. Fast architecture and the second one is our proposed Squeeze Net architecture. Both systems share the same post-processing and refinement modules as they were described in the previous sections, differing only in the cost calculation module composed by different CNN network architectures. However different executions were performed removing some modules in order to determine how a reduced input was affecting in the different parts of the system. A table of the different system configurations is shown in the table 5.

The resizing of the inputs' dimensions on both networks shown a decrement of the execution time using both Kitti 2012 and 2015 datasets. The ZBontar et al. architecture best result reached a 4.36 times reduction factor in terms of execution time of the system running only the cost calculation and

Table 5: Configurations used to run the experiments with reduced input size

Configuration	Cost Calculation	Post-Processing	Refinement
C1	•		
C2	•		•
C3	•	•	
C4	•	•	•

postprocessing modules when an input of 50% size was used. In case of the Squeeze Net architecture there was a 3.39 times reduction factor executing the same modules with the same input size. Both systems were close to real time solutions having 0.15 and 0.26 seconds of execution time respectively. Similar but better results were obtained using Kitti 2015 dataset. Under the same configuration and input size the Zbontar et al. architecture best result reached 5.52 times reduction factor while the Squeeze Net architecture had a 4.14 times reduction factor. Also, both system got close to real time solutions having 0.17 and 0.28 seconds of execution time respectively. The rest of the results of the different executions are shown in the table 6 and table 7 for Kitti 2012 dataset, and table 8 and table 9 for Kitti 2015.

The accuracy error had a incremental factor in the cases the post-processing module was executed according with the amount of reduction performed in the input. However there was a reduction of the error in the cases where cost calculation only or cost calculation plus refinement was used. A complete list of the results obtained is shown in the table 10 and table 11 for the ZBontar et al. and Squeeze Net architectures respectively using Kitti 2012 dataset, and table 12 and table 13 using Kitti 2015 dataset.

The memory consumption was also taken into consideration after each execution, recording the reduction factor on every input reduction over the different configurations of both systems using Kitti 2012 and 2015 datasets. In both systems there was a higher memory saving compared with the reduction performed on the inputs reaching a factor of 2.93 and 2.33 for the ZBontar et al. and Squeeze Net architectures respectively with both datasets. The rest of the results of the different executions are shown in the table 14 and table 15 for Kitti 2012 dataset, and table 16 and table 17 for Kitti 2015 dataset.

Table 6: Execution times of the ZBontar Fast architecture using different input sizes on different modules over Kitti 2012 dataset

Execution Time in Seconds (Reduction Factor)						
	Input Size					
Conf.	100%	90%	80%	70%	60%	50%
C1	0.33	0.27 (1.22)	0.22 (1.52)	0.16 (2.14)	0.14 (2.40)	0.11 (3.06)
C2	0.34	0.28 (1.22)	0.22 (1.57)	0.17 (1.98)	0.13 (2.55)	0.11 (3.00)
C3	0.65	0.50 (1.30)	0.38 (1.71)	0.26 (2.53)	0.21 (3.06)	0.15 (4.36)
C4	0.67	0.52 (1.29)	0.39 (1.70)	0.30 (2.26)	0.22 (2.09)	0.16 (4.12)

Table 7: Execution times of the Squeeze Net architecture using different input sizes on different modules over Kitti 2012 dataset

Execution Time in Seconds (Reduction Factor)						
	Input Size					
Conf.	100%	90%	80%	70%	60%	50%
C1	0.58	0.48 (1.21)	0.41 (1.42)	0.29 (2.01)	0.22 (2.61)	0.22 (2.68)
C2	0.59	0.49 (1.20)	0.41 (1.45)	0.34 (1.74)	0.28 (2.15)	0.23 (2.58)
C3	0.89	0.72 (1.25)	0.57 (1.56)	0.39 (2.30)	0.29 (3.11)	0.26 (3.39)
C4	0.91	0.73 (1.25)	0.59 (1.55)	0.46 (1.97)	0.35 (2.56)	0.29 (3.13)

Table 8: Execution times of the ZBontar Fast architecture using different input sizes on different modules over Kitti 2015 dataset

Execution Time in Seconds (Reduction Factor)						
	Input Size					
Conf.	100%	90%	80%	70%	60%	50%
C1	0.33	0.28 (1.18)	0.22 (1.53)	0.16 (2.10)	0.11 (2.96)	0.11 (3.01)
C2	0.33	0.28 (1.21)	0.22 (1.50)	0.18 (1.89)	0.13 (2.50)	0.12 (2.90)
C3	0.92	0.72 (1.28)	0.53 (1.73)	0.35 (2.64)	0.26 (3.49)	0.17 (5.52)
C4	0.94	0.73 (1.29)	0.54 (1.74)	0.38 (2.45)	0.27 (3.54)	0.19 (5.01)

Table 9: Execution times of the Squeeze Net architecture using different input sizes on different modules over Kitti 2015 dataset

Execution Time in Seconds (Reduction Factor)						
	Input Size					
Conf.	100%	90%	80%	70%	60%	50%
C1	0.59	0.49 (1.19)	0.41 (1.44)	0.29 (2.02)	0.22 (2.72)	0.23 (2.53)
C2	0.59	0.49 (1.20)	0.41 (1.44)	0.34 (1.72)	0.28 (2.13)	0.23 (2.59)
C3	1.17	0.92 (1.27)	0.73 (1.62)	0.48 (2.44)	0.32 (3.66)	0.28 (4.14)
C4	1.19	0.93 (1.27)	0.73 (1.61)	0.55 (2.14)	0.41 (2.93)	0.31 (3.85)

Table 10: Accuracy errors of the ZBontar Fast architecture using different input sizes on different modules over Kitti 2012 dataset

Accuracy Error (Increase Factor)						
	Input Size					
Conf.	100%	90%	80%	70%	60%	50%
C1	15.33%	14.89% (0.97)	13.11% (0.86)	11.59% (0.76)	10.52% (0.69)	9.96% (0.65)
C2	8.39%	7.38% (0.88)	6.59% (0.79)	6.05% (0.72)	6.17% (0.74)	7.42% (0.88)
C3	4.00%	4.09% (1.02)	4.17% (1.04)	4.36% (1.09)	4.66% (1.16)	5.25% (1.31)
C4	2.93%	3.17% (1.08)	3.40% (1.16)	3.88% (1.32)	4.78% (1.63)	6.87% (2.34)

Table 11: Accuracy errors of the Squeeze Net architecture using different input sizes on different modules over Kitti 2012 dataset

Accuracy Error (Increase Factor)						
	Input Size					
Conf.	100%	90%	80%	70%	60%	50%
C1	28.77%	31.30% (1.09)	28.60% (0.99)	26.12% (0.91)	24.27% (0.84)	23.37% (0.81)
C2	11.87%	11.21% (0.94)	10.05% (0.85)	9.29% (0.78)	9.17% (0.77)	10.15% (0.86)
C3	5.27%	5.45% (1.03)	5.77% (1.10)	6.26% (1.19)	7.00% (1.33)	8.22% (1.56)
C4	3.69%	4.10% (1.11)	4.57% (1.24)	5.36% (1.45)	6.76% (1.83)	9.51% (2.58)

Table 12: Accuracy errors of the ZBontar Fast architecture using different input sizes on different modules over Kitti 2015 dataset

Accuracy Error (Increase Factor)						
	Input Size					
Conf.	100%	90%	80%	70%	60%	50%
C1	15.24%	14.72% (0.97)	13.22% (0.87)	12.09% (0.79)	13.87% (0.91)	11.21% (0.74)
C2	8.17%	7.44% (0.91)	6.92% (0.85)	6.61% (0.81)	6.78% (0.83)	7.67% (0.94)
C3	4.64%	4.76% (1.03)	4.91% (1.06)	5.19% (1.12)	5.72% (1.23)	6.55% (1.41)
C4	4.04%	4.30% (1.06)	4.53% (1.12)	4.98% (1.23)	5.77% (1.43)	7.41% (1.83)

Table 13: Accuracy errors of the Squeeze Net architecture using different input sizes on different modules over Kitti 2015 dataset

Accuracy Error (Increase Factor)						
	Input Size					
Conf.	100%	90%	80%	70%	60%	50%
C1	28.734%	29.78% (1.04)	27.26% (0.95)	25.13% (0.87)	23.57% (0.82)	21.96% (0.76)
C2	12.048%	11.22% (0.93)	10.23% (0.85)	9.60% (0.80)	9.50% (0.79)	10.22% (0.85)
C3	5.991%	6.27% (1.05)	6.68% (1.11)	7.27% (1.21)	8.26% (1.38)	9.83% (1.64)
C4	5.046%	5.49% (1.09)	5.98% (1.19)	6.74% (1.34)	7.95% (1.58)	10.32% (2.04)

Table 14: Memory usage of the ZBontar Fast architecture using different input sizes on different modules over Kitti 2012 dataset

Memory Usage in Mb (Reduction Factor)						
	Input Size					
Conf.	100%	90%	80%	70%	60%	50%
C1	2029	1617 (1.25)	1283 (1.58)	1003 (2.02)	827 (2.45)	693 (2.93)
C2	2039	1627 (1.25)	1293 (1.58)	1029 (1.98)	833 (2.45)	695 (2.93)
C3	2033	1617 (1.26)	1287 (1.58)	1119 (1.82)	835 (2.43)	723 (2.81)
C4	2039	1627 (1.25)	1293 (1.58)	1029 (1.98)	835 (2.44)	695 (2.93)

Table 15: Memory usage of the Squeeze Net architecture using different input sizes on different modules over Kitti 2012 dataset

Memory Usage in Mb (Reduction Factor)						
	Input Size					
Conf.	100%	90%	80%	70%	60%	50%
C1	2031	1617 (1.26)	1389 (1.46)	1241 (1.64)	1145 (1.77)	893 (2.27)
C2	2041	1631 (1.25)	1389 (1.47)	1159 (1.76)	1007 (2.03)	877 (2.33)
C3	2035	1631 (1.25)	1389 (1.47)	1249 (1.63)	1145 (1.78)	893 (2.28)
C4	2041	1631 (1.25)	1389 (1.47)	1173 (1.74)	1045 (1.95)	877 (2.33)

Table 16: Memory usage of the ZBontar Fast architecture using different input sizes on different modules over Kitti 2015 dataset

Memory Usage in Mb (Reduction Factor)						
	Input Size					
Conf.	100%	90%	80%	70%	60%	50%
C1	2031	1619 (1.25)	1285 (1.58)	1249 (1.58)	833 (2.44)	763 (2.66)
C2	2041	1629 (1.25)	1295 (1.58)	1031 (1.98)	835 (2.44)	697 (2.93)
C3	2035	1619 (1.26)	1289 (1.58)	1249 (1.63)	837 (2.43)	771 (2.64)
C4	2041	1629 (1.25)	1295 (1.58)	1029 (1.98)	837 (2.44)	697 (2.93)

Table 17: Memory usage of the Squeeze Net architecture using different input sizes on different modules over Kitti 2015 dataset

Memory Usage in Mb (Reduction Factor)						
	Input Size					
Conf.	100%	90%	80%	70%	60%	50%
C1	2031	1631 (1.25)	1389 (1.46)	1303 (1.56)	1151 (1.76)	893 (2.27)
C2	2041	1631 (1.25)	1389 (1.47)	1173 (1.74)	1045 (1.95)	851 (2.40)
C3	2035	1631 (1.25)	1389 (1.47)	1303 (1.56)	1151 (1.77)	919 (2.21)
C4	2039	1631 (1.25)	1389 (1.47)	1185 (1.72)	1045 (1.95)	877 (2.32)

4.0.8 Erode-Dilate module results

With the purpose of testing the efficiency of the Erode-Dilate module, new system configurations were added to the ZBontar et al. and Squeeze Net architectures. These configurations can be observed in the table 18.

Table 18: New configurations with the proposed Erode-Dilate module

Configuration	Cost Calculation	Post-Processing	Erode-Dilate	Refinement
ED1	•		•	
ED2	•		•	•
ED3	•	•	•	•

In this experiments the systems were tested with full sized images in order to see the direct impact of the new module on the execution time and accuracy. The GPU consumption was not taken into consideration since it's depreciable compared with the cost calculation module of the system. Also, the threshold disparity t for the disparity mask process was set to 25, denoting any disparity below this value as background. The results are depicted in the table 19 and table 20 for Kitti 2012 and 2015 respectively.

The experiments shown that the Erode-Dilate module over both datasets was able to improve the accuracy of the solution with in the configurations ED1 and ED2 adding approximately only 0.01 seconds to the execution time. However, the module is not recommended with the configuration ED3 as it added errors compared with the configuration C4.

Table 19: Erode-Dilate module accuracy and execution time comparison with standard configurations over Kitti 2012 dataset

	Execution Time in seconds (Difference with C1)		Accuracy Error (Difference with Previous Conf.)	
Conf.	Squeeze Net	ZBontar Fast	Squeeze Net	ZBontar Fast
C1	0.57 sec (-)	0.33 sec (-)	28.77% (-)	15.33% (-)
ED1	0.58 sec (0.01 sec)	0.34 sec (0.01 sec)	23.59% (5.18%)	13.52% (1.81%)
C2	0.58 sec (0.01 sec)	0.34 sec (0.01 sec)	11.87% (11.73%)	8.39% (5.12%)
ED2	0.59 sec (0.02 sec)	0.35 sec (0.02 sec)	9.64% (2.22%)	7.70% (0.69%)
C4	0.90 sec (0.33 sec)	0.66 sec (0.33 sec)	3.69% (5.95%)	2.93% (4.77%)
ED3	0.92 sec (0.35 sec)	0.68 sec (0.35 sec)	3.73% (-0.04%)	2.94% (-0.01%)

Table 20: Erode-Dilate module accuracy and execution time comparison with standard configurations over Kitti 2015 dataset

	Execution Time in seconds (Difference with C1)		Accuracy Error (Difference with Previous Conf.)	
Conf.	Squeeze Net	ZBontar Fast	Squeeze Net	ZBontar Fast
C1	0.59 sec (-)	0.33 sec (-)	28.73% (-)	15.24% (-)
ED1	0.59 sec (-)	0.34 sec (0.01 sec)	24.20% (4.53%)	13.60% (1.64%)
C2	0.59 sec (-)	0.33 sec (-)	12.05% (12.15%)	8.17% (5.43%)
ED2	0.60 sec (0.01 sec)	0.35 sec (0.02 sec)	10.18% (1.87%)	7.79% (0.38%)
C4	1.18 sec (0.59 sec)	0.94 sec (0.61 sec)	5.05% (5.13%)	4.04% (3.75%)
ED3	1.20 sec (0.61 sec)	0.95 sec (0.62 sec)	5.06% (-0.01%)	4.05% (-0.01%)

5 Conclusions and Future Work

The long term objective of this research is to reduce the computational resources required by these models, to make their deployment on smaller devices feasible. These resources include memory, execution time, storage and communications size. In this paper we presented a Cost Calculation CNN based module built as a Squeeze Network to generate an initial disparity map. This network was combined with post-processing and refinement algorithms to improve the final disparity map quality. In the tests performed the system showed a reduction of almost 30% of the number of parameters. The cost of such a reduction was less than 1.5% of accuracy and less than 250 ms of execution time when compared to state of the art networks. The GPU memory used was comparable with the fast architecture and consumed less than a half the accurate architecture. Thus, the results show the utility of our architecture in terms of reducing the size of the network, and it is a first step towards the more general goal of reducing the execution time and memory required.

The model proposed might be improved to obtain better quality on disparity maps and less execution time by experimenting with different architectures and hyperparameters, like the combination of squeeze nets and residual networks. Also, different techniques of parameter size reduction like pruning, as mentioned in Iandola et al[15], to minimize even more the network size, will be explored in future work.

In addition, the 50% reduction of the input parameters shown a decrement of 4.36 and 3.39 times the execution time using Kitti 2012, or 5.52 and 4.14 times the execution time using Kitti 2015 in the ZBontar et al. and Squeeze Net architectures respectively with a cost of only 1.31 and 1.56 times increment of the accuracy error in the C3 architectures configuration using Kitti 2012, or 1.41 and 1.64 times using Kitti 2015, denoting the potential of such technique to make these architectures closer to real time applications. In other configurations like C1, this technique not only reduced the execution time but also improved the accuracy of the system reaching a 5.37% and 5.4% less error rate and a 66.6% and 62.1% reduction on the execution time using Kitti 2012. In our tests using Kitti 2015 the same configuration reached 4.03% and 6.77% less error rate and similar reduction on the execution time. In all the cases, the reduction of the input size helped to consume less GPU memory making it feasible to use on smaller devices with less resources. We believe that this technique in combination with parameter size reduction, can help even more to decrement the amount of GPU memory required by these systems.

Finally, taking into consideration the execution cost of the post-processing module, the Erode-Dilate module offered a low execution cost alternative to improve the solution quality when the post-processing module is not used.

6 Acknowledgements

This research was supported by CONICET (National Council of Scientific and Technological Research), and LABSIN (Intelligent Systems Laboratory) of the School of Engineering (National University of Cuyo). We also want to gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

References

[1] Hannah, M.J.: Computer matching of areas in stereo images. Technical, Stanford University (1974)

[2] Marr, D., Poggio, T.: Cooperative computation of stereo disparity. Science **194**(4262), 283–287 (1976)

[3] Scharstein, D., Szeliski, R.: A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. International Journal of Computer Vision **47**(1–3), 7–42 (2002)

[4] Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature **323**(6088), 533–536 (1986)

[5] LeCun, Y., Boser, B.E., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W.E., Jackel, L.D.: Backpropagation Applied to Handwritten Zip Code Recognition. Neural Computation **1**(4), 541–551 (1989)

[6] Hinton, G.E.: Learning multiple layers of representation. Trends in Cognitive Sciences **11**(10), 428–434 (2007)

[7] Torch home page, http://torch.ch/

[8] Tensorflow home page, https://www.tensorflow.org/

[9] Theano home page, http://deeplearning.net/software/theano/

[10] Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet Classification with Deep Convolutional Neural Networks. Advances in Neural Information Processing Systems **25**(2), 1097–1105 (may 2012)

[11] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR **abs/1512.03385** (2015), http://arxiv.org/abs/1512.03385

[12] Žbontar, J., LeCun, Y.: Stereo Matching by Training a Convolutional Neural Network to Compare Image Patches. Journal of Machine Learning Research **17**, 1–32 (2016)

[13] ImageNet Large Scale Visual Recognition Challenge (ILSVRC) home page, http://image-net.org/challenges/LSVRC/

[14] The KITTI Vision Benchmark Suite evaluation page, http://www.cvlibs.net/datasets/kitti/eval_stereo.php

[15] Iandola, F.N., Moskewicz, M.W., Ashraf, K., Han, S., Dally, W.J., Keutzer, K.: Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <1mb model size. CoRR **abs/1602.07360** (2016), http://arxiv.org/abs/1602.07360

[16] Hamzah, R.A., Ibrahim, H.: Literature Survey on Stereo Vision Disparity Map Algorithms. Journal of Sensors **2016**, 1–23 (2016)

[17] Zeiler, M.D., Fergus, R.: Visualizing and Understanding Convolutional Networks. Computer Vision-ECCV 2014 **8689**, 818–833 (2014)

[18] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision **115**(3), 211–252 (2015)

[19] Mei, X., Sun, X., Zhou, M., Jiao, S., Wang, H., Xiaopeng Zhang: On building an accurate stereo matching system on graphics hardware. In: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops). pp. 467–474. IEEE, Barcelona (nov 2011)

[20] Shaked, A., Wolf, L.: Improved stereo matching with constant highway networks and reflective confidence learning. CoRR **abs/1701.00165** (2017), http://arxiv.org/abs/1701.00165

[21] Yang, M., Lv, X.: Learning both matching cost and smoothness constraint for stereo matching. Neurocomputing **314**, 234 – 241 (2018)

[22] Ke Zhang, Jiangbo Lu, Lafruit, G.: Cross-Based Local Stereo Matching Using Orthogonal Integral Images. IEEE Transactions on Circuits and Systems for Video Technology **19**(7), 1073–1079 (jul 2009)

[23] Hirschmüller, H.: Stereo Processing by Semiglobal Matching and Mutual Information. IEEE Transactions on Pattern Analysis and Machine Intelligence **30**(2), 328–341 (feb 2008)

[24] Yao, G., Liu, Y., Lei, B., Ren, D.: A rapid stereo matching algorithm based on disparity interpolation. In: Proceedings of 2010 Conference on Dependable Computing. pp. 5–10 (2010)

[25] Williem, W., Kyu Park, I.: Deep self-guided cost aggregation for stereo matching. Pattern Recognition Letters **112** (07 2018)

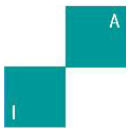
[26] Miclea, V.C., Vancea, C.C., Nedevschi, S.: New sub-pixel interpolation functions for accurate real-time stereo-matching algorithms. In: 2015 IEEE International Conference on Intelligent Computer Communication and Processing (ICCP). vol. 20, pp. 173–178. IEEE (sep 2015)

[27] Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: Conference on Computer Vision and Pattern Recognition (CVPR) (2012)

[28] Menze, M., Heipke, C., Geiger, A.: Joint 3d estimation of vehicles and scene flow. In: ISPRS Workshop on Image Sequence Analysis (ISA) (2015)

[29] Menze, M., Heipke, C., Geiger, A.: Object scene flow. ISPRS Journal of Photogrammetry and Remote Sensing (JPRS) (2018)

Inteligencia Artificial, 22(63) (2019), 39-60
doi: 10.4114/intartif.vol22iss63pp39-60



INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

Intelligent Classification of Supernovae Using Artificial Neural Networks

Francisca Joamila Brito do Nascimento[1,A], Luis Ricardo Arantes Filho[1,B], Lamartine Nogueira Frutuoso Guimarães[1,2,C]

[1]INPE - Instituto Nacional de Pesquisas Espaciais, São José dos Campos 12227-010, Brasil

[2]IEAv - Instituto de Estudos Avançados, São José dos Campos 12228-001, Brasil

[A]francisca.nascimento@inpe.br, [B]luisricardoengcomp@gmail.com, [C]guimarae@ieav.cta.br

Abstract The classification of supernovae (explosions of certain stars) divides them into two main types, those of type I do not present Hydrogen in the spectrum while those of type II present. In addition to the division into these two types, there is still a subdivision that establishes types Ia, Ib and Ic. In practice, the classification of supernovae requires specialized knowledge of astronomers and data (light spectra) of good quality. Some automatic/intelligent classifiers have been developed and are reported in the literature, one of them is CIntIa, which uses 4 Artificial Neural Networks to classify supernovae types Ia, Ib, Ic and II. The objective of this work is to improve CIntIa, so that it has more diversity in its learning, proposing CIntIa 2.0. In this way, this work is a hierarchical learning structure that connects Artificial Neural Networks in an integrated system that allows a more secure and unambiguous classification. The computational improvement of this new version included the increased amount of data used at all stages of development of intelligent classifier and a new approach to filtering and processing of spectral data, ensuring better quality of information that are to be trained networks. The results achieved were good, especially in the classification of types Ia and II. A comparison with the works found in the literature shows that CIntIa 2.0 is superior in quantity and diversity of data and achieves higher classification indices than the other classifiers.

Resumo A classificação das supernovas (explosões de certas estrelas) as divide em dois tipos principais, as do tipo I não apresentam Hidrogénio no espectro enquanto as do tipo II apresentam. Além da divisão nesses dois tipos, há ainda uma subdivisão que estabelece os tipos Ia, Ib e Ic. Na prática, a classificação das supernovas exige o conhecimento especializado de astrónomos e dados (espectros de luz) de boa qualidade. Alguns classificadores automáticos/inteligentes foram desenvolvidos e são reportados na literatura, um deles é a CIntIa, que usa 4 Redes Neurais Artificiais individuais para classificar as supernovas nos tipos Ia, Ib, Ic e II. O objetivo deste trabalho é aperfeiçoar a CIntIa, a fim de que ela tenha mais diversidade em seu aprendizado, propondo a CIntIa 2.0. Dessa maneira, este trabalho propõe uma estrutura de aprendizado hierárquica que conecta as Redes Neurais Artificiais individuais em um sistema integrado permitindo uma classificação mais segura e não ambígua. O aprimoramento computacional desta nova versão compreendeu o aumento da quantidade de dados usados em todas as fases de desenvolvimento do classificador inteligente e uma nova abordagem na filtragem e processamento dos dados espectrais, garantindo mais qualidade nas informações que são submetidas ao treinamento das redes. Os resultados obtidos com este aprimoramento demonstram um bom desempenho, principalmente na classificação dos tipos Ia e II. Uma comparação com os trabalhos encontrados na literatura mostra que a CIntIa 2.0 é superior em quantidade e diversidade de dados e alcança índices de classificação superiores aos demais classificadores.

Keywords: Artificial Neural Networks, Intelligent Classification, Supernovae.
Palavras-Chave: Redes Neurais Artificiais, Classificação Inteligente, Supernovas.

ISSN: 1137-3601 (print), 1988-3064 (on-line)

1 Introdução

As supernovas (SNs) são grandes explosões que caracterizam o fim da vida de estrelas muito massivas. Outro mecanismo gerador de SNs é o acréscimo de massa em uma anã-branca que acontece quando ela atrai para si massa de outra estrela, sua companheira em um sistema binário. O brilho intenso das SNs tem uma grande importância nos estudos atuais de Cosmologia, por exemplo, a descoberta da expansão acelerada do Universo, que rendeu o Prêmio Nobel de Física de 2011 para os astrónomos Saul Perlmutter [23], Adam G. Riess e Brian P. Schmidt [24]. As SNs usadas na descoberta da expansão acelerada do Universo foram as do tipo Ia (SNIa), que tem a característica de manter o pico de luminosidade constante. Essa característica permite que sejam utilizadas como vela-padrão, um objeto astronómico que por sua luminosidade conhecida é usado para medir distâncias astronómicas.

Segundo [22], as SNs são classificadas em dois tipos principais propostos por Rudolph Minkowski em 1941, as SNs do tipo I não possuem Hidrogénio (H) e as do tipo II (SNII) possuem esse elemento. Há ainda uma sub-classificação do tipo I que resulta nos tipo Ia (SNIa), Ib (SNIb) e Ic (SNIc). Atualmente, a classificação de SNs segue um consenso de separar entre as de origem termonuclear e as outras, originadas do colapso do núcleo. [30] apresenta um esquema de classificação (Figura 1) que é seguido por grande parte dos classificadores automáticos e inteligentes descritos na literatura, que classificam usando a técnica da espectroscopia. Os tipos Ia, Ib, Ic e II estão destacados nos quadrados pretos e são os tipos passíveis de classificação pelo sistema CIntIa. As siglas fora dos quadrados são os elementos químicos buscados para que a classificação seja realizada: Hidrogénio (H); Silício duas vezes ionizado (Si II); e Hélio uma vez ionizado (He I). O I no quadrado branco à esquerda é o tipo I, que se subdivide nos tipos Ia, Ib e Ic. Os tipos IIb, IIL e IIP (quadrados brancos à direita), são sub-tipos do tipo II, sendo que o IIL e o IIP são diferenciados apenas pela curva de luz (fotometria). E os sub-tipos IIb pec e IIn não são considerados supernovas, mas hipernovas.

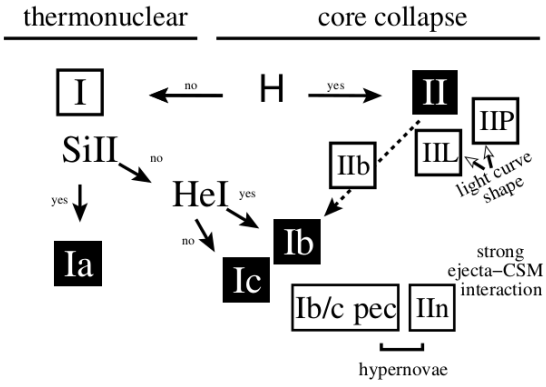


Figura 1: Esquema de classificação das SNs de acordo com os elementos químicos presentes no espectro. Os tipos Ia, Ib, Ic e II, destacados em quadrados pretos, são objectos de classificação da CIntIa. Fonte: [30].

A espectroscopia é o estudo da luz, pela sua decomposição em comprimento de ondas ou frequências (cores). Os espectros das SNIa costumam exibir picos e vales largos atribuídos aos elementos Oxigénio, Magnésio, Silício (Si), Enxofre (S), Cálcio, Ferro e Cobalto neutros ou uma vez ionizados [11]. Ainda segundo [11], os elementos Si, S, e Cálcio estão presentes principalmente na fase de máximo brilho da SN e o Ferro após essa fase, aproximadamente duas semanas depois. A fase espectral de brilho máximo é quando o brilho da SN atinge o seu valor máximo e é um período que dura apenas alguns dias. O dia em que o brilho da SN atinge o seu ápice é o dia 0, os dias precedentes tem valores negativos (a partir do -1) e os dias que sucedem esse momento tem valores positivos (a partir do +1).

O esquema de Turatto [30] foi desenvolvido considerando espectros de SNs na sua fase de máximo, isto significa que todas as características de elementos utilizadas para a classificação são aquelas que se apresentam no curto período de brilho máximo, como o Si II em espectros de SNs de tipo Ia. Destacamos

que as características espectrais, como a intensidade, a localização de picos e vales nos espectros mudam no decorrer do tempo. Essa mudança pode acarretar em classificações ambíguas, por exemplo, em SNIa a presença de linhas de elementos como o Fe I e Fe II (Ferro uma e duas vezes ionizado), bem como de Níquel e Cobalto são mais fortes e ocorrem dias após a SN atingir seu brilho máximo. Estes elementos mais complexos não são avaliados pelo esquema de Turatto, e, desta forma, a classificação torna-se difícil.

A classificação de SNs por espectroscopia, baseada na presença e ausência de elementos químicos, é normalmente rápida e pode ser feita assim que a SN é descoberta pelo telescópio, como esclarece [31]. Um dos classificadores inteligentes que usa dados espectroscópicos é a CIntIa, proposto por [21], que usa um conjunto de Redes Neurais Artificiais (RNAs) para classificar os tipos principais: Ia, Ib, Ic e II. Usar RNAs para prover classificações é vantajoso porque o sistema aprende os padrões e o aprendizado pode continuar acontecendo facilmente, em oposição a métodos puramente matemáticos. Outro classificador importante para o presente trabalho é a SUZAN, desenvolvido por [2], que utiliza lógica nebulosa e regras de aprendizado para identificar Si e S em SNs do Tipo Ia. CIntIa e SUZAN são parte de uma mesma iniciativa para fornecer sistemas de classificação automática de SNs para o projeto Kulun Dark Universe Survey Telescope (KDUST), os dois classificadores são melhor explicados na secção de Revisão Bibliográfica. O projeto KDUST está construindo um grande telescópio para o observatório de Kulun na Antártida, previsto para ser instalado em 2020 [34]. Devido a localização da estação e a grande quantidade de observações previstas para o KDUST a automatização do processo de classificação se mostra essencial. No entanto, a CIntIa e a SUZAN não estão restritas ao projeto KDUST, é possível usá-las para classificar espectros obtidos por outros telescópios. Inclusive,um classificador como a CIntIa, que usa RNAs, pode ser implementado, depois de treinado, em um sistema embarcado com custo energético muito baixo.

O objetivo deste trabalho é aperfeiçoar a CIntIa a fim de que ela tenha mais diversidade no aprendizado dos padrões de espectros de SNs Ia, principalmente, o que significa incrementar a sua capacidade de generalização. Para cumprir esse objetivo, algumas actividades, que são descritas neste artigo, foram realizadas:

- Aumento da quantidade de dados usados no treinamento e teste das RNAs. A quantidade de espectros de luz foi incrementada em mais de 1300% (de 649 para 9156);
- Aumento da amplitude de fases espectrais que podem ser reconhecidas pelo classificador. De -3 a +7 dias para -10.9 a +10.9 dias, para os espectros de SNs do tipo Ia;
- Nova abordagem na filtragem dos espectros, que utiliza o esquema de Dupla-Filtragem, o mesmo da SUZAN, proposto por [2], garantindo mais qualidade de informação;
- Desenvolvimento de uma arquitetura que integra as 4 RNAs para o aprendizado de SNs, gerando um aprendizado hierárquico e eliminando as possibilidades de classificação ambígua existente na primeira versão da CIntIa.

Com a adoção dessas medidas, a quantidade de dados passível de classificação aumentou drasticamente. Além disso, a análise dos resultados se apresenta mais robusta em relação à primeira versão da CIntIa. É importante destacar que a arquitetura para a integração das RNAs foi gerada como um resultado dos testes e dos aprimoramentos desenvolvidos para a melhoria da CIntIa 1.0.

Este artigo está organizado da seguinte maneira. Na Secção 2, explanamos o tema Supernovas e sua classificação. A Secção 3 é uma revisão da literatura de sistemas computacionais classificadores de SNs que usam espectros de luz. Em seguida, na Secção 4, tratamos dos materiais e métodos aplicados para obtenção dos resultados, que por sua vez, são apresentados na Secção 5. Finalmente, a Secção 6 apresenta a conclusão do trabalho com uma síntese do que foi realizado e as expectativas dos trabalhos futuros.

2 Supernovas

2.1 Evolução Estelar

Supernovas (SNs) são eventos caracterizados por grandes explosões que correspondem à fase final da vida de algumas estrelas. “[...] as grandes estrelas, ao sucumbir, superam galáxias inteiras em brilho. Seus clarões podem ser vistos por toda a extensão do Universo por alguns dias.” [7]. Por ocorrerem apenas em

grandes estrelas, ou seja, as muito massivas, SNs são eventos raros, aproximadamente um ocorrência por galáxia a cada século, segundo [6]. Cada estrela segue uma sequência evolutiva que depende da massa que ela possui ao ser formada e se está sozinha ou em um sistema binário ou múltiplo. As estrelas com massa maior do que $10\odot$ são as que geram eventos catastróficos. A Figura 2 apresenta um esquema de evolução das estrelas massivas proposto inicialmente por Peter Conti em 1976, como conta [6], e depois modificado, em todos os cenários a estrela explode marcando assim o fim de sua vida. Esse tipo de explosão é conhecido como colapso do núcleo.

$$
\begin{aligned}
m > 85\,\mathcal{M}_{\odot}&\text{: O} \longrightarrow \text{LBV} \longrightarrow \text{WN} \longrightarrow \text{WC} \longrightarrow \text{SN}\\
40 > m > 85\,\mathcal{M}_{\odot}&\text{: O} \longrightarrow \text{WN} \longrightarrow \text{WC} \longrightarrow \text{SN}\\
25 > m > \ 40\,\mathcal{M}_{\odot}&\text{: O} \longrightarrow \text{RSG} \longrightarrow \text{WN} \longrightarrow \text{WC} \longrightarrow \text{SN}\\
20 > m > 25\,\mathcal{M}_{\odot}&\text{: O} \longrightarrow \text{RSG} \longrightarrow \text{WN} \longrightarrow \text{SN}\\
10 > m > 20\,\mathcal{M}_{\odot}&\text{: OB} \longrightarrow \text{RSG} \longrightarrow \text{BSG} \longrightarrow \text{SN}.
\end{aligned}
$$

Figura 2: Esquema de evolução das estrelas massivas. Fonte: [6].

Onde:

- LBV: Variável azul luminosa;
- WN: Tipo de estrela Wolf-Rayet que contém linhas de emissão de He e Níquel;
- WC: Tipo de estrela Wolf-Rayet que contém linhas de emissão de He e Carbono e não possui Níquel e H;
- RSG: Estrela super-gigante vermelha;
- BSG: Estrela super-gigante azul;
- OB: Estrela OB;
- SN: Supernova.

A explosão por colapso do núcleo acontece porque a estrela na sequência principal converte todo o H do núcleo em He, transformando-se em gigante. Em seguida, consome Carbono e Oxigénio passando assim à fase de super-gigante, quando consome Neônio, Magnésio e Si até restar apenas o núcleo de Ferro, as camadas da estrela são mostradas na Figura 3. Nessa etapa, a super-gigante ejecta a maior parte da sua massa originando uma SN.

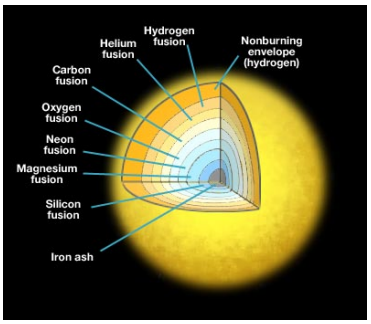


Figura 3: Camadas de uma estrela massiva. Fonte: [8].

Além das SNs originadas pelo colapso do núcleo, existem as originadas por explosões termonucleares, que são aquelas criadas em sistemas binários, onde pelo menos uma das estrelas é uma anã branca com

massa superior a 0.8⊙. A explosão é provocada pela instabilidade no núcleo provocada pelo acréscimo de massa na anã branca, que excede o limite de Chandrasekhar (1.38⊙). Segundo [33], a ignição de Carbono ou He sob condições extremamente degeneradas queima uma massa substancial no núcleo que provoca a desintegração da estrela em altas velocidades. De acordo com [22], cerca de 0.6⊙ é ejectada ao meio interestelar na forma de Ferro, produzido durante a explosão, sendo esta a maior fonte de Ferro conhecida. Na Figura 4 somos apresentados a um esquema que mostra a evolução estelar até a explosão da SN pelos dois processos discutidos acima.

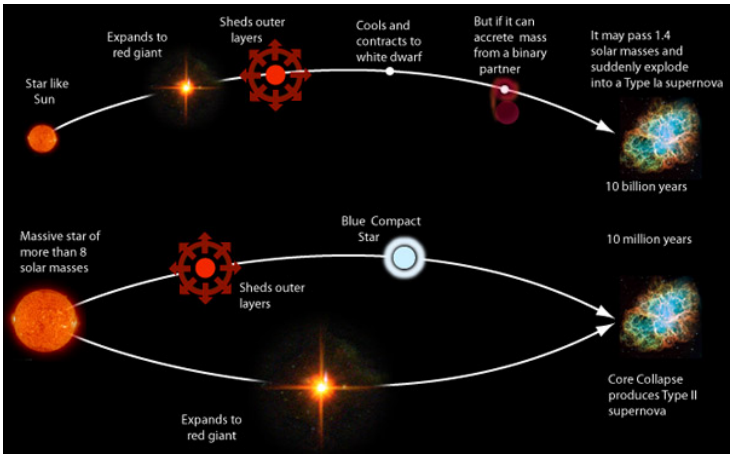


Figura 4: Evolução estelar até o estágio de SN. Fonte: [9].

2.2 Classificação das Supernovas

"As SNs são classificadas em dois tipos principais [..]: as SNs tipo I, que não apresentam H no espectro, e as SNs tipo II, que apresentam linhas de emissão ou absorção de H no espectro [...]" [22]. Segundo [6], o tipo I pode ser subdividido de acordo com seu espectro. As SNs do tipo Ia contam com uma forte presença de linhas de Si. As do tipo Ib apresentam linhas de He, enquanto as do tipo Ic não apresentam Si nem He. As SNs do tipo Ia, de acordo com [30], são as originadas das explosões termonucleares. Por outro lado, as SNs de tipo Ib, Ic e II são provenientes do colapso do núcleo das estrelas massivas. O esquema de Turatto para a classificação de SNs foi introduzido na Figura 1.

Os tipos Ia, Ib, Ic e II são os tipos que podem ser identificados a partir da análise do espectro de luz usando a técnica de espectroscopia. A espectroscopia é o estudo da luz, pela sua decomposição em comprimento de ondas ou frequências (cores). As três leis empíricas da espectroscopia foram formuladas por Gustav Kirchhoff e são postuladas como segue, segundo [22]:

- Espectro contínuo: um corpo opaco quente, sólido, líquido ou gasoso, emite um espectro contínuo;
- Espectro de emissão: um gás transparente produz um espectro de linhas brilhantes (de emissão). O número e a cor (posição) dessas linhas depende dos elementos químicos presentes no gás; e
- Espectro de absorção: se um espectro contínuo passar por um gás a temperatura mais baixa, o gás frio causa a presença de linhas escuras (absorção). O número e a posição dessas linhas depende dos elementos químicos presentes no gás.

A classificação de SNs por espectroscopia, baseada na presença e ausência de elementos químicos, é normalmente rápida e pode ser feita assim que a SN é descoberta pelo telescópio, como esclarece [31]. A Figura 5 mostra os aspectos mais comuns dos tipos principais de SNs que podem ser classificadas por espectroscopia. As regiões destacadas em cores por [20] são aquelas em que há emissão ou absorção dos elementos químicos característicos formando picos e vales. Essas regiões são as que recebem maior atenção na classificação por RNA proposta neste trabalho.

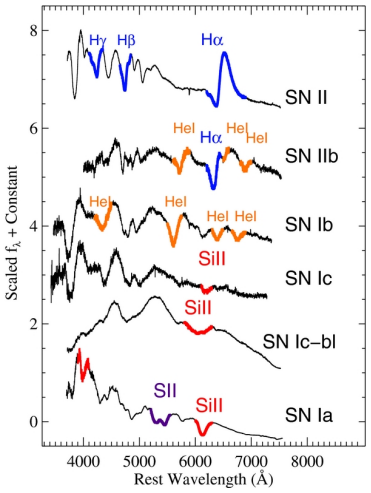


Figura 5: Padrão dos espectros de SNs dos tipos Ia, Ib, Ic, Ic-bl, II e IIb. Fonte: [20].

A fase do espectro (fase espectral) se refere ao dia, em relação ao dia em que a SN alcançou seu brilho máximo, em que aquele espectro foi auferido. O dia em que o brilho foi o mais intenso é a fase espectral 0 dias, se o espectro foi medido em dias anteriores, a fase tem valor negativo e em dias posteriores ela tem valor positivo. A fase espectral é importante para a classificação porque a forma do espectro muda com o tempo e apenas para alguns períodos existe consenso quanto à forma que o espectro de cada tipo deve ter. Por isso, a classificação automática tende a ser confiável apenas em fases mais próximas do 0 (brilho máximo) quando o padrão, principalmente para o tipo Ia, já é bem conhecido pelos especialistas. O inconveniente dessa abordagem é que é necessário traçar a curva de luz da SN para inferir a fase de cada espectro, o que pode levar alguns meses, impedindo assim a classificação rápida em plataformas semelhantes ao KDUST. Um dos avanços alcançado neste trabalho é a classificação correta de espectros em fases além do período de brilho máximo, definido na CIntIa 1.0 como -3 a +7 dias.

3 Revisão da Literatura

A classificação automática de SNs costuma ser feita de dois modos, analisando o espectro de luz (comprimento de onda em função do fluxo de radiação) ou a curvas de luz (magnitude em função do tempo). As curvas de luz permitem a análise no decorrer de 40 a 60 dias após a explosão da estrela. Em contrapartida, pode se analisar o tipo de SN pelo espectro óptico em poucos dias, no momento em que a explosão atinge o pico de luminosidade máxima [11] [4]. A classificação por espectroscopia costuma ser imediata porque a análise pode ser feita com apenas um espectro. Com dados de fotometria, a classificação só pode ser feita muitos dias depois da explosão da SN porque é necessário estabelecer o perfil temporal da intensidade do brilho. Portanto, neste trabalho, focaremos na classificação usando dados espectroscópicos. Foram selecionados 6 classificadores de SNs que usam espectro de luz para abordagem nesta secção.

3.1 Supernova Identification (SNID)

O SNID é uma ferramenta desenvolvida por [5] com a finalidade de determinar idade, *redshift* e classificar SNs. A identificação das SNs, que resulta em sua classificação é feita usando técnicas estatísticas. O espectro que se deseja classificar, cujo *redshift* deve ser conhecido, é correlacionado com outros espectros previamente identificados, o espectro recebe a mesma classificação daquele com quem mais se correlacionou. Foram usados 879 espectros de 65 SNIa, 322 de 19 SNIb/c e 353 de 10 SNII no desenvolvimento do classificador. Os espectros foram previamente pré-processados para serem submetidos à correlação com fluxo normalizado entre 0 e 1 e filtrados. Os autores consideram que o classificador alcança algum sucesso, mas os resultados não estão explícitos no trabalho.

3.2 Generic Classification Tool (GELATO)

O GELATO, apresentado por [16], é um classificador automático de SNs que pretende mitigar a subjetividade das classificações feitas por especialistas humanos. A ferramenta usa um método matemático para comparar novos espectros com outros previamente classificados em um banco de dados. Primeiramente, os espectros são pré-processados com correção do *redshift*, suavização, reamostragem e a divisão em 11 intervalos do comprimento de onda. A comparação entre os espectros é feita em cada um desses intervalos. A ferramenta computa para cada intervalo a distância relativa entre o novo espectro e todos os espectros do banco de dados. Por fim, calcula-se a média das distâncias relativas, o menor valor se refere ao espectro mais parecido, portanto os espectros recebem a mesma classificação. O autor relata que o GELATO é usado no dia-a-dia do seu grupo de pesquisa e que dá bons resultados, mas não são apresentadas métricas de avaliação no trabalho para comparações futuras.

3.3 Classificador Inteligente de Supernovas do tipo Ia (CIntIa 1.0)

A CIntIa 1.0 é um classificador inteligente de SNs dos tipos Ia, Ib, Ic e II e foi desenvolvida por [21]. O sistema é composto por 4 RNAs para reconhecer cada um dos tipos principais de SNs e apresenta bons resultados na classificação dos tipos, com ênfase nas SNIa.

A CIntia 1.0 usa 559 espectros de 192 SNIa, 33 espectros de 12 SNIb, 44 espectros de 12 SNIc e 13 espectros de 5 SNII, totalizando 649 espectros. Todos os espectros selecionados estão entre 3 dias antes (-3) e 7 dias depois (+7) do brilho máximo das SNs. Os espectros passam por um pré-processamento antes de serem submetidos à RNA, que inclui: correção do *redshift*, suavização, interpolação, e normalização do fluxo. A etapa seguinte é extrair intervalos do espectro que são usados como entradas nas RNAs. Cada intervalo corresponde a região onde se encontram ou não os elementos H, Si, S e He. As regiões escolhidas são as mais usadas pelos especialistas humanos para efetuar a classificação. Para treinar cada uma das RNAs foram selecionados 80% dos espectros, dos quais 20% são usados na validação. Foram testados 20% dos espectros, a quantidade e os resultados são apresentados na Tabela 1.

Tabela 1: Avaliação das 4 RNAs que compõem o sistema CIntIa 1.0.

	Tipo Ia	Tipo Ib	Tipo Ic	Tipo II
Espectros testados	106	5	6	1
Acurácia	0.99	0.97	0.97	0.98
Precisão	0.99	1	1	0.33
Recall	1	0.2	0.5	1
Índice Kappa	0.95	0.32	0.65	0.49

Nessa versão da CIntIa (1.0), existe a possibilidade de um espectro de SN ser classificado de duas maneiras diferentes. Por exemplo, uma rede neural para classificar SNIa e SNIb pode classificar um espectro como sendo dos dois tipos. Dessa forma, o mesmo espectro é classificado como pertencendo a duas classes diferentes, a ambiguidade ocorre devido a falta de integração das RNAs no momento de classificar o espectro. Outra característica que procuramos melhorar neste trabalho é a confiança nos testes. A CIntIa 1.0 apresenta uma pequena quantidade de espectros testados, por exemplo, a RNA que classifica em tipo II ou tipo não-II testa apenas um espectro do tipo II. Neste trabalho, aumentamos consideravelmente a quantidade de padrões de teste, tornando assim, a análise mais segura.

3.4 Dimensionality Reduction and Clustering for Unsupervised Learning in Astronomy (DRACULA)

O DRACULA é um sistema desenvolvido por [25] para identificar subtipos do tipo Ia. Para isso o DRACULA primeiro reduz a dimensionalidade dos dados e em seguida usa aprendizado não-supervisionado para realizar a classificação. Foram usados 3677 espectros do tipo Ia na fase de redução de dimensionalidade que foi realizada com uma técnica de *Deep Learning* (DL). Espectros dentro do intervalo 4000Å a 7000Å foram inicialmente interpolados gerando espectros com 300 pontos cada. Os 300 pontos foram

transformados em apenas 4 características pelo algoritmo de DL. Em seguida, dos 3677 espectros foram selecionados 486 que foram auferidos entre -3 e +3 dias do brilho máximo das SNs. Os 486 espectros selecionados após a redução da dimensionalidade foram submetidos ao algoritmo de aprendizado não-supervisionado *K-Means*. O parâmetro necessário para a execução do algoritmo é o número de classes, os autores optaram por 4 classes de acordo com a divisão em subtipos do tipo Ia proposta por [32]. A classificação das SNs não ficou totalmente em acordo com a proposta de [32] e não há métricas de avaliação no trabalho suficientes para a discussão.

3.5 Quantitative Classification of Type I Supernovae

A proposta de [29] é a classificação quantitativa dos subtipos do tipo I de SNs. O método proposto pelos autores para classificar quantitativamente é medir a profundidade de linhas de absorção dos elementos Si II (em λ 6150Å) e O I (em λ 7774Å). Foram usados 146 espectros do tipo Ia, 12 do tipo Ib, 19 do tipo Ic e 4 do tipo Ib/c. As profundidades medidas resultaram em um critério desenvolvido e aplicado pelos autores para efetivar a classificação. O resultado da profundidade das regiões foi comparado com o apresentado por [27] e foi constatado que as medidas resultantes foram um pouco mais baixas do que a dos autores consultados, mas apenas 5% do total apresentaram grande discrepância, 119 medidas do Si II e 40 medidas do O I foram concordantes. Assim, os autores afirmam que a maioria das SNs do tipo I podem ser classificadas pelos critérios apresentados por eles, exceto por semelhantes à SN2002cx Ia-pec, SNs Ic-BL e as SN1991T.

3.6 Sistema Fuzzy Avaliador de Supernovas (SUZAN)

A SUZAN, apresentada por [2] é um sistema classificador de SNs que diferencia entre as SNIa e as SN não-Ia. O sistema usa lógica *fuzzy* e foi desenvolvido para atuar em redundância com a CIntIa. No total foram usados 3697 espectros de 588 SNs diferentes, dos quais 3082 espectros são de SNIa, 217 do tipo Ib, 282 do tipo Ic e 116 do tipo II. Os espectros passaram pela etapa de pré-processamento que consistiu na Dupla-Filtragem usando o filtro de Savitzky-Golay, ou seja, os espectros foram filtrados duas vezes seguidas e depois o *redshift* foi ajustado. Esta amostra de dados compreende espectros em diversas fases de observação compreendendo o intervalo de -20 dias (antes do brilho máximo da explosão) até +2959 dias (após o brilho máximo da explosão), desta forma, os espectros possuem muitas variações no decorrer do tempo.

Os elementos H, He, Si e S são buscados, então a SUZAN extrai características dos espectros para em seguida submetê-los ao crivo das regras criadas com conhecimento especialista. Primeiro, a SUZAN seleciona as linhas de emissão e absorção (linhas candidatas) que se adequam aos padrões esperados para as linhas dos elementos químicos buscados. Então, realiza-se uma nova etapa de processamento por regras *fuzzy* que avalia a distância relativa de cada linha candidata. Assim, ao final do processo é verificada a diversidade de elementos em cada espectro e aplicada a classificação proposta por [30]. A (Tabela 2) mostra os índices de avaliação considerando a classificação em tipo Ia ou tipo não-Ia.

Tabela 2: Métricas de avaliação da classificação entre tipo Ia e tipo não-Ia realizada pela SUZAN.

Acurácia	Precisão	Recall	F1-Score	Índice Kappa
0.73	0.93	0.72	0.82	0.34

O melhor desempenho da SUZAN acontece na classificação de espectros de SNIa na fase de brilho máximo, assim os elementos como o S e o Si são identificados pelas regras *fuzzy*, é neste período que os astrónomos especialistas classificam os espectros de SNIa. Dessa forma, o autor propõe uma separação de espectros de SNIa apenas na fase de brilho máximo considerando uma correlação de Spearman entre modelos de SNIa no período entre -2.5 dias a +2.5 dias, período proposto por [3] que considera ser o período em que as características do Si e do S estão mais evidentes. Foram correlacionadas 772 espectros de SNIa dos 3082 espectros originais, atingindo os índices de 96.63% na classificação entre SNIa e SN Não-Ia.

4 Materiais e Métodos

Nesta secção, são apresentados os materiais e os métodos usados neste trabalho. Tratamos das Redes Neurais Artificiais, dos bancos de espectros usados, do pré-processamento pelo qual todos os espectros foram submetidos, com ênfase na Dupla-Filtragem, da extração de características para as entradas do sistema e das configurações da CIntIa que tornaram possíveis a obtenção de resultados que são apresentados na próxima secção.

4.1 Redes Neurais Artificiais

Uma Rede Neural Artificial (RNA) é uma rede que conecta unidades de processamento simples, os neurónios (Figura 6), e tem a capacidade de aprendizado por meio de exemplos. Segundo [17], os elementos que compõem o neurónio são os seguintes:

- Peso sináptico: é um valor negativo ou positivo que representa a força de conexão entre um dado de entrada e o neurónio. O primeiro índice de um peso se refere ao neurónio e o segundo índice é o mesmo da entrada;
- Somador ou Junção Aditiva: é responsável por somar todas as entradas ponderadas por seus pesos sinápticos;
- Bias: é um fator que aumenta ou diminui o valor da saída do somador;
- Função de ativação: é a função que define a saída do neurónio.

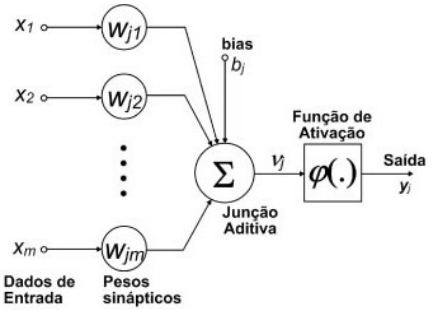


Figura 6: Modelo de um neurónio artificial. Fonte: [12].

A conexão entre mais de um neurónio molda a arquitetura de uma RNA. A ligação entre os neurónios pode ser feita em apenas uma camada, na qual as entradas são multiplicadas pelos pesos, passam pelo somador e logo em seguida pela função de ativação. Ou a RNA pode ter mais de uma camada, assim a saída de uma camada é usada como entrada para os neurónios da camada posterior sendo propagada até a saída.

O aprendizado da RNA, por sua vez, pode ser do tipo supervisionado ou não-supervisionado. No aprendizado supervisionado, em oposição ao não-supervisionado, um agente externo apresenta à RNA alguns padrões de entrada e seus correspondentes padrões de saída. Portanto, é necessário ter um conhecimento prévio do comportamento que se deseja ou se espera da rede, como esclarece [10]. O aprendizado depende de um algoritmo que modifica os pesos de acordo com as saídas da RNA em cada sessão de treinamento.

O algoritmo de atualização dos pesos, conhecido como *backpropagation* é o mais usado com a RNA Perceptron de Múltiplas Camadas (MLP, do inglês *Multi-Layer Perceptron*). Esse tipo de RNA consiste de um conjunto de unidades que constituem a camada de entrada, uma ou mais camadas ocultas (intermediárias) e uma camada de saída, como descreve [17]. "A abordagem adotada pelo algoritmo [*backpropagation*] consiste em iniciar na camada de saída e propagar o erro retroativamente através das

camadas ocultas." [19]. O treinamento com *backpropagation* usa a regra delta generalizada. O cálculo do fator delta de cada neurónio depende se o neurônio está na camada de saída ou em uma camada intermediária. As Equações 1 e 2 são as fórmulas para calcular o erro e o delta, respetivamente, em neurónios da camada de saída. Enquanto a Equação 3 é a fórmula para calcular o delta em neurónios de camadas ocultas. As equações são as mesmas apresentadas por [17].

$$e_j = d_j - y_j \tag{1}$$

Onde:

- e_j: erro no neurónio j;
- d_j: resultado esperado do neurónio j;
- y_j: resultado obtido no neurónio j.

$$\delta_j = e_j \dot{\varphi}_j(v_j) \tag{2}$$

Onde:

- δ_j: delta do neurónio j;
- e_j: erro no neurónio j;
- $\dot{\varphi}_j(v_j)$: derivada da função de ativação do neurónio j.

$$\delta_j = \dot{\varphi}_j(v_j) \sum_k \delta_k w_{kj} \tag{3}$$

Onde:

- δ_j: delta do neurónio j;
- e_j: erro no neurónio j;
- $\dot{\varphi}_j(v_j)$: derivada da função de ativação do neurónio j;
- $\sum_k \delta_k(n) w_{kj}$: somatório ponderado dos deltas calculados para os neurónios da camada posterior à qual o neurónio j está conectado.

Em síntese, o algoritmo segue dois passos principais. O primeiro passo é a propagação para a frente, durante a qual os pesos não são alterados e o sinal de erro é calculado ao fim da propagação. No segundo passo, os deltas derivados do sinal de erro são propagados para trás e os pesos são atualizados. Cada vez que esses dois passos são executados se dá uma sessão de treinamento, também chamada de época de treinamento. São executadas quantas épocas forem necessárias até que a rede tenha aprendido suficientemente de acordo com algum parâmetro chamado condição de parada.

4.2 Banco de Espectros

Os espectros de luz das SNs foram obtidos no repositório aberto The Open Supernova Catalog [14], atualmente mantido por dois pesquisadores do Harvard-Smithsonian Center for Astrophysics (CfA). O acervo é uma coletânea dos dados de 17 bases de espectros mais contribuições individuais. Nele, estão disponíveis espectros de SNs dos tipos Ia, Ib, Ic, II e os outros tipos que são classificados por fotometria. A Tabela 3 discrimina a quantidade total de dados que foram obtidos para utilização no classificador, porém nem todos os dados foram utilizados nos treinamentos e testes porque passaram por etapas de limpeza e pré-processamento de dados que desabilita a utilização de alguns deles. Essas etapas são exploradas nas próximas subsecções.

Tabela 3: Quantidade total de dados obtidos no The Open Supernova Catalog.

Tipo	Quantidade de espectros	Quantidade de SNs
Ia	9405	3751
Ib	627	105
Ic	742	153
II	2128	874
Total	12902	4883

4.3 Pré-Processamento

Os espectros utilizados neste trabalho passaram por uma etapa de pré-processamento para que as informações extraídas fossem mais confiáveis. Após essa etapa, a quantidade de dados disponíveis para o desenvolvimento do classificador foi reduzida. Isso aconteceu porque foram descobertos alguns arquivos sem as informações de comprimento de onda e de fluxo de radiação necessárias para a caracterização do espectro.

4.3.1 Correção do Redshift

[13] explica que em 1929, Edwin P. Hubble observou um *redshift* (desvio para o vermelho) nos espectros de luz de galáxias distantes da Terra, que indicavam que elas estavam se afastando de nós a uma velocidade proporcional à sua distância. O desvio para o vermelho pode ser creditado principalmente à expansão do Universo. Portanto, a correção do *redshift* é necessária para que o espectro capturado seja analisado como se o evento correspondente estivesse em repouso. Dessa forma, as linhas espectrais podem ser comparadas com as linhas medidas em laboratório. A correção é feita com a Equação 4.

$$\lambda_0 = \frac{\lambda}{z+1} \tag{4}$$

Onde:

- λ_0: comprimento de onda do objeto em repouso;
- λ: comprimento de onda observado;
- z: *redshift*.

4.3.2 Dupla-Filtragem com Filtro Savitzky-Golay

O sistema de Dupla-Filtragem de dados espectrais foi proposto por [2], e é caracterizado por ser um sistema de ajuste de dados para eliminar ruídos e inconsistências em sinais. Este sistema foi testado particularmente em dados espectrais e apresenta bom desempenho no sentido de remoção de ruídos e preservação das características de sinais. A Figura 7 ilustra como estas inconsistências estão presentes nos sinais de SNs ao se realizar uma busca gradiente pelos picos e vales, no esquema de Dupla-Filtragem o objetivo é eliminar estas inconsistências e utilizar apenas os pontos mais importantes dos espectros, os picos e vales reais.

A Figura 7 revela uma busca por picos em vales em uma região (5500-7000 Å) do espectro original da SN1998dx. Esse trecho apresenta apenas um vale (linha de absorção) útil na análise de elementos, entretanto uma busca simples pelos picos e vales identificou uma grande quantidade de pontos. Esses pontos quando presentes em uma análise por RNAs, ou por outras técnicas de aprendizagem de máquina podem interferir severamente na análise identificando falsos positivos ou propagando o erro proveniente do ruído para as etapas finais da análise. No esquema de Dupla-Filtragem estas inconsistências são removidas e uma análise segura pode ser feita, pois todos os dados apresentam apenas os picos e vales reais dos espectros.

O sistema de Dupla-Filtragem consiste em 3 etapas diferentes denominadas: Normalização, Filtragem Simples e Filtragem Dupla.

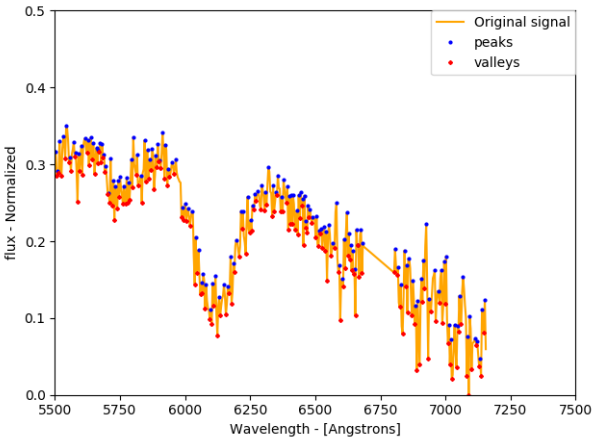


Figura 7: Busca gradiente de picos e vales em um espectro de SN.

- Normalização: Consiste em uma interpolação linear do sinal em 1000 pontos. Em seguida, é feita uma normalização nos valores de fluxo dos espectros para vector de magnitude 1, nesta etapa o sinal está normalizado e pode ser submetido a uma filtragem.
- Filtragem Simples: Consiste em aplicar o filtro de Savitzky-Golay com o tamanho da janela de 71 pontos e com polinómio de grau 9 nos dados normalizados. Esta etapa resulta em um dado suavizado pelo filtro, em que ruídos e resíduos não estão mais presentes. Entretanto, apenas uma filtragem simples não foi capaz de eliminar os ruídos dos espectros e desta forma é que se desenvolve a próxima etapa.
- Filtragem Dupla: Consiste em aplicar o filtro de Savitzky-Golay com o tamanho de janela de 71 pontos e com polinómio de grau 9. Neste ponto, foi possível identificar apenas os picos e vales reais do espectro removendo as inconsistências e ruídos do espectro. A Figura 8 ilustra o espectro normalizado que é utilizado para o treinamento da CIntIa 2.0.

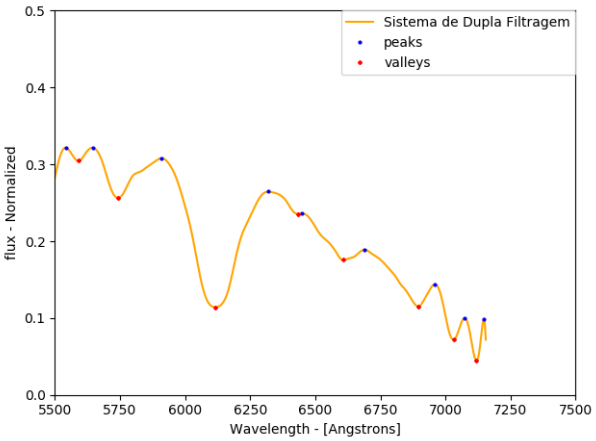


Figura 8: Espectro de uma SN resultante da Dupla-Filtragem.

A construção do sistema de Dupla-Filtragem ocorreu após a comparação com outras estratégias da

literatura ([1], [15], [26]), que não obtiveram desempenho suficiente para o tratamento destes dados. Os parâmetros escolhidos para a Dupla-Filtragem foram derivados de testes das abordagens citadas a fim de escolher um parâmetro de janela de pontos e de grau de polinómio que se aproximasse de uma resposta optimizada. A Figura 9 ilustra a comparação destes sistemas de filtragem com relação ao sistema de Dupla-Filtragem.

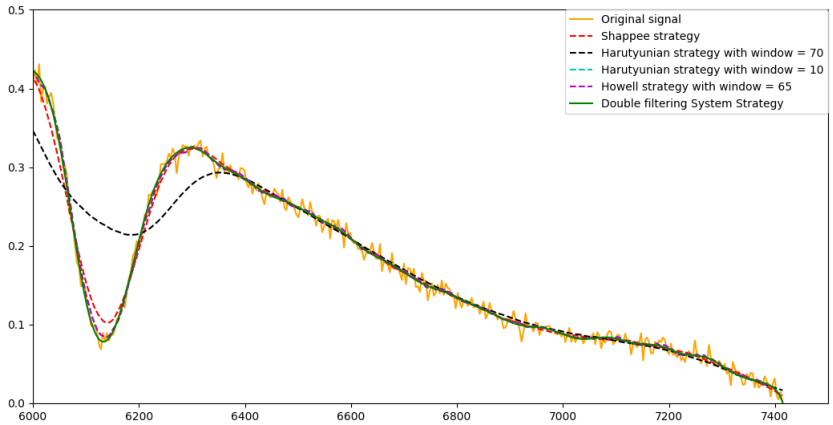


Figura 9: Comparação da Dupla-Filtragem com os sistemas de filtragem para espectros de SNs consultados na literatura.

4.3.3 Seleção de Intervalo de Comprimento de Onda

Como supracitado, a quantidade de espectros efetivamente utilizados é menor que o total apresentado na Tabela 3. Além dos arquivos sem informação, outra razão para o decremento no número de espectros, foi a seleção apenas dos que possuem intervalo de comprimento de onda entre 4000Å e 7000Å, podendo variar para mais e/ou para menos. A escolha desse intervalo decorre da observação das linhas espectrais importantes para a avaliação dos espectros, as quais estão dentro da faixa de luz visível. Anteriormente, a CIntIa 1.0 aceitava espectros com intervalos maiores (3800-7400 Å), com essa mudança conseguimos inserir mais espectros porque a exigência em relação ao tamanho do espectro é menor. A Figura 10 apresenta a quantidade de dados remanescente após a aplicação desse critério de seleção em comparação com a quantidade de dados usados nos treinamentos e teste da primeira versão da CIntIa.

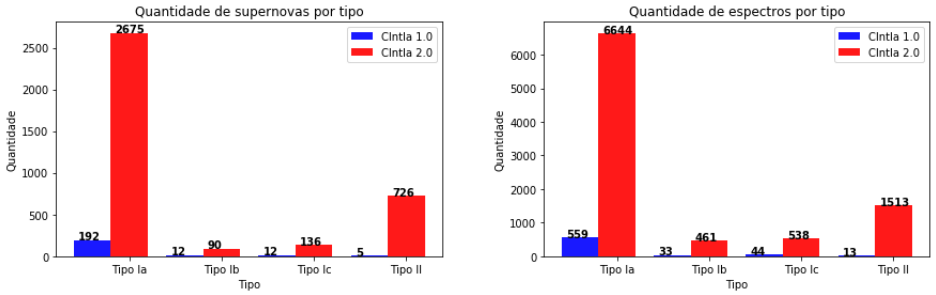


Figura 10: Quantidade de espectros de SNs utilizados no desenvolvimento da CIntIa 2.0 em comparação à quantidade de dados da CIntIa 1.0.

4.4 Extração de Características

Após o pré-processamento, as características necessárias para o treinamento das RNAs que compõem a CIntIa são extraídas. Primeiramente, cada espectro passa por uma interpolação a cada 8 pontos começando em 4000Å até 7000Å, resultando em 375 pontos. As entradas são os pontos contidos em intervalos definidos originalmente por [21], que se referem às regiões em que os elementos H, Si, S e He se manifestam e são as mesmas regiões analisadas por especialistas humanos. Os intervalos são os seguintes:

- 5000Å a 6500Å: para classificar em tipo Ia ou não;
- 5500Å a 7000Å: para classificar em tipo Ib ou não;
- 5500Å a 6500Å: para classificar em tipo Ic ou não;
- 4000Å a 5000Å e 6000Å a 7000Å: para classificar em tipo II ou não.

Cada espectro origina quatro conjuntos de entrada, um para cada RNA. A Figura 11 mostra uma representação gráfica desses intervalos. O primeiro intervalo (acima à esquerda) é a representação dos 188 pontos, na faixa de comprimento de onda 5000-6500 Å, que constituem a entrada da RNA que classifica entre espectros Ia e não-Ia. O segundo intervalo (acima à direita) é a representação dos 187 pontos, na faixa de comprimento de onda 5500-7000 Å, que constituem a entrada da RNA que classifica entre espectros Ib e não-Ib. O terceiro intervalo (abaixo à esquerda) é a representação dos 125 pontos, na faixa de comprimento de onda 5500-6500 Å, que constituem a entrada da RNA que classifica entre espectros Ic e não-Ic. E o quarto intervalo (abaixo à direita) é a representação dos 251 pontos, nas faixas de comprimento de onda 4000-5000 Å e 6000-7000Å, que constituem a entrada da RNA que classifica entre espectros II e não-II.

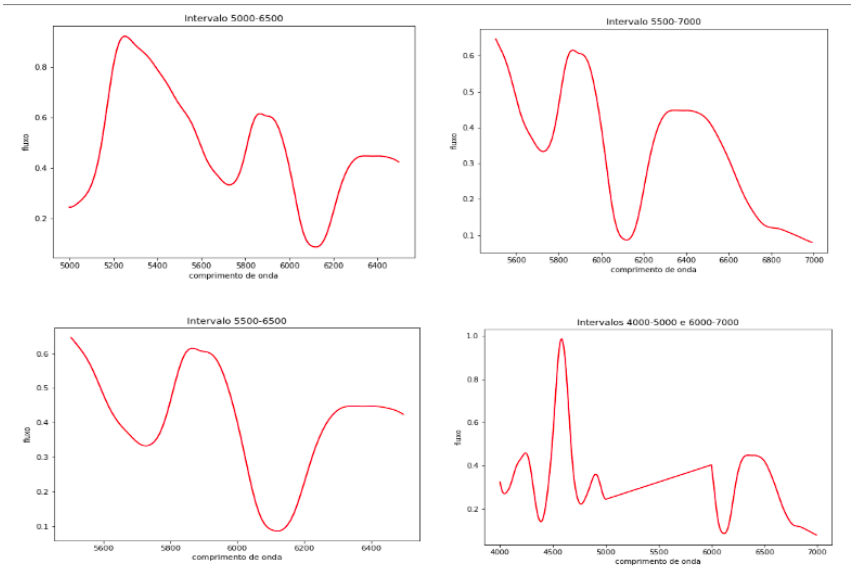


Figura 11: Intervalos escolhidos como entradas das 4 RNAs.

4.5 Parâmetros de Treinamento

Os parâmetros necessários para definir cada RNA da CIntIa são os mesmos apresentados por [21]:

- Taxa de aprendizado = 0.5;
- Momentum = 0.3;
- Bias = 1;

- Erro tolerado = 0.001;
- Função de ativação = Sigmoide.

Executou-se 7 experimentos na CIntIa para cada RNA, como mostrado na Tabela 4, a fim de escolher a topologia mais adequada para cada uma delas. O máximo de camadas ocultas permitidas pelo sistema são duas. A quantidade de espectros selecionados para o treinamento equivale a 60% do total, enquanto 20% é usado na validação e 20% no teste.

Tabela 4: Topologias testadas para escolha da mais adequada para cada RNA do sistema.

Id. da topologia	Neurónios na camada 1	Neurónios na camada 2
1	5	0
2	10	0
3	25	0
4	40	0
5	10	10
6	20	4
7	40	8

5 Resultados e Discussão

Os resultados alcançados neste trabalho redundam no aperfeiçoamento da primeira versão da CIntIa. As topologias das quatro RNAs que compõe o sistema foram ajustadas e uma arquitetura é proposta. Nesta secção tratamos dos resultados dos testes, a nova arquitetura da CIntIa e a discussão decorrente daí, seguida das comparações possíveis com os trabalhos relacionados tratados na revisão da literatura.

5.1 Resultados da Classificação: Tipo Ia

A Tabela 5 exibe a quantidade de espectros usados neste teste, por tipo. Percebemos que a quantidade de espectros do tipo Ia é bastante superior. Isso acontece porque a quantidade de dados do tipo Ia no banco de espectros também é muito maior do que os outros tipos. Optou-se por usar todos os espectros disponíveis, assim o classificador está mais preparado para a diversidade do mundo real, principalmente do tipo Ia. Os espectros do tipo Ia utilizados nesta classificação são aqueles auferidos nas fases espectrais $\geq$-10.9 e $\leq$+10.9, ou seja, entre -10 e +10 dias do brilho máximo. Os espectros dos outros tipos estão sem restrição no que concerne à fase espectral.

Tabela 5: Quantidade de espectros usados no teste da classificação entre Tipo Ia e Tipo Não-Ia.

Tipo	Quantidade de espectros
Ia	703
Ib	90
Ic	106
II	296

As Tabelas 6 e 7 mostram a matriz de confusão e as métricas escolhidas para avaliação dos classificadores binários, de acordo com [28]. Para avaliação dos resultados, também é usado o Índice Kappa que mede a concordância entre duas ou mais classificação [18].

Tabela 6: Matriz de confusão da classificação entre Tipo Ia e Tipo Não-Ia.

	Tipo Ia	Tipo Não-Ia
Tipo Ia	609	94
Tipo Não-Ia	6	486

Tabela 7: Métricas de avaliação da classificação entre Tipo Ia e Tipo Não-Ia.

Acurácia	0.92
Precisão	0.99
Recall	0.87
F1-Score	0.92
Especificidade	0.99
AUC	0.93
Índice Kappa	0.83

A matriz de confusão deixa claro que a maioria dos erros são falsos negativos, ou seja, o classificador não reconhece alguns espectros de SNs do tipo Ia. Essa característica se reflete na métrica Recall, que indica a eficácia para encontrar os positivos. Enquanto isso, a Especificidade, que indica a eficácia para encontrar os negativos é bastante alta. No entanto, o Índice Kappa atribuído é interpretado textualmente como Quase Perfeita (concordância da classificação feita pela CIntIa e a classificação dos especialistas humanos), a interpretação máxima dentre as possíveis, que são em ordem crescente: Pobre (<0.0); Fraca (>0.0 e <0.2); Razoável (>0.2 e <0.4); Moderada (>0.4 e <0.6); Substancial (>0.6 e <0.8); Quase Perfeita (>0.8).

5.2 Resultados da Classificação: Tipo Não-Ia

Nesta subsecção, agrupamos os resultados das 3 RNAs que classificam os espectros em Ib, Ic e II, ou seja, os Não-Ia. A Tabela 8 exibe a quantidade de espectros usados nos 3 testes, por tipo. Todos os espectros estão em fases espectrais diversas.

Tabela 8: Quantidade de espectros usados nos 3 testes de classificação Tipo Não-Ia.

	Espectros Ia	Espectros Ib	Espectros Ic	Espectros II
Classificação Ib	1294	89	106	295
Classificação Ic	1291	90	106	295
Classificação II	1293	90	106	297

As Tabelas 9, 10 e 11 mostram a matrizes de confusão de cada RNA, enquanto a Tabela 12 apresenta as métricas de avaliação.

Tabela 9: Matriz de confusão da classificação entre Tipo Ib e Tipo Não-Ib.

	Tipo Ib	Tipo Não-Ib
Tipo Ib	52	37
Tipo Não-Ib	8	1687

Tabela 10: Matriz de confusão da classificação entre Tipo Ic e Tipo Não-Ic.

	Tipo Ic	Tipo Não-Ic
Tipo Ic	20	86
Tipo Não-Ic	9	1667

Tabela 11: Matriz de confusão da classificação entre Tipo II e Tipo Não-II.

	Tipo II	Tipo Não-II
Tipo II	230	67
Tipo Não-II	8	1481

Tabela 12: Métricas de avaliação das 3 RNAS que classificam os tipos Não-Ia.

	Classificação Ib	Classificação Ic	Classificação II
Acurácia	0.97	0.95	0.96
Precisão	0.87	0.69	0.97
Recall	0.58	0.19	0.77
F1-Score	0.70	0.30	0.86
Especificidade	0.99	0.99	0.99
AUC	0.79	0.59	0.88
Índice Kappa	0.69	0.28	0.84

Nas 3 classificações, a maioria dos erros são falsos negativos. Os valores mais baixos de Recall, F1-Score (indica a qualidade do classificador) e AUC (capacidade do classificador de evitar classificações falsas) deixam claro essa deficiência. O Índice Kappa atribuído concede a qualidade Quase Perfeita ao classificador de tipo II, Substancial para o de tipo Ib e Razoável para o de tipo Ic.

A grande quantidade de falsos negativos, principalmente nas classificações de Ib e Ic podem ser explicadas pela formação comum dos dois tipos de SNs. SNs dos tipo Ib e Ic são ambas de colapso de núcleo o que as separa é apenas a presença do elemento He nas do tipo Ib que está ausente no tipo Ic, isso ocorre porque antes de explodir as estrelas que originam SNs tipo Ic expelem a camada de He, enquanto as estrelas que originam o tipo Ib explodem ainda com essa camada. Portanto, não há um padrão bem definido para os espectros de SNs dos tipos Ib e Ic, o que dificulta a sua classificação por métodos automáticos.

5.3 CIntIa 2.0

Os resultados dos testes realizados em cada uma das RNAs do sistema levaram ao desenvolvimento de uma arquitetura para a segunda versão da CIntIa, que está sendo proposta neste trabalho. Cada uma das RNAs é um módulo na visão geral do classificador, a Figura 12 mostra as topologias das RNAs em cada módulo. Enquanto a Figura 13 apresenta a visão geral da arquitetura da CIntIa 2.0.

O fluxo da classificação inicia identificando se o espectro, previamente pré-processado, é do tipo II ou não, assim a negativa pode ser interpretada como tipo I. Em seguida, são classificados os do tipo Ia. Na sequência os espectros devem ser classificados como tipo Ib ou tipo Ic, que é a diferenciação mais desafiadora a ser feita pelo sistema inteligente, assim como é para os especialistas humanos. Caso o espectro não seja classificado em nenhum dos módulos ele recebe o rótulo de tipo não identificado. A Tabela 13 mostra as métricas de avaliação da CIntIa 2.0 considerando o sistema como um classificador multi-classes, as métricas foram calculadas de acordo com [28].

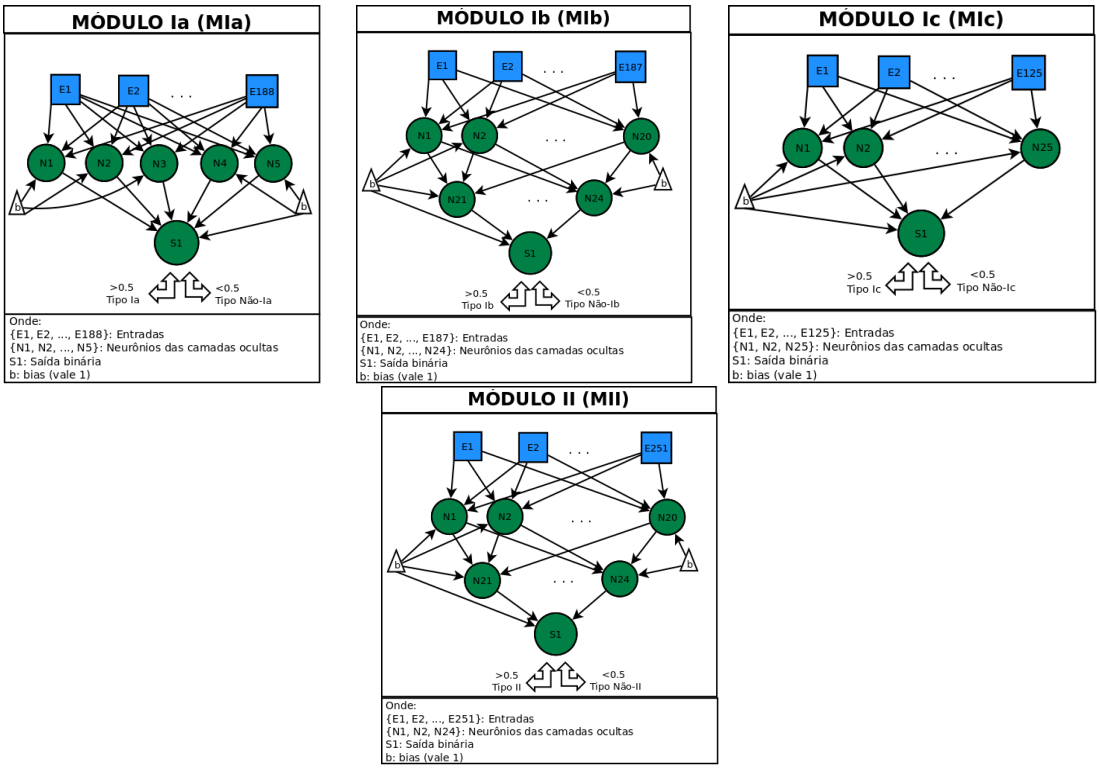


Figura 12: Módulos que compõem a CIntIa 2.0.

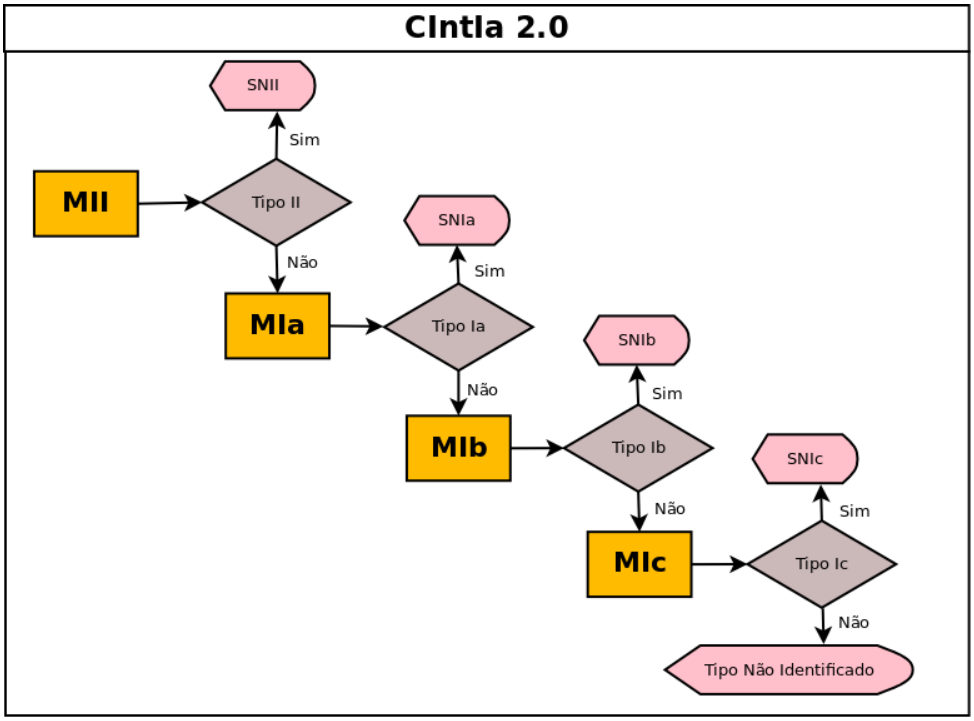


Figura 13: Arquitetura da CIntIa 2.0.

Tabela 13: Métricas de avaliação da CIntIa 2.0.

Acurácia Média	0.95
Taxa de Erro	0.05
Precisão M	0.88
Precisãoμ	0.97
Recall M	0.60
Recallμ	0.76
F1-Score M	0.71
F1-Scoreμ	0.85

A acurácia média tem um excelente resultado, pois a acurácia dos módulos individuais também é muito boa. A taxa de erro é baixa, mas os erros que acontecem são maioritariamente de falsos negativos, logo isso se reflete nos valores de Recall. Apesar disso, o F1-Score (principalmente o μ) confere ao classificador um bom índice.

5.4 Comparação da CIntIa 2.0 com Revisão da Literatura

Uma comparação entre a CIntIa 2.0 e os outros classificadores consultados na literatura é importante para avaliar em que aspectos ela representa uma melhoria e depreender o quão vantajosa é a sua utilização. A Tabela 14 apresenta alguns critérios escolhidos para efetuar essa verificação.

Tabela 14: Comparação entre a CIntIa 2.0 e os classificadores descritos na Revisão da Literatura. Os valores NC indicam que a informação não consta no trabalho. Para as duas versão da CIntIa, as métricas Acurácia, Precisão, Recall e F1-Score se referem aos valores μ.

Classificador	[5]	[16]	[21]	[25]	[29]	[2]	CIntIa 2.0
Inteligência Artificial?	Não	Não	Sim	Sim	Não	Sim	Sim
Espectros Ia	879	1009	559	486	146	3082	6644
Espectros não-Ia	675	1699	90	NC	35	615	2512
Menor Fase Espectral	-15	-15	-3	-3	-5	-15	-1435
Maior Fase espectral	+70	+600	+7	+3	+5	+2959	+8303
Acurácia	NC	NC	0.98	NC	0.95	0.73	0.95
Precisão	NC	NC	0.97	NC	NC	0.93	0.97
Recall	NC	NC	0.94	NC	NC	0.72	0.76
F1-Score	NC	NC	0.95	NC	NC	0.83	0.85

Considerando a quantidade de dados e a abrangência das fases espectrais, a segunda versão da CIntIa é superior a todos os outros classificadores, ou seja, ela compreende uma diversidade maior de espectros. Apesar dos dados de teste não serem exactamente os mesmos testados nos outros classificadores, podemos dizer que a CIntIa 2.0 transcende todos os classificadores consultados, que apresentam bons resultados apenas para espectros na fase de brilho máximo. Considerando que a expectativa de tornar o sistema um classificador atuante em conjunto com telescópios, essa é uma característica importante, ou seja, a CIntIa 2.0 tem grande capacidade de generalização pois o seu treino foi realizado com uma gama de padrões bem maior do que os classificadores comparados. A sua acurácia média e a precisão μ são equiparáveis aos sistemas que também apresentam essa informação. Em relação à CIntIa 1.0, o sistema que adaptamos promovendo melhorias, a CIntIa 2.0 alcança a interpretação "Quase Perfeita"do Índice Kappa, a mesma de sua predecessora. Além disso, a quantidade de dados melhora a análise realizada anteriormente, evidenciando a capacidade das RNAs em aprender padrões dos tipos Ia e II e a dificuldade em diferenciar espectros de SNs Ib e Ic as quais têm a mesma origem física.

6 Conclusão

O trabalho descrito neste artigo, apresenta a CIntIa 2.0, a segunda versão do Classificador Inteligente de Supernovas do tipo Ia, cuja primeira versão foi proposta em 2016. A CIntIa 1.0 foi treinada com uma quantidade limitada de dados, provenientes de apenas uma base de espectros dentre as disponíveis online. A atualização desse sistema é uma necessidade atual, pois cada vez mais cresce a quantidade de telescópios varrendo os céus em busca de SNs. Nesse contexto, a CIntIa 2.0 apresenta melhorias adequadas ao classificador original. Aumentou-se a quantidade e variedade de dados, a amplitude das fases espectrais considerada foi alargada e os resultados, consequentemente, são bons. A CintIa 2.0 faz distinção, com excelência, entre espectros de SNs do tipo II e os outros, do tipo I. As SNs do tipo Ia, as mais utilizadas nas pesquisas de Cosmologia, também são adequadamente reconhecidas pelo classificador. A classificação do tipo Ib tem bons resultados, mas não excelentes, enquanto a classificação do tipo Ic é apenas regular. Uma comparação com outros sistemas relatados na literatura atestam a qualidade da CIntIa 2.0. Uma arquitetura que conecta as RNAs individuais (módulos) em um sistema completo ainda não havia sido proposta para a primeira versão e é um avanço na garantia de que a classificação não será ambígua. Todos os espectros recebem apenas um rótulo identificando o tipo da SN a qual pertencem, ao fim da execução do sistema.

Futuramente, espera-se definir as topologias e outros parâmetros de cada módulo da CIntIa automaticamente, assim é possível melhorar a qualidade da classificação, já que mais testes serão feitos em um tempo menor. Outro desdobramento almejado deste trabalho é avançar na classificação de SNs, propondo formas inteligentes de concluir em que fase espectral a SN está. Torna-se, assim, menos custoso, em termos temporais, descobrir se a SN está na fase de brilho máximo no momento da obtenção do espectro, informação de grande relevância para os estudos que usam SNIa como vela-padrão.

Agradecimentos

Agradecemos à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) pelo suporte financeiro, Código de Financiamento 001. Agradecemos ao Instituto Nacional de Pesquisas Espaciais (INPE) e ao Instituto de Estudos Avançados (IEAv) por disponibilizar espaço físico e recursos computacionais.

Referências

[1] D Andrew Howell, Peter Hoeflich, Lifan Wang, and J Craig Wheeler. Evidence for asphericity in a subluminous type ia supernova: Spectropolarimetry of sn 1999by. *The Astrophysical Journal*, 556, 01 2001. DOI: `https://doi.org/10.1086/321584`.

[2] L. R. Arantes Filho. Classificação inteligente de supernovas utilizando sistemas de regras nebulosas. Dissertação (mestrado em computação aplicada), Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2018.

[3] S. Blondin, K. S. Mandel, and R. P. Kirshner. Do spectra improve distance measurements of type ia supernovae? *Astronomy & Astrophysics*, 526(A81), February 2011. DOI: `https://doi.org/10.1051/0004-6361/201015792`.

[4] S. Blondin, T. Matheson, R. P. Kirshner, K. S. Mandel, P. Berlind, M. Calkins, P. Challis, P. M. Garnavich, S. W. Jha, M. Modjaz, A. G. Riess, and B. P. Schmidt. The spectroscopic diversity of type ia supernovae. *The Astronomical Journal*, 143(5):126, 2012.

[5] Stéphane Blondin and John L. Tonry. Determining the type, redshift, and age of a supernova spectrum. *The Astrophysical Journal*, 666(2), 2007.

[6] B. W. Carroll and D. A. Ostlie. *An Introduction to Modern Astrophysics*. Pearson Addison-Wesley, San Francisco, 2007. 1358 p.

[7] A. Damineli and J. Steiner. *Fascínio do Universo*. Odysseus Editora, São Paulo, 2010. 120 p.

[8] Departamento de Física da Southern Methodist University, ?

[9] Departamento de Física e Astronomia da Georgia State University, 1998.

[10] E. Ferneda. Redes neurais e sua aplicação em sistemas de recuperação de informação. *Ci. Inf. Brasília*, 35(1):25–30, 2006. DOI: http://dx.doi.org/10.1590/S0100-19652006000100003.

[11] Alexei V. Filippenko. Optical spectra of supernovae. *Annual Review of Astronomy and Astrophysics*, 35(1):309–355, 1997. DOI: https://dx.doi.org/10.1146/annurev.astro.35.1.309.

[12] D. V. Fiorin, F. R. Martins, N. J. Schuch, and E. B. Pereira. Aplicações de redes neurais e previsões de disponibilidade de recursos energéticos solares. *Revista Brasileira de Ensino de Física*, 33:01 – 20, 2011. DOI: http://dx.doi.org/10.1590/S1806-11172011000100009.

[13] Juan Garcia-Bellido. Astrophysics and cosmology. In *High-energy physics. Proceedings, European School, ESHEP'99, Casta-Papiernicka, Slovak Republic, August 22-September 4, 1999*, pages 109–186, 1999.

[14] J. Gillochon, J. Parrent, L. Z. Kelley, and R. Margutti. An open catalog for supernova data. *The Astrophysical Journal*, 835(1):64, 2017. DOI: http://dx.doi.org/10.3847/1538-4357/835/1/64.

[15] A H. Harutyunyan, Patrick Pfahler, Andrea Pastorello, S Taubenberger, M Turatto, E Cappellaro, S Benetti, Nancy Elias-Rosa, H Navasardyan, Stefano Valenti, Vallery Stanishev, F Patat, Marco Riello, Giuliano Pignata, and W Hillebrandt. Esc supernova spectroscopy of non-esc targets. *Astronomy and Astrophysics*, 488, 09 2008. DOI: https://doi.org/10.1051/0004-6361:20078859.

[16] Avet Harutyunyan. *Automatic Objective Classification of Supernovae*. PhD thesis, Universitá degli Studi de Padova, 2008.

[17] S. Haykin. *Redes neurais: princípios e práticas*. Bookman, Porto Alegre, 2001.

[18] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, 1977. DOI: https://doi.org/10.2307/2529310.

[19] G. F. Luger. *Inteligência Artificial*. Pearson Education do Brasil, São Paulo, 2013.

[20] M. Modjaz, S. Blondin, R. P. Kirshner, T. Matheson, P. Berlind, F. B. Bianco, M. L. Calkins, P. Challis, P. Garnavich, M. Hicken, S. Jha, Y. Q. Liu, and G. H. Marion. Optical spectra of 73 stripped-envelope core-collapse supernovae. *The Astronomical Journal*, 147(5), 2014.

[21] Marcelo Módolo. *Classificação Automática de Supernovas Usando Redes Neurais Artificiais*. PhD thesis, Instituto Nacional de Pesquisas Espaciais, 2016.

[22] K. S. Oliveira Filho and M. F. O. Saraiva. *Astronomia e Astrofísica*. UFRGS, Rio Grande do Sul, 2014. 784 p.

[23] S. Perlmutter, G. Aldering, G. Goldhaber, R. A. Knop, P. Nugent, P. G. Castro, S. Deustua, S. Fabbro, A. Goobar, D. E. Groom, I. M. Hook, A. G. Kim, M. Y. Kim, J. C. Lee, N. J. Nunes, R. Pain, C. R. Pennypacker, R. Quimby, C. Lidman, R. S. Ellis, M. Irwin, R. G. McMahon, P. Ruiz-Lapuente, N. Walton, B. Schaefer, B. J. Boyle, A. V. Filippenko, T. Matheson, A. S. Fruchter, N. Panagia, H. J. M. Newberg, W. J. Couch, and The Supernova Cosmology Project. Measurements of omega and lambda from 42 high-redshift supernovae. *The Astrophysical Journal*, 517(2):565, 1999. DOI: https://doi.org/10.1086/307221.

[24] Adam G. Riess, Alexei V. Filippenko, Peter Challis, Alejandro Clocchiatti, Alan Diercks, Peter M. Garnavich, Ron L. Gilliland, Craig J. Hogan, Saurabh Jha, Robert P. Kirshner, B. Leibundgut, M. M. Phillips, David Reiss, Brian P. Schmidt, Robert A. Schommer, R. Chris Smith, J. Spyromilio, Christopher Stubbs, Nicholas B. Suntzeff, and John Tonry. Observational evidence from supernovae for an accelerating universe and a cosmological constant. *The Astronomical Journal*, 116(3):1009, 1998.

[25] M. Sasdelli, E. E. O. Ishida, R. Vilalta, M. Aguena, V. C. Busti, H. Camacho, A. M. M. Trindade, F. Gieseke, R. S. de Souza, Y. T. Fantaye, and P. A. Mazzali. Exploring the spectroscopic diversity of type ia supernovae with dracula: a machine learning approach. *Monthly Notices of the Royal Astronomical Society*, 461(2):2044–2059, 2016. DOI: https://doi.org/10.1093/mnras/stw1228.

[26] Stanek K. Z. Pogge R. W. Garnavich P. M. Shappee, B. J. No stripped hydrogen in the nebular spectra of nearby type ia supernova 2011fe. *The Astrophysical Journal Letters*, 762(1):L5, 2012. DOI: https://doi.org/10.1088/2041-8205/762/1/l5.

[27] J. M. Silverman, J. J. Kong, and A. V. Filippenko. Berkeley supernova ia program – ii. initial analysis of spectra obtained near maximum brightness. *Monthly Notices of the Royal Astronomical Society*, 425(3):1819–1888, 2012. DOI: https://doi.org/10.1111/j.1365-2966.2012.21269.x.

[28] M. Sokolova and G. Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing and Management*, 45:427–437, 2009. DOI: https://doi.org/10.1016/j.ipm.2009.03.002.

[29] Fengwu Sun and Avishay Gal-Yam. Quantitative classification of type i supernovae using spectroscopic features at maximum brightness. *ArXiv e-prints*, 2017.

[30] M. Turatto. *Classification of Supernovae*, pages 21–36. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003. DOI: https://doi.org/10.1007/3-540-45863-8_3.

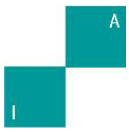
[31] Massimo Turatto, Stefano Benetti, and Andrea Pastorello. Supernova classes and subclasses. *AIP Conference Proceedings*, 937(1):187–197, 2007. DOI: https://doi.org/10.1063/1.3682902.

[32] X. Wang, A. V. Filippenko, M. Ganeshalingam, W. Li, J. M. Silverman, L. Wang, R. Chornock, R. J. Foley, E. L. Gates, B. Macomber, F. J. D. Serduke, T. N. Steele, and D. S. Wong. Improved distances to type ia supernovae with two spectroscopic subclasses. *The Astrophysical Journal Letters*, 699(2):L139, 2009. DOI: https://doi.org/10.1088/0004-637X/699/2/L139.

[33] S. E. Woosley and Thomas A. Weaver. The physics of supernova explosions. *Annual Review of Astronomy and Astrophysics*, 24(1):205–253, 1986. DOI: https://doi.org/10.1146/annurev.aa.24.090186.001225.

[34] Y. Zhu, L. Wang, Y. Xiangyan, B. Gu, X. Li, S. Yang, X. Gong, F. Du, Y. Qi, and L. Xu. Kunlun dark universe survey telescope. volume 9145, page 9145, 2014. DOI: https://doi.org/10.1117/12.2055768.

Inteligencia Artificial, 22(63) (2019), 61-80
doi: 10.4114/intartif.vol22iss63pp61-80



INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

A Flexible Supervised Term-Weighting Technique and its Application to Variable Extraction and Information Retrieval

Mariano Maisonnave[1], Fernando Delbianco[2], Fernando Tohmé[2], Ana Maguitman[1]

[1] Departamento de Ciencias e Ingeniería de la Computación, Universidad Nacional del Sur,
Instituto de Ciencias e Ingeniería de la Computación (UNS-CONICET),
Bahía Blanca, Argentina
{mariano.maisonnave,agm}@cs.uns.edu.ar

[2] Departamento de Economía, Universidad Nacional del Sur,
Instituto de Matemática de Bahía Blanca (UNS-CONICET),
Bahía Blanca, Argentina
fernando.delbianco@uns.edu.ar, ftohme@criba.edu.ar

Abstract

Successful modeling and prediction depend on effective methods for the extraction of domain-relevant variables. This paper proposes a methodology for identifying domain-specific terms. The proposed methodology relies on a collection of documents labeled as relevant or irrelevant to the domain under analysis. Based on the labeled document collection, we propose a supervised technique that weights terms based on their descriptive and discriminating power. Finally, the descriptive and discriminating values are combined into a general measure that, through the use of an adjustable parameter, allows to independently favor different aspects of retrieval such as maximizing precision or recall, or achieving a balance between both of them. The proposed technique is applied to the economic domain and is empirically evaluated through a human-subject experiment involving experts and non-experts in Economy. It is also evaluated as a term-weighting technique for query-term selection showing promising results. We finally illustrate the applicability of the proposed technique to address diverse problems such as building prediction models, supporting knowledge modeling, and achieving total recall.

Keywords:Term Weighting, Variable Extraction, Information Retrieval, Query-Term Selection

1 Introduction

A great number of machine learning and data science applications require identifying domain- or topic-relevant terms. For instance, automatic query formulation requires selecting good query terms; classification requires extracting good features, and in general, any modeling and prediction task requires mechanisms for variable extraction as an initial step to build useful representations. Also, term weighting is a crucial component of these representations since the importance of a term for a domain or topic can usually be numerically estimated and such weights have an impact on the task to be carried out.

Several term-weighting schemes have been proposed in the literature with varying degree of success. Most of these methods apply an unsupervised approach to determine term importance. This is the case of the widely-used TF-IDF weighting scheme, where terms are weighted based on local (TF) and global (IDF) term frequencies, but no class-label information is used to compute these weights. This scheme is limited when it comes to identifying terms that are important for general topics or domains because it has the constraints of being document dependent

ISSN: 1137-3601 (print), 1988-3064 (on-line)

(as it is based on the document and not on the general topic or domain) and label independent (as it is independent of the topic or domain label). Other term weighting methods take a supervised approach to assess the importance of a term in a class. However, term importance is typically taken as a fixed value independent of the task at hand. This represents a limitation because the importance of a term depends on whether the term is needed for query construction, clustering, classification, document summarization, among other tasks. Even for a specific task, such as is the case of query construction, a term may be more or less effective depending on whether the application requires high recall (e.g., looking for all relevant literature about a given topic) or high precision (e.g., looking for a specific piece of information such as a date, place or name). For example, a term that is a useful descriptor for a topic of interest, and therefore useful for attaining high recall, may lack discriminating power, resulting in low precision, unless it is combined with other terms that can discriminate between good and bad results.

This paper proposes a methodology that can be applied to identify domain- or topic-relevant variables from labeled documents. Two forms of relevance are distinguished, namely the relevance of a term as a descriptor, or *descriptive relevance*, and the relevance of a term as a discriminator, or *discriminative relevance*. Guided by this distinction, we propose two weighting schemes that account for these two notions of relevance. These weights are then combined into a parameter-dependent measure to which we refer to as FDD_β, accounting for a general notion of relevance. As we will show in the experiments, the FDD_β measure offers an advantage over several state-of-the-art term-weighting schemes as its parameter can be adjusted to emphasize different aspects of relevance (i.e., descriptive and discriminative relevance). As a consequence, the FDD_β measure has the practical implication of being able to favor either precision or recall, as well as to achieve a balance between both.

The paper is structured as follows. Section 2 briefly describes background concepts and reviews existing term-weighting schemes. Section 3 presents our novel term-weighting scheme, to which we refer to as FDD_β. Section 4 describes the data collection used in our analysis and evaluates FDD_β through a user study and as a query-term selection mechanisms. Section 5.1 illustrates the application of the proposal in variable extraction, knowledge modeling and information retrieval. Finally, section 6 presents the conclusions and outlines future research work.

2 Background and Related Work

Term weighting has been widely used in text classification and information retrieval. For historical reasons, term-weighting methods in text classification were originally borrowed from the information retrieval area, which traditionally applied unsupervised techniques. These traditional term-weighting schemes were designed to improve both recall and precision in the retrieval task. Based on these considerations, Salton and Buckley (1988) claimed that at least three main factors are required in any term weighting scheme. The first is a local factor that stands for the presence of the term in the document. This factor represents whether the term appears at all, and how many times it does. It represents the idea that frequent terms are semantically close to the content of the document. Such a factor is designed to improve recall. The second factor is a global value associated with each term, which represents how frequent the term is in the document collection, in such a way that frequent terms are penalized. The rationale for using this penalizing factor is that common terms are poor discriminators, and as a consequence, they are not useful to tell apart among different documents containing them. It is known that using this factor helps to achieve higher precision. Note, however, that this might be at the expenses of a drop in recall. Finally, the terms are sometimes corrected by a normalization factor.

The simplest local factor is the binary one, which only measures the presence or absence of the term in the document (with values 1 or 0). Another simple and highly-used factor is *term frequency* (TF), which counts the number of times a term appears in a document. It relies on the assumption that most frequent terms are closely related to the content of the document. Leopold and Kindermann (2002) propose *inverse term frequency* (ITF) as an alternative to the classic TF. The ITF weight is based on Zipf's Law and normalizes the local factor to the interval [0,1]. On the other hand, Debole and Sebastiani (2004) propose another variation for the local factor, with a logarithmic transformation in which the terms that are extremely frequent do not increase at the same rate as in TF. Hassan et al. (2007) present a new local factor using a variant of *TextRank* [23] as a scoring function, which recursively increases the importance of a term by determining the degree of connectivity between other terms using co-occurrence as a way to measure connectivity. *TextRank* is based on the renowned *PageRank* algorithm [25].

A simple global factor can be computed by counting the number of documents in the corpus that contain the term. We refer to this factor as *term global frequency* (TGF). The best known global factor is the *inverse document frequency* (IDF) function [28], which relies on the assumption that terms that occur in many documents are not good for discrimination. The TF-IDF formulation is a widely used weighting scheme because it reaches a good balance between the local (TF) and the global (IDF) factor. Tokunaga and Makoto (1994) propose a variant of IDF named *weighted inverse document frequency* (WIDF) that penalizes frequent terms by taking into

account the number of times they occur in each document of a collection. A variant of TF-IDF called *modified inverse document frequency* (MIDF) that combines TF and WIDF is proposed in [4]. According to the authors, MIDF outperforms TF-IDF in text classification. Also, they remark the ability of MIDF to adapt to dynamic document corpora.

While unsupervised weighting schemes have proved to be useful in many scenarios, these methods do not take full advantage of class information, which is available as part of the training set in a class-labeled collection. The design of term-weighting methods that exploit class information gained increasing attention, giving rise to different forms of supervised term-weighting schemes [2, 3, 6, 8, 9, 14, 31, 32]. A simple method that uses class information can be computed by counting the number of documents in a class that contain the term. We use TGF* to refer to this method. Another supervised weighting scheme is the *inverse class frequency factor* (ICF), which relies on the assumption that a term that occurs in documents from a single class are good discriminants of that class. Conversely, terms that appear in documents from different classes contribute poorly to the identification of the class of the documents. So, this factor penalizes a term proportionally to the number of different classes in which the term appears. Other functions from traditional information theory such as *mutual information* (MI), *chi-squared* (χ^2), *information gain* (IG) and *gain ratio* (GR) can be used as supervised term-weighting scores to capture the idea that the most valuable terms for categorization under a class are those that are distributed most differently in the sets of positive and negative examples of the class. A classic feature scoring function that is commonly used as a global term-weighting factor is the *odds ratio* (OR) [30]. This score is based on the conditional probability of a term occurring given a class. Another supervised technique known as *category relevance factor* (CRF) computes a factor that stands for the discriminating power of a feature to a class [5]. Some feature selection techniques that were adapted for term weighing are the *Galavotti-Sebastiani-Simi* coefficient (GSS) [11] and *entropy-based category coverage difference* (ECCD) [16]. Liu et al. (2009) propose a probabilistic-based technique (Prob) that involves two ratios directly related to the term's strength in representing a category. These ratios are such that one of them increases if the term appears in a lot of documents of the class (descriptive power), while the other tends to be higher if the term appears only in documents of the class (discriminating power). Another scheme uses a *relevancy frequency factor* (RF) [15] that takes into account term distribution across classes. According to this scheme, the higher the concentration of high-frequency terms in the positive category than in the negative one, the greater the contribution to classification. Domeniconi et al. (2015) propose a supervised variant of IDF called *inverse document frequency excluding category* (IDFEC). Similar to IDF, IDFEC penalizes frequent terms, but different form IDF it avoids penalizing those terms occurring in several documents belonging to the same class. Another variant also proposed in [7] results from combining IDFEC and RF, resulting in the IDFEC_B scheme.

Table 1 shows the definitions of the main scores presented above using the following notation [7, 14]:

- A denotes the number of documents that belong to class c_k and contain term t_i.
- B denotes the number of documents that belong to class c_k but do not contain the term t_i.
- C denotes the number of documents that do not belong to class c_k but contain the term t_i.
- D denotes the number of documents that do not belong to class c_k class and do not contain the term t_i.
- N denotes the total number of documents in the collection (i.e., $N = A + B + C + D$).

Note that some formulations include the expression $\max(X, 1)$ to prevent the possibility of undefined values, such as divisions by zero or $\log(0)$.

3 A Novel Supervised Term-Weighting Score

Based on the idea that class labels convey useful information for term weighting and on the fact that the importance of a term in a topic or domain depends on the specific objectives at hand (e.g., attaining high recall, high precision or both), we distinguish two relevancy scores. The first score represents the importance of a term to describe the class or topic, and we refer to it as *descriptive relevance* (DESCR). Given a term t_i and a class c_k the DESCR score is expressed as:

$$\text{DESCR}(t_i, c_k) = \frac{|d_j : t_i \in d_j \wedge d_j \in c_k|}{|d_j : d_j \in c_k|},$$

which is equivalent to $A/(A + B)$, using the notation adopted in the previous section. The descriptive relevance of a term in a class stands for a simple idea: those terms that occur in many documents of a given class are good descriptors of that class. As a consequence, we compute it as the portion of documents in the class that contain the given term.

Name	Formulation
TGF	$A + C$
IDF	$\log(N/(A + C))$
TGF*	A
MI	$\log((N \times \max(A, 1))/((A + B)(A + C)))$
χ^2	$N((AD - BC)^2/((A + C)(B + D)(A + B)(C + D)))$
OR	$\log((\max(A, 1) \times D)/\max(B \times C, 1))$
IG	$(A/N)\log(\max(A, 1)/(A+C)) - ((A+B)/N)\log((A+B)/N) + (B/N)\log(B/(B+D))$
GR	$IG/(-((A + B)/N)\log((A + B)/N) - ((C + D)/N)\log((C + D)/N))$
GSS	$\log(2 + ((A + C + D)/(\max(C, 1))))$
Prob	$\log(1 + (A/B)(A/C))$
RF	$\log(2 + (A/\max(C, 1))$
IDFEC	$\log((C + D)/\max(C, 1))$
TGF-IDFEC	$(A + C)(\log((C + D)/\max(C, 1)))$
TGF*-IDFEC	$A \times (\log((C + D)/\max(C, 1)))$
IDFEC_B	$\log(2 + (A + C + D)/(\max(C, 1)))$

Table 1: Definitions of term weighting schemes.

The second relevancy score represents the importance of a term to discriminate a class or topic, and we call it *discriminative relevance*. For a term t_i and a class c_k the DISCR score is expressed as:

$$\text{DISCR}(t_i, c_k) = \frac{|d_j : t_i \in d_j \wedge d_j \in c_k|}{|d_j : t_i \in d_j|},$$

which is equivalent to $A/(A + C)$. The discriminative relevance of a term in a class is based on the idea that a term is a good discriminator of a class if it tends to occur only in documents of that class. We compute it as the portion of documents that contain the given term that belong to the class. The DESCR and DISCR scores can be seen as the supervised versions of the semi-supervised techniques proposed in [21, 22] to compute the descriptive and discriminative power of a term in a topic.

To better understand the notions of descriptors and discriminators, the best terms based on these metrics were analyzed for the Economy domain using an expert manually labeled collection (described in section 4.1). The terms with the highest descriptive power (DESCR > 0.2) and the ones with the highest discriminating power (DISCR > 0.85) are shown in the word clouds of figures 1 and 2, respectively. It is possible to observe that descriptors are terms that are common in the Economy domain, while discriminators are more specific terms that can be found in particular areas in the Economy domain.

We propose to combine the DESCR and DISCR scores by means of the following general term relevancy formula:

$$\text{FDD}_\beta(t_i, c_k) = (1 + \beta^2)\frac{\text{DISCR}(t_i, c_k) \times \text{DESCR}(t_i, c_k)}{(\beta^2 \times \text{DISCR}(t_i, c_k)) + \text{DESCR}(t_i, c_k)}.$$

The FDD_β measure is derived from the F_β formula traditionally used in information retrieval to give β times more importance to recall than to precision:

$$\text{F}_\beta(t_i, c_k) = (1 + \beta^2)\frac{\text{precision}(t_i, c_k) \times \text{recall}(t_i, c_k)}{(\beta^2 \times \text{precision}(t_i, c_k)) + \text{recall}(t_i, c_k)}.$$

By using a β value higher than 1 in the FDD_β function we can weight descriptive relevance higher than discriminative relevance (by placing more emphasis on terms that help achieving good recall) while a β smaller than 1 weights descriptive relevance lower than discriminative relevance (by placing more emphasis on terms that help achieving good precision).

We will show next that FDD_β, can serve the purpose of approximating term relevancy in a topic. This score can be computed for any collection of documents labeled as relevant or irrelevant to the given topic.

We will show next that despite the simplicity of the FDD_β score, it is highly effective both as an estimator of expert assessments of relevance and for guiding the selection of good query terms. In particular, we will show how the tunable parameter β offers a means to favor different objectives in the information retrieval task.

Figure 1: Word cloud based on the top-ranked terms according to the DESCR weighting scheme (only words with DESCR > 0.02 are shown).

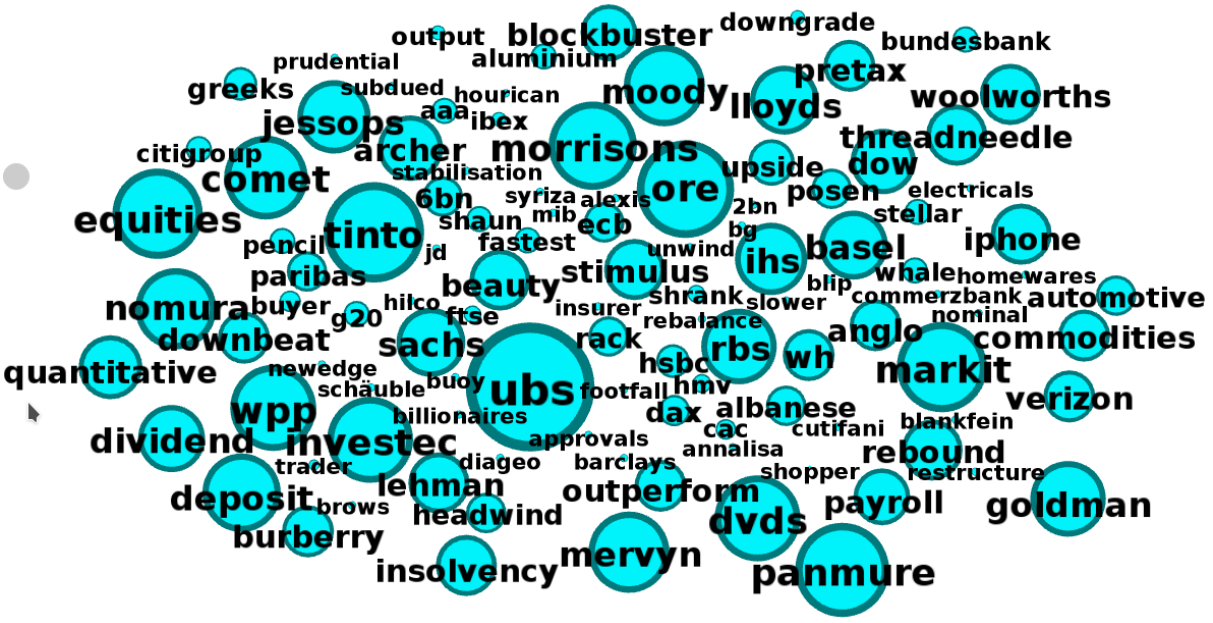


Figure 2: Word cloud based on the top-ranked terms according to the DISCR weighting scheme (only words with DISCR > 0.85 are shown).

4 Evaluation

The goal of this section is to compare the FDD_β weighting score against other term-weighting schemes. The evaluation comprises a human-subject study and an experiment for assessing the effectiveness of the evaluated techniques in information retrieval. The evaluations were carried out using a labeled collection of news articles and human subjects' relevance assessments, as described next. The labeled collection of news articles and the

human subjects' relevance assessments are available for download at http://ir.cs.uns.edu.ar/datasets.

4.1 Data Collection

The *The Guardian* newspaper (https://www.theguardian.com/) was selected as a source to collect a set of digital news. *The Guardian* is a British daily newspaper with an open platform that allows accessing over 1.9 million pieces of content, including full-text news articles. A simple Python script was developed to collect news articles through an API provided by the newspaper. Only news coming from the Politics, World news, Business and Society sections were collected. Although several fields are available for each news article, only the news titles and full body text were used. A simple preprocessing step was carried out to eliminate stopwords and punctuation marks, as well as to transform the text into a sequence of lowercase terms. A total of 1689 news articles corresponding to January 2013 were manually labeled by two experts in Economy as relevant (537) or irrelevant (1152) to the economy. News were considered relevant to the economy if the described event has some impact on the economy.

To complete the labeling task, both experts read the news articles and agreed on whether each of them was relevant or not. It is worth mentioning that the manual labeling stage was important due to the fact that news identified by the experts as relevant to the economy do not exactly correspond to those from the "Business" section (418 out of 512) but also some of them were in the Politics (39 out of 290), World news (43 out of 650), and Society (37 out of 237) sections.

To better understand the discrepancies between the news in the "Business" section and what was identified by the experts as relevant to the economy we present in table 2 some examples of news titles that were labeled by the experts. In these example it is possible to see that many news that belong to the "Business" section have to do with the *World Economic Forum* that took place in Davos in 2013. While this social event may be of interest to the Business community it was considered by the experts as non-relevant to the economy. Similarly, other news that describe social events related to the area of Economy were labeled as non-relevant by the experts. On the other hand, the analysis of news outside the "Business" section allowed to identify some news with impact on local and/or global economies. This is the case of many news in the "Politics" section that involved the announcement of political decisions of countries. Other news in the "Society" section that were identified as relevant to the economy include news about retirements, social benefits, public health, among other topics. While these news have a focus on society, and hence belong to the "Society" section, many of them have a direct impact on the economy. As is illustrated by the selected examples, the "World news" section also included several news from countries different from Great Britain that were identified by the expert as news with an impact on the economy.

The collection of 1689 expert-labeled news articles was used as the training set. Also, a reduced set consisting of 100 expert-labeled news articles (not included in the training set) was used as the validation set. The total number of terms in these news articles is 38511. However, to reduce the dimensionality of the dataset only those terms that occur in at least six news articles were considered, resulting in a set of 10373 terms.

4.2 Validation by User Study

Eight volunteer subjects were recruited for an experiment conducted online. The group of subjects included four volunteers with no background in Economy and four others with a Ph.D. degree in Economy. We refer to the first group as *non-experts* and to the second group as *experts*. The motivation for examining and comparing the assessments of both expert and non-expert users was to determine if both populations exhibit different behaviors and to evaluate the discrepancy between our tool and the two groups. A set of 50 terms (10 lists of 5 terms each) and another set of 100 terms (20 lists of 5 terms each) were strategically selected from the 10373 terms of the dataset. The selection was made based on the distribution of term frequency in light of Zipf's law. The goal was to avoid providing low-frequency terms (which are many) more chances of being selected than high-frequency terms (which are a few). To complete an initial parameter-adjusting stage, two of the experts were asked to agree on the economic relevance of each of the words from the 50-term set. The experts were asked to rate these terms with a score ranging from 0 (economic irrelevant) to 5 (economic very relevant). We used these ratings and the labeled collection to learn the best β value for the FDD_β method. As can be seen in figure 3 the highest Pearson correlation between the expert ratings and the FDD_β values was 0.797671, which was achieved for $\beta = 0.477$.

To complete the validation stage we asked the eight volunteer subjects to rate the 100 terms using a 0-5 scale, and we computed DESCR, DISCR, $\text{FDD}_{0.477}$ and the 15 weighting schemes listed in table 1 for these terms. In the first place, we tested the level of agreement between pairs of users belonging to the non-expert group and between pairs of users in the expert group. Table 3 presents the means and standard deviations obtained as a result of such analysis. It is possible to observe that there is a high level of agreement in both groups, being this agreement higher in the expert group.

Table 2:

News in the Business section but not relevant to the economy		
Title	Date	Section
Jessops goes into administration: staff and customers react	2013-01-10	Business
How to spot a fake indie business	2013-01-03	Business
Women's rights activists protest at Davos - in pictures	2013-01-26	Business
Davos diary: Paul Coelho becomes most retweeted attendee	2013-01-24	Business
Davos 2013: day two - as it happened	2013-01-24	Business
News outside the Business section but relevant to the economy		
Title	Date	Section
Spain's economy shrinks again and remains deep in recession	2013-01-30	World news
Britain leaving EU major threat to global economy, says Sir Martin Sorrell	2013-01-23	Politics
Benefits and child credits squeeze pushes 200,000 children into poverty	2013-01-17	Society
Cameron faces unfriendly fire from military chiefs over defence budget	2013-01-31	Politics
News in the Business section and relevant to the economy		
Title	Date	Section
Shell dividend pleases shareholders but profits disappoint City	2013-01-31	Business
US recovery stalls after first quarter of negative growth in three years	2013-01-30	Business
US economy shrinks unexpectedly despite improving job market	2013-01-30	Business
UK GDP shrank by 0.3% in fourth quarter	2013-01-25	Business
News outside the Business section and not relevant to the economy		
Title	Date	Section
Four US states considering laws that challenge teaching of evolution	2013-01-31	World news
Syria may respond to Israeli air strike, says ambassador	2013-01-31	World news
David Cameron arrives in Libya on surprise visit	2013-01-31	Politics
Queen Beatrix of the Netherlands - in pictures	2013-01-28	World news
Hugo Chávez fights for life as supporters pray in Venezuela	2013-01-04	World news
Man arrested in East Sussex over Nepal war crimes	2013-01-03	World news
Criminals should spend longer in jail, says Chris Grayling	2013-01-05	Society

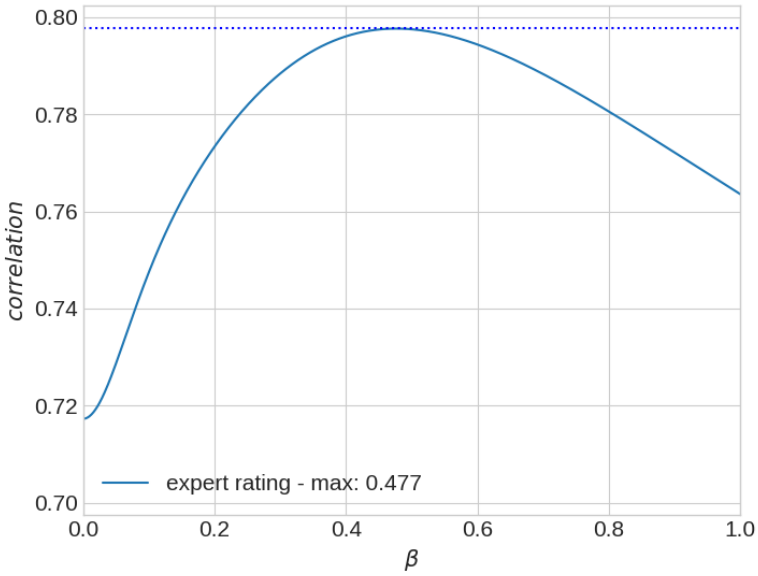


Figure 3: Learning the optimal β value (maximum correlation equal to 0.797671 for $\beta = 0.477$).

non-expert	experts
$\mu = 0.839475$, $\sigma = 0.037791$	$\mu = 0.876390$, $\sigma = 0.009438$

Table 3: Means (μ) and standard deviations (σ) of correlations to test agreement among non-experts and among experts.

Table 4 presents results on the level of agreement between the two different groups of users (non-experts and experts), and on the level of agreement between each of these groups (non-experts and experts) and $\text{FDD}_{0.477}$. The level of agreement is given by the averaged Pearson correlation coefficients.

non-experts and experts	non-experts and $\text{FDD}_{0.477}$	experts and $\text{FDD}_{0.477}$
$\mu = 0.80383$, $\sigma = 0.053205$	$\mu = 0.685598$, $\sigma = 0.054969$	$\mu = 0.752352$, $\sigma = 0.018904$

Table 4: Means (μ) and standard deviations (σ) of correlations computed between non-experts and experts, non-experts and $\text{FDD}_{0.477}$, and experts and $\text{FDD}_{0.477}$.

Finally, to compare the effectiveness of the weighting schemes as predictors of subjects' judgments of term relevancy we computed the Pearson correlation coefficients between the averaged ratings assigned by the subjects and those computed by each of the weighting schemes. Table 5 summarizes these correlations. The reported values correspond to the correlations between each of the methods and the different groups of users. In all these cases we observe that $\text{FDD}_{0.477}$ outperforms the other methods, being TGF*-IDFEC the second most effective one in estimating human subjects' relevance assessments.

4.3 Retrieval Effectiveness

In this section, we analyze the performance of FDD_β as a mechanism for query-term selection, and we compare it with other state-of-the-art weighting schemes. In the first place, the training set described in section 4.1 was used to select the top-rated terms based on FDD_β by assigning different values to the parameter β. Simple queries were generated using the selected terms and then evaluated by means of the classical recall, precision and F_1 metrics. The results are shown in figure 4. As expected, the highest recall using the FDD_β-based term selection mechanisms is obtained with larger values of β while the highest precision is obtained for smaller values. Note, for instance, that terms such as `uk` occur often in relevant news articles given the fact that news were collected from a British newspaper. As a result, the term `uk` results in a high-recall query. However, `uk` is not a good discriminator for the Economy domain, resulting in a low-precision query. On the other hand, terms such as `adp`,

Method	non-expert (averaged)	expert (averaged)	non-expert and expert (averaged)
TGF	0.283553	0.365037	0.332324
IDF	-0.488816	-0.563704	-0.539138
TGF*	0.574110	0.642607	0.623198
MI	0.697053	0.659659	0.694604
χ^2	-0.164537	-0.087771	-0.128992
OR	0.432627	0.306599	0.378188
IG	0.663296	0.705736	0.701123
GR	0.663296	0.705736	0.701123
GSS	0.722761	0.757015	0.757807
Prob	0.654187	0.697007	0.691990
RF	0.472824	0.407394	0.450543
IDFEC	-0.226397	-0.325872	-0.283050
TGF-IDFEC	0.603975	0.676551	0.655882
TGF*-IDFEC	0.721871	0.774026	0.766110
IDFEC_B	-0.221061	-0.320304	-0.277466
DESCR	0.574110	0.642607	0.623198
DISCR	0.662481	0.610804	0.651848
$FDD_{0.477}$	**0.735456**	**0.791969**	**0.782264**

Table 5: Correlations between methods and ratings obtained by averaging non-expert, expert and all human subjects' scores.

`jp`, `ubs`, `forecasts` and `ftse` are not good descriptors but tend to occur only in relevant news articles. This means that they are good discriminators, offering a mechanism to ensure high precision, although usually at the expense of low recall. Other terms, such as `sales`, `growth` and `business` achieve a balance between descriptive and discriminative relevance, resulting in a good F_1 score.

The query term with the highest F_1 score is `growth`, which is the term achieving the best FDD_β for a range of β values that begins approximately at 0.4 and ends close to 1.2. Note that this range includes 0.477, which is the value that yields the highest correlation between FDD_β scores and experts' relevance assessments. Based on this preliminary analysis the best FDD_β achieves an F_1 score as high as the one obtained with the two most effective state-of-the-art weighting schemes (TGF-IDFEC and TGF*-IDFEC). The top-rated term according to the three weighting schemes is `growth`. It is interesting to note that for small β values FDD_β outperforms these two methods in terms of precision while for large β values FDD_β outperforms these two methods in terms of recall.

The validation set described in section 4.1 was used to determine if the best queries identified using the training set were effective on a different set. The resulting recall, precision and F_1 metrics computed on the validation set are shown in figure 5. Given that the validation set was small, some of the most discriminating terms identified during the training stage (`adp` and `ubs`) were absent from the validation set, resulting in an empty answer set when used as query terms. However, those terms with a good balance between descriptive and discriminative relevance (`sales`, `growth` and `business`) achieve the highest F_1 scores when used as query terms on the validation set. This preliminary analysis indicates that the proposed method does not overfit the training data.

To further investigate into the effectiveness of FDD_β, we used the top-ranked terms based on different β values to formulate different types of queries. The evaluated queries for FDD_β include single-term queries (FDD_β), disjunctive queries with two terms (FDD_β (OR(2))), disjunctive queries with three terms (FDD_β (OR(3))), conjunctive queries with two terms (FDD_β (AND(2))) and conjunctive queries with three terms (FDD_β (AND(3))). For comparison purposes, we used TGF*-IDFEC, which based on our previous analysis represents one of the most effective state-of-the-art weighting schemes, and proceeded to formulate different queries based on the top-ranked terms according to this scheme. In this sense, the evaluated queries for TGF*-IDFEC include single-term queries (TGF*-IDFEC), disjunctive queries with two terms (TGF*-IDFEC (OR(2))), disjunctive queries with three terms (TGF*-IDFEC (OR(3))), conjunctive queries with two terms (TGF*-IDFEC (AND(2))) and conjunctive queries with three terms (TGF*-IDFEC (AND(3))).

Figure 6 shows the effectiveness of these queries on the training set based on recall, precision and F_1. These

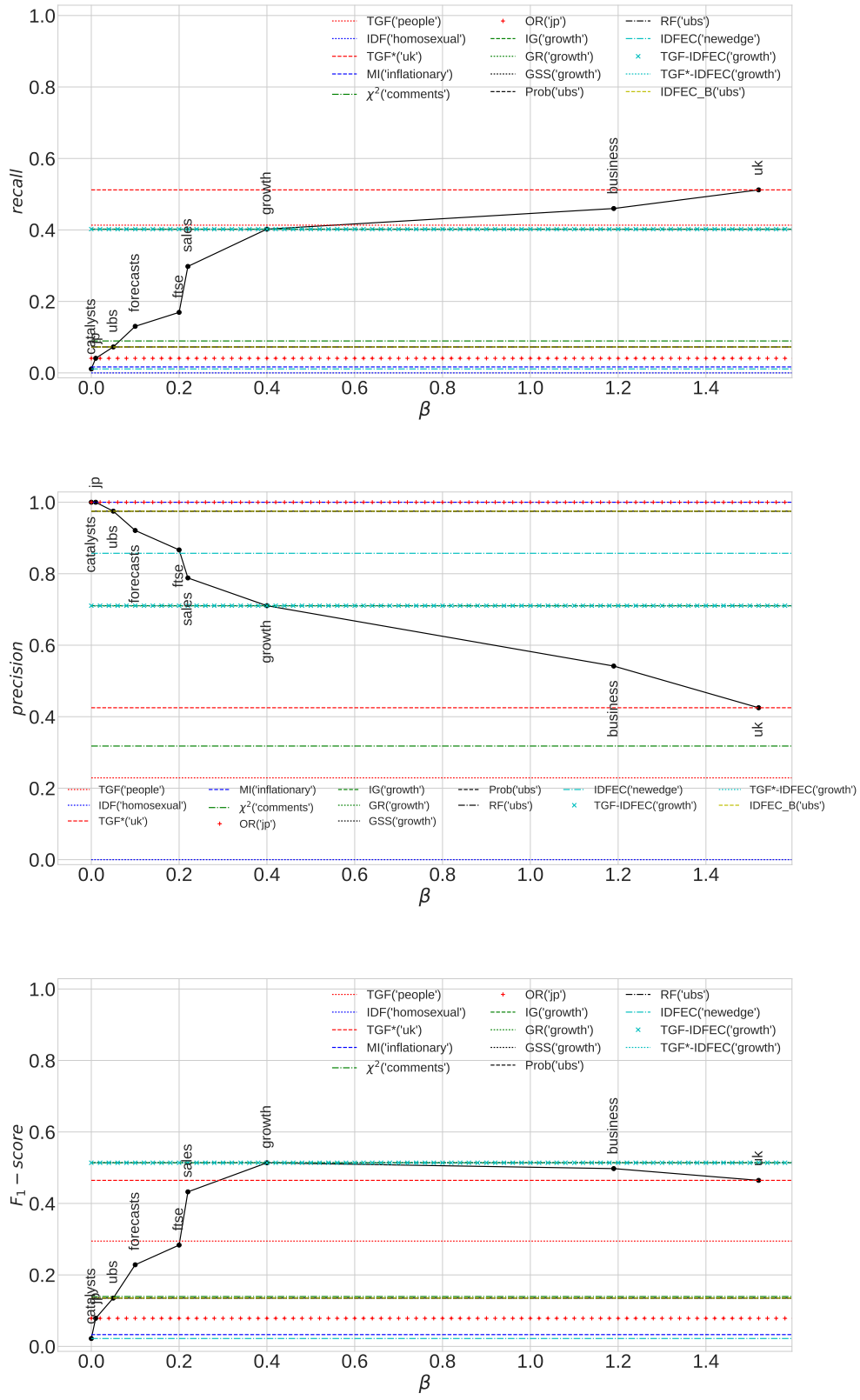


Figure 4: Effectiveness on the training set of queries generated based on term weighting schemes. The black solid curve corresponds to the effectiveness of query terms selected using FDD_β on the training with different β values.

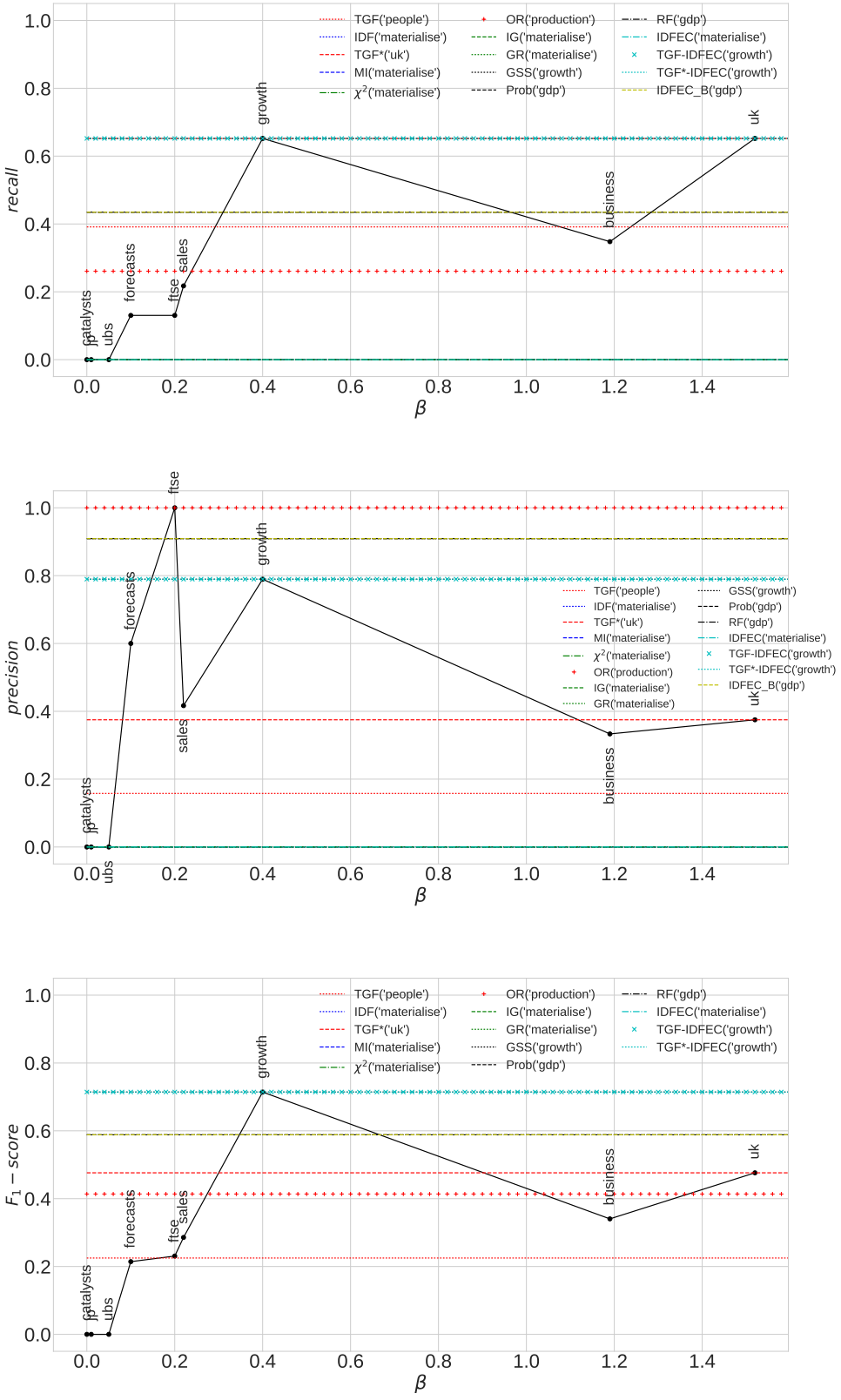


Figure 5: Effectiveness on the validation set of queries generated based on term weighting schemes. The black solid curve corresponds to the effectiveness of query terms selected using FDD_β on the training set with different β values

results indicate that disjunctive queries with three terms selected based on FDD_β (FDD_β (OR(3))) achieve the same F_1 as the best TGF*-IDFEC-based query when the FDD_β-based queries take a β-value in an interval that includes 0.477 (which is the β with the highest correlation between FDD_β scores and experts' relevance assessments). Also, it is interesting to note that for high β values, the "FDD_β (OR(3))" queries outperform all the TGF*-IDFEC queries in terms of recall, while for small β values several combinations of FDD_β-based queries outperform the TGF*-IDFEC-based queries in terms of precision.

Figure 7 shows the performance of the queries described above on the validation set. These results show that for some ranges of β, the single-term queries (FDD_β) and some of the disjunctive queries with three terms (FDD_β (OR(3))) achieve an F_1 score equal to that achieved by the best TGF*-IDFEC-based query. Once again, some of the queries generated based on the "FDD_β (OR(3))" scheme outperform all other queries in terms of recall. On the other hand several FDD_β and TGF*-IDFEC-based schemes achieve the highest possible precision value (1.0). Once again, the results indicate that the proposed query generation schemes do not overfit the training data.

5 Application in Variable Extraction, Modeling and Retrieval

As can be seen in the reported results, FDD_β performs consistently well, not only as an estimator of human subjects' relevance assessments but also as a method for guiding the selection of good query terms. This opens numerous opportunities for applying the proposed scheme on different scenarios. This section describes some possible applications that may benefit from the proposed term-weighting technique.

5.1 Building Prediction Models

The proposed technique can be used to extract variables from digital media with the ultimate goal of building models of prediction, explanation and description. Figure 8 shows a word-cloud visualization with the top-ranked terms based on the training data using $FDD_{0.477}$ as weighting scheme. To avoid overcharging the figure only terms with $FDD_{0.477} > 0.6$ are shown.

It is worth mentioning that the proposed FDD_β measure can be computed not only for words in a collection (as illustrated so far) but also on other types of lexical units, such as stems, n-grams, named entities, noun phrases, among others. Figure 9 presents a word cloud with the top-ranked named entities based on $FDD_{0.477}$ identified in the manually labeled data collection used in our previous analysis. The *Stanford Named Entity Recognition* tagger (Stanford NER) [10] was used to identify these named entities.

A subsequent modeling step would be to identify different types of dependency relations between these variables. For instance, some *causal relations* that can be recognized are `investment-growth-gdp`, `spending-market-recovery` and `sales-companies-investment-gdp`. Other types of relations, such as *close associations* are illustrated by `credit-debt-banks`, `recession-decline`, `trading-stock-ftse` and `debt-bank-investors-trading-recession`. A possible *simultaneity relation* is given by *market-prices*. It is also interesting to note that `christmas`, one of the words selected by the method, may capture *seasonality in a casual series*. Automatically identifying these types of relations is a challenging problem that we plan to address as part of our ongoing research work. In particular, we plan to investigate into the problem of finding causal relations with the purpose of automatically building different types of networks, such as Bayesian networks [26].

5.2 Supporting Knowledge Modeling

Building knowledge models is a difficult and costly task. There are several initiatives aimed at providing intelligent support to facilitate the construction of knowledge models, as is the case of the family of intelligent suggesters for concept mapping described in [17, 20].

Concept mapping [24] is a vehicle for knowledge modeling that was first proposed in education, to enable students to externalize their knowledge by constructing a graphical representation of concepts and their relationships. Concept mapping systems have been used by users ranging from elementary school students to scientists to support generation, storage of, and access to concept maps in electronic form. In addition to providing basic operations needed to draw concept maps, these systems can be augmented with methods aimed at facilitating knowledge extension. In particular, the automatic identification of terms relevant to the modeled domain allows to extend a knowledge model beyond information that has already been captured. The visualizations presented in figures 8 and 9 can support the construction of concept maps, by helping in the process of choosing relevant terms in a particular domain, which is typically the initial step in any knowledge modeling task.

Concept maps are usually organized in a hierarchical fashion, where more general terms tend to appear on the top of the concept map (i.e., in or close to the root node), while more specific terms tend to occur toward the bottom (i.e, in or close to the leaf concepts). The term-weighting method proposed in this work may offer a novel solution to the problem of identifying different terms and entities that can be suggested for addition at

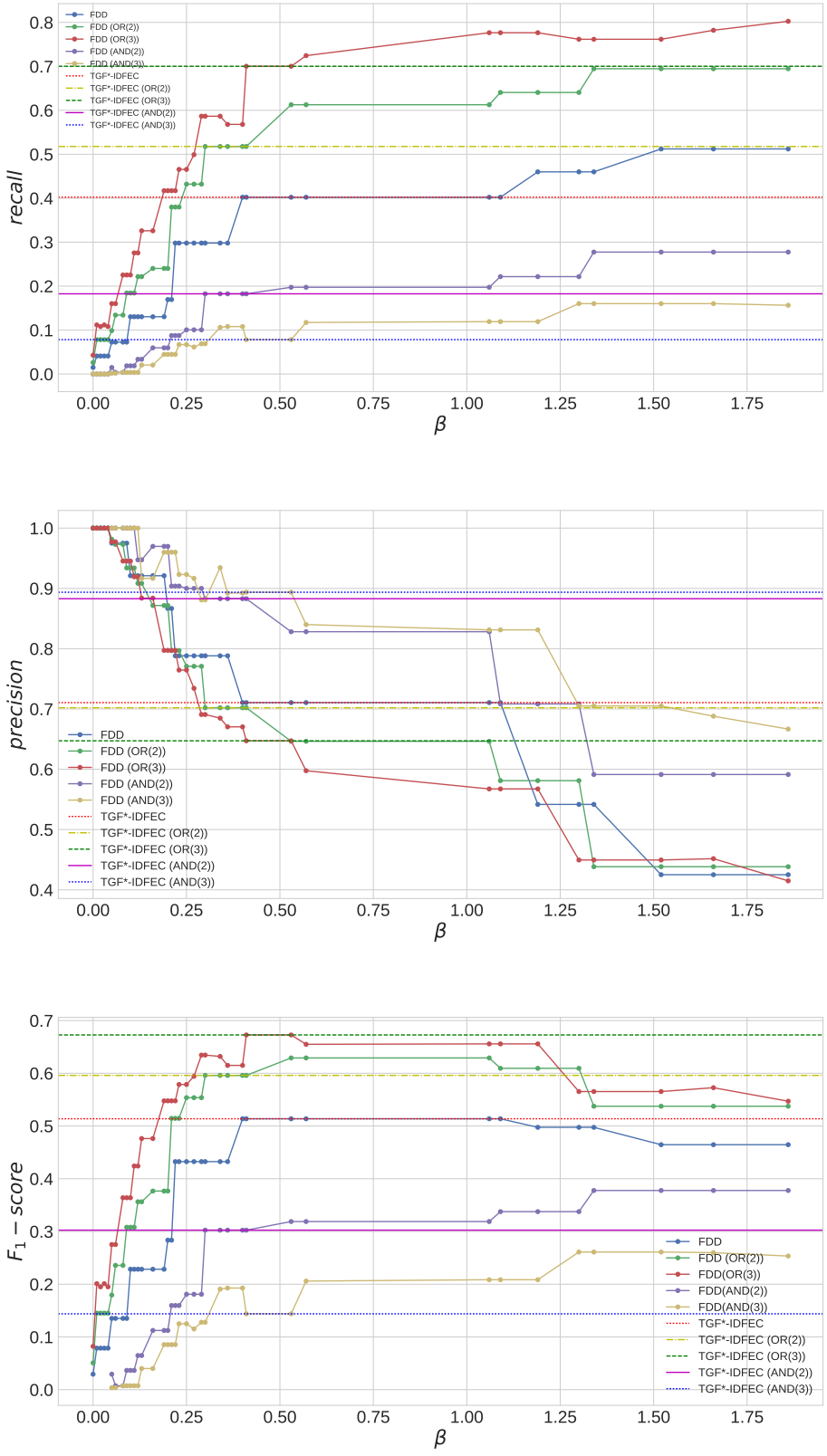


Figure 6: Effectiveness on the training set of one-, two- and three-term conjunctive and disjunctive queries generated based on the FDD_β and TGF*-IDFEC term weighting schemes. The solid curves correspond to the effectiveness of query terms selected using FDD_β with different β values.

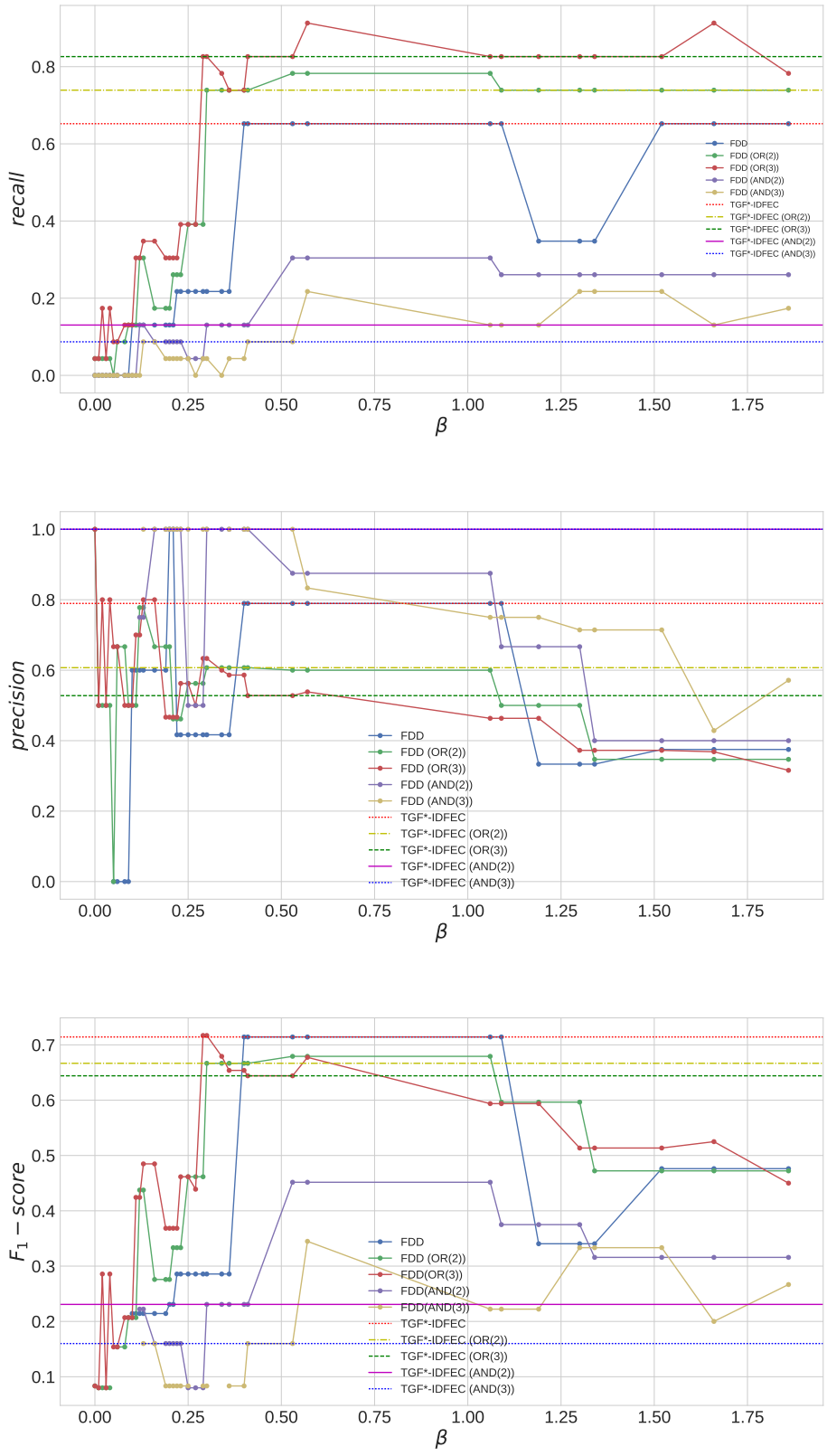


Figure 7: Effectiveness on the validation set of one-, two- and three-term conjunctive and disjunctive queries generated based on the FDD_β and TGF*-IDFEC term weighting schemes. The solid curves correspond to the effectiveness of query terms selected using FDD_β with different β values.

Figure 8: Word cloud based on the top-ranked terms according to the $FDD_{0.477}$ weighting scheme (only words with $FDD_{0.477} > 0.6$ are shown).

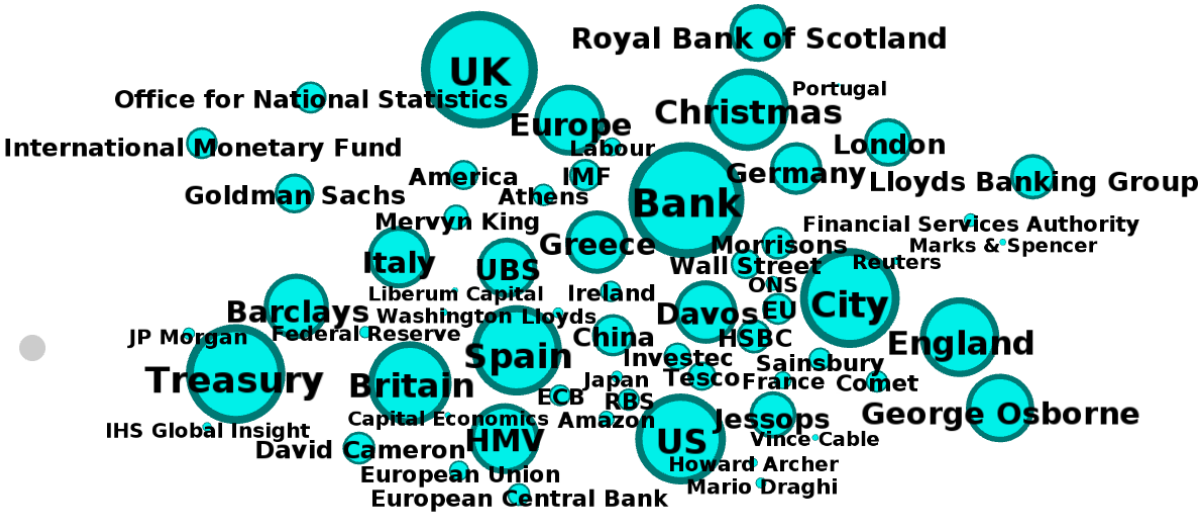


Figure 9: Word cloud based on the top-ranked named entities according to the $FDD_{0.477}$ weighting scheme (only words with $FDD_{0.477} > 0.15$ are shown).

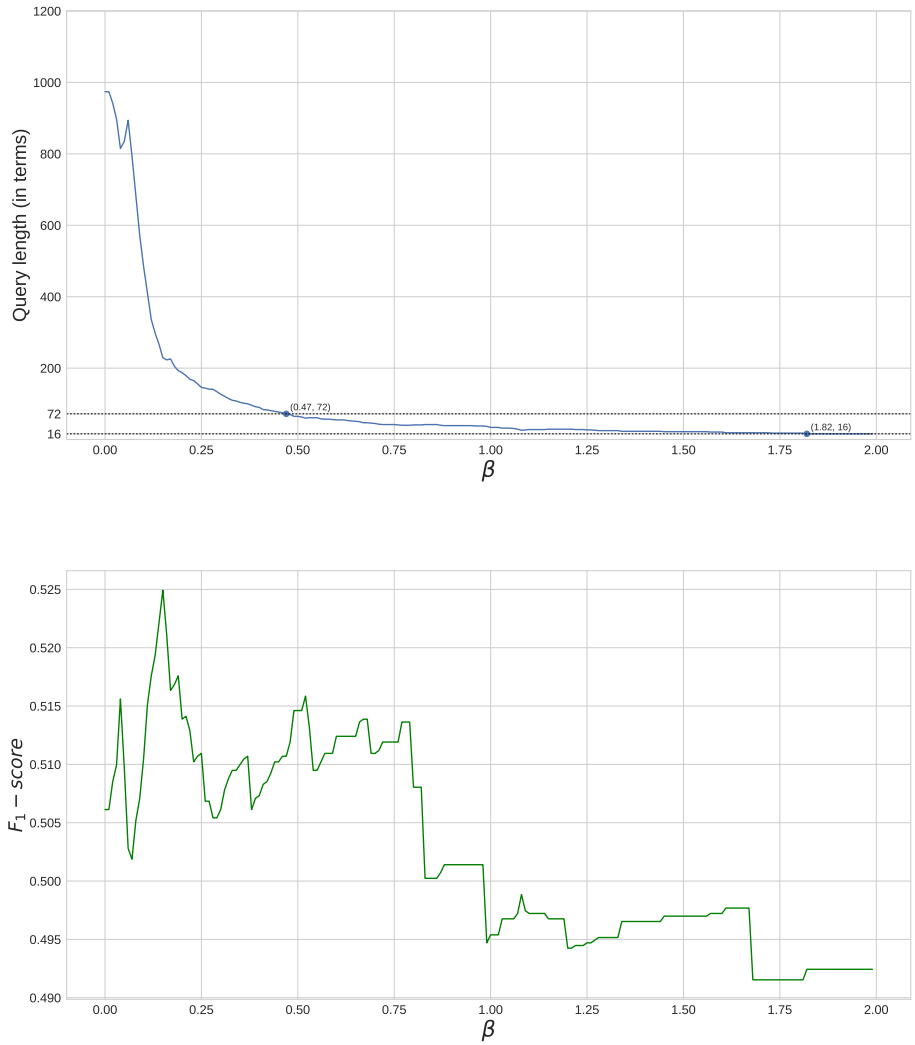


Figure 10: Number of top-ranked terms based on different β values needed in a disjunctive query to achieve total recall (i.e., recall =1) on the training set (top) and F_1 values achieved for these queries on the same set (bottom).

different levels of a concept maps. Since descriptors tend to represent terms describing a general domain or topic while discriminators tend to be specific terms, the use of an adjustable β parameter in the FDD_β measure can help focus the term selection process to favor generality or specificity, depending on the user needs.

5.3 Achieving Total Recall

The problem of total-recall (or high-recall) retrieval [1, 12, 27] is to find all (or nearly all) relevant documents for a search topic. As opposed to those scenarios where the goal is to retrieve a few high quality relevant documents (e.g., answering a specific consultation need with high precision), total recall scenarios are those where the focus is on retrieving all relevant documents, without a significant loss in precision (e.g., collecting all documents relevant to a topic or domain to build a topical or domain-based web portal).

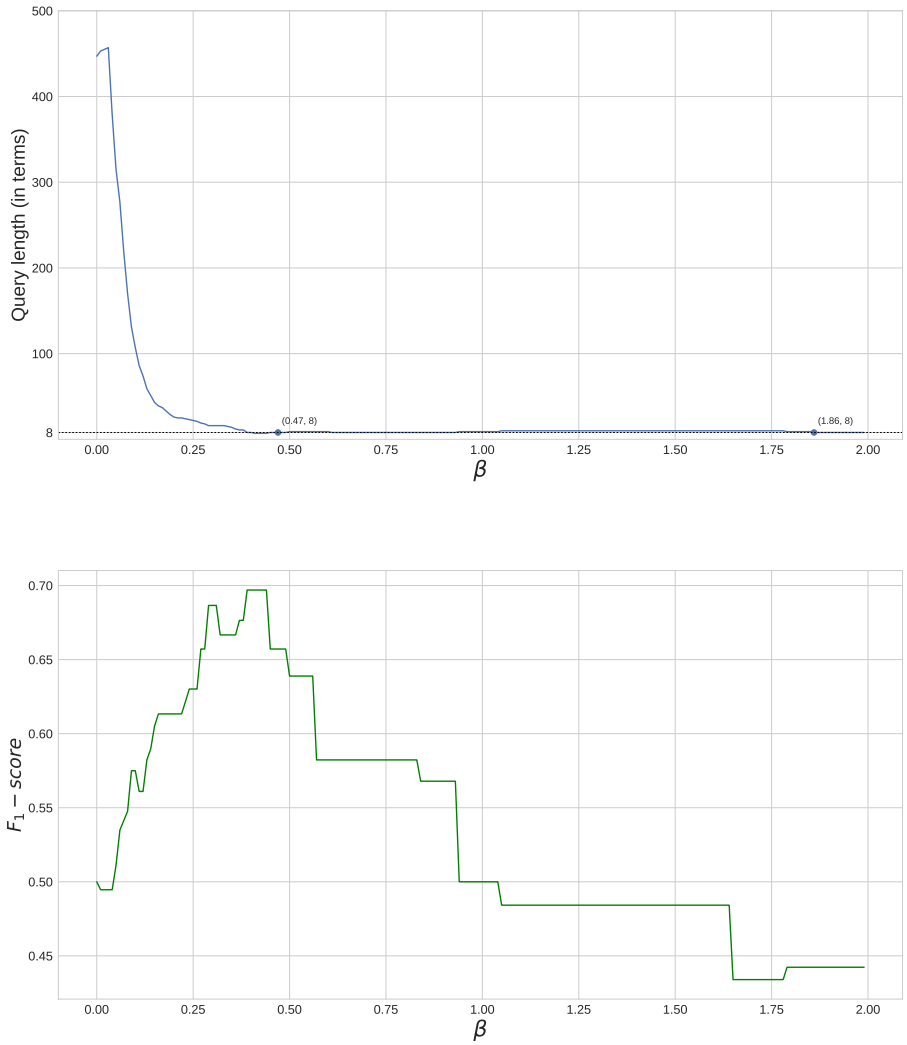


Figure 11: Number of top-ranked terms (learned from the training set) based on different β values needed in a disjunctive query to achieve total recall (i.e., recall =1) on the validation set (top) and F_1 values achieved for these queries on the same set (bottom).

A question that arises when addressing the total recall problem is how many terms are necessary to build a disjunctive query that achieves 100% recall. To look into this question we used our training data collection to incrementally construct disjunctive queries by adding the top-ranked terms based on their FDD_β score for different β values. As discussed earlier, terms with high descriptive power are those terms that tend to occur often in relevant documents. Hence, we expected that when incrementally longer queries are built with a focus on the descriptive power of the query terms (i.e., large β values) it should be possible to construct shorter high-recall queries than when the focus is placed on the discriminative power of the terms (i.e., small β values). This intuition is verified by the results shown in figure 10, where it is possible to see that as β increases, the number of terms needed to achieve total recall reduces significantly.

A similar analysis on the validation set is presented in figure 11. Once again the results indicate that the larger the β value in the selection of top-ranked terms the smaller is the number of terms needed to form a total-recall query. Note that the query length required to achieve total recall is highly dependent on the number of relevant documents in the collection. Hence, total-recall queries for the testing set are significantly shorter than total-recall queries for the training set.

To analyze how total-recall queries impact the F_1 scores we also report F_1 for the evaluated queries both on the training set (bottom of figure 10) and the validation set (bottom of figure 11). For both collections it is possible to see that for certain values of β, total recall is possible without a significant loss in F_1. These results point to the potential of the proposed technique as a mechanism to further investigate the total recall problem.

6 Conclusions and Future Work

In this paper we presented a methodology for identifying domain-specific terms. As part of the proposed methodology we defined a novel supervised term-weighting scheme called FDD_β, which is based on the notions of descriptive and discriminative relevance. Preliminary evaluations show that FDD_β achieves good performance as an estimator of human subjects' relevance judgments and as a mechanism for selecting good query terms. Also, it offers the flexibility of adapting to different goals, such as achieving high recall, high precision, or a balance between both. This flexibility represents an important advantage over the analyzed state-of-the-art weighting schemes. In particular, it offers a novel mechanism for query refinement by guiding the selection of more descriptive query terms to retrieve more general results when initial results are to narrow. Similarly, it can help identifying more discriminative query terms to retrieve more specific results when initial results are too broad.

While the analysis presented here is focused on retrieval, the proposed term-weighting technique may also be applied to classification. This will open new challenges such as analyzing strategies for combining the FDD_β score with local weighting schemes, such as TF.

The proposed technique was evaluated on the economic domain with promising results and we anticipate that it will also achieve good performance on other domains. Also, we plan to test FDD_β on specific topics (as is the case of the topic of a news article), as opposed to general domains (as is the case of Economy). Another important future task will be to validate FDD_β on larger data sets, such as those available as part of the TREC collection (https://trec.nist.gov/data/test_coll.html). Finally, we plan to investigate the definition of non-supervised versions of the proposed weighting techniques, where topic relevance will be approximated by clustering and other non-supervised approaches.

Acknowledgment

This work was supported by CONICET (PIP 11220120100487), MinCyT (PICT 2014-0624) and Universidad Nacional del Sur (PGI-UNS 24/N029).

References

[1] Mustafa Abualsaud, Nimesh Ghelani, Haotian Zhang, Mark D Smucker, Gordon V Cormack, and Maura R Grossman. A system for efficient high-recall retrieval. In *SIGIR*, pages 1317–1320, 2018.

[2] Kewen Chen, Zuping Zhang, Jun Long, and Hao Zhang. Turning from tf-idf to tf-igm for term weighting in text classification. *Expert Systems with Applications*, 66:245–260, 2016.

[3] Franca Debole and Fabrizio Sebastiani. Supervised term weighting for automated text categorization. In *Text mining and its applications*, pages 81–97. Springer, 2004.

[4] C Deisy, M Gowri, S Baskar, SMA Kalaiarasi, and N Ramraj. A novel term weighting scheme midf for text categorization. *Journal of Engineering Science and Technology*, 5(1):94–107, 2010.

[5] Zhi-Hong Deng, Shi-Wei Tang, Dong-Qing Yang, Ming Zhang, Xiao-Bin Wu, and Meng Yang. A linear text classification algorithm based on category relevance factors. In *International Conference on Asian Digital Libraries*, pages 88–98. Springer, 2002.

[6] Zhi-Hong Deng, Kun-Hu Luo, and Hong-Liang Yu. A study of supervised term weighting scheme for sentiment analysis. *Expert Systems with Applications*, 41(7):3506–3513, 2014.

[7] Giacomo Domeniconi, Gianluca Moro, Roberto Pasolini, and Claudio Sartori. A study on term weighting for text categorization: A novel supervised variant of tf. idf. In *DATA*, pages 26–37, 2015.

[8] MA Fattah and MG Sohrab. Combined term weighting scheme using ffnn, ga, mr, sum, and average for text classification. *International Journal of Scientific and Engineering Research*, 7(8):2031–2040, 2016.

[9] Guozhong Feng, Shaoting Li, Tieli Sun, and Bangzuo Zhang. A probabilistic model derived term weighting scheme for text classification. *Pattern Recognition Letters*, 110:23–29, 2018.

[10] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics. doi: 10.3115/1219840.1219885. URL `https://doi.org/10.3115/1219840.1219885`.

[11] Luigi Galavotti, Fabrizio Sebastiani, and Maria Simi. Experiments on the use of feature selection and negative evidence in automated text categorization. In *International Conference on Theory and Practice of Digital Libraries*, pages 59–68. Springer, 2000.

[12] Maura R Grossman, Gordon V Cormack, and Adam Roegiest. Trec 2016 total recall track overview. In *TREC*, 2016.

[13] Samer Hassan, Rada Mihalcea, and Carmen Banea. Random walk term weighting for improved text classification. *International Journal of Semantic Computing*, 1(04):421–439, 2007.

[14] Man Lan, Sam-Yuan Sung, Hwee-Boon Low, and Chew-Lim Tan. A comparative study on term weighting schemes for text categorization. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 1, pages 546–551. IEEE, 2005.

[15] Man Lan, Chew Lim Tan, Jian Su, and Yue Lu. Supervised and traditional term weighting methods for automatic text categorization. *IEEE transactions on pattern analysis and machine intelligence*, 31(4):721–735, 2009.

[16] Christine Largeron, Christophe Moulin, and Mathias Géry. Entropy based feature selection for text categorization. In *Proceedings of the 2011 ACM Symposium on Applied Computing*, pages 924–928. ACM, 2011.

[17] David Leake, Ana Maguitman, and Thomas Reichherzer. Experience-based support for human-centered knowledge modeling. *Knowledge-Based Systems*, 68(0):77 – 87, 2014. ISSN 0950-7051. URL `http://dx.doi.org/10.1016/j.knosys.2014.01.013`.

[18] Edda Leopold and Jörg Kindermann. Text categorization with support vector machines. how to represent texts in input space? *Machine Learning*, 46(1-3):423–444, 2002.

[19] Ying Liu, Han Tong Loh, and Aixin Sun. Imbalanced text classification: A term weighting approach. *Expert systems with Applications*, 36(1):690–701, 2009.

[20] Carlos Lorenzetti, Ana Maguitman, David Leake, Filippo Menczer, and Thomas Reichherzer. Mining for topics to suggest knowledge model extensions. *ACM Transactions on Knowledge Discovery from Data*, 11 (2):23:1–23:30, 2016. ISSN 1556-4681. doi: 10.1145/2997657. URL `http://dl.acm.org/authorize?N37228`.

[21] Carlos M Lorenzetti and Ana G Maguitman. A semi-supervised incremental algorithm to automatically formulate topical queries. *Information Sciences*, 179(12):1881–1892, 2009.

[22] Ana Maguitman, David Leake, Thomas Reichherzer, and Filippo Menczer. Dynamic extraction topic descriptors and discriminators: towards automatic context-based topic search. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 463–472. ACM, 2004.

[23] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, 2004.

[24] Joseph D Novak. Concept mapping: A useful tool for science education. *Journal of research in science teaching*, 27(10):937–949, 1990.

[25] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[26] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier, 2014.

[27] Adam Roegiest, Gordon V Cormack, Maura R Grossman, and Charles Clarke. Trec 2015 total recall track overview. *Proc. TREC-2015*, 2015.

[28] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

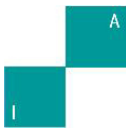
[29] Takenobu Tokunaga and Iwayama Makoto. Text categorization based on weighted inverse document frequency. In *Special Interest Groups and Information Process Society of Japan (SIG-IPSJ*. Citeseer, 1994.

[30] CJ Van Rijsbergen, David J Harper, and Martin F Porter. The selection of good search terms. *Information Processing & Management*, 17(2):77–91, 1981.

[31] Suzan Verberne, Maya Sappelli, Djoerd Hiemstra, and Wessel Kraaij. Evaluation and analysis of term scoring methods for term extraction. *Information Retrieval Journal*, 19(5):510–545, 2016.

[32] D. Wang and H. Zhang. Inverse-category-frequency based supervised term weighting schemes for text categorization. *Journal of Information Science and Engineering*, 29(2):209–225, 2013. URL `https://www.scopus.com/inward/record.uri?eid=2-s2.0-84876262286&partnerID=40&md5=00897fb85a83bfd704cb2428254e1950`.

Inteligencia Artificial, 22(63) (2019), 81-100
doi: 10.4114/intartif.vol22iss63pp81-100



INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

An experimental study on feature engineering and learning approaches for aggression detection in social media

Antonela Tommasel, Juan Manuel Rodriguez, Daniela Godoy
ISISTAN, CONICET-UNICEN, Argentina
antonela.tommasel@isistan.unicen.edu.ar
juanmanuel.rodriguez@isistan.unicen.edu.ar
daniela.godoy@isistan.unicen.edu.ar

Abstract With the widespread of modern technologies and social media networks, a new form of bullying occurring anytime and anywhere has emerged. This new phenomenon, known as cyberaggression or cyberbullying, refers to aggressive and intentional acts aiming at repeatedly causing harm to other person involving rude, insulting, offensive, teasing or demoralising comments through online social media. As these aggressions represent a threatening experience to Internet users, especially kids and teens who are still shaping their identities, social relations and well-being, it is crucial to understand how cyberbullying occurs to prevent it from escalating. Considering the massive information on the Web, the developing of intelligent techniques for automatically detecting harmful content is gaining importance, allowing the monitoring of large-scale social media and the early detection of unwanted and aggressive situations. Even though several approaches have been developed over the last few years based both on traditional and deep learning techniques, several concerns arise over the duplication of research and the difficulty of comparing results. Moreover, there is no agreement regarding neither which type of technique is better suited for the task, nor the type of features in which learning should be based. The goal of this work is to shed some light on the effects of learning paradigms and feature engineering approaches for detecting aggressions in social media texts. In this context, this work provides an evaluation of diverse traditional and deep learning techniques based on diverse sets of features, across multiple social media sites.

Resumen Con la difusión de nuevas tecnologías y los sitios de redes sociales surgió una nueva forma de acoso, que puede ocurrir en cualquier momento y lugar. Este nuevo fenómeno es denominado cyber agresión o acoso cibernético y hace referencia a actos agresivos e intencionales, cuyo objetivo es causar repetidamente daños a otras personas mediante comentarios insultantes, ofensivos, burlones o desmoralizadores a través de las redes sociales. Dado que estas agresiones representan una experiencia amenazadora para los usuarios de Internet, especialmente los niños y adolescentes, es crucial comprender cómo se produce el acoso cibernético para evitar que se intensifique. Teniendo en cuenta la gran cantidad de información que se comparte y distribuye en la Web, en los últimos tiempos ha cobrado importancia el desarrollo de técnicas inteligentes para la detección automática del contenido dañino. Esto potencialmente permite el monitoreo a gran escala de redes sociales, y la detección temprana de situaciones agresivas o no deseadas. A pesar de que en los últimos años se han desarrollado diversos enfoques basados tanto en técnicas tradicionales como en técnicas de aprendizaje profundo, diversas preocupaciones han surgido respecto a la duplicación de investigación y la dificultad para comparar resultados. Asimismo, no existe aún acuerdo respecto a qué tipo de técnica es mejor para la tarea, ni el tipo de características en las que se debe basar el aprendizaje. El objetivo de este trabajo es analizar el efecto de los diferentes paradigmas de aprendizaje y enfoques de ingeniería de características para la detección de agresión en redes sociales. En este contexto, este trabajo proporciona una

ISSN: 1137-3601 (print), 1988-3064 (on-line)

evaluación en múltiples redes sociales de diversas técnicas tradicionales y de aprendizaje profundo, basadas en diversos conjuntos de características.

Keywords: Cyberaggression, Social Media, Aggression detection, Feature Engineering, Machine Learning, Deep Learning
Palabras Clave: Cyber aggression, Medios sociales, Detección de agresión, Ingeniería de Características, Aprendizaje de Máquina, Aprendizaje Profundo

1 Introduction

People have fully embraced the Web and social media sites for socialising and communicating, and interact through different sites, such as *Facebook*, *Twitter*, *Instagram* and *YouTube* at the same time. These sites do not only allow users to create content, publish photos, comment on content other users have shared or tag content, but also foster the social connections between users. Nonetheless, alongside this vast exchange of information, ideas and friendships, undesirable phenomena and behaviours have appeared, this leads to the widespread dissemination of aggressive and potentially harmful content over the web. Even tough most of the time Internet use is safe and enjoyable, there are risks involving the online communications through social media. As the real world could be a dangerous place, social media sites are not the exception. Users might have to deal with threatening situations like cyberaggression, cyberbullying, suicidal behaviour or grooming [Whittaker and Kowalski, 2015].

Cyberbulling and cyberaggression are serious issues increasingly affecting Internet users. With the "help" of the widespread of social media networks, bullying once limited to particular places or times of the day (e.g. schools), can now occur anytime and anywhere [Chatzakou et al., 2017] and have a wider range of audience. Cyberaggression can be defined as aggressive online behaviour that intends to cause harm to another person [Hosseinmardi et al., 2015], involving rude, insulting, offensive, teasing or demoralising comments through online social media that target educational qualifications, gender, family or personal habits [Chavan and S, 2015]. Cyberbullying is one of the many forms of cyberaggression and is characterised by an act of online aggression, the existence of a power imbalance between the individuals involved (including diverse forms, such as physical, social, relational or psychological), and repetitions across time [Hosseinmardi et al., 2015]. This problem is aggravated by the persistence and durability of online materials, which gives these incidents an unprecedented power and influence to affect the lives of billions of people.

Links were found between experiences of cyberbullying and negative outcomes, such as decreased performance at school, dropping out and violent behaviour, in combination with devastating mental and psychological effects such as depression, low self-esteem, and even suicide [Hosseinmardi et al., 2015]. In recent years, there have been several high-profile cases involving teenagers taking their own lives in part for being harassed and mistreated over the internet. Additionally, cyberaggressive comments make their targets feel demoralised and frustrated, thus acting as a barrier for participation and socialisation. While these incidents are still isolated and do not represent the norm, their gravity demands deeper understanding [Hinduja and Patchin, 2010].

Considering the severity of the consequences that cyberaggression has on its victims and its rapid spread amongst internet users (specially kids and teens), there is an imperious need for research aiming at understanding how cyberbullying occurs, in order to prevent it or at least to decrease the harassing and bullying incidents in the cyberspace. Moreover, cyberaggression detection can be used to provide better support and advice for the victims as well as monitoring and tracking the bullies [Dadvar and de Jong, 2012]. Even though one incident cannot be a certain indication that the involved users are victims or bullies, following their behaviours after the incidents across different social networks during a period can provide a more accurate description of their profiles. Other important application of the detection of cyberaggression or aggressive content is the detection of cyberextremism, cybercrime and cyberhate propaganda [Agarwal and Sureka, 2015]. Most networking sites today prohibit the usage of offensive and insulting comments [Van Hee et al., 2015], which is partially being carried out and filtered to a limited extent. On the strategies to cope with aggressive behaviour online is to manually monitor and moderate the user-generated content. However, given the massive information overload on the Web and the pace at which new posts are being shared, it is unfeasible for human moderators to manually track and flag each insulting and offensive comments [Chavan and S, 2015]. Thereby, it is crucial to develop

intelligent techniques to automatically detect harmful content, which would allow the large-scale social media monitoring and early detection of undesired situations.

Despite the seriousness of the problem, there are few successful efforts to detect abusive behaviour on social media data. This is related to the existence of several challenges [Chatzakou et al., 2017, Nobata et al., 2016], not only related to the nature of posts and the environment in which the aggression occurs, but also to technical limitations of the generalisation and comparability of the proposed approaches. One concern that arises from the approaches in the literature is related to the duplication of research and the difficulty of comparing results. Most works are evaluated over different datasets, which hinders the generalisation of their results. To advance towards solving this complex phenomenon, it is crucial to reach an agreed understanding of the different aspects of the problem, and the creation of standardised datasets [Kumar et al., 2018a], that would allow the comparison of approaches. In this context, this paper builds on a previous work [Tommasel et al., 2018] and focuses on the detection of aggressive content in the context of multiple and heterogeneous social media sites. In this context, the performance of several feature sets in the context of both traditional and deep learning techniques is evaluated in four publicly available datasets. Additionally, it is explored the feasibility of the selected feature sets and techniques for the identification of different types of accounts dedicated to distributing aggressive content. The goal is to both provide a comparison between different feature sets and techniques proposed in the literature over the same datasets to analyse the generalisation of results, and to shed some light on the usefulness or adequacy of the different techniques for the task, and the generalisation of models trained for a specific social media site to other sites.

The remainder of this paper is organised as follows. Section 2 describes recent related work regarding the detection of aggressive content. Section 3 describes the features considered in the analysis. Section 4 describes the experimental settings, including the selected datasets and implementation details. Section 5 analyses the obtained results for each of the selected datasets and their combinations. Finally, Section 6 presents the conclusions drawn from the study, and outlines future lines of research.

2 Related Work

Cyberaggression detection has captured the interest of researchers in the last years due to its proliferation across social media and its harmful effect on people. Consequently, automatic approaches for cyberbullying detection, mostly based on machine learning and natural language processing techniques, are being extensively developed. The detection of cyberbullying and online harassment is often formulated as a classification problem [Salawu et al., 2017], in which techniques traditionally used for document classification, topic detection, and sentiment analysis are used to detect electronic bullying based on the characteristics of messages, senders, and recipients. Thereby, feature engineering (i.e., the process of analysing and designing predictive features) becomes an important step in this process, as it allows to enhance the performance of techniques. Features can be broadly categorised into four groups [Salawu et al., 2017], namely content-based (e.g. bag-of-words, n-grams and part-of-speech tagged words), sentiment-based (e.g. positive and negative emotions), user-based (e.g. demographic information) and network-based features (e.g. number of friends and activity). Several sets of these four types of features have been used in the literature with both classical machine learning methods and methods in the deep learning paradigm. The former methods, such as Support Vector Machines (SVM), Naïve Bayes or Logistic Regression, rely on manually engineered features that are then used for training the models, whereas the latter methods employ neural networks to automatically learn features from raw data.

Most efforts related to cyberaggression detection focus on the detection of the actual aggressive events and their classification. In this regard, Van Hee et al. [2015] explored the identification and fine-grained classification of events into seven categories (non-aggressive, threat/blackmail, insult, curse, defamation, sexual talk, defence and encouragement to the harasser) based on two types of lexical features. First, bag-of-words features including unigrams, bigrams and character trigrams. Second, polarity features including the number of positive, negative and neutral lexicon words averaged over text length and the overall post polarity. In total, the authors created more than $300k$ features. Experimental evaluation was based on approximately $80k$ Dutch posts belonging to *Ask.fm*, which were manually labelled with a Kappa score of 0.69. Results achieved an average F-Measure of 0.55 and a minimum of 0.07 for the defamation class. The authors hypothesised that the discrepancy of results arose from the differences in

post lexicality. For example, insults are generally highly lexicalised, whereas threats are often expressed in an implicit way.

Nobata et al. [2016] focused on the detection of hate speech on 2 million online comments from two domains (*Yahoo!* Finance and News) based on four types of features: n-grams, linguistic, syntactic and distributional semantics. Linguistic features explicitly look for inflammatory words and non-abusive language elements, such as the usage of politeness words or modal verbs. Distributional semantic features refer to features derived from word embeddings. The online comments were pre-processed by normalising numbers, replacing unknown words with the same token and replacing repeated punctuation. The best F-Measure results were obtained when combining all features. Interestingly, the individual sets of features obtaining the best results varied according to the dataset domain. For the finance dataset, the best individual results were obtained by n-grams, whilst for the news dataset, the best results were obtained by the embedded features. The authors hypothesised that the selected features could achieve good performance in other languages, although it remains to be evaluated.

Similarly to [Nobata et al., 2016], Chavan and S [2015] distinguished bullying and non-bullying comments by means of TF-IDF weighted n-grams, the presence of pronouns and skip-grams. Feature selection was then apply to select only the $3,000$ highest features according to χ^2. Experimental evaluation was based on approximately $6.5k$ comments from an unspecified site. Pre-processing was applied by removing non-word characters, hyphens and punctuation and applying a spell-checker. The best classification performance was achieved by selecting pronouns and skip-grams, with differences up to a 4.8% regarding the other features.

All previously described approaches are based on training traditional supervised models such as SVM, Naïve Bayes and Logistic Regression. Nonetheless, other approaches based on lexicons [Pérez et al., 2012, Bretschneider et al., 2014] and rules [Chen et al., 2012, Bretschneider et al., 2014] have been also proposed. For example, Pérez et al. [2012], Bretschneider et al. [2014] proposed using established patterns of aggression to detect bullying on lexically processed text. Particularly, Bretschneider et al. [2014] formulated rules to recognise word patterns indicating relationships between profane words and personal pronouns. Experimental evaluation was based on publicly available *Facebook* posts[1]. Chen et al. [2012] computed the offensiveness score of *YouTube* comments posted by over 2 million users based on a set of rules and syntactic features mined using the Stanford parser[2]. According to the authors, their rule-based approach was able to improve the results obtained with SVM and Naïve Bayes classifiers. Additionally, Fahrnberger et al. [2014] aimed at not only detecting the aggressive content in chat rooms, but also to replace it with alternatives suitable for minors extracted from WordNet[3].

The cyberbullying detection task does not only focus on the detection of the actual aggressive content, but also on the detection of users sharing it. In this context, Chatzakou et al. [2017] identified aggressive users by combining user, text, and network-based features. Experimental evaluation was based on the *#GamerGate* controversy in *Twitter*, including approximately $650k$ tweets, which were manually labelled into three categories (aggressor, bully and spammer). Tweets were pre-processed by removing numbers, stopwords, punctuations and converting the remaining words to lower case. According to the authors, their approach achieved an overall precision of 0.89, and precisions of 0.295 and 0.411 for the aggressive and bully classes, exposing the difficulties of the task. Results showed that features did not have the same relevance for the task. For example, considering session statistics, average emotional scores, hate score, average word embedding and community information added noise to the classification, whilst most text features did not contribute to the improvement of results. Conversely, the most effective features were the network-based ones. Moreover, the authors found that no statistical difference was found for concrete sentiment features (e.g. anger, disgust, fear, joy) amongst the abusive and normal posts.

Recently, the importance of detecting aggressive content motivated the development of competitions and shared tasks, like TRAC 2018 Shared Task on Aggression Identification[4] [Kumar et al., 2018a] and the MEX-A3T track at IberEval 2018[5] [Alvarez-Carmona et al., 2018]. The former aimed at discriminating between overtly aggressive, covertly aggressive (which resulted the most difficult class to predict), and non-aggressive texts. To that end, participants were provided with a training dataset of $15k$ aggression-

[1] http://www.ub-web.de/research/index.html
[2] https://nlp.stanford.edu/software/lex-parser.shtml
[3] https://wordnet.princeton.edu/
[4] https://sites.google.com/view/trac1/shared-task
[5] https://sites.google.com/view/ibereval-2018

annotated *Facebook* Posts and Comments in both Hindi and English, and two test sets. One of the text sets was extracted purely from *Facebook* and the other mixed posts from *Facebook* and other social media site. In turn, the MEX-A3T track was oriented to discriminate between aggressive and non-aggressive tweets in Mexican Spanish language. In this case, participants were provided with an annotated corpus of more than $11k$ tweets. Similar alternatives were proposed for both tasks, ranging from using traditional features for training SVM and random forest classifiers, to using more sophisticated deep neural network techniques.

For example, Samghabadi et al. [2018] combined lexical and semantic features. Lexical features included TF-IDF weighted word n-grams, char n-grams, and k-skip n-grams, whilst semantic features were extracted using vectors trained using Google News. Additional feature vectors included sentiment features, LIWC [Pennebaker et al., 2001] features, and the gender probabilities of each message. Ramiandrisoa and Mothe [2018] combined random forest and logistic regression classifiers. The former classifier was based on different set of features adapted from depression detection tasks and included common features like punctuation or emotions and others tending to analyse the readability and comprehensibility of texts. On the other hand, the latter classifier was based on document vectorization with doc2vec [Le and Mikolov, 2014].

As regards the deep learning based techniques, Aroyehun and Gelbukh [2018] leveraged on pre-trained word embeddings including word2vec[Mikolov et al., 2013], GloVe [Pennington et al., 2014], SSWE and fastText [Joulin et al., 2017], which exhibited the highest vocabulary coverage. Galery and Charitos [2018] combined pre-trained English and Hindi fastText word embeddings by means of pre-computed SVD matrices to join the representations from both languages into a single space. Roy et al. [2018] proposed an ensemble of classifiers that included a SVM classifier trained with TF-IDF vectors of unigrams, and a Convolution Neural Network (CNN) with pre-trained GloVe vectors. Risch and Krestel [2018], Frenda and Banerjee [2018], Gómez-Adorno et al. [2018] explored the usage of traditional features (e.g. unigrams, bigrams, and character n-grams, POS tags, named entities and sentiment polarity) for training deep learning models. Particularly, Risch and Krestel [2018] focused on logistic regression and bidirectional short-term memory (LSTM) networks learning. Frenda and Banerjee [2018] also included features related to style and writing density as well as the extraction of syntactic patterns, lists of aggressive words and affective features, and Gómez-Adorno et al. [2018] included morphological features.

The reviewed approaches in most cases show several of the challenges that have yet to be tacked by the selected features and techniques. For example, the lack of grammar and syntactic structure of social media posts, which hinders the usage of natural language processing tools. For example, the intentional obfuscation of words and phrases to evade checks. On the other hand, abusive content might span over multiple sentences. Second, the limited context provided by each individual post, causing that an individual post might be deemed as normal text, whilst it might be aggressive in the context of a series of posts. Third, the fact that aggression could occur in multiple forms, besides the obvious abusive language. For example, the usage of irony and sarcasm. Fourth, the difficulty of tracking racial and minority insults, which might be unacceptable to one group, but acceptable to another. Moreover, the difficulty for detecting aggression might depend on the language in which the aggression is made.

In summary, the main concern of the summarized works was to improve classification results through the application of both classical machine learning algorithms as well as deep learning approaches. However, a conclusion regarding which approach is more adequate to the task at hand has not yet been reached, as the performance of the different approaches did not seem to differ much. For example, if features are carefully selected, then classifiers like SVM and even random forest and logistic regression perform at par with deep neural networks. Similarly, if deep learning approaches are not carefully designed they might perform poorly.

From the reviewed approaches, concerns over the duplication of research and the difficulty of comparing results arise. Most works are evaluated over different datasets, which hinders the generalisation of results. To advance towards solving this complex phenomenon, it is crucial to reach an agreed understanding of the different aspects of the problem and the creation of standardised datasets [Kumar et al., 2018a], that would allow the comparison of approaches. In this context, this paper builds on a previous article [Tommasel et al., 2018] to provide a more extensive evaluation of both traditional and deep learning techniques over not only four publicly available datasets, but also on their cross-combinations.

3 Characterising Aggression

Many studies about aggression or bullying detection have assessed the capabilities of content, sentiment, user, network-based features or a combination of them. However, there is still no consensus regarding which features perform better for characterising and detecting acts of aggression. This situation is evidenced by the variability of results and the diversity of social media sites, which have their own intrinsic characteristics. The main goal of this work is to study different feature sets and their performance for detecting aggression in different social media sites. In particular, this work focuses on four type of features: character-based, word-based, sentiment-features, and irony-features. Additionally, this work also aims at shedding some light on the generalisation of features and models trained for a specific social media site to other sites, i.e. for cross social media cyberaggression detection.

Character-, syntactic- and word-based features have been traditionally used in most text related tasks. Character-based features usually include the number and ratio of punctuation marks (e.g. question marks, exclamation marks, period, commas and ellipses), the number and ratio of upper case letters, and the number and ratio of emoticons/emojis. Syntactic features are associated to the part-of-the-speech (POS) tagging of text, and usually include the number and ratio of nouns, verbs, adverbs and adjectives, or the selection of only a particular type of word. For example, only words tagged as nouns, verbs, adjectives and adverbs could be selected. Word-based features can include stemming, lemmatisation, name entity recognition, average word length, number of synonyms base on *WordNet*[6], commonness of words based on the American National Corpus[7] and frequency of rarest word. Word Embeddings can also be considered be used to define features.

Since cyberbullying has been associated to negative emotions, such as anger, irritation, disgust and depression, sentiment-based features might be useful for characterising aggression and cyberbullying. Sentiment detection could refer either to identifying the overall polarity of texts, i.e. whether the text yields a positive, negative or neutral polarity, or to identifying specific emotions, such as anger, joy, love or hate, amongst others. For the purpose of the aggression detection task, social media post were characterised according to their overall polarity. Polarity is associated to the diverse syntactic structures of posts, the polarity of their individual words, the number and ratio of curse words. Two pre-trained sentiment models are considered: *StandordNLP* and *SentiWordNet*[8]. In addition to words, emoticons and emojis are an integral part of social media language and they are considered to deliver sentiment information. In this context, the analysis includes the average sentiment polarity of emojis in posts based on the Emoji Sentiment Ranking [Kralj Novak et al., 2015][9].

Finally, irony based features might be helpful for detecting cyberbullying and cyberaggression. As irony is meant to communicate the opposite of the literal interpretation of the expressions, it might be used to masked aggression. For example, a congratulation might be actually a mocking about a bad outcome. Ironic statements can elicit affective reactions [Hernańdez Farías et al., 2016] For example, ironic criticism has been recognised as offensive and associated with particular negative affective states, which could enhance negative emotions such as anger, irritation or disgust. In this context, the feature sets defined in [Barbieri and Saggion, 2014, Hernańdez Farías et al., 2016] are also considered. Such feature sets focus on both character- and word-based features, and emotive word lists and lexicons (e.g. AFFIN[10], the lexicon created by [Hu and Liu, 2004] and the Whissell's Dictionary of Affect in Language[11]).

4 Experimental Settings

This section describes the experimental settings considered for evaluating the capabilities of of the selected features for cyberaggression detection in social media, and is organised as follows. Section 4.1 outlines the data collections used. Then, Section 4.2 describes the process for extracting the features and creating the posts representations. Finally, Section 4.3 describes implementation details.

[6] https://wordnet.princeton.edu/
[7] http://www.anc.org/
[8] http://sentiwordnet.isti.cnr.it/
[9] http://kt.ijs.si/data/Emoji_sentiment_ranking/
[10] http://neuro.imm.dtu.dk/wiki/AFINN
[11] https://www.god-helmet.com/wp/whissel-dictionary-of-affect/index.htm

4.1 Data Collections Used

The performance of the aggression detection was evaluated considering four data collections gathered from diverse social media sites. Table 1 summarises the general characteristics of the selected collections. Unless data collections were already separated into training and test set, they were randomly split 70% training and 30% test sets.

Kumar et al. [Kumar et al., 2018b] It was made public as part of the challenge of the TRAC 2018 Shared Task on Aggression Identification[12] and comprises more than $17k$ posts extracted from *Twitter* and *Facebook*. Posts were collected from Hindi pages related to news, forums, political parties, student's organisations, and groups in support or in opposition to recent incidents. Most of the posts are in English, some contains Hindi word or expressions, and a minority are completely in Hindi. Human annotators assigned posts to either one of three classes, overtly aggressive, covertly aggressive and non aggressive. According to the authors, the best classification achieved a F-Measure of 0.7. However, the authors did not specify which features were used. The collection is divided into four different sets of posts. The first two only includ $15k$ *Facebook* posts, and are intended as training and validation datasets. The other two collections were intended as testing datasets and comprise posts from different social media sites. The first one includes 916 *Facebook* posts, whilst the other $1,257$ *Twitter* posts.

Davidson et al. [Davidson et al., 2017] It comprises approximately $25k$ tweets containing terms compiled by *Hatebase*[13]. Tweets were assigned to one of three classes (hate speech, offensive but not hate speech and neither hate speech nor offensive) by human annotators. The agreement of the labelling was 92%. Interestingly, only 5% of the tweets were coded as hate speech by the majority of coders, showing the limitations of the *Hatebase* lexicon. According to the authors, the best classification achieved an F-Measure of 0.9, when considering n-grams and POS information.

Reynolds et al. [Reynolds et al., 2011] It comprises approximately $3k$ questions and answers extracted from the ask and answer site *FormSpring.me*[14]. Posts are manually labelled into three categories (strongly aggressive, weakly aggressive and non-aggressive). Each post was labelled by three different taggers to improve the labelling quality. According to the authors, the best classification achieved an overall accuracy of 81%, when considering features related to the number and intesity of curse words.

Chatzakou et al. [Chatzakou et al., 2017] it comprises tweets gathered between June and August 2016, in relation to the *GamerGate* controversy, which is one of the most well documented and mature, large-scale instances of aggressive behaviour. It focuses on the classification of users instead of posts. Collection started with the "#GamerGate" hashtag and continued with co-ocurring hashtags. Additionally, a random set of tweets was crawled, as it was assumed less likely to contain offensive behaviour. Unlike the other selected collections, this one focused on classifying users according to their behaviour instead of classifying each independent post.

4.2 Feature Extraction and Post Representation

For the purpose of feature extraction, posts were first pre-processed by removing all non-standard characters, such as non-printable and control characters. Syntactic-based features, such as character- or word-based features, required text tokenisation, which was carried out using two tools, one specifically designed for social media (*twokeniser*[15]), and one of general purpose (*StanfordNLP* library[16]) English stopwords were also removed. The set of extracted features and their combinations in presented in Table 2.

Once the tokens were obtained, post representations were built according to the defined feature sets. Two strategies were followed for describing posts. The first strategy mapped posts to feature vectors. This strategy represents posts considering all character-, syntactic- and sentiment-based features, and most of the word-based features. In this case, each feature represented a dimension of the vector. In case features represented actual words in posts, they were weighted by means of TF-IDF. This kind of representation considers global characteristics of the post, such as the terms in each post and their

[12] First Workshop on Trolling, Aggression and Cyberbullying: `https://sites.google.com/view/trac1/home`
[13] `https://www.hatebase.org/`
[14] `https://spring.me/`
[15] `http://www.cs.cmu.edu/~ark/TweetNLP/`
[16] `https://stanfordnlp.github.io/CoreNLP/`

	Kumar et al.	*Davidson et al.*	*Reynolds et al.*	*Chatzakou et al.*
# of classes	non aggressive, overtly aggressive, covertly aggressive	hate, offensive, neither	strongly, weakly, non aggressive	normal, aggressor, bully, spammer
# of posts	14,984: 6283, 3417, 5284	24,783: 1430, 19190, 4163	12,773: 799, 1224, 10,750	4954: 3562, 59, 24, 1300
average number of words per post	27: 23.83, 32.26, 27.40	16.84: 16.06 16.76 17.49	33.20: 34.29, 32.10, 33.25	17.55: 17.87, 16.42, 16.125, 15.95
average number of nouns per class	4.19, 5.57, 7.75	3.92, 3.90, 3.97	7.95, 7.11, 6.51	5.07, 4.33, 4.58, 3.89
average number of verbs per class	1.15, 1.46, 1.41	0.73, 0.76, 0.62	1.66, 1.44, 1.54	0.45, 0.37, 0.41, 0.30
average number of adverbs per class	1.06, 1.57, 1.46	0.62, 0.72, 0.69	1.47, 1.34, 1.57	0.31, 0.28, 0.25, 0.20
average number of adjectives per class	1.53, 2.24, 1.77	1.03, 0.82, 0.96	1.65, 1.36, 1.42	0.65, 0.64, 0.75, 0.48
average number of punctuation per class	1.29, 1.77, 1.51	0.72, 0.61, 0.84	1.95, 1.80, 1.97	0.56, 0.57, 0.66, 0.39
average number of emoticons-emojis per class	0.05, 0.01, 0.02	0.01, 0.01, 0.02	0.59, 0.73, 0.72	0.07, 0.08, 0.12, 0.02

Table 1: Data Collection Characteristics

TF-IDF	Tokenisation, stopword removal and TF-IDF weighting.	*Stanford Sentiment*	Overall sentiment of the post and sentiment of each detected syntactic structure.
Char	The defined char-based features.	*word2vec*	Matrix representation based on word2vec.
Lemma	Only the lemma of the tokenised terms are kept.	*GloVe*	Matrix representation based on GloVe.
NER	Only the recognised types of entities are kept.	*Barbieri*	Irony detection features based on Barbieri and Saggion [2014].
POS-NVAA	Only noun, verbs, adjectives and adverbs are kept.	*Hernandez*	Irony detection features based on Hernańdez Farías et al. [2016].
POS Tags	Instead of considering the actual terms, it considers their POS tags.	*TF-IDF + SentiWordNet*	TF-IDF + sentiment polarity of the post extracted with SentiWordnet.
POS-NVAA + POS-Frequencies	POS-NVAA + frequency of the different POS tags.		

(a) Simple Feature Sets

TF-IDF + SentiWordNet + Emoji	*TF-IDF + Hernandez*	*TF-IDF + Stemmer + Barbieri*
TF-IDF + Stemmer + Hernandez	*TF-IDF + Barbieri*	word2vec + GloVe
TF-IDF + Stemmer + Char	*TF-IDF + POS Tags*	*TF-IDF + Stemmer + POS Tags*
TF-IDF + Stemmer	*TF-IDF + Char*	*TF-IDF + Char + POS Tags*

(b) Combined Feature Sets

Table 2: Summary of the Evaluated Feature Sets

frequency, but losses local information, such as word order. This type of representation is suitable for any traditional classification technique.

The second strategy involves representing posts in the form of matrices as a sequence of vectors each representing a term according to the selected word embedding model. This kind of representation preserves local information, such as which adjective is modifying which noun. In this case, posts were represented considering the average number of words per post in the collection. For example, for *Kumar et al.*, each post was represented by their last 23 words. Then, each word was replaced by its corresponding 300-dimension vector (as suggested in [Mikolov et al., 2013]), resulting in a matrix representation of posts of dimensionality 23×300. The word-vector mapping was performed using a pre-trained word embedding model. Particularly, two commonly used models are considered: word2vec [Mikolov et al., 2013] (trained with Google News data) and GloVe[17] (trained with *Twitter* data). Additionally, the matrix representation also considered the sentiment of words. Each word was associated to the corresponding *WordNet* synset. For each sense associated to the synset, it was retrieved its negative, positive and neutral polarity. Finally, each word was represented by its positive, negative and neutral average polarity and standard deviation.

4.3 Implementation Details

Two experimental methodologies were followed, which were closely related to the feature extraction strategy used. For the vector representation of posts, evaluation was performed using different configurations of three traditional classification algorithms. First, two variations of the SVM, one with a poly kernel and the other with a RBF kernel, both setting $\gamma = 0.1$. Second, Random Forest using 10 and 20 estimators, and third Naïve Bayes. In each case, the implementation provided by Sklearn[18] was used. Finally, the performance of multi-layer perceptrons was also evaluated. The number of hidden layers ranged between 0 and 2 hidden layers. Note that having 0 hidden layers corresponds to performing a logistic regression. Training was performed by means of rmsprop, and loss was analysed in terms of the categorical cross entropy. Hidden layers were activated with the RELU function and had $features/(\#layer + 1)$ neurons. Three normalisation alternatives were applied: no normalisation, feature scaling (minimum and maximum values were computed from the training set) and standardisation.

On the other hand, for the matrix representation, classification was based on recurrent neural networks. Two neural network architectures were evaluated. First, a stacked LSTM network including: a dropout layer with a probability of 0.5, two LSTM layers with 150 and 50 neurons, a RELU layer with $10 \times \#classes$ neurons and finally a softmax activated layer. Second, a hybrid architecture that concatenated the results obtained for word2vec, GloVe and *SentiWordnet* in combination with the first architecture. After concatenation, four layers were added: a RELU layer with $10 \times \#classes$ neurons, dropout with a probability of 0.5, another RELU layer with $10 \times \#classes$ neurons, and finally a softmax layer with $\#classes$ neurons. Neural networks were implemented with Keras[19], using a Theano[20] backend. In all cases, performance was assessed considering the traditional precision and recall metrics, summarised by means of F-Measure.

5 Experimental Results

This section presents the empirical evaluation results of the capabilities of the selected feature sets and their combinations for the task of cyberbullying and cyberaggression detection. This section is organised as follows. Section 5.1 discusses the results obtained for each individual data collection. This means that each evaluation pertains to a single social media site, i.e. to one particular data collection. Then, Section 5.2 presents a cross-social site evaluation. The main goal of this section is to evaluate whether a model trained for a particular social media site is suitable for detecting aggression in another one.

For both empirical evaluations, the statistical significance of performance differences was assessed based on the Wilcoxon test [Corder and Foreman, 2009] for related samples. The test was performed over

[17] https://nlp.stanford.edu/projects/glove/
[18] http://scikit-learn.org/
[19] https://keras.io/
[20] http://deeplearning.net/software/theano/

the results observed for the different feature sets, where samples corresponded to the results obtained for each classification alternative. Two hypothesis were defined. The null hypothesis stated that no difference existed amongst the results of the different samples, i.e. every evaluated feature set performed similarly across the different classification techniques. On the contrary, the alternative hypothesis stated that the differences amongst the results obtained for each feature set were significant and non-incidental.

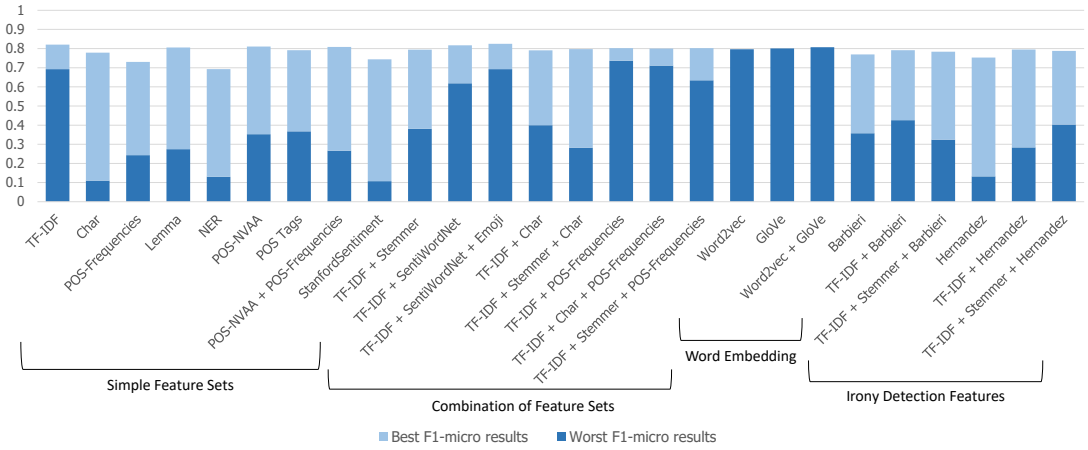
5.1 Single Social Media Site Evaluation

Empirical evaluation consisted of training the classifiers described in Section 4.3 with the selected feature sets described in Section 4.2. The evaluation was independently performed per each of the collections described in Section 4.1. For *Kumar et al.*, the training was performed using the train collection, while testing was performed using the validation dataset. It is worth noting that evaluations considering feature selection techniques were also performed. Particularly, Information Gain was used for retaining the 75% of the most important features. Nonetheless, although feature selection allowed improving results up to a 2%, such differences were statistically insignificant. Hence, those results are not reported.

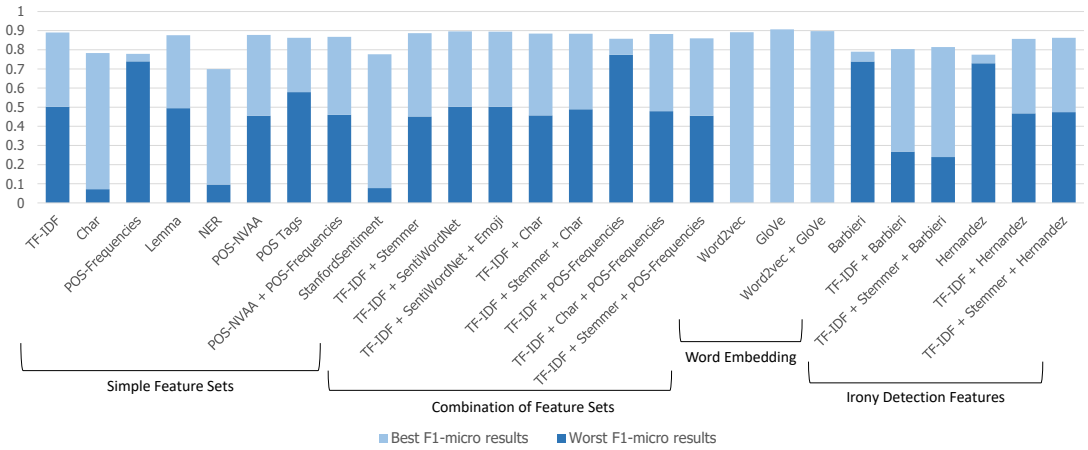
Figure 1 presents the results obtained for each data collection. Each stacked bar reports the worst and best results obtained for the corresponding feature set. In most cases, the worst results were obtained for Naïve Bayes, followed by SVM with a polynomial kernel, regardless of the data collection under analysis. On the other hand, the best results were mostly obtained with either SVM with a RBF kernel or a neural network with 0 hidden layers. An exception was observed for *Chatzakou et al*, for which the Random Forest technique outperformed the others. This might be related with the fact that, during data collection, authors filtered tweets based on a specific hashtag known to be related to an aggressive and violent controversy. As a result, there was a high predominance of hashtags in the text (more than the 6% of terms were actually hashtags), which could have worked as labels [Huang et al., 2018], despite not being completely accurate. As a result, they could have introduced bias to the classification techniques, especially to decision trees. Interestingly, even though neural networks with hidden layers were the most computationally complex techniques, they did not achieve the best results, probably due to overfitting.

As it can be observed, the results for *Kumar et al.* are lower than those observed for the other collections, regardless of the evaluated feature set. Moreover, in many cases, the worst results observed for *Chatzakou et al., Davidson et al.,* and *Reynolds et al.* are higher than the best results observed for *Kumar et al.* It is worth noting that the results obtained for *Reynolds et al.* and *Chatzakou et al.* are higher than those originally reported in [Reynolds et al., 2011, Chatzakou et al., 2017], reaching the same range of scores than *Davidson et al.* [Davidson et al., 2017]. This highlights the complexity of detecting aggression, and how prediction quality depends not only on the selected features, but also on the intrinsic characteristics of data. For example, despite being written in English, posts in *Kumar et al.* were gathered from Hindi sites. Thereby, they could encompass idiomatic expressions that could differ from those used by Occidental users, or with those presenting a more colloquial usage of English. Additionally, due to cultural differences and that the concept of aggression is subjective, the criteria for defining what is and what is not an aggression could differ, hence it could also occur that posts might have a hidden sense that might not be captured by the English language. Furthermore, word embeddings or corpus-based techniques for extracting features might be biased by the origin of the training data, or by how such training corpus was created.

For both *Reynolds et al.* and *Kumar et al*, the best results were obtained when considering *TF-IDF + SentiWordNet*. In the case of *Chatzakou et al.*, the best results were observed for *TF-IDF + SentiWordNet+ Emoji*, whilst for *Davidson et al.*, they were observed for *GloVe. StanfordSentiment* consistently obtained the worst results for *Reynolds et al.*, *Chatzakou et al.* and *Davidson et al.* Conversely, in the case of *Kumar et al.*, the worst results were obtained when considering *POS Tags*. It is worth noting that the best and worst results differed at most in a 3%, 13%, 20% and 34% for *Reynolds et al, Chatzakou et al., Davidson et al., and Kumar et al.* respectively. This evidences that features are not the only variable that affects the performance of classifiers. In the case of *Reynolds et al.*, the implications of these observations might be two-fold. First, results could indicate that there is a clear differentiation between the post types, implying that the different feature sets could correctly classify most posts. Nonetheless, as neither precision nor recall were perfect, there also exists a set of posts that is similar to posts belonging to the other categories, thus misguiding the classifier. Second, as results are similar for most of the evaluated

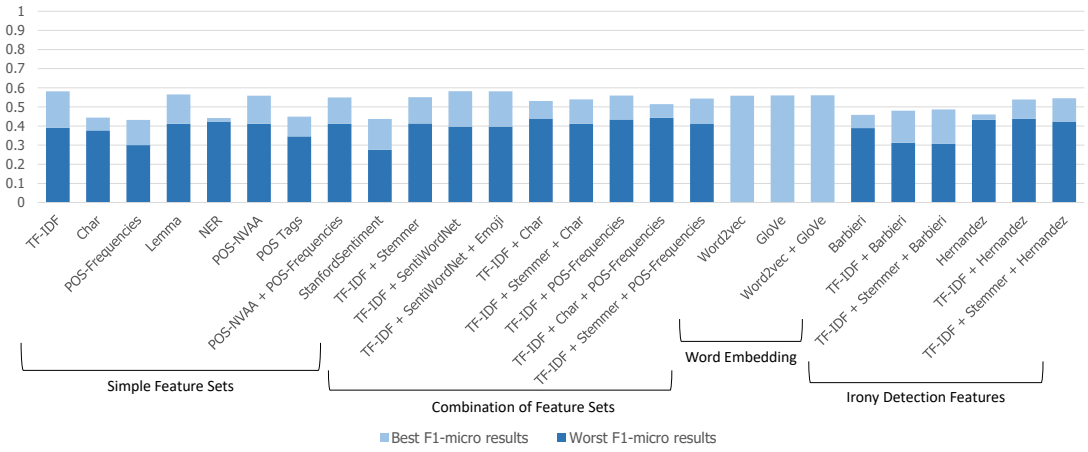


(a) *Chatzakou et al.*

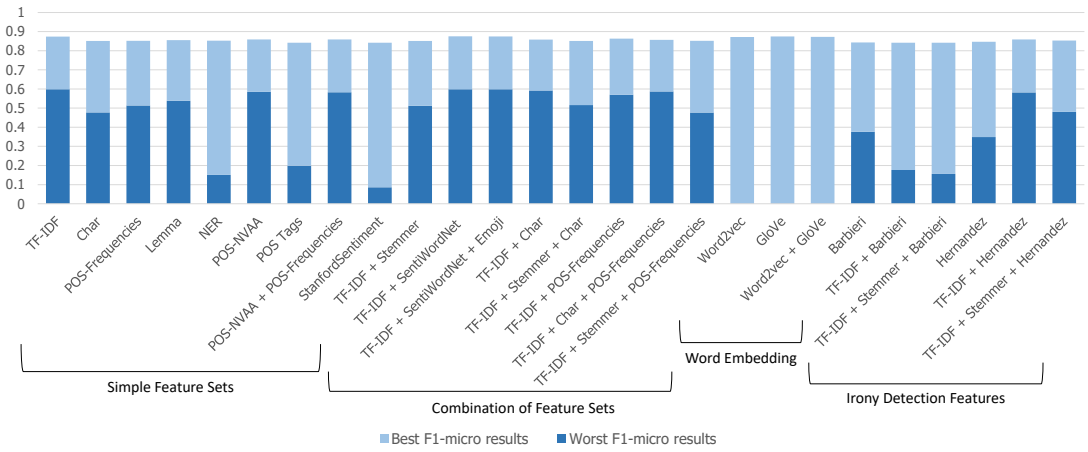


(b) *Davidson et al.*

Figure 1: Aggression Detection Results



(a) *Kumar et al.*



(b) *Reynolds et al.*

Figure 1: Aggression Detection Results (cont.)

feature sets, it might seem that for this data collection, despite providing different characterisations of posts, the diverse feature types might not contribute with new information. Conversely, in the cases of *Chatzakou et al., Davidson et al.,* and *Kumar et al.*, the high variability of results might indicate the difficulties for differentiating similar posts belonging to different classes, and the fact that the different combinations of feature sets provide complementary posts characterisation. For example, the result obtained for *Kumar et al.* when combining *TF-IDF* with sentiment features are higher that those obtained for the individual *TF-IDF* and sentiment features.

As regards the different types of features, their behaviour was similar for all data collections. For example, simply considering the textual features achieved high results in every collection. Feature sets including the *POS* tags or their frequencies did not achieve high results. Similarly, applying lemmatisation did not improve results of applying stemming. Interestingly, representing posts by their word embeddings only improved the results of simply considering the content of posts in one case, and for a marginal difference. Moreover, some features seemed to misguide the classifier, as exposed by the results of *TF-IDF + SentiWordNet + Emoji* that were slightly lower than those observed for *TF-IDF + SentiWordNet*. These results show that adding more features does not necessarily lead to a quality improvement of classifications. Finally, the feature sets for identifying irony were amongst the worst performing ones, which could imply that aggression does not convey irony.

The performed statistical analysis showed in the case of *Kumar et al.* with a confidence of 0.01 that the differences between most pairs of features sets were not statistically different. Nonetheless, statistically significant differences were observed for *Barbieri* and *StanfordSentiment*, which were shown to be statistically lower than feature sets involving *TF-IDF*. Similarly, for *Chatzakou et al.*, there were no statistical significant differences for most cases, with the exception of *TD-IDF* and *StanfordSentiment,* and *TF-IDF + SentiWordNet + Emoji* and *StanfordSentiment*. On the other hand, in the cases of *Davidson et al.* and *Reynolds et al.*, no statistical differences were observed for any of the feature sets. These results imply that more evaluations are needed to truly assess the descriptive power of features, and thus to improve the quality of results.

5.2 Cross Social Media Site Evaluation

Another evaluated scenario was the capabilities of models trained with data belonging to a particular social media site to detect aggression in other social media sites. In particular, the models were trained using a combination of the *Kumar et al.* training and validation collections. For the sake of brevity, this data collection will be called *Kumar et al. - Training*. As described above, this collection was built using English posts gathered from Indian *Facebook* pages. To assess the capabilities of the trained models, three testing collections were used. The first one was *Kumar et al. - Facebook testing*. This collection has similar characteristics to the training one, as it also comprises English posts gathered from Indian *Facebook* pages. The main goal of this evaluation is to act as a baseline for the cross social media aggression detection. The second dataset was *Kumar et al. - Twitter testing*. This collection comprises English post of Indian users gathered from *Twitter*. Note that the cultural context of the published posts in both collections is the same, but the social media and its interaction mechanisms are different. Finally, the third testing collection was *Reynolds et al.*, which comprised posts extracted from the American social network *FormSprin*. As a result, this collection differs from the training one in both cultural context and social network.

As regards the ground truth, all *Kumar et al.* datasets were labelled following the same conventions. As previously described, the classes in this dataset are non-aggressive, covertly aggressive, and overtly aggressive. The main difference between covertly and overtly aggressive is that in the former, aggression's are masked, e.g by irony, whilst the latter has openly aggressive or violent terms. On the other hand, *Reynolds et al.* was labelled following a different strategy. Posts were independently tagged by three Amazon Mechanical Turk workers. In this case, posts were classified as non-aggressive, weakly aggressive, and strongly aggressive. Therefore, there is nodirect correspondence between *Kumar et al.* and *Reynolds et al.* classes. For the purpose of the evaluation, a mapping between the classes in both collections was made, by which the weakly aggressive class was mapped to the covertly aggressive one (CAG), and the strongly aggressive class was mapped to the overtly aggressive one (OAG). Table 3 depicts the class distributions of data collections. In addition to the number of posts per class, the table also presents the

Dataset	*NAG*	*CAG*	*OAG*	*KL divergence*
Kumar et al. - Training	6,284	5,297	3,419	N/A
Kumar et al. - *Facebook* Testing	630	142	144	0.0672
Kumar et al. - *Twitter* Testing	483	413	361	0.0041
Raynolds et al.	10,837	1,160	776	0.1715

Table 3: Data Collections Class Distribution

Dataset	*Posts*	*Word Stems*	*Shared Word Stems*
Kumar et al. - Training	15,000	17,636	N/A
Kumar et al. - *Facebook* Testing	916	3,406	2694 (79.09%)
Kumar et al. - *Twiteeter* Testing	1,257	2,532	1,725 (68.12%)
Raynolds et al.	12,773	11,416	3,966 (34.74%)

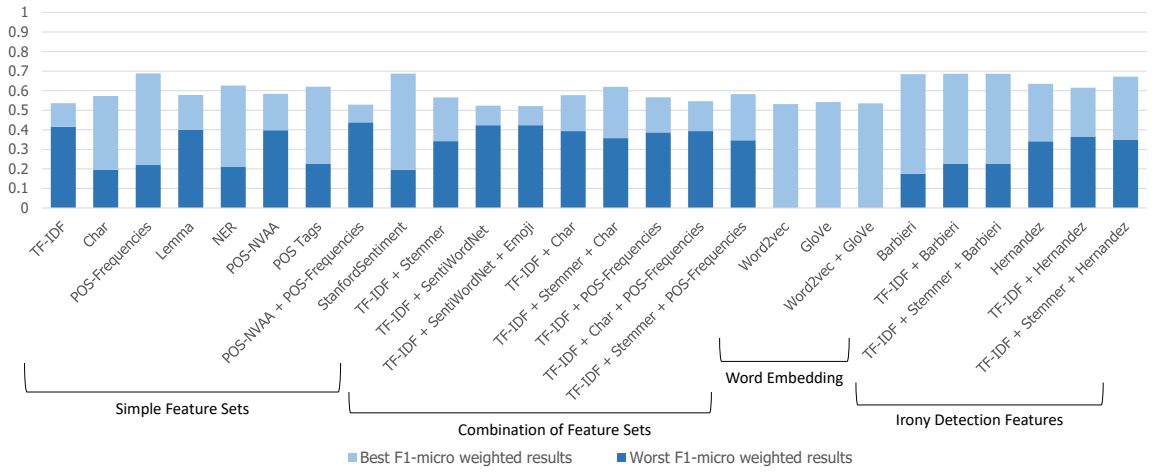
Table 4: Dataset Word Stems

Kullback-Leibler (KL) divergence (i.e. entropy difference) between the class distribution in the test sets against the train set. As it can be observed, *Kumar et al. - Twitter* testing is the most similar collection to the training dataset, followed by *Kumar et al. - Facebook* testing and *Reynolds et al.*

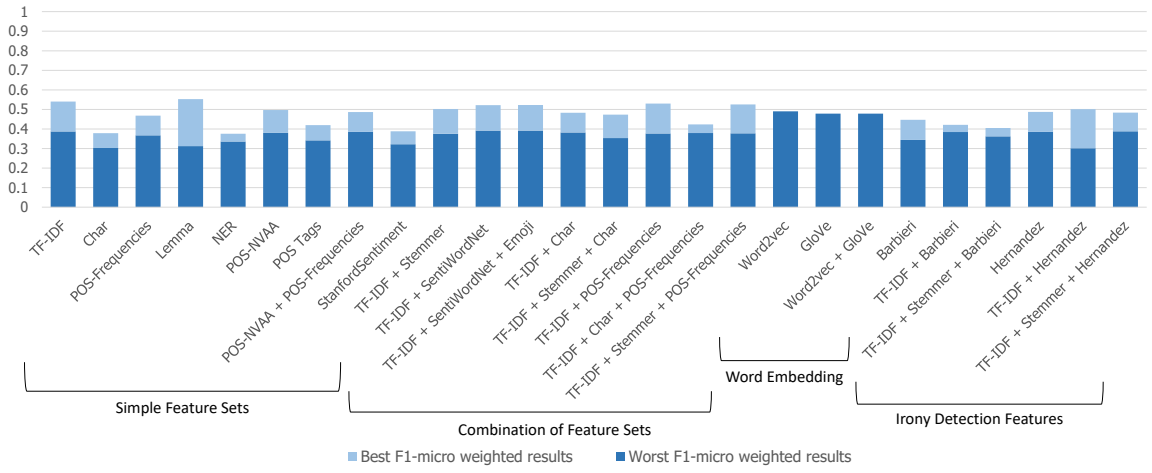
In this evaluation, models were trained considering the same feature sets previously described. One potential issue of using training and testing collections belonging to different data sources is that they might not share the same terms. Table 4 presents the number of word stems in each collection and how many of them were shared with the training collection. As it can be observed, approximately 79% of the word stems in *Kumar et al. - Facebook testing* are in the *Kumar et al. - Training.* When considering *Kumar et al. - Twitter testing*, only 68% of the word stems are shared with the training dataset. This might be related to the difference on the usage of language in both social media sites. However, further study on this topic is required. Finally, when considering *Reynolds et al.* dataset, even though this collection comprises approximately the same number of posts and terms than *Kumar et al. - Training*, the overlapping between them reached only the 35% of word stems.

Figure 2 shows the weighted F-measure according to the inverse class frequency of each instance. The stacked bars represent the worst and best results observed for each feature set. Figure 2a presents the results for *Kumar et al. - Facebook testing*, for which the best results were relatively stable across feature sets. Figure 2b depicts the results for *Kumar et al. - Twitter testing*, for which the best results obtained for this collection were slightly lower than the ones for *Kumar et al. Facebook - testing.* However, the worst results showed a higher estability and spanned over a similar range than the best ones. This indicates that the selected classification technique might have a low effect over the classification results. Finally, Figure 2c presents the results for *Reynolds et al.* In this case, the best and worst results were very dissimilar across feature sets. Nonetheless, in most cases, the best results were higher than the ones observed for *Kumar et al. - Facebook testing.*

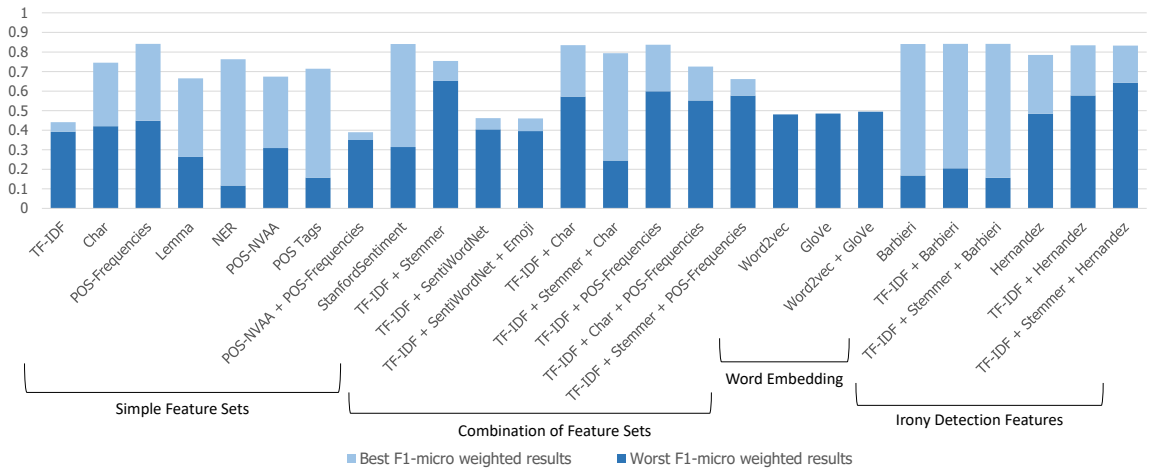
In the case of *Kumar el al. - Facebook* testing, the best results were obtained for *POS-Frequencies*, followed by *StanfordSentiment*, whilst the worst ones were observed for both *TF-IDF + SentiWordNet* and *TF-IDF + SentiWordNet + Emoji.* For *Reynolds et al.* the best results were observed for *TF-IDF + Stemmer + Barbieri* and *TF-IDF + Barbieri,* highlighting the relevance of irony related features for this dataset. On the other hand, the worst results were observed for both *POS-NVAA + POS-Frequencies* and *TF-IDF.* Note that even though combinations including *TF-IDF* achieved the highest results, considering *TF-IDF* alone did not achieve high results, which reinforces the importance of analysing how the different feature types interact. This could be related to the differences in language usage between both datasets. For example, Indians commonly use British spelling, hence it would be more likely to find the spelling "travelled" rather than "traveled". On the other hand, as *FormString* was mostly an American social site, the spelling "traveled" would be more likely to occur. In this context, simply considering the terms in post might not find many matchings, thus affecting the quality of predictions. This mismatching might be overcome by applying stemming or lemmatisation techniques. In the example, both "travelled" and "traveled" are mapped to the same stem "travel", which could help to improve the quality of classifications



(a) *Kumar et al. - Facebook* testing (Baseline)



(b) *Kumar et al. - Twitter* testing



(c) *Reynolds et al.*

Figure 2: Cross Social Media Site Evaluation Results

Dataset	*Best combination*	*Max. difference best results*	*Avg. Best - Worst*
Kumar et al. - Facebook testing	0.6879	0.1667	0.2807 (±0.1367)
Kumar et al. - Twitter testing	0.5529	0.1767	0.1089 (±0.0490)
Reynolds et al.	0.8416	0.4519	0.3295 (±0.2154)

Table 5: Summary of Results

as shown by the superiority of results of *TF-IDF + Stemmer* over *TF-IDF*. Finally, for *Kumar el al. - Twitter testing*, the best results were observed for both *Lemma* and *TF-IDF,* whilst the worst ones were observed for *NER* and *Char*.

For *Kumar et al. - Facebook testing* results were slightly higher (on average 14%) than the ones obtained for *Kumar et al.* in the individual evaluation. On the other hand, results for *Kumar et al. - Twitter testing* where slightly lower with differences up to a 9%. Despite this performance difference, the best results observed for each feature set were fairly stable, with standard deviations of 9.06% and 10.48% regarding the best results for *Kumar et al. - Facebook* and *Twitter testing* respectively. Similarly, the standard deviation for the previous evaluation was 9.45%. These results might have multiple implications. First, they could indicate that the feature sets have a similar capability for characterising aggression in both training and testing sets. Conversely, they could indicate that the selected feature sets are not able to grasp on the characteristics of data. Finally, the results might imply that even though feature sets aim at describing aggressions from different points of view, feature sets might not actually represent complementary points of view. This situation leads to the necessity of continuing the exploration and analysis of not only the characteristics and capabilities of the selected feature sets, but also the characteristics of the texts being analysed. Unlike the previous evaluation, results observed for *Reynolds et al.* have a greater variance across features. Whilst for the previous evaluation, considering both training and test sets from *Reynolds et al.* yielded a standard deviation of the best results of 0.01 (representing a 1.27% of the best average results), when training with *Kumar et al.* and testing with *Reynolds et al.*, the standard deviation was 0.153 (representing a 22.15% of the best average results). These observations might indicate that feature sets have different capabilities for characterising aggressions across different social media sites.

Another discrepancy with previous results was found in those observed for the different classification techniques. Whilst for *Kumar et al.* the multilayer perceptron with 0 hidden layers and standardised features performed statistically better than the majority of classifiers, for *Reynolds et al.* the Naïve Bayes classifier achieved statistically significant better results than the majority of classifiers. Particularly, Naïve Bayes achieved the best results for some of the low dimensional feature sets, such as *StanfordSentiment* (51 features), *POS Tags* (45 features), or high dimensional feature sets that integrate low dimensional feature sets, such as *TF-IDF + Char + POS-Frequencies* (29,439 for *TF-IDF*, 30 for *char* and 7 for *POS-Frequencies*) or *TF-IDF + Stemmer + Hernandez* (17,636 for *TF-IDF + Stemmer* and 14 for *Hernandez*). Regarding the combinations of low and high dimensional feature sets, the high dimensional feature sets that are mainly based on textual features (such as *TF-IDF + Stemmer*) are highly sparse. For instance, when mapping the *TF-IDF + Stemmer* features defined for *Kumar et al.*, only the 33% of them could be mapped to features in *Reynolds et al.* On the other hand, for feature sets like *Hernandez*, the sparseness was lower, and all features appeared in at least one instance. As a result, for the *TF-IDF + Stemmer + Hernandez* feature set, the *TF-IDF + Stemmer* features could be acting as noise, thus having a low impact on Naïve Bayes [El Hindi, 2014] models, which would be guided by the *Hernandez* features. Conversely, for the *Reynolds et al.* individual evaluation, Naïve Bayes did not perform well for these feature sets. This differences might be caused by changes in the balanced or unbalanced nature of datasets [Frank and Bouckaert, 2006].

6 Conclusions

Cyberbulling and cyberaggression are serious and widespread issues increasingly affecting Internet users. With the "help" of the widespread of social media networks, bullying once limited to particular places or

times of the day (e.g. schools), can now occur anytime and anywhere and have a wider range of audience. Cyberaggression can be defined as aggressive online behaviour that intends to cause harm to another person, involving rude, insulting, offensive, teasing or demoralising comments through online social media that target educational qualifications, gender, family or personal habits. This problem is aggravated by the persistence and durability of online materials, which gives these incidents an unprecedented power and influence to affect the lives of billions of people. In this context, cyberaggressions can have deeper and longer-lasting effects in comparison to physical bullying.

Links were found between experiences of cyberbullying and negative outcomes, such as decreased performance at school, dropping out and violent behaviour, in combination with devastating mental and psychological effects such as depression, low self-esteem, and even suicide. Considering the severity of the consequences that cyberaggression has on its victims, there is an imperious need for research aiming at understanding how cyberbullying occurs, in order to prevent it from escalating. Moreover, cyberaggression detection can be used to provide better support and advice for the victims as well as monitoring and tracking the bullies. Other important application of the detection of aggressive content is the detection of cyberextremism, cybercrime and cyberhate propaganda. Given the massive information overload on the Web and the pace at which new posts are being shared, it is unfeasible for human moderators to manually track and flag each insulting and offensive comment. Thereby, it is crucial to develop intelligent techniques to automatically detect harmful content, which would allow the large-scale social media monitoring and early detection of undesired situations.

Despite the seriousness of the problem, there are few successful efforts to detect abusive behaviour on social media data, due to the existence of several challenges, not only related to the nature of posts and the environment in which the aggression occurs, but also to technical limitations of the generalisation and comparability of the proposed approaches. This paper focused on the detection of aggressive content in the context of multiple and heterogeneous social media sites and analysed the capabilities of diverse feature sets for such task. Additionally, it was explored the feasibility of the selected feature sets and techniques for the identification of different types of accounts dedicated to the distribution of aggressive content. Feature sets included char, word and emotional-based features, features used for detecting irony and word-embeddings. The goal was to both provide a comparison between different feature sets and techniques proposed in the literature over the same datasets to analyse the generalisation of results, and to shed some light on the usefulness or adequacy of the different techniques for the task, and the generalisation of models trained for a specific social media site to other sites. Experimental evaluation conducted on four real-world social media dataset showed the difficulties for accurately detecting aggression in social media posts. Moreover, results exposed the limitations of the selected features in relation to the characteristics of the social media sites.

This work also studied the feasibility of using models trained for a specific social media site for classifying the content generated in other social media sites. In this regard, results showed the dependence of the trained models on the characteristics of the users sharing content (e.g. their culture and language usage), and hence the difficulty of generalising models and results to different social media sites. Moreover, another difficulty arose from the fact that there was no pre-defined agreement on the tagging of datasets. This worsens when considering datasets pertaining users belonging to different communities or cultures. This hints the necessity of studying how different demographic groups use, define and express aggressions through language in social media.

Considering the observed results, there still are open issues and challenges that could be tackled in future work. First, given the effect that the intrinsic characteristics of social media sites have on the performance of the detection task, it could be studied how such characteristics impact on each selected feature set. In this regard, it could be also studied how the information belonging to multiple and diverse social media sites could be integrated into a unified model to overcome the observed difficulties. Second, considering that nowadays social media sites are not limited to only textual posts, additional features such as images and videos could also be explored. In addition, it would be interesting to analyse the importance of considering the neighbourhood of users and how they behave to detect aggressions. Finally, given the unbalanced distribution of aggressive and non-aggressive posts, the performance of semi-supervised learning techniques could be studied.

References

S. Agarwal and A. Sureka. Applying social media intelligence for predicting and identifying on-line radicalization and civil unrest oriented threats. *arXiv preprint arXiv:1511.06858*, 2015. 1

M. A. Alvarez-Carmona, E. Guzmán-Falcón, M. Montes y Gómez, H. J. Escalante, L. Villaseñor-Pineda, V. Reyes-Meza, and A. Rico-Sulayes. Overview of MEX-A3T at IberEval 2018: Authorship and aggressiveness analysis in Mexican spanish tweets. In *Proceedings of the 3rd Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018)*, pages 74–96, Sevilla, Spain, 2018. 2

S. Taofeek Aroyehun and A. Gelbukh. Aggression detection in social media: Using deep neural networks, data augmentation, and pseudo labeling. In *Proceedings of the 1st Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pages 90–97, Santa Fe, USA, 2018. 2

F. Barbieri and H. Saggion. Modelling irony in Twitter. In *Proceedings of the Student Research Workshop at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 56–64, 2014. 3, 2a

U. Bretschneider, T. Wöhner, and R. Peters. Detecting online harassment in social networks. In *International Conference on Information Systems - Building a Better World through Information Systems, ICIS 2014*, 2014. 2

D. Chatzakou, N. Kourtellis, J. Blackburn, E. De Cristofaro, G. Stringhini, and A. Vakali. Mean birds: Detecting aggression and bullying on Twitter. In *Proceedings of the 2017 ACM on Web Science Conference (WebSci '17)*, pages 13–22, New York, NY, USA, 2017. ISBN 978-1-4503-4896-6. 1, 2, 4.1, 5.1

V. S. Chavan and S. S S. Machine learning approach for detection of cyber-aggressive comments by peers on social media network. In *Proceedings of the 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 2354–2358, Aug 2015. 1, 2

Y. Chen, Y. Zhou, S. Zhu, and H. Xu. Detecting offensive language in social media to protect adolescent online safety. In *Proceedings of the 2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, pages 71–80, Sept 2012. 2

G. W. Corder and D. I. Foreman. *Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach*. John Wiley & Sons, Inc., 2009. 5

M. Dadvar and Franciska M.G. de Jong. Cyberbullying detection; a step toward a safer internet yard. In *Proceedings of the 21st International World Wide Web Conference, WWW 2012 - PhD-Symposium*, pages 121–125, United States, 4 2012. Association for Computing Machinery. ISBN 978-1-4503-1323-0. doi: 10.1145/2187980.2187995. 1

T. Davidson, D. Warmsley, M. Macy, and I. Weber. Automated hate speech detection and the problem of offensive language. *arXiv preprint arXiv:1703.04009*, 2017. 4.1, 5.1

Khalil El Hindi. A noise tolerant fine tuning algorithm for the naïve bayesian learning algorithm. *Journal of King Saud University-Computer and Information Sciences*, 26(2):237–246, 2014. 5.2

G. Fahrnberger, D. Nayak, V. S. Martha, and S. Ramaswamy. Safechat: A tool to shield children's communication from explicit messages. In *2014 14th International Conference on Innovations for Community Services (I4CS)*, pages 80–86, June 2014. 2

Eibe Frank and Remco R Bouckaert. Naive bayes for text classification with unbalanced classes. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 503–510. Springer, 2006. 5.2

S. Frenda and S. Banerjee. Deep analysis in aggressive mexican tweets. In *Proceedings of the 3rd Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018)*, pages 108–113, Sevilla, Spain, 2018. 2

T. Galery and E. Charitos. Aggression identification and multi-lingual word embeddings. In *Proceedings of the 1st Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pages 74–79, Santa Fe, USA, 2018. 2

H. Gómez-Adorno, G. Bel-Enguix, G. Sierra, O. Sánchez, and D. Quezada. A machine learning approach for detecting aggressive tweets in Spanish. In *Proceedings of the 3rd Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2018)*, pages 102–107, Sevilla, Spain, 2018. 2

D. I. Hernańdez Farías, V. Patti, and P. Rosso. Irony detection in Twitter: The role of affective content. *ACM Transaction of Internet Technology*, 16(3):19:1–19:24, July 2016. ISSN 1533-5399. doi: 10.1145/2930663. 3, 2a

S. Hinduja and J. W. Patchin. Bullying, cyberbullying, and suicide. *Archives of Suicide Research*, 14(3): 206–221, 2010. 1

H. Hosseinmardi, S. A. Mattson, R. Ibn Rafiq, R. Han, Q. Lv, and S. Mishra. Analyzing labeled cyberbullying incidents on the instagram social network. In Tie-Yan Liu, Christie Napa Scollon, and Wenwu Zhu, editors, *Social Informatics*, pages 49–66, Cham, 2015. Springer International Publishing. ISBN 978-3-319-27433-1. 1

M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '04)*, pages 168–177, New York, NY, USA, 2004. ACM. ISBN 1-58113-888-1. doi: 10.1145/1014052.1014073. 3

H.-H. Huang, C.-C. Chen, and H.-H. Chen. Disambiguating false-alarm hashtag usages in tweets for irony detection. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 771–777, 2018. 5.1

A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017)*, volume 2, pages 427–431, Valencia, Spain, 2017. 2

P. Kralj Novak, J. Smailović, B. Sluban, and I. Mozetič. Sentiment of emojis. *PLoS ONE*, 10(12): e0144296, 2015. 3

R. Kumar, A. Kr. Ojha, S. Malmasi, and M. Zampieri. Benchmarking aggression identification in social media. In *Proceedings of the 1st Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pages 1–11, Santa Fe, USA, 2018a. 1, 2

R. Kumar, A. N. Reganti, A. Bhatia, and T. Maheshwari. Aggression-annotated corpus of hindi-english code-mixed data. *arXiv preprint arXiv:1803.09402*, 2018b. 4.1

Q. Le and T. Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages II–1188–II–1196. JMLR.org, 2014. 2

T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. 2, 4.2

C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang. Abusive language detection in online user content. In *Proceedings of the 25th International Conference on World Wide Web (WWW '16)*, pages 145–153, 2016. ISBN 978-1-4503-4143-1. 1, 2

J. W. Pennebaker, M. E. Francis, and R. J. Booth. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, 71(2001):2001, 2001. 2

J. Pennington, R. Socher, and C. D. Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, 2014. 2

P. J. C. Pérez, C. J. L. Valdez, M. Ortiz, J. P. S. Barrera, and P. F. Pérez. MISAAC: Instant messaging tool for ciberbullying detection. In *Proceedings on the International Conference on Artificial Intelligence (ICAI)*, page 1, 2012. 2

F. Ramiandrisoa and J. Mothe. IRIT at TRAC 2018. In *Proceedings of the 1st Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pages 19–27, Santa Fe, USA, 2018. 2

K. Reynolds, A. Kontostathis, and L. Edwards. Using machine learning to detect cyberbullying. In *Proceedings of the 10th International Conference on Machine Learning and Applications and Workshops*, volume 2, pages 241–244, Dec 2011. 4.1, 5.1

J. Risch and R. Krestel. Aggression identification using deep learning and data augmentation. In *Proceedings of the 1st Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pages 150–158, Santa Fe, USA, 2018. 2

A. Roy, P. Kapil, K. Basak, and A. Ekbal. An ensemble approach for aggression identification in English and Hindi text. In *Proceedings of the 1st Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pages 66–73, Santa Fe, USA, 2018. 2

S. Salawu, Y. He, and J. Lumsden. Approaches to automated detection of cyberbullying: A survey. *IEEE Transactions on Affective Computing*, pages 1–1, 2017. ISSN 1949-3045. 2

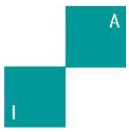
N. Safi Samghabadi, D. Mave, S. Kar, and T. Solorio. RiTUAL-UH at TRAC 2018 shared task: Aggression identification. In *Proceedings of the 1st Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pages 12–18, Santa Fe, USA, 2018. 2

A. Tommasel, J. M. Rodriguez, and D. L. Godoy. Features for detecting aggression in social media: An exploratory study. In *XIX Simposio Argentino de Inteligencia Artificial (ASAI)-JAIIO 47 (CABA, 2018)*, 2018. 1, 2

C. Van Hee, E. Lefever, B. Verhoeven, J. Mennes, B. Desmet, G. De Pauw, W. Daelemans, and V. Hoste. Automatic detection and prevention of cyberbullying. In *Proceedings of the International Conference on Human and Social Analytics*, pages 13–18. IARIA, 2015. ISBN 978-1-61208-447-3. 1, 2

E. Whittaker and R. M. Kowalski. Cyberbullying via social media. *Journal of School Violence*, 14(1): 11–29, 2015. 1

Inteligencia Artificial, 22(63) (2019), 101-120
doi: 10.4114/intartif.vol22iss63pp101-120



INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

Hand Vein Biometric Recognition Approaches Based on Wavelet, SVM, Artificial Neural Network and Image Registration

Daniel Felix de Brito[1,A] and Lee Luan Ling[1,B]

[1]Faculty of Electrical and Computer Engineering (FEEC), State University of Campinas (UNICAMP), Campinas-SP, Brazil.

[A]fbrito.daniel@gmail.com

[B]lee@decom.fee.unicamp.br

Abstract This paper describes in detail different hand vein recognition methods based on Wavelet-SVM, Wavelet-ANN and Image Registration. A new image segmentation and processing algorithm is proposed to efficiently locate vein regions and suitable for feature extraction (wavelet coefficients and normalized vein imagens) and classification (SVM, ANN and Image Registration). For real time recognition and high recognition rate, we proposed an integrated system which combines three above mentioned classification methods. The simulation results reveal that the proposed integrated system achieves 1% *false rejection rate* (FRR) and 0.02% *false acceptance rate* (FAR).

Resumo Este artigo descreve em detalhes diferentes métodos para o reconhecimento biométrico com base nas veias das mãos. Os métodos têm como base Wavelets com SVM, Wavelets com Rede Neural Artificial e Registro de Imagens. Um novo método de segmentação de imagens é proposto para localizar de modo adequado e eficiente a região das veias, permitindo a extração de características (com Wavelets ou a partir de uma imagem normalizada) e a classificação (com SVM, Rede Neural e Registro de Imagens). Para uma alta taxa de reconhecimento realizada em tempo real nós propomos um sistema híbrido que combina os três métodos de classificação mencionados. Os resultados de testes revelam que o sistema híbrido fornece uma *taxa de falsa rejeição* (FRR) de 1% para uma *taxa de falsa aceitação* (FAR) de 0.02%.

Keywords: Hand vein biometric, Support Vector Machine, Artificial Neural Network, Image Registration, Wavelet.

Palavras-chave: Biometria de veias das mãos, Máquina de vetores de suporte, Rede Neural Artificial, Registro de Imagens.

1 Introdução

Biometria pode ser definida como uma técnica de reconhecimento de padrões que identifica um indivíduo através de suas características fisiológicas ou comportamentais [1]. Entre as características biométricas mais utilizadas estão: impressão digital, íris, assinaturas, reconhecimento facial, veias das mãos, sendo que cada tipo biométrico apresenta certas vantagens e desvantagens quando aplicadas na prática [2].

Atualmente, o uso da biometria para identificação pessoal é uma técnica amplamente aplicada em diversas áreas envolvendo o acesso à segurança de dados e a confirmação de identidade pessoal no cotidiano. Podemos citar alguns exemplos de seu uso: caixas eletrônicos; smartphones, tablets e notebooks; controle

ISSN: 1137-3601 (print), 1988-3064 (on-line)

eletrônico de acesso aos locais de trabalho; identificação de usuários de planos de saúde. Em alguns casos, para aumentar ainda mais as taxas de correta identificação individual e o grau de confiabilidade no sistema que identificará o indivíduo, são utilizados sistemas biométricos multimodais, envolvendo mais de uma característica biométrica na identificação, como por exemplo, um sistema capturar a imagem da íris e da impressão digital e utilizar ambas para a identificação [3][4][5][6].

Neste projeto, utilizamos as veias das mãos como característica biométrica.Um sistema biométrico que utiliza a imagem das veias de uma das mãos considera o fato da hemoglobina, presente no sangue que flui nas veias, ser sensível à luz infravermelha. Sendo assim, ao iluminar a mão com uma luz infravermelha faz com que a hemoglobina do sangue absorva parte da luz e destaque o formato e a localização das veias, o que pode ser observado em uma imagem capturada por uma câmera infravermelha [7]. Logo, a variação dos formatos e das localizações das veias na região iluminada com a luz infravermelha permite a diferenciação entre os indivíduos, formando padrões que permitem determinar exclusivamente um indivíduo com base nessa característica biométrica. Na literatura, diversos estudos têm sido realizados com base nas veias das mãos para a autenticação pessoal [8][9][10][11].

O intenso interesse por essa área de pesquisa é justificado por algumas vantagens do uso da biometria de veias em relação aos demais tipos biométricos (íris, impressão digital, face, entre outros), incluindo: alta variabilidade inter-classes e baixa variabilidade intra-classes; baixo custo para a montagem de um protótipo para aquisição de imagens das veias; sem contato físico entre o equipamento de aquisição e a característica biométrica (veias), o que poderia interferir na correta captura, como é o caso da impressão digital; alto desempenho em termos de taxas de acerto no processo de reconhecimento.

Um sistema biométrico com base nas veias das mãos normalmente é composto de quatro unidades funcionais em cascata, ilustradas na Figura 1.

Figura 1: Fluxograma das etapas de um sistema biométrico de veias.

- **Aquisição da imagem**: através de um dispositivo de captura, posicionar a mão e iluminar a região do dorso ou da palma com luz infravermelha e capturar uma imagem digital da região iluminada;
- **Localização das veias na imagem**: utilizando técnicas de processamento de imagens digitais, dividir a imagem capturada em regiões e encontrar a região que representa as veias;
- **Normalização:** mapear a região das veias, que pode ter diferentes dimensões de acordo com cada captura, sempre para uma imagem de dimensões NxN;
- **Extração das características:** utilizar uma técnica de reconhecimento de padrões que possibilite a diferenciação entre as imagens das veias entre indivíduos distintos;
- **Armazenamento de características de referências**: armazenar em uma base de dados as características extraídas para posterior comparação;
- **Identificação ou autenticação:** comparar o padrão atual com o padrão capturado previamente e armazenado em uma base de dados.

O processo realizado em cada unidade é extremamente relevante para o sistema como um todo, influenciando o resultado das próximas unidades e o desempenho global do sistema. Métodos e algoritmos de processamento mais simples requerem a colaboração do usuário no correto posicionamento da mão para uma correta captura da imagem. Por outro lado, as abordagens mais sofisticadas consideram algum tipo de posicionamento incorreto pelo usuário e efetuam os ajustes necessários para identificar corretamente.

As seções seguintes apresentam em detalhes os procedimentos para cada etapa. Na seção 2 são apresentadas características gerais de um dispositivo de captura e uma ilustração de sua estrutura interna. Na seção 3 são descritos os filtros utilizados para melhorar a qualidade da imagem digital adquirida, os

procedimentos para dividir a imagem em regiões e assim localizar o contorno da mão seguida da localização das veias. Na seção 3.6 são descritos os procedimentos para transformar as regiões de veias de imagens capturadas em momentos distintos, seja da mesma pessoa ou de pessoas diversas, sempre em uma região de mesma dimensão. A seção 4 apresenta as seguintes técnicas para extrair padrões de características das regiões das veias: Transformada Wavelet com SVM, Transformada Wavelet com Redes Neurais e Registro de Imagens. Por fim, as seções 5 e 6 apresentam os procedimentos adotados na implementação de todos os métodos apresentados na seções anteriores, bem como discussões sobre os resultados obtidos e as conclusões finais.

2 Aquisição da Imagem

A unidade de aquisição da imagem consiste na captura de imagens das veias da parte superior (dorso) ou inferior (palma) da mão utilizando um dispositivo de captura. A estrutura geral de um dispositivo de captura é ilustrada na Figura 2, onde leds infravermelhos (normalmente na faixa de 850 nm) são usados para iluminar a região de interesse das veias. Nessa faixa de frequência, a hemoglobina absorve parte da energia da luz fazendo com que as veias sejam mais nitidamente realçadas na imagem capturada por uma câmera CCD comum ou uma câmera infravermelha. Uma câmera CCD comum tem menor custo, porém menor qualidade de imagem, em comparação com o custo e a qualidade de uma câmera infravermelha.

A Figura 3 ilustra imagens capturadas por uma câmera CCD[1]. Para uma câmera CCD, é necessário remover o filtro infravermelho que está presente nas câmeras comerciais e utilizar um filtro de luz visível. A distância entre a mão e a câmera costuma variar entre 8 a 20 cm.

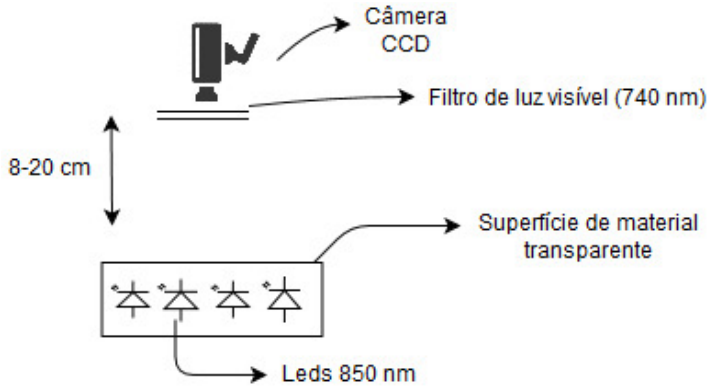


Figura 2: Visão geral de um protótipo da estrutura de um dispositivo de captura de imagens de veias da mão. Nesse caso, a palma da mão deve ser posicionada sobre a superfície de material transparente, com a palma para baixo.

Detalhes de alguns protótipos de dispositivos de aquisição podem ser encontrados em [7][9][10][11]. Um estudo mais completo sobre a aquisição de imagens das veias e o uso de feixes de luz próximos ao infravermelho é feito por Wang *et al.*[13].

Alternativamente ao uso de um dispositivo de captura, pode-se obter a imagem a partir de um banco de imagens já capturadas. Este segundo caso normalmente ocorre nos estudos na literatura quando o foco do estudo não é a aquisição da imagem em si, mas uma ou mais das outras etapas do processo de reconhecimento, como, por exemplo, apresentar uma nova técnica para a extração de características.

Neste estudo, utilizamos as imagens do banco de dados compilado por Badawi [12], que possui imagens capturadas da região do dorso das mãos. Três motivos principais levaram a escolha desse banco de imagens dentre os disponíveis:

1. O banco de imagens é utilizado por outros pesquisadores na literatura, facilitando a comparação dos resultados de testes com artigos do Estado da Arte;

[1]Imagens do banco de dados [12]

2. As imagens foram capturadas por uma imagem CCD, o que apresenta um maior desafio na identificação uma vez que as imagens são de menor qualidade em relação a imagens de outros banco de dados disponíveis;

3. A quantidade de imagens disponíveis por pessoa permite separá-las em dois conjuntos: um conjunto de treino e um conjunto de teste;

4. Alta variabilidade dos voluntários: homens e mulheres com idades entre 18 e 65 anos.

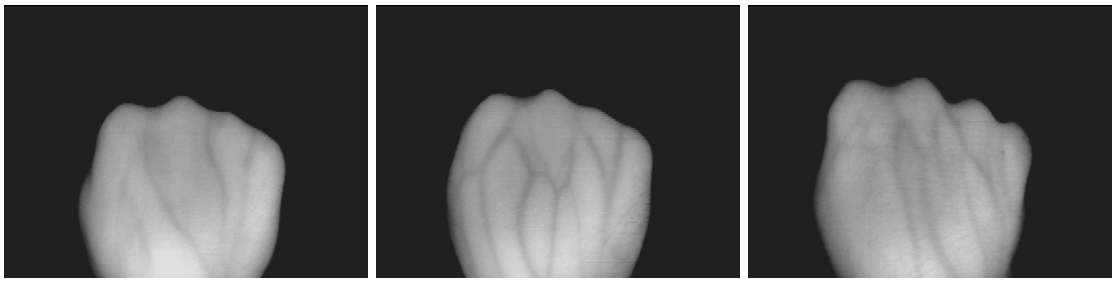


Figura 3: Exemplos de imagens de veias do dorso da mão.

3 Pré-Processamento: Segmentação e Normalização

A segunda etapa do sistema biométrico desenvolvido trata-se da segmentação da imagem capturada com o objetivo de localizar e delimitar a região de veias na imagem capturada. Para atingir este objetivo, a técnica desenvolvida, em linhas gerais, determina o contorno da mão na imagem e posteriormente, com base neste contorno localizado, identifica os pixels da imagem que contêm a região das veias. Conforme ilustradas na Figura 4, o procedimento completo implementado neste trabalho subdivide-se em quatro fases: Filtragem para melhoria da imagem (Filtro Mediana, filtragem no domínio da frequência e ajuste de contraste); Geração de uma imagem binária; Determinação da região de interesse; e Eliminação de ruídos.

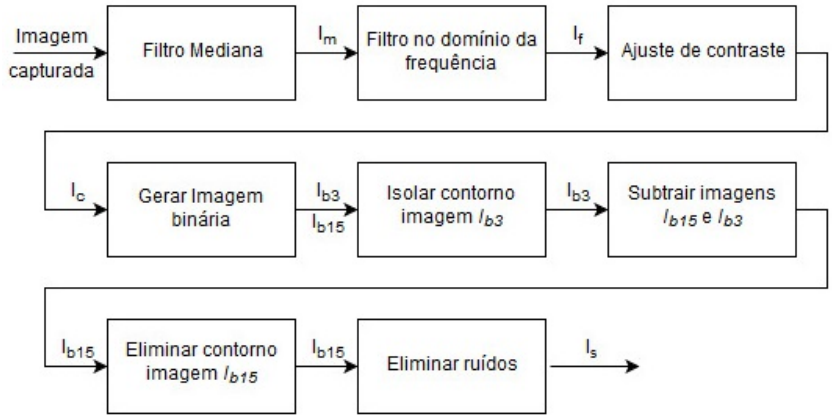


Figura 4: Visão geral das etapas do processo de segmentação da imagem.

3.1 Filtragem

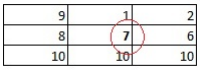
Esta etapa de filtragem tem como objetivos melhorar a qualidade geral da imagem através da redução de ruídos gerados na fase de aquisição e realçar as bordas na imagem, em especial as bordas que formam o contorno da mão e as veias. Este procedimento de realce é fundamental para o próximo passo de processamento da imagem que é a determinação da região da mão utilizando bordas realçadas presentes na imagem, bem como localizar as veias. O processo de melhoria da qualidade da imagem consiste na aplicação de dois filtros e no ajuste de contraste.

3.1.1 Filtro Mediana

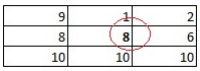
O primeiro filtro aplicado é um Filtro Mediana. Este filtro tem como objetivo eliminar ruídos gerados no processo de aquisição da imagem, ocasionados principalmente por: variações da luz ambiente e da incidência de luz solar; variações de temperatura; acúmulo de poeira na lente; defeitos de fábrica em transdutores fotossensíveis que formam o sensor de captura. Em geral, o ruído gera valores de brilho que tendem a ser muito diferentes dos valores de brilho de seus vizinhos adjacentes.

Basicamente, o procedimento de filtragem é feito da seguinte forma: para cada pixel p da imagem, ordena-se, em ordem crescente, uma lista com os valores de brilho dos nxn pixels adjacentes a p. Considera-se n um valor ímpar. O valor de brilho do pixel p é substituído pelo valor de brilho da posição mediana da lista que foi ordenada.

A Figura 5 ilustra este procedimento, onde o pixel central da região em análise, com valor de brilho igual a 7 (Figura 5a) receberá o valor 8 (Figura 5b), após a ordenação dos valores de brilho de sua região vizinha (Figura 5c).



(a) Valores de brilho do pixel central em análise e região vizinha.



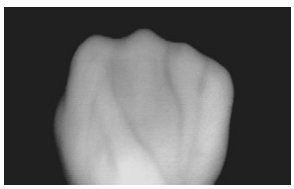
(b) Novo valor de brilho para o pixel central.

(c) Valores de brilho da região ordenados. Valor mediano selecionado.

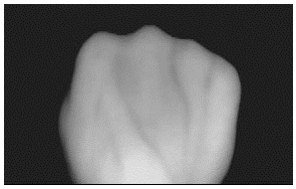
Figura 5: Exemplo da escolha de um novo valor de brilho para um Filtro Mediana de dimensão 3x3.

Foram testados diversos valores de n para a dimensão do filtro da mediana e feita a análise visual das imagens resultantes. Os melhores resultados foram obtidos com o valor de n igual a 3, ou seja, um filtro de dimensão 3x3, o qual reduziu os ruídos sem distorcer as bordas presentes na imagem, o que ocorreu com a aplicação de filtros de dimensões maiores, como 15x15, 21x21.

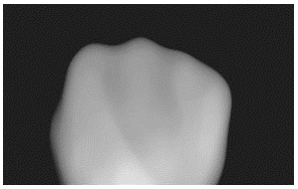
Ao final desta etapa, temos uma imagem com os ruídos reduzidos, com regiões mais uniformes, ou seja, os valores de brilhos dos pixels de cada região estão muito próximos ao valor de brilho médio da região. A imagem resultante é denominada I_m, conforme o fluxograma apresentado na Figura 4. As Figuras 6a e 6b reproduzem uma imagem original e a imagem após a aplicação do Filtro da Mediana (imagem I_m), respectivamente.



(a) Imagem original.

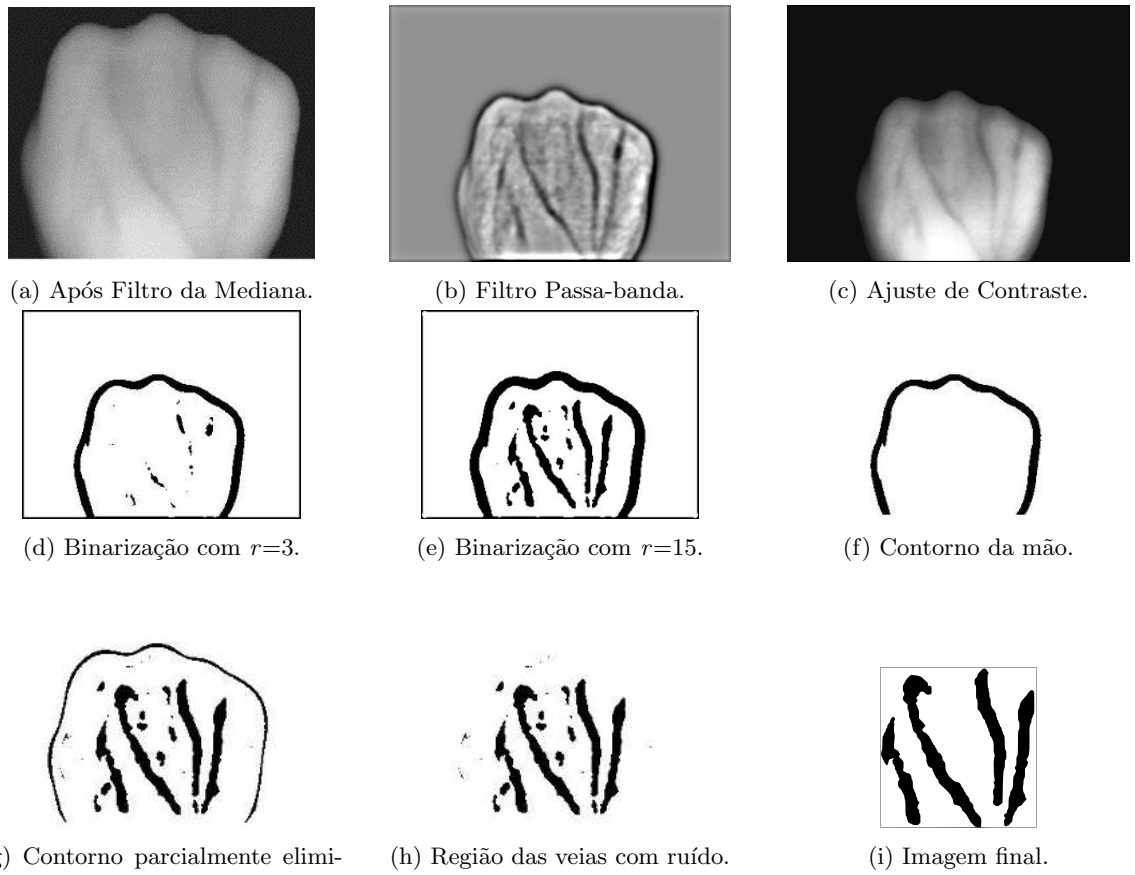


(b) Filtro Mediana de dimensão 3x3.



(c) Filtro Mediana de dimensão 21x21. Imagem borrada.

Figura 6: Exemplos de imagens com o resultado da aplicação de filtros Mediana.



(a) Após Filtro da Mediana. (b) Filtro Passa-banda. (c) Ajuste de Contraste.

(d) Binarização com r=3. (e) Binarização com r=15. (f) Contorno da mão.

(g) Contorno parcialmente eliminado. (h) Região das veias com ruído. (i) Imagem final.

Figura 7: Imagens resultantes de cada etapa do processo de segmentação.

3.1.2 Filtragem no domínio da frequência

O segundo filtro aplicado tem como objetivo realçar as bordas da região que formam as veias. Trata-se de um filtro passa-banda ou passa-faixa (*FFT bandpass filter*), sendo aplicado no domínio da frequência. Este filtro permite a passagem das frequências de uma certa faixa de valores, enquanto rejeita o restante. Nesse projeto utilizamos a implementação *open-source* disponibilizada no projeto ImageJ [14]. Para utilizar esta implementação, é necessário definir alguns parâmetros. Os dois principais parâmetros são os valores que determinam a faixa de frequências permitidas pelo filtro, denominados no ImageJ como *Filter Large Structures Down* e *Filter Large Structures Up*. Após testes com diversos valores, adotou-se para os dois parâmetros os valores 5 e 20, respectivamente. Para os demais parâmetros, os valores utilizados não foram alterados, ou seja, foram utilizados valores *default* definidos pela própria implementação. O filtro é aplicado sobre a imagem I_m (resultante da etapa anterior), produzindo como resultado a imagem I_f, conforme ilustrado no fluxograma da Figura 4. A Figura 7b ilustra o resultado desta etapa. Comparando-se com a Figura 7a, nota-se que as veias foram realçadas.

3.1.3 Ajuste de contraste

Esta etapa utiliza dois conceitos de processamento de imagens: saturação e contraste. A saturação indica a pureza ou a intensidade de uma cor em particular. O contraste é a diferença entre a região clara e escura da imagem. Deste modo, esta etapa tem como objetivo reduzir o efeito de regiões muito claras ou muito escuras, em relação as demais. Este efeito é causado, principalmente, pelo excesso de luz em algumas regiões e pela falta de luz em outras, além de reflexos, durante a captura da imagem. Em outras palavras, esta etapa busca equalizar a imagem, ou seja, que a diferença entre os valores de brilhos das

regiões mais claras e mais escuras não seja tão grande, mas mais uniforme.

O ajuste de contraste tem como base as informações do histograma dos pixels da imagem. Utilizando a implementação *open-source* ImageJ [14] para o ajuste de contraste, é possível definir uma porcentagem do número de pixels da imagem que serão saturados, bem como se ocorrerá uma equalização do histograma e uma normalização dos valores de brilho.

Foram testados diferentes valores de porcentagem de saturação e foi feita uma análise visual dos resultados obtidos. O valor de porcentagem de saturação de 0,3% apresentou o melhor resultado em termos de realce da região das veias na imagem. Foram realizadas a equalização do histograma e a normalização dos valores de brilho. Estes procedimentos foram aplicados na imagem I_f, resultante da etapa anterior. Como resultado, temos a imagem I_c, ilustrada na Figura 7c, com o realce das veias do dorso da mão.

3.2 Geração da imagem binária

Esta etapa é responsável pela produção de duas imagens binárias, denominadas I_{b_3} e $I_{b_{15}}$. Ambas as imagens são obtidas de modo independente a partir da imagem I_c, conforme ilustrado no fluxograma da Figura 4. Uma imagem binária consiste em uma imagem com apenas dois valores possíveis para cada pixel. A partir de um valor referência T, denominado limiar de corte, os pixels com valores acima de T formarão o objeto e receberão valor v_1. Caso contrário, serão classificados como fundo e receberão valor v_2.

O valor T pode ser um único valor, denominado limiar global, ou um valor calculado de modo independente para cada posição ou região da imagem, denominado limiar local. Nesse projeto, utilizou-se um limiar local, calculado a partir da Fórmula 1, proposta por Phansalkar *et al.* [15]. O valor de T na Fórmula 1 é calculado para cada pixel da imagem, considerando seu valor de brilho e os valores de brilho de seus pixels vizinhos. Apesar de Phansalkar *et al.* terem sugerido uma área retangular para delimitar a vizinhança em torno da posição corrente, nesse projeto delimitou-se a vizinhança a uma área circular de raio 0,5.

Os parâmetros da Fórmula 1 são: m é a média e d o desvio padrão dos valores dos brilhos dos pixels dentro da área circular delimitada; p e q são constantes cujos valores recomendados por Phansalkar *et al.* são 2 e 10, respectivamente, e foram os valores utilizados; k é um parâmetro cujo valor recomendado por Phansalkar *et al.* é 0,25 e foi o valor utilizado; $T(x,y)$ é o valor de limiar local para a posição em análise. O parâmetro r possui valor 3 para gerar a imagem I_{b_3} e 15 para gerar a imagem $I_{b_{15}}$, ilustradas nas Figuras 7d e 7e, respectivamente.

$$T = m * (1 + p * exp(-q * m) + k * ((d/r) - 1)) \quad (1)$$

3.3 Determinação da região de interesse

Esta etapa tem como objetivo isolar a região que compõe as veias. Para isto, são utilizadas as imagens binárias I_{b_3} e $I_{b_{15}}$. Este procedimento é composto de três passos, descritos a seguir. A seção 3.5 descreve detalhadamente porque são utilizadas duas imagens binárias e não apenas uma.

3.3.1 Isolamento do contorno das mãos na imagem I_{b_3}

O primeiro passo utiliza a imagem binária I_{b_3}. Conforme é possível perceber ao analisar a Figura 7d, ao aplicar a Fórmula 1 para valor de r baixo, restaram os pixels que compõem o contorno da mão e pixels em outras áreas. Sendo assim, é necessário eliminar esses pixels de outras áreas e selecionar apenas o contorno da mão.

O método utilizado para atingir este objetivo consiste em localizar cada região contínua da imagem binária I_{b_3}. Denomina-se região contínua uma região em que qualquer pixel dessa região cumpre dois critérios: o primeiro, possuir um valor de brilho igual a v_1, ou seja, pertencer a um objeto e não ao fundo; e ao menos um de seus vizinhos imediatos também tenha valor de brilho v_1.

Para cada região, são calculados o ponto médio (X_m,Y_m) e o desvio-padrão, considerando as coordenadas x e y de cada pixel da região. A região escolhida como sendo a que representa o contorno da mão é àquela que possui o maior valor de desvio-padrão. A Figura 7f é o resultado desta operação na

imagem I_{b_3} (Figura 7d). Como pode ser notado ao analisar visualmente as Figuras 7f e 7d a região do contorno da mão é a região que possui os pixels menos agrupados em determinada área, motivo pelo qual adotou-se o critério de escolha a região com o maior valor de desvio-padrão.

3.3.2 Subtração das imagens $I_{b_{15}}$ e I_{b_3}

O segundo passo tem como objetivo afinar a região que representa o contorno da mão na imagem $I_{b_{15}}$, ilustrada na Figura 7e, utilizando a informação do contorno da mão isolado na imagem I_{b_3}, ilustrado na Figura 7f. Esse processo de afinar uma região é conhecido em Morfologia como Erosão.

Para atingir este objetivo, a imagem $I_{b_{15}}$ é subtraída da imagem I_{b_3}: caso um pixel pertença ao contorno da mão na imagem I_{b_3} e o pixel na posição correspondente na imagem $I_{b_{15}}$ seja um pixel de algum objeto, o pixel na imagem $I_{b_{15}}$ deixará de pertencer ao objeto e passará a ser considerado como fundo. Em termos de implementação, o valor de brilho do pixel será alterado de v_1 para v_2.

O resultado deste passo é ilustrado na Figura 7g. Ao analisar a Figura 7g, comparando-se com a Figura7e, nota-se que a imagem $I_{b_{15}}$ foi afinada parcialmente na região do contorno da mão.

3.3.3 Eliminação do contorno da mão na imagem $I_{b_{15}}$

Este terceiro passo tem como objetivo eliminar a região do contorno da mão na imagem $I_{b_{15}}$. O procedimento adotado é similar ao passo 3.3.1. No entanto, ao invés de manter somente a região com o maior valor de desvio-padrão, elimina-se justamente essa região que representa a região do contorno da mão. O resultado é ilustrado na Figura 7h. Conforme pode ser observado na Figura 7h o resultado final consiste em uma imagem com as regiões das veias, sem o contorno da mão, embora ainda persistam alguns pixels que não pertencem às veias. Esses pixels são considerados ruídos e precisam ser eliminados.

3.4 Eliminação de ruídos e imagem final

Esta é a última etapa do processo de segmentação com o objetivo de eliminar ruídos ainda presentes após a localização dos pixels que compõem as veias. Para atingir este objetivo, localizamos cada uma das regiões contínuas da imagem da Figura 7h. Remove-se da imagem as regiões que tiverem um total de pixels menor do que *10%* da região com o maior número de pixels. O resultado desta operação é uma imagem em que estarão presentes apenas os pixels das veias, ilustrada na Figura 7i.

É importante ressaltar, para fins de implementação, que a imagem final deve possuir dimensões equivalente apenas a região final das veias, conforme ilustrada pela moldura ao redor da região das veias na Figura 7i. Para isto, é suficiente verificar a localização (coordenadas) dos pixels mais extremos em cada um dos lados (direito, esquerdo, superior e inferior). Este detalhe é importante para a etapa de normalização em que poderão ser realizadas alterações na escala da imagem final. Destaca-se que a imagem final contém apenas a região das veias. Sendo assim, além de eliminar os ruídos, Estes procedimentos encerram a etapa de Segmentação da Imagem e a imagem resultante é utilizada na etapa seguinte de Normalização.

3.5 Subtração das Imagens $I_{b_{15}}$ e I_{b_3}

Antes de continuar o processo de reconhecimento biométrico, convém explicar porque no processo de segmentação foram utilizadas duas imagens binárias, I_{b_3} e $I_{b_{15}}$, bem como a etapa de subtração entre elas.

Quando utiliza-se valores maiores para r na Fórmula 1, a região das veias na imagem binária são melhor preservadas. No entanto, em algumas imagens, as regiões que formam as veias conectam-se à região que forma o contorno da mão, formando uma única região, conforme ilustrado na Figura 8. Deste modo, ao se determinar a região com o maior valor de desvio-padrão, eliminava-se o contorno da mão e também os pixels que compõem as veias, pois estavam todos unidos.

A solução encontrada foi aplicar a técnica de liminar local para r igual a 3 e gerar uma imagem binária em que o contorno da mão é preservado, enquanto a região das veias é praticamente eliminada. Além disso, o uso de r igual a 3 faz com que o contorno da mão seja uma borda mais fina, quando comparada ao contorno produzido na imagem gerada para r igual a 15, o que é possível verificar ao

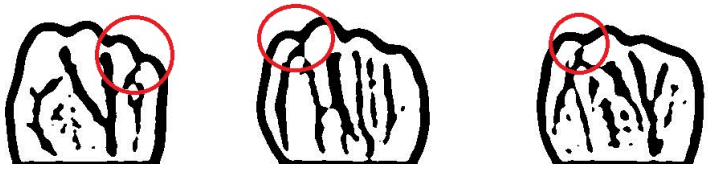


Figura 8: Exemplos de imagens em que o contorno da mão e as veias formam uma única região.

analisar visualmente as Figuras 7d e 7e). Deste modo, ao efetuar a subtração dos contornos, elimina-se justamente essa conexão entre o contorno da mão e as veias na imagem que preserva melhor as veias, $I_{b_{15}}$. Por este motivo foram utilizadas duas imagens binárias com valores de r distintos.

3.6 Normalização

A imagem final produzida no processo de segmentação, I_{final}, que contém apenas a região das veias, conforme ilustrado na Figura 7i, pode possuir dimensões diferentes para imagens tanto do mesmo indivíduo quanto de pessoas distintas. Isso ocorre tanto pela possibilidade de alterações de escala durante o processo de aquisição da imagem quanto pelo processo de segmentação em si, que pode resultar em maior ou menor quantidade de pixels ao seu final. Sendo assim, convém que seja feita uma normalização da imagem I_{final}, para que o processo de extração de características, etapa seguinte, atue sobre imagens padronizadas quanto às suas dimensões.

Assim, a etapa de normalização tem como objetivo produzir uma imagem de dimensões nxn a partir da imagem I_{final}. Para atingir este objetivo, a técnica de Transformação Afim ou *affine transformation* foi utilizada. Esta técnica consiste basicamente em efetuar as operações geométricas básicas através de um mapeamento linear aplicando uma matriz de transformação sobre os valores dos pixels da imagem. Foi utilizada uma implementação disponível em Java na classe *AffineTransform* que utilizada a matriz ilustrada na Figura 9. Os parâmetros s_x e s_y indicam, respectivamente, os fatores de alteração de escala nas direções x e y. O valor desses fatores dependem das dimensões da imagem inicial e final.

$$\begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Figura 9: Matriz de Transformação para normalização da imagem I_{final}.

Para os estudos experimentais, os valores de dimensões nxn utilizados para a imagem normalizada foram de 100x100 e 25x25. As imagens normalizadas em 100x100 são utilizadas no Registro de Imagens (seção 4.4), enquanto as imagens 25x25 são utilizadas na SVM e na Rede Neural (seções 4.2 e 4.3).

4 Extração de Características e Classificação

Esta seção descreve os métodos de classificação usados para identificação pessoal com base nas características das veias das mãos. Dois conjuntos de características são formados, um a partir da Transformada Wavelet aplicada na imagem das veias e o outro a partir da própria imagem da veias. Três métodos de classificação são avaliados: SVM com Transformada Wavelet, Redes Neurais com Transformada Wavelet e Registro de Imagem usando imagens das veias.

A seção está organizada da seguinte forma: a seção 4.1 faz uma revisão sucinta sobre Transformada Wavelet, SVM, Rede Neural e os conceitos de *FAR* e *FRR*. As seções 4.2 e 4.3 apresentam os métodos de classificação SVM e Rede Neural com a Transformada Wavelet. A seção 4.4 descreve o procedimento de classificação de Registro de Imagem.

4.1 Resumo teórico

4.1.1 Transformada Wavelet

Wavelets são um conjunto de funções obtidas a partir de operações de dilatações e translações de uma função principal, conhecida como Wavelet mãe [16]. Wavelets possuem formatos variantes contínua e discreta. Exemplos de famílias de funções Wavelets muito utilizadas na literatura são: Haar, Daubechies, Symlet e Coiflet [17][18][19].

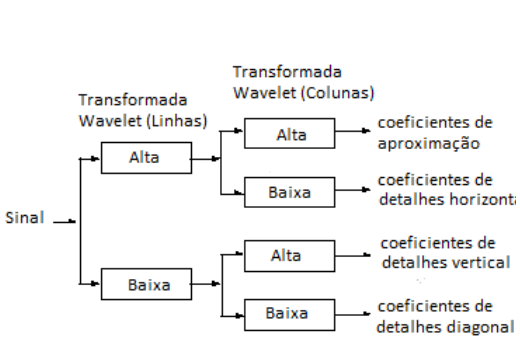
A Transformada Wavelet é uma ferramenta de análise de funções temporais que faz a decomposição de um sinal em termos de wavelets. Uma vantagem da Transformada de Wavelet é que as funções wavelets adaptam-se a cada ponto do sinal, permitindo a análise em diferentes resoluções temporais. Ao aplicar a Transformada Wavelet em um sinal, gera-se um novo sinal que contém dois componentes: um de baixa frequência e um de alta frequência. O componente de baixa frequência contém os denominados *coeficientes de aproximação*, que representam uma versão suavizada do sinal original. Por outro lado, o componente de alta frequência é composto pelos *coeficientes de detalhes*, que representam os detalhes do sinal original.

Na área de Processamento Digital de Imagens, a Transformada Wavelet pode ser utilizada para a decomposição de uma imagem em sub-imagens, também chamadas de componentes ou sub-bandas, sendo que cada componente possui informações e níveis distintos de detalhes. Normalmente, utiliza-se a Transformada Wavelet discreta em duas dimensões, em um processo conhecido como Decomposição de Wavelet em duas dimensões, ou *2D DWT* (*2-D discrete wavelet decomposition*).

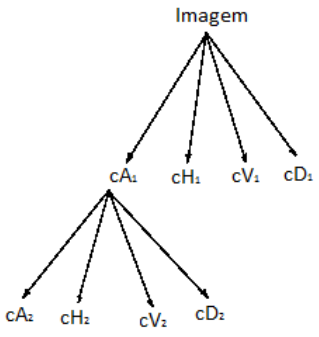
O processo *2D DWT* consiste basicamente na aplicação duas vezes da Transformada Wavelet em um sinal uni-dimensional. Assim, considerando uma imagem representada por uma matriz bi-dimensional, em que cada elemento da matriz indica o valor de brilho local da imagem, a primeira aplicação da transformada utiliza os valores de brilho das linhas da matriz. Sobre o resultado da primeira operação de Transformada Wavelet, aplica-se novamente a transformada, mas desta vez considerando os valores de brilho das colunas. Como consequência da aplicação da *2D DWT* são gerados quatro componentes: os coeficientes de aproximação, coeficientes de detalhes horizontal, coeficientes de detalhes vertical e coeficientes de detalhes diagonal. Detalhes sobre os coeficientes gerados podem ser consultados em [20].

A Figura 10a ilustra o processo de aplicação da Transformada Wavelet, em que primeiramente aplica-se a transformada considerando os valores de brilho das linhas, e para cada uma das duas imagens resultantes, aplica-se a transformada considerando os valores de brilho das colunas. Na Figura 10a, *Alta* indica o componente de alta frequência e *Baixa* o componente de baixa frequência.

Diferentes níveis de decomposição podem ser realizados, sendo o componente que representa os *coeficientes de aproximação* de um nível decomposto no nível seguinte, formando uma decomposição multi-nível. A Figura 10b ilustra a decomposição multi-nível, em que cA_i, cH_i, cV_i, cD_i representam, respectivamente, os coeficientes de aproximação, de detalhes horizontal, de detalhes vertical e de detalhes diagonal no nível i.



(a) Componentes resultantes da aplicação da *2D DWT*.

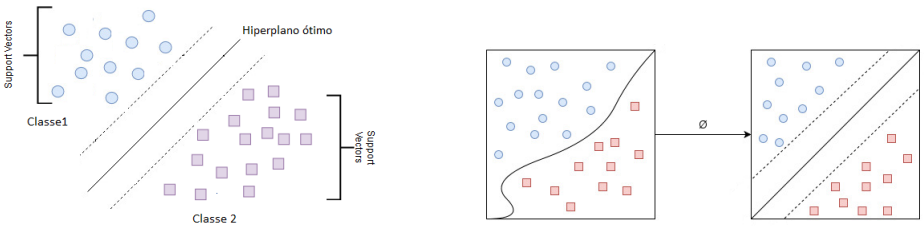


(b) Representação de uma decomposição multi-nível da Transformada Wavelet.

Figura 10: Organogramas das estruturas gerais da aplicação da Transformada de Wavelets.

4.1.2 SVM - Support Vector Machine

SVM é uma técnica de aprendizado de máquina supervisionado que pode ser utilizada para separar dados em classes. O SVM tem como base a teoria de aprendizado estatístico de Vapnik and Chervonenkis [21]. O funcionamento básico de uma SVM consiste em encontrar o hiperplano que separa um conjunto de dados de treinamento em duas classes. Neste processo, o SVM busca o hiperplano que maximiza a distância entre as classes, ou seja, o SVM não busca apenas um hiperplano, mas um hiperplano ótimo. A Figura 11a ilustra a representação de duas classes e o hiperplano ótimo para um conjunto de dados.



(a) Representação de duas classes em uma SVM e o hiperplano.

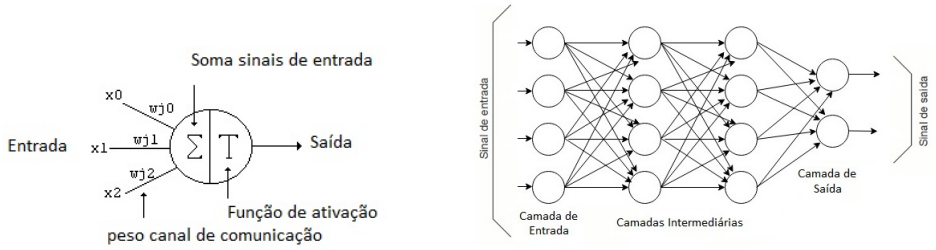
(b) Representação de um problema não linear mapeado para um problema linear utilizando uma função ϕ.

Figura 11: Representação gráfica de conceitos relativos à SVM.

O SVM pode ser usado para solucionar um problema com dados não linearmente separáveis. Para isto, o conjunto de dados é mapeado de seu espaço original para um novo espaço de maior dimensão, denominado *espaço de características* e a separação de dados é feita no espaço expandido. Esse novo conjunto de dados é separado por uma SVM linear. Na literatura o termo *Kernel* é utilizado para designar uma função que faz este mapeamento entre espaços. A Figura 11b ilustra a transformação ou expansão de espaço, transformando um problema não linearmente separável em linearmente separável, através da execução de uma função ϕ. Detalhes sobre o mapeamento entre espaços, bem como de diversas funções de mapeamento podem ser obtidos em [22].

4.1.3 Redes Neurais

Rede Neural Artificial é um modelo computacional composto por diversos elementos de processamento chamados de neurônios. Este modelo é amplamente utilizado no aprendizado de máquina e reconhecimento de padrões. Em sua forma mais tradicional, os neurônios são estruturados em camadas, dividindo-se em: uma camada de entrada, onde são recebidos os dados; uma ou mais camadas intermediárias e uma camada de saída. Uma rede neural com camadas intermediárias é denominada de *Multilayer Perceptron* (MLP).



(a) Estrutura de um neurônio.

(b) Estrutura de uma Rede Neural Artificial multicamadas (MLP).

Figura 12: Ilustrações de elementos de uma Rede Neural Artificial.

A saída de um neurônio de uma camada é uma das entradas para cada um dos neurônios da camada seguinte. A ligação entre dois neurônios é chamada de canal de comunicação e possui um peso associado.

Assim, o sinal de saída de um neurônio será multiplicado pelo respectivo peso do canal antes de servir de entrada para o neurônio seguinte correspondente. Pesos distintos podem ser atribuídos a cada canal de comunicação. A entrada de um neurônio será a soma ponderada de todos os sinais de entrada, conforme ilustrado na Figura 12a. O neurônio possui internamente uma função associada, denominada função de ativação. A função de ativação produzirá um sinal de saída caso o somatório dos sinais de entrada ultrapasse um determinado valor limiar.

A Figura 12b ilustra uma estrutura geral de Rede Neural Artificial multicamadas (MLP). Um algoritmo para treinar uma rede MLP é o *Backpropagation* [23]. Nesse algoritmo, os dados são apresentados na camada de entrada e os resultados são analisados na camada de saída. O sinal de saída é analisado de acordo com o padrão desejado, e caso diferente, contabiliza-se o erro. Assim, durante a fase de treinamento da rede neural, a partir da camada de saída, os valores de erros são propagados pelas camadas intermediárias até a camada de entrada, fazendo com que os pesos dos canais de comunicações sejam alterados conforme o nível de erro propagado. Esse processo de ajuste de pesos nos canais de comunicações é repetido até obter-se um resultado de saída desejado.

4.1.4 Conceitos de FAR e FRR

No processo de autenticação, um erro de Falsa Aceitação ocorre quando o sistema aceita a amostra testada como legítima, embora a amostra não pertença a um membro da classe genuína. Nesse caso, o sistema aceita um impostor como usuário legítimo. O erro de Falsa Rejeição ocorre quando o sistema rejeita uma amostra da classe genuína, negando acesso a um usuário legítimo. Neste trabalho o desempenho do sistema de autenticação pessoal é avaliado de acordo com as taxas de Falsa Aceitação (FAR - *False Accept Rate*) e Falsa Rejeição (FRR - *False Reject Rate*).

4.2 Extração de características com Transformada Wavelet e classificação com SVM

Nesta seção apresentamos como utilizamos os métodos a Transformada Wavelet e SVM para extrair as características discriminantes da imagem e efetuar a classificação individual. Neste trabalho, utilizamos a família de funções wavelets Daubechie e Coiflet, aplicando a Transformada Wavelet multi-nível. Os melhores resultados obtidos foram com 4 níveis de decomposição. Assim, todos os *coeficientes de aproximação* gerados formam um único *vetor de características* que identifica exclusivamente uma pessoa, sendo este vetor usado como dados de entrada para o classificador SVM.

4.2.1 Banco de dados

Para implementar e avaliar o classificador SVM, desenvolvemos um sistema utilizando a versão multi-classes da SVM disponível em Python[2], particularmente a versão multi-classes *one-versus-one*. Para o treinamento e os testes utilizamos as imagens da base de dados fornecidas por [12]. Essa base de dados possui um total de 500 imagens de 50 pessoas distintas, sendo 10 imagens por pessoa divididas em 5 imagens da mão direita e 5 imagens da mão esquerda. As imagens foram coletadas de forma diversificada entre homens e mulheres de 16 a 65 anos. Foram utilizadas 3 imagens para treinamento e 2 para testes, das 50 pessoas disponíveis, totalizando assim 150 imagens para treinamento e 100 amostras para testes, sendo todas as imagens utilizadas da mão esquerda.

4.2.2 Treinamento da SVM

Em termos de implementação da SVM, para o treinamento, o *vetor de características* é fornecido como parâmetro para a função *SVC*, juntamente com qual classe ele pertence, ou seja, de qual pessoa é a amostra. O principal parâmetro dessa função é qual função *kernel* utilizar. A função kernel que apresentou os melhores resultados foi a *rbf*. Os valores dos parâmetros *gamma* e C (parâmetro de penalidade de erro) dependem da função Wavelet utilizada e podem ser consultados na Tabela 3 na seção 5. Os demais parâmetros foram mantidos os valores *default*. Detalhes sobre a teoria envolvida e a implementação

[2]http://scikit-learn.org/ Acessado em 01/06/2018.

da função *SVC* podem ser consultados em [24]. Foram utilizadas 3 amostras para cada pessoa para o treinamento, totalizando 150 amostras.

4.2.3 Testes da SVM

Após o treinamento do modelo SVM, foram efetuados dois testes separadamente, sendo o primeiro para avaliar a taxa FRR o segundo para avaliar a taxa FAR. Para os testes, o sistema define dois parâmetros de entrada: a amostra de teste e a pessoa a quem alega-se ser a amostra. Como resultado, o sistema indica se a pessoa é aceita ou rejeitada.

Para avaliar a taxa FRR, o seguinte procedimento foi efetuado:

1. Dentre as 100 imagens disponíveis para testes, uma amostra foi aleatoriamente sorteada, sendo denominada a_m. Conhecemos a priori a pessoa p a quem pertence a amostra a_m. Assim, a_m e p são entradas do sistema;
2. A amostra a_m é apresentada ao classificador SVM que irá indicar pertencer à pessoa p_i;
3. Caso p seja igual a p_i, houve um acerto; caso contrário, um usuário legítimo será negado pelo sistema e, portanto, um erro de FRR.

Para a avaliação da FRR, são efetuados 100 testes: são apresentadas como entrada para o sistema cada uma das 100 amostras, sempre indicando p como entrada a pessoa a quem de fato pertence a amostra.

Para avaliar a taxa FAR, o seguinte procedimento foi efetuado:

1. Dentre as 100 imagens disponíveis para testes, uma amostra foi aleatoriamente sorteada, sendo denominada a_m. Conhecemos a priori a pessoa p a quem pertence a amostra a_m. Assim, a_m e p são entradas do sistema;
2. A amostra a_m é apresentada ao classificador SVM que indica pertencer à pessoa p_i;
3. Caso p seja diferente de p_i, houve um acerto, já que um impostor foi negado pelo sistema. Caso contrário, um usuário impostor foi aceito pelo sistema e, portanto, um erro de FAR.

Para a avaliação da FAR, para cada amostra sorteada são testadas 49 possibilidades: são fornecidos como entrada para o sistema a amostra sorteada e o valor de p como sendo cada uma das outras 49 pessoas que não sejam a pessoa a qual de fato a amostra pertence. Sendo 100 amostras disponíveis para teste, há um total de 4900 testes.

4.3 Extração de características com Transformada Wavelet e classificação com Rede Neural

Os procedimentos adotados para gerar o *vetor de características*, efetuar o treinamento e os testes no caso da Rede Neural foram os mesmos utilizados na SVM, descritos na seção 4.2. No caso da Rede Neural, a função Python utilizada foi a *MLPClassifier*, cujos detalhes teóricos e de implementação podem ser consultados em [25]. Os melhores valores encontrados para os parâmetros *alpha* e *hidden_layer_sizes* (total de camadas intermediárias para a Rede Neural multicamadas) da função *MLPClassifier* dependem da função Wavelet utilizada e podem ser consultados na Tabela 3 na seção 5. Os demais parâmetros foram mantidos os valores *default*.

4.4 Classificação por Registro de Imagens com base em imagem de veia

A técnica de Registro de Imagens é um processo de alinhamento de duas imagens no mesmo sistema de coordenadas espaciais através da aplicação das operações geométricas básicas de alterações de escala, rotação e translação, cujo objetivo é determinar se duas imagens pertencem a mesma classe. O processo de alinhamento possui duas etapas: na primeira etapa efetuam-se as operações geométricas enquanto a segunda etapa verifica-se o resultado do alinhamento.

4.4.1 Primeira etapa: operações geométricas para o alinhamento de imagens

Particularmente nesse projeto, para esta etapa, em termos de implementação, utilizamos o seguinte procedimento: sejam duas imagens I_1 e I_2 com as mesmas dimensões, que contêm pixels de veias e de fundo; seja uma imagem I_3 que contém apenas pixels de fundo. Efetua-se o mapeamento de I_1 e I_2 para I_3. A imagem I_1 é mapeada sem alterações, enquanto sobre a imagem I_2 efetua-se operações geométricas de alteração de escala e rotação antes do mapeamento e de translação durante o mapeamento. A dimensão da imagem I_3 deve ser ao menos 4 vezes maior do que a dimensão de I_2 para que não sejam perdidos pixels devido aos procedimento realizados sobre I_2.

O processo de mapeamento consiste em dois passos: primeiramente calcula-se o centroide da região de veias para cada imagem I_1 e I_2; posteriormente, de acordo com a distância de cada pixel de veia em relação ao centroide, calcula-se a coordenada correspondente em relação ao centro da imagem I_3. O cálculo do centroide ($\mathrm{X_c}$,$\mathrm{Y_c}$) é feito de acordo com as Equações 2 e 3, em que x e y são as coordenadas horizontal e vertical, respectivamente, de um pixel de veia da imagem e N é o número total de pixels que são pixels de veia na imagem.

$$X_c = \frac{\sum_{i=1}^{N} x_i}{N} \tag{2}$$

$$Y_c = \frac{\sum_{i=1}^{N} y_i}{N} \tag{3}$$

As novas coordenadas de cada pixel (x,y) são (x',y') e calculadas de acordo com as Equações 4 e 5, em que: ($\mathrm{X_c}$,$\mathrm{Y_c}$) são as coordenadas do centroide da imagem a ser mapeada ($\mathrm{X_{c_1}}$ e $\mathrm{X_{c_2}}$); ($\mathrm{X_{c_3}}$,$\mathrm{Y_{c_3}}$) são as coordenadas do ponto central da imagem I_3; Δ_x e Δ_y indicam valores de deslocamento no intervalo de $[-10, 10]$.

$$x' = |x - X_c| + X_{c_3} + \Delta_x \tag{4}$$

$$y' = |y - Y_c| + Y_{c_3} + \Delta_y \tag{5}$$

A escala da imagem I_2 é alterada nas seguintes proporções da dimensão original: 0,8; 0,9; 1,1 e 1,2. A rotação é alterada, em graus, de {-8, -6, -4, -2, 2, 4, 6, 8}. A alteração de escala e a rotação são implementadas em Java utilizando as funções *getScaleInstance* e *rotate*, respectivamente, da classe *AffineTransform*, cujos detalhes podem ser consultados em [26].

4.4.2 Segunda etapa: verificação do resultado do processo de alinhamento

Durante o alinhamento ocorre uma sobreposição quando um pixels de veia de I_1 e um pixel de veia de I_2 são mapeados para a mesma posição em I_3. A verificação do resultado do processo de alinhamento ocorre de acordo com o valor de três taxas de sobreposição, determinadas pelas Equações 6, 7 e 8, onde N_1 e N_2 correspondem ao número total de pixels de veias das imagens I_1 e I_2, respectivamente. Assim T_3 indica o total de pixels sobrepostos em relação a soma dos pixels de veias de ambas as imagens, enquanto T_1 e T_2 indicam o total de pixels sobrepostos em relação aos pixels de veias das imagens I_1 e I_2, respectivamente.

$$T_1 = \frac{(Número\ total\ de\ pixels\ sobrepostos)}{N_1} \tag{6}$$

$$T_2 = \frac{(Número\ total\ de\ pixels\ sobrepostos)}{N_2} \tag{7}$$

$$T_3 = \frac{(Número\ total\ de\ pixels\ sobrepostos)}{N_1 + N_2} \tag{8}$$

Após cada cálculo das taxas T_1, T_2 e T_3, é verificado se os percentuais estão de acordo com limiares pré-definidos, e caso estejam, o processo de Registro de Imagens é encerrado e o sistema aceita como legítimo o usuário. Se após todas as tentativas não forem atingidos os limiares, o sistema rejeita o

usuário. Após diversos testes, foi definido o limiar de 56% para cada uma das taxas T_1, $T2$ e T_3. A Figura 13 exemplifica o conceito de Registro de Imagem através do alinhamento entre duas imagens de veias de pessoas distintas, assumindo que quanto maior o número de pixels sobrepostos, maior o indicativo das duas amostras pertencerem ao mesmo indivíduo. A decisão da classificação depende de limiares de sobreposição conforme discutido.

(a) Imagem I_1. (b) Imagem I_2. (c) Sobreposição das imagens das pessoas p_1 e p_2.

Figura 13: Exemplo de amostras para o Registro de Imagens.

5 Estudo Experimental

Para o estudo experimental, um sistema completo com todas as etapas (Segmentação, Normalização, Extração de Características e Classificação) foi desenvolvido. Os métodos de Segmentação, Normalização e Registro de Imagens foram implementados em Java, enquanto os métodos de Transformada Wavelet, SVM e Rede Neural foram desenvolvidos em Python utilizando as ferramentas Scikit-learn[3] e PyWavelets[4]. Para os testes, foram utilizadas as imagens fornecidas pela base de dados [12], descrito na seção 4.2.1.

Primeiramente, todas as imagens do conjunto de dados foram segmentadas e normalizadas. Nenhuma imagem foi descartada. Posteriormente, foram feitas quatro avaliações distintas sobre a etapa de Extração de Características e Classificação das veias: SVM, Rede Neural e Registro de Imagens individualmente e uma avaliação híbrida envolvendo em conjunto as 3 técnicas mencionadas anteriormente. Para as técnicas de SVM e Rede Neural, conforme mencionado na seção 4.2, os dados consistem em um vetor de características formado pela concatenação dos *coeficientes de aproximação* resultantes da aplicação da Transformada Wavelet sobre a imagem normalizada em 4 níveis. Já a técnica de Registro de Imagens utiliza a própria imagem normalizada. As seções a seguir descrevem os resultados obtidos com cada técnica testada.

5.1 Resultados experimentais da *2D DWT* com SVM e Rede Neural

As Tabelas 1 e 2 apresentam os resultados obtidos considerando 2 tipos de funções wavelets. Foram testados de 1 a 6 níveis de decomposição para a *2D DWT*, sendo que com 4 níveis os melhores resultados foram obtidos. Para escolher os melhores modelos de classificadores, foram testados diversos valores para os parâmetros exigidos para as funções de classificação *SVC*, no caso do SVM, e *MLPClassifier*, para a Rede Neural. A Tabela 3 apresenta os valores testados, bem como aqueles que permitiram obter os melhores resultados.

Tabela 1: Taxa de acertos utilizando a função wavelet Coiflet 7 na Transformada Wavelet.

	Coiflet 7		
	FRR (%)	FAR (%)	Acertos (%)
SVM	8	0,16	91,84
Rede Neural	9	0,18	90,82

[3] `http://scikit-learn.org/` Acessado em 22/02/2019.
[4] `https://pypi.org/project/PyWavelets/` Acessado em 22/02/2019

Tabela 2: Taxa de acertos utilizando a função wavelet Daubechies 12 na Transformada Wavelet.

	Daubechies 12		
	FRR (%)	FAR (%)	Acertos (%)
SVM	7	0,14	92,86
Rede Neural	8	0,16	91,84

Tabela 3: Valores de parâmetros testados para os classificadores SVM e Rede Neural. Foram testados de 1 a 6 níveis de decomposição para a *2D DWT* para cada um dos valores de parâmetros.

Classificador		Melhores Parâmetros (com 4 níveis de decomposição da 2D DWT)	
		Coiflet7	Daubechies 12
SVM	$10^{-5} \leqslant gamma \leqslant 10^5$, $\forall$ *gamma múltiplo de* 10; Para cada valor de $gamma$, $10^{-5} \leqslant c \leqslant 10^5$, $\forall$ *c múltiplo de* 10.;	$gamma = 0.01$; $c = 0.0001$.	$gamma = 0.0001$; $c = 10$.
Rede Neural	$alpha = 0.1$ e $alpha = 0.01$; $10 \leqslant hidden\_layer\_sizes \leqslant 120$, $\forall$ $hidden\_layer\_sizes$ múltiplo de 10.	$alpha = 0.1$; $hidden\_layer\_sizes = 60$.	$alpha = 0.1$; $hidden\_layer\_sizes = 80$.

5.2 Resultados experimentais da classificação com Registro de Imagens

No caso do Registro de Imagens, 3 imagens do mesmo indivíduo são usadas como amostras de referência e 2 imagens para testes. Assim, no processo de avaliação, cada imagem de teste deve ser comparada com 3 imagens de referência do suposto indivíduo, com a decisão de aceitar ou rejeitar baseada no critério majoritário. Isto é, a autenticação positiva é declarada quando a imagem de teste é validada, com as taxas de sobreposições maiores do que o limiar estabelecido, com 2 ou mais imagens de referência.

Diversos valores de limiar de sobreposição foram avaliados. O valor de limiar de 56% de sobreposição oferece o melhor desempenho com a taxa de Falsa Rejeição (FRR) igual a 1% com uma taxa de Falsa Aceitação (FAR) nula (zero), sendo que estas taxas foram obtidas através de 100 e 4900 comparações, respectivamente.

5.3 Resultados experimentais da classificação com um Sistema Híbrido

No Sistema Híbrido, as três técnicas são consideradas conjuntamente. No processo de autenticação híbrido, o teste da amostra é feito simultaneamente pelos classificadores SVM e Rede Neural. Caso ambos os classificadores indiquem o mesmo resultado, esta será a resposta final do sistema híbrido (positivo ou negativo). Quando as respostas divergirem, o sistema híbrido executa a técnica de Registro de Imagens, que fornece a resposta definitiva do teste.

A aplicação do sistema híbrido tem como objetivo otimizar a performance do sistema de autenticação em termos de rapidez computacional fornecida pelas técnicas SVM e Rede Neural combinada com a alta taxa de acerto fornecida pelo Registro de Imagem. A melhor taxa de acertos obtida para o sistema híbrido foi utilizando a SVM com Wavelet Daubechies 12 e a Rede Neural com Coiflet 7. A Tabela 4 compara os desempenhos das técnicas de autenticação estudadas.

5.4 Resultados experimentais do desempenho computacional

Os testes foram realizados com uma configuração computacional simples (Windows 7, processador Intel i5 com 8Gb de memória RAM). O tempo médio da etapa de segmentação foi de aproximadamente 0,4 segundos. O tempo médio no caso do Registro de Imagens foi de 0,84 segundos para autenticar positivamente um usuário legítimo e 5,27 segundos para negar um usuário impostor. Note que o tempo médio

Tabela 4: Comparativo de desempenho entre os diversos métodos propostos.

	FRR (%)	FAR (%)	Acertos (%)
SVM com Daubechies 12	7	0,14	92,86
Rede Neural com Coiflet 7	9	0,18	90,82
Registro de Imagens	1	0	99,00
Sistema híbrido	1	0,02	98,98

para detectar um impostor é consideravelmente maior devido a necessidade de se testar e executar todas as possíveis alterações de escalas, rotações e translações antes da aplicação rejeitar a amostra, negando acesso ao indivíduo. Por outro lado, no caso de um usuário genuíno, normalmente, apenas pequenos ajustes e poucas operações são exigidos, o que indica que um usuário legítimo pode ser rapidamente validado.

Tabela 5: Desempenho em termos de processamento computacional.

	Coiflet 7	
	Tempo de treinamento (s)	Tempo de teste (s)
SVM	11,28	2,51
Rede Neural	64,73	2,32

Tabela 6: Desempenho em termos de processamento computacional.

	Daubechies 12	
	Tempo de treinamento (s)	Tempo de teste (s)
SVM	4,55	1,07
Rede Neural	23,32	1,01

Os tempos médios de treinamento e testes para os classificadores SVM e Rede Neural são apresentados nas Tabelas 5 e 6, respectivamente. Ao analisar estas tabelas, é possível verificar que os tempos de treinamento da SVM são bem inferiores aos exigidos para a Rede Neural Artificial, apesar de ambos possuírem os tempos de testes muito próximos e relativamente curtos. Nota-se que para autenticar um usuário legítimo, os tempos dos três métodos utilizados são muito próximos. No entanto, para negar um usuário impostor, o tempo no caso do Registro de Imagens é praticamente 5 vezes superior ao tempo necessário para a SVM e para a Rede Neural. Este fato motivou a proposta do sistema híbrido, integrando as técnicas estudadas para otimizar o ganho de desempenho geral no sistema de autenticação, uma vez que o Registro de Imagem será utilizado somente quando os classificadores SVM e Rede Neural tiverem resultados divergentes.

5.5 Comparativo entre os resultados obtidos e de métodos da literatura

As Tabelas 7 e 8 apresentam um comparativo de resultados entre os métodos propostos e os métodos frequentemente usados na literatura. O melhor desempenho apresentado por [27] é de 3% de FRR, e uma FAR igual a 0,0202% utilizando a mesma base de dados usada neste projeto. Isso significa que a nossa bordagem utilizando apenas o Registro de Imagens permite uma redução da taxa de Falsa Rejeição (FRR) de 3% para 1% e ao mesmo tempo um decremento da taxa de Falsa Aceitação (FAR) de 0,0202% para zero. Quando utilizado o sistema híbrido, também reduzimos a taxa de falsa rejeição para 1% mantendo praticamente a mesma taxa de falsa aceitação.

Em relação a outros métodos de reconhecimento do Estado da Arte que utilizam diferentes bases de dados, o método descrito possui um desempenho, em termos de taxas de erro, compatível com os melhores métodos da literatura. O trabalho de Wang *et al.* [28] também utiliza decomposição wavelet. No entanto,

os autores escolheram a função wavelet Haar e apresentam taxas de erros iguais (EER, quando FAR é igual a FRR) de 2,067%, sendo que para uma taxa FAR nula (zero), a taxa FRR é superior a 6%, enquanto para uma taxa FAR próxima de 0,02%, a taxa FRR é superior a 4%.

Tabela 7: Comparativo do desempenho entre o sistema híbrido proposto e métodos da literatura para o mesmo banco de dados de imagens.

	FRR (%)	FAR (%)
Mohamed Shahin *et al.* [27]	3	0,02
Registro de Imagens	1	0
Sistema híbrido	1	0,02

Tabela 8: Comparativo do desempenho entre o sistema híbrido proposto e métodos da literatura para banco de dados diferente.

	FRR (%)	FAR (%)
Wang *et al.* [28] - Wavelet Haar	> 6	0
Registro de Imagens	1	0
Sistema híbrido	1	0,02

As melhores performances obtidas nesse trabalho podem ser justificadas, principalmente, por dois fatores. O primeiro fator é a melhoria na etapa de localização das veias na imagem, através de uma melhor combinação e sequência de aplicação de filtros considerando as especificidades de uma imagem do dorso de uma das mãos. Além disso, houve uma melhor escolha de parâmetros dos filtros utilizados. Estas melhorias permitiram que a região adicional não pertencente às veias fosse eliminada e as regiões efetivamente relevantes preservadas.

O segundo fator deve-se ao método de extração de características. O método de Registro de Imagens apresentado engloba a maioria das possíveis variações de escala, de rotação e de translação da imagem das veias. Já o sistema híbrido. quando comparado com o Registro de Imagens, mantém baixas taxas de erro e simultaneamente reduz o tempo de processamento computacional necessário para negar um usuário impostor para 3,58 segundos, embora aumente o tempo para aceitar um usuário legítimo para 3,14 segundos. Ao compararmos o sistema híbrido com o uso dos classificadores SVM ou Rede Neural individualmente, nota-se uma expressiva queda nas taxas de erro, mantendo-se um tempo computacional reduzido.

6 Conclusões

Ao analisar os dados obtidos pode-se concluir que funções wavelets, das famílias Daubechies e Coiflet, integradas com SVM e Rede Neural Artificial multicamadas podem ser utilizadas no reconhecimento biométrico das veias das mãos com relativa eficiência em termos de taxas de acertos e alta eficiência em questões de tempo computacional. No entanto, ainda são necessários ajustes na etapa de Segmentação da imagem para que uma melhor localização e extração das regiões das veias não atrapalhe os métodos de aprendizado de máquina e deste modo, possam ser atingidas taxas de erros muito baixas obtidas com o método de Registro de Imagens e de métodos apresentados na literatura. Observa-se também que o método de Registro de Imagens é menos susceptível a não perfeita localização das veias na etapa de Segmentação, quando comparado aos métodos de aprendizado de máquina.

É possível concluir também que pode haver a utilização de um sistema híbrido para aproveitar a ótima eficiência em termos de taxas de reconhecimento do método de Registro de Imagens com o ótimo desempenho em termos de tempo de reconhecimento apresentado pelos métodos de aprendizado de máquina.

Referências

[1] S. Srivastava, S. Bhardwaj, S. Bhargava, *et al.*, "Fusion of palm-phalanges print with palmprint and dorsal hand vein," *Applied Soft Computing*, vol. 47, pp. 12–20, 2016.

[2] A. K. Jain, K. Nandakumar, and A. Ross, "50 years of biometric research: Accomplishments, challenges, and opportunities," *Pattern Recognition Letters*, vol. 79, pp. 80–105, 2016.

[3] R. Parkavi, K. C. Babu, and J. A. Kumar, "Multimodal biometrics for user authentication," in *Intelligent Systems and Control (ISCO), 2017 11th International Conference on*, pp. 501–505, IEEE, 2017.

[4] G. Arora, P. L. Pavani, R. Kohli, and V. Bibhu, "Multimodal biometrics for improvised security," in *Innovation and Challenges in Cyber Security (ICICCS-INBUSH), 2016 International Conference on*, pp. 1–5, IEEE, 2016.

[5] V. Mhaske and A. Patankar, "Multimodal biometrics by integrating fingerprint and palmprint for security," in *Computational Intelligence and Computing Research (ICCIC), 2013 IEEE International Conference on*, pp. 1–5, IEEE, 2013.

[6] M. K. Shahin, A. M. Badawi, and M. E. Rasmy, "Multimodal biometric system based on near-infrared dorsal hand geometry and fingerprints for single and whole hands," *World Academy of Science, Engineering and Technology*, vol. 56, pp. 1107–1122, 2011.

[7] P.-O. Ladoux, C. Rosenberger, and B. Dorizzi, "Palm vein verification system based on sift matching," in *International Conference on Biometrics*, pp. 1290–1298, Springer, 2009.

[8] R. B. Trabelsi, A. D. Masmoudi, and D. S. Masmoudi, "Hand vein recognition system with circular difference and statistical directional patterns based on an artificial neural network," *Multimedia Tools and Applications*, vol. 75, no. 2, pp. 687–707, 2016.

[9] Y. Wang, W. Xie, X. Yu, and L.-K. Shark, "An automatic physical access control system based on hand vein biometric identification," *IEEE Transactions on Consumer Electronics*, vol. 61, no. 3, pp. 320–327, 2015.

[10] R. B. Trabelsi, A. D. Masmoudi, and D. S. Masmoudi, "A novel biometric system based hand vein recognition," *Journal of Testing and Evaluation*, vol. 42, no. 4, pp. 809–818, 2013.

[11] C.-B. Hsu, S.-S. Hao, and J.-C. Lee, "Personal authentication through dorsal hand vein patterns," *Optical Engineering*, vol. 50, no. 8, p. 087201, 2011.

[12] A. M. Badawi, "Hand vein biometric verification prototype: A testing performance and patterns similarity.," *IPCV*, vol. 14, pp. 3–9, 2006.

[13] L. Wang, G. Leedham, and S.-Y. Cho, "Infrared imaging of hand vein patterns for biometric purposes," *IET computer vision*, vol. 1, no. 3, pp. 113–122, 2007.

[14] `https://imagej.nih.gov/ij/docs/guide/146-29.html`. Acessado em 10/05/2018.

[15] N. Phansalkar, S. More, A. Sabale, and M. Joshi, "Adaptive local thresholding for detection of nuclei in diversity stained cytology images," in *Communications and Signal Processing (ICCSP), 2011 International Conference on*, pp. 218–220, IEEE, 2011.

[16] T. Dócusse, A. Pereira, N. Marranghello, R. Guido, J. Furlani, P. Maturana, and R. Romano, "Aplicação da transformada wavelet para melhoria de visualização de microcalcificações em mamografias digitais."

[17] I. Daubechies, *Ten lectures on wavelets*, vol. 61. Siam, 1992.

[18] R. Singh, R. E. Vasquez, and R. Singh, "Comparison of daubechies, coiflet, and symlet for edge detection," in *Visual Information Processing VI*, vol. 3074, pp. 151–160, International Society for Optics and Photonics, 1997.

[19] `https://www.mathworks.com/help/wavelet/gs/introduction-to-the-wavelet-families.htmll`. Acessado em 10/05/2018.

[20] G. O. F. da Silva, G. A. Boggione, and L. M. G. Fonseca, "Fusao de imagens de sensoriamento remoto utilizando a transformada wavelet haar," *Simpósio Brasileiro de Sensoriamento Remoto*, vol. 12, pp. 6175–6182, 2007.

[21] V. N. Vapnik and A. Y. Chervonenkis, "On the uniform convergence of relative frequencies of events to their probabilities," in *Measures of complexity*, pp. 11–30, Springer, 2015.

[22] S. Khellat-kihel, N. Cardoso, J. Monteiro, M. Benyettou, *et al.*, "Finger vein recognition using gabor filter and support vector machine," in *Image Processing, Applications and Systems Conference (IPAS), 2014 First International*, pp. 1–6, IEEE, 2014.

[23] H. B. Demuth, M. H. Beale, O. De Jess, and M. T. Hagan, *Neural Network Design*. USA: Martin Hagan, 2nd ed., 2014.

[24] `scikit-learn.org/stable/modules/svm.html`. Acessado em 10/05/2018.

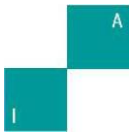
[25] `http://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html`. Acessado em 10/05/2018.

[26] `https://docs.oracle.com/javase/7/docs/api/java/awt/geom/AffineTransform.html`. Acessado em 10/05/2018.

[27] M. Shahin, A. Badawi, and M. Kamel, "Biometric authentication using fast correlation of near infrared hand vein patterns," *International Journal of Biological and Medical Sciences*, vol. 2, no. 3, pp. 141–148, 2007.

[28] Y.-D. Wang, Q.-Y. Yan, and K.-F. Li, "Hand vein recognition based on multi-scale lbp and wavelet," in *Wavelet Analysis and Pattern Recognition (ICWAPR), 2011 International Conference on*, pp. 214–218, IEEE, 2011.

Inteligencia Artificial 22(63), 121-134
doi: 10.4114/intartf.vol22iss63pp121-134



INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

Users Activity Gesture Recognition on Kinect Sensor Using Convolutional Neural Networks and FastDTW for Controlling Movements of a Mobile Robot

Miguel Pfitscher[1], Daniel Welfer[2], Evaristo José do Nascimento[3], Marco Antonio de Souza Leite Cuadros[4] and Daniel Fernando Tello Gamarra[5]
[1,2,3,5] Universidade Federal de Santa Maria (UFSM)
Santa Maria RS 97105-900, Brazil
miguel.pfitscher@gmail.com, welfer@gmail.com, evaristo.nascimento@ecomp.ufsm.br, daniel.gamarra@ufsm.br
[4] Instituto Federal do Espirito Santo
Serra, ES 29173-087, Brazil
marcoantonio@ifes.edu.br

Abstract In this paper, we use data from the Microsoft Kinect sensor that processes the captured image of a person using and extracting the joints information on every frame. Then, we propose the creation of an image derived from all the sequential frames of a gesture the movement, which facilitates training in a convolutional neural network. We trained a CNN using two strategies: combined training and individual training. The strategies were experimented in the convolutional neural network (CNN) using the MSRC-12 dataset, obtaining an accuracy rate of 86.67% in combined training and 90.78% of accuracy rate in the individual training. Then, the trained neural network was used to classify data obtained from Kinect with a person, obtaining an accuracy rate of 72.08% in combined training and 81.25% in individualized training. Finally, we use the system to send commands to a mobile robot in order to control it.

Keywords: Human gestures recognition, convolutional neural networks, Microsoft Kinect, MSRC-12 dataset, Mobile robot.

1 Introduction

Human physical activity recognition from skeleton data has attracted increasing attention in signal and image processing due to the variety of applications in which it could be used. The process of recognizing human actions implies processing a sequence of frames, looking one frame at a time. In order to help to solve this problem the emergence of a new generation of optimized frameworks has made Convolutional Neural Networks (CNN) popular and efficient for solving many problems in image recognition. In this paper, we present a way to use CNN to recognize gestures and human actions more efficiently in order to use them in real time. We also present a comparison of two forms of training for the CNN structure.

The Kinect sensor from Microsoft processes, collects and recognizes human joints. This processing facilitates the recognition of actions, the collected joints information is processed to facilitate gesture recognition. In the other hand, different databases have been created using the Kinect sensor in order to test different machine learning algorithms to recognize human gestures or activities for different purposes. Mo et al. [1] uses the dataset CAD-60, otherwise, in this article, we used the MSRC-12 dataset [2] for training and evaluation, the referred dataset has 6244 gesture instances of 12 actions. In order to make that all the gestures of the dataset could have fixed size of frames, a Fast-dynamic time warping (FastDTW) algorithm was used [3]. After training the neural

ISSN: 1137-3601 (print), 1988-3064 (on-line)

network, we used a Kinect sensor to collect data and, thus, it is possible to make that the employed network could recognize the gestures of a person and control a mobile robot.

The article is divided in seven sections, after a brief introduction, the second section explores some related works, the third section is a short summary about convolutional neural networks, the fourth section explains the Fast Dynamic time warping algorithm, the fifth section describes the experimental setup, and the sixth section depicts the method proposed in this work, the seventh section shows the obtained results, and finally, conclusions are summarized in the last section.

2 Related Work

In this section, we review some related works for skeleton-based action detection, and also papers that address the robot control problem using the Kinect sensor.

2.1 Deep Learning for Gesture Recognition

Mo et al. [1] used a computer vision model based on the deep learning algorithm to recognize human physical activity from the Microsoft Kinect. The skeleton data from the CAD-60 dataset was used for training and testing. First, the data is processed and prepared to be use in the deep learning algorithm. The CAD data set was labelled and grouped in small sets of 48 frames. Thus, it was possible to get approximately 3500 samples for training and evaluation. The model uses a convolutional neural network and a multilayer perceptron to classify twelve human activities. Moreover, the model structure has an input data size of 144x48 and the architecture of the network alternates three convolutional layers with three pooling layers. Finally, a multi-layer perceptron was used to generate the output. As a result, an accuracy of 81.8% is obtained on the validation set.

Hou et al. [4] proposed a structure using convolutional neural networks to recognize human actions, where three datasets were used for training and evaluation, namely: the MSRC-12 Kinect Gesture, the G3D and the UTD-MHAD datasets. Each action, in this skeleton dataset, was divided in three scatter plots, which creates spectral distributions of the joints. Each of the three spectral distributions creates an image which is used to train the neural network. Thus, they have three outputs scores that will be fused to get a score. So, the process keeps the basic spatial information, but the temporal information is lost [4]. As seen before, MSRC-12 Kinect Gesture dataset is a relatively large data set for action recognition. It contains 594 sequences with 12 gestures, 6244 gestures instances in total.

Jiang et al. [5] show a new approach for human action recognition. The main difference between this approach and a traditional method is that Jiang divides each action into some human segments as pre-processing and then use a k-nearest neighbours (KNN) classifier to classify each group. This approach can be better for handling complex motion variations. After segmentation, the data is classified in motion vectors and for online recognition the approach is similar with the addition of spatial temporal features. As a result, this model achieved at least 90% for every gesture using the MSRC-12 dataset.

Ke et al. [6], considered a CNN model for 3-D action recognition. The joints information is transformed into images, and feed to the deep learning network. Instead of treating the features of all frames as a time series, it is generated a discriminative and compact representation for action recognition to learn robust temporal information. Using NTU RGB+D Dataset, they got an accuracy of 75.9%.

Sharaf et al. [7] proposed the use of a support vector machine (SVM) to recognize human activities in real-time from 3D skeleton data. They used multivariate statistical methods for encoding the relationship between the extracted features. Besides, it was proposed a multi-scale action detector to process a sequence of frames at different scales.

Nguyen and Le [8], demonstrate that relevance vector machines (RVMs) can also achieve a good performance. They use the MSRC-12 data set, obtaining an accuracy of 93.6%. Pan and Li [9], adapt the dynamic time warping (DTW) algorithm to reduce the pathological alignment in resource extraction and model generation, caused by the traditional DTW. This algorithm was used for gesture recognition in the MSRC-12 data set. They get a hit rate of 75.1%.

Other approaches that have also been explored for the problem of gesture recognition are the Hidden Mark Models as in the work of Xia et al. [10] and Piyathilaka et al. [11]; multi kernel as in the work of Althloothi et al. [12]; or the use of hierarchical Recurrent Neural Networks (RNN) as in the work of Du [13]; Oyedotun et al. [14] used Convolutional Neural Networks to recognize static hand gesture.

2.2 Gesture Recognition with Kinect for the Control of Mobile Robots

We also review papers that address the robot control problem using the Kinect sensor. Among these works, we could mention the work developed by Borja et al. [15] that presented an algorithm which makes tracking of the hand using just images of depth captured with a Kinect sensor, it makes them invariant to light and skin colour conditions. To obtain the Kinect images, the OpenNi library was used. They use this tracking to control the speed and angular position of a mobile robot. The algorithm uses the previous frame for the segmentation of the current one, thus, a user must extend the hand in front of the camera and leave it for a while until the program recognizes the hand. Also, Borja, design a PID control for controlling the wheel speed of the robot.

Wang et al. [16] propose a simple method to control the movement of a robot using Kinect skeleton data. They used the coordinate of joints, which were obtained by Kinect SDK to make a gesture recognition of eleven simple gestures, and, then, control the movements of a Khepera III robot. For gesture recognition, they used a method based on angles of each gesture.

Zhao et al. [17] show a calling gesture recognition for taking order service of an elderly care robot. It was designed mainly for helping non-expert users like elderly to call a service robot. They used a skeleton based gesture recognition and, also, an Octree based gesture recognition. This method was implemented on a service robot developed for elderly care.

Limin et al. [18] used a Kinect camera to track the human skeleton points and capture human actions in real time. They used this information to send commands to the robot through the Bluetooth communication and make some movements as turning and forward.

Fadli et al. [19] proposed a system which allows us to instruct a robot to imitate what we are doing. They used a Kinect for capture information about its skeleton model and, then, a humanoid robot gets commands to move based on obtained angle data and imitate a body posture.

Other works applying the gesture recognition with the Kinect sensor with a mobile robot or manipulator robot have been use, such in the work of Kundu et al. [20], that propose an omnidirectional drive system to detect and pass a port; The work of AbdelKrim et al. [21], which show a reactive navigation system for internal environment using the Kinect sensor; Kameyama and Hidaka [22] which demonstrate an autonomous exploration algorithm of unknown environment for mapping; and Bellarbi et al. [23], that show a navigation method on a mobile robot and a human-robot interaction.

3 Theoretical Background

3.1 Convolutional Neural Networks (CNNs)

Deep learning is a subfield of machine learning, which uses hierarchical architectures to attempt to learn high levels of data abstraction without the need to detail how the algorithm will work or explicitly describe the characteristics of the data or the solution required. The deep learning technique is defined by a neural network architecture composed of several layers, and every layer has neurons (i.e. processing units) in their structure. Convolutional Neural Networks is one of the most popular categories of deep learning neural networks for being efficient in image recognition. As referred in Guo et al [24] It has demonstrated a high accuracy in diverse computer vision applications. This neural network mainly uses three types of layers, a convolutional layer, a pooling layer and a fully connected layer. Generally, the convolutional layers and the pooling layers are interleaved and the use of the full connected layers is limited to the end of the neural network. Figure 1 shows an example of the architecture of a deep convolutional network where we can see the input, pooling or convolutional layers, the fully connected layers and the output layer.

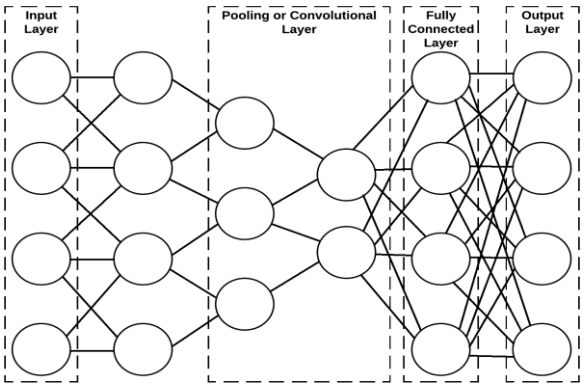


Figure 1: CNN architecture.

3.1.1 Convolutional Layer

This layer executes the operation of convolution of the feature maps of the input layer or those of previous layers using filters; each filter is used to detect a type of feature. This layer has a kernel with two dimensions that corresponds to the size of the convolution window, that is, we can size the kernel according to the size of the feature that we want to map. There are three main advantages of the convolution operation: 1) the weight sharing mechanism in the same feature map reduces the number of parameters 2) The local connectivity learns correlations among neighbouring pixels 3) The invariance to the location of the object as referred in [24].

3.1.2 Pooling Layer

The pooling layers are simple, but really useful to reduce the size of the feature maps, for this, it usually follows a convolutional layer. A pooling layer with the kernel size of 2×2, for example, reduces a path with 2×2 pixels to one pixel, choosing the maximum pixel or the average pixel for max-pooling or average-pooling respectively. it is easy to infer that the convergence of max-pooling is faster than that of average pooling which makes max-pooling to be used in many applications.

3.1.3 Fully Connected Layer

Fully-connected layers are widely used at the end of CNNs to reshape the previous layer (usually 2D) to a single dimension. The fully-connected layers contain about 90% of the total parameters in a CNN and are responsible for most of the training computational cost [25].

3.2 The Fast Dynamic Time Warping Algorithm (FastDTW)

Dynamic time warping (i.e. DTW) is a technique that finds the optimal alignment between two time series, if one time series may be “warped” non-linearly by stretching or shrinking it along its time axis as referred in [3]. Chu et al. [26] show that this similarity measurement can be computationally expensive. A traditional DTW has a time and spatial complexity of O (N^2) which make it considerably time consuming to execute for each gesture of a large dataset. Thus, we use an algorithm that is an approximation of DTW, the algorithm used was the FastDTW [3], an algorithm that performs well with a O (N) time and memory complexity. the algorithm uses three main treatments: Coarsening, which is responsible to decrease a time series into a smaller one that represents the same curve with fewer data points; Projection is used to find a minimum distance warp path at a lower resolution, and use it as an initial guess for a higher resolution warp path; Refinement, by locally adjusting the warp path, it will refine the warp path projected from a lower resolution.

The FastDTW algorithm uses a multilevel graph bisection algorithm, which will split a graph and get smaller graphs as possible. This multilevel approach is used to find an optimal solution for each small graph and makes

the algorithm linear in time and space, as shown by Stan [3]. Defining two time series as X and Y, equation (1) and (2) respectively, where x_k and y_k are each time series element:

$$X = x_1, x_2, \ldots, x_k \quad (1)$$

$$Y = y_1, y_2, \ldots, y_k \quad (2)$$

and constructing a wrap path as W given by equation (3):

$$W = w_1, w_2, \ldots, w_k \; max\,(|X|, |Y|) \leq K < |X| + |Y| \quad (3)$$

An optimal warp path is the minimum distance warp path, where the distance of a warp path W is given in equation (4), where Dist(w_{ki}, w_{kj}) is the distance between the two data point indexes (i.e. one from X and the other one from Y).

$$\text{Dist (W)} = \sum_{k=1}^{k=K} \text{Dist}(w_{ki}, w_{kj}) \quad (4)$$

4 Experimental Setup

4.1 The MSRC-12 Dataset

In this article, we used the MSRC-12 dataset [2] which has 6244 gesture instances of 12 actions. These actions can have a variable number of frames and that could make it hard to train in a fixed neural network with a fixed-size input. The dataset contains 594 sequences in 719359 frames that were collected from 30 different people performing 12 actions, providing in total 6244 gestures. We can see all the gestures obtained from this dataset in Figure 2.

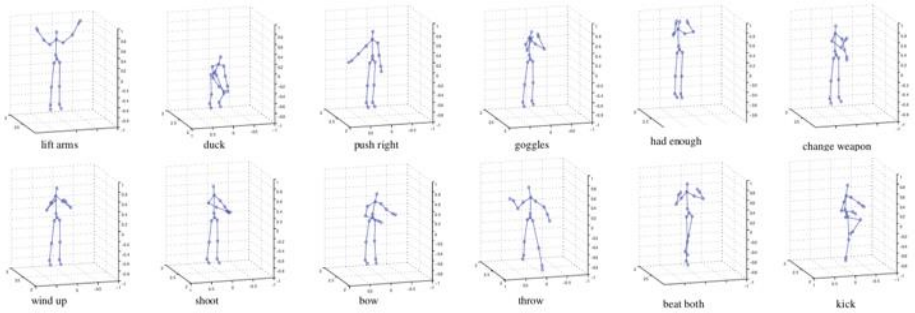


Figure 2: Gestures in MSRC-12 dataset.

4.2 Sensor Kinect

The Kinect sensor is a device manufactured by Microsoft with the intention of collecting data in RGBD for the purpose of using them in the recognition of the most diverse human movements. It was developed specifically for the Xbox videogame which brought greater interactivity and immersion into your games. Kinect consists of some sensors that make it possible to capture data from people, or objects mapping the environment in three dimensions. This mapping is due to the set of sensors: a camera, where the image data is obtained; An infrared (IR) projector, which works to obtain depth data from the site; An IR camera, which, along with the infrared projector, get the depth data being read by the sensor.

The depth sensor consists of an infrared emitter and an infrared camera. The infrared emitter emits an array of infrared rays in such a way that the environment is filled by these rays. The moment a ray encounters an object, it is reflected, and thus captured by the IR camera. With this, it is possible to calculate the distance of the object to Kinect, and, from there, to be able to map the environment in three dimensions. The sensor calculates the time that elapses between the time the beam was emitted and the time it was captured by the camera, so that it can approximate the distance of the object. With this data we can obtain the third dimension of the objects in the image, improving the performance of the recognition of human movements and actions. In Figure 3 we can see a Kinect sensor.

Figure 3: Kinect Sensor.

4.3 Mobile Robot

The mobile robot, which received the gesture recognition system, consists of a robot driven by wheels. It features two front wheels that perform the traction to exert the movement of the robot and a third independent rear wheel with the function of giving balance and stability to the robot besides aiding it in lateral movements. Two servo motors are used (i.e. one for each front wheel), which have built-in encoders of the EMG49 model. The control of the robot is performed by the controller module Arduino Mega 2560 that uses the ATMega 2560 controller from Atmel. Figure 4 shows us a picture of the Mobile Robot.

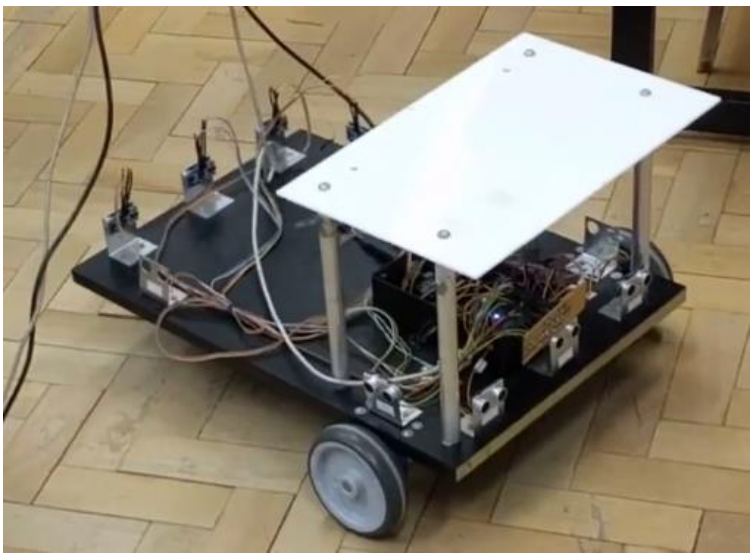


Figure 4: Mobile Robot.

4.4 Software Architecture

The proposed architecture was implemented in an Intel Core i7 3.06 GHz 4 Cores/16GB- DDR4/Ubuntu14.04-x64, and it is carried out by using python 3.6 to program in the TensorFlow framework. The joints positions were obtained using a Kinect sensor (first version). We use the Kinect SDK framework together with the PyKinect library for data collection and storage in the Python application. We send commands for the mobile robot using serial communication with the Pyserial library in python 2.7.

TensorFlow [27] is a platform for research and deployment of machine learning systems in many areas, such as computer vision and robotics. Its computational model is based on graphs of data flow with changeable state. It is widely used in research and is effective in applications involving machine learning.

The Kinect Software Development Kit (SDK) is a development framework that has come to assist developers in using Kinect, bringing integrated methods and functions not previously seen. This SDK was developed by Microsoft for Windows with the goal of enhancing the effects of Kinect with its automatic detection of joints, hands, people and objects. This has transformed human-computer interaction into a number of areas, such as education, the medical field, and transportation.

5 General System Architecture

The action detection method proposed in this work relies on a deep CNN architecture to classify an image where each row represents a frame and each column a specific joint. In order to implement our approach, it is necessary to pre-process the data by fixing its size (i.e. which is necessary for the CNN input).

The general system architecture is shown in Figure 5, and it could be summarizes in four main steps, in the first step, a Kinect sensor captures images of a person that is executing a gesture; in the second step, this image feeds a convolutional neural network, the neural network has been previously trained with data generated from the joints of the MSRC-12 data set, the convolutional network is written in python using the TensorFlow library, in the third step the network recognizes the gesture using the data collected by us. In the third step, we add a mobile robot to the system, the robot is controlled according to the results of the classification gesture algorithm. we were able to use data collected by us, also, each gesture is related to one robot movement, such as robot go forward, robot go backwards, etc. And finally, in the fourth step, we add a mobile robot to the system to control it according to the classified gesture. we were able to use data collected by us to the system to obtain tests with people in a controlled environment. We can see, in the Figure 5, the flow of the system.

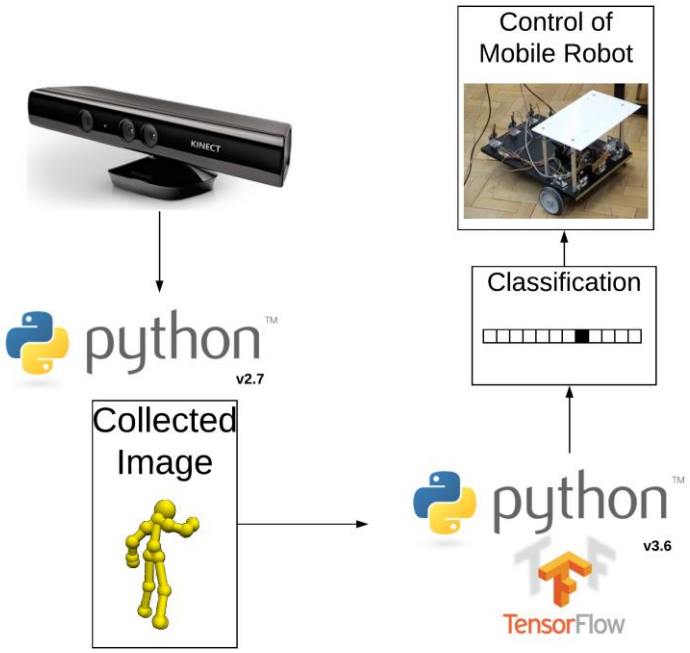


Figure 5: Flow of the gesture recognition system.

A gesture can have a different number of frames because it could be seen as an aperiodic signal, so in order to make that all the gestures could have a fixed size we used the FastDTW algorithm [2], so we created a fixed size image for the neural network input of 667x80 resolution, where 667 correspond to the number of frames and 80 represents that number of frames for every frame. The number of frames of 80 for our 20 joints is calculated using the 3D Cartesian coordinates of the 20 joints that gives 60 plus a separation of 20 between each joint that is added, so we have 60. Fig. 6 shows an example of the matrix that we created for every gesture that we used as an input to the convolutional network.

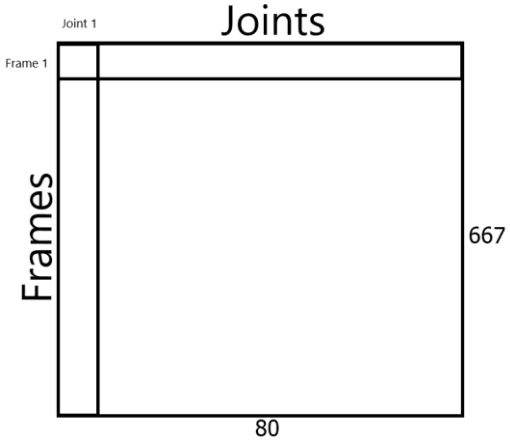


Figure 6: Input to the convolutional network.

6 Convolutional Neural Network Strategies: Combined and Individual Training

We tested two strategies for gesture recognition. In the first one, combined training, all the gestures were trained with one neural network. Then, on the second strategy, called individual training, we trained 12 neural networks, one for each gesture.

The neural network has six hidden layers, three convolutional layers with 3x3 kernel with ReLu activation functions, two pooling layers with 3x3 kernel and three strides, and one pooling layer with 2x2 kernel and two strides. In the end, it has a dense layer with 9472 units and a dropout operation with a dropout rate of 0.4 to prevent overfitting, as we can see in the Figure 7.

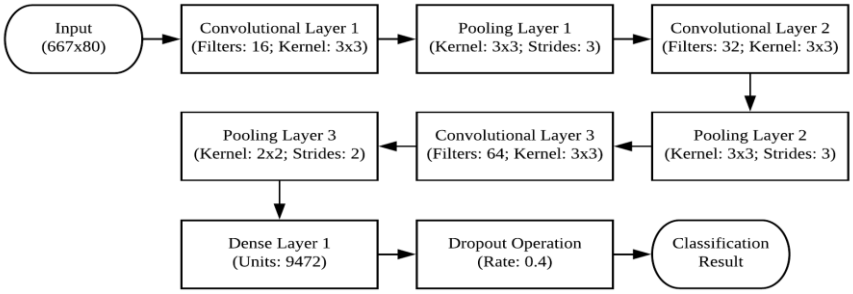


Figure 7: CNN Structure.

7 Results

In this section, we present the results regarding the experiments of training the model with the 12 gestures and the experiments with the model using the individual and combined training of the gestures in the MSRC-12 data set. In addition, we present results from data collected experimentally by us using a Kinect sensor applying also both strategies, and the results of using all the proposed architecture acquire a person gestures, classifying it and controlling a mobile robot all at the same time.

7.1 Results in the MSRC-12 data set

In a fairly large dataset, the number of frames for each action is unlikely to be repeated even when the gesture is executed by the same person, we can see the variation of that number. Using traditional convolutional networks in images we expect them all to have a fixed size and if this does not happen, we can resize them without major losses.

These actions can have a variable number of frames and that could make it hard to train the data in a fixed neural

network with a fixed-size input. Considering this problem, as pre-processing, it was necessary to use an algorithm to have all gestures with a fixed size of frames, therefore, we use the FastDTW [2] algorithm.

So, the Microsoft Research Cambridge-12 database (MSRC-12) was pre-processed using the FastDTW algorithm [3] to create images of 667x80 where 667 is the number of frames normalized by the algorithm of dynamic time warping and 80 represents the number of variables per frame. The dataset used has the positions collected from the 20 joints where each joint has three coordinates (x, y z), that is, 20 x 3 = 60 variables. Also, a separation between each joint is added, thus, we have 60 + 20 = 80 variables. So, we've shaped the network to have a fixed input of that size.

we present the results regarding the use of the MSRC-12 data set in training and validation using the proposed CNN model. We will also present the results regarding the training of the model trained with the 12 gestures in a convolutional network and the results of the individual training of the actions (12 trainings).

7.1.1 Combined Training

The dataset used has 6244 gesture samples, which we can use for training and validation. In this training, after the creation of the images in the pre-processing, we separated 33.33% of the gestures for validation and left 66.66% of the data for the training. We obtained a total of 4162 images of 667x80 for training and 2082 images of the same size for evaluation. Considering that the number of samples is relatively large for gesture recognition and with the number of 12 actions for recognition, a CNN model takes several steps to converge and even more with the use of the dropout operation. Thus, the training was performed with small batches of 10 samples and 60000 steps obtaining an accuracy of 86,67% for the validation data. Figure 8 shows the accuracy of the combined training.

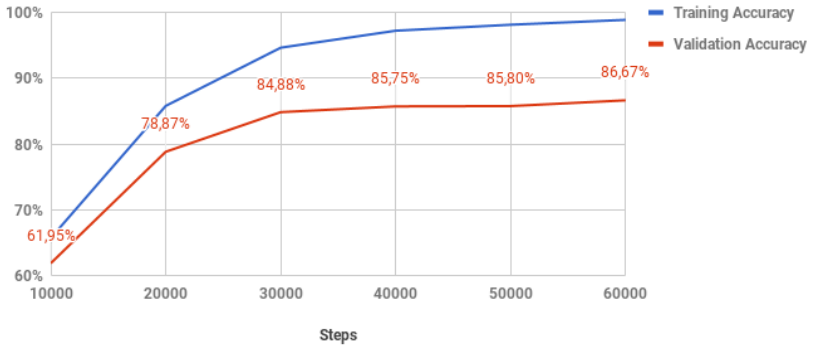


Figure 8: Combined training accuracy.

7.1.2 Individual Training

In the individual training, we used the same model for the individual training of each of the 12 gestures, that is, we trained 12 separate networks. The training was done using the number of samples of a gesture plus the same number of samples from other randomly selected gestures. Thus, each individual network was trained with approximately 1000 samples. Figure 9 shows the validation and training accuracy of one of the individual training networks. Besides, we can notice in the Figure 10 the accuracy of each individual training, with 30000 steps, together with the general accuracy obtained by the weighting of the individual accuracies.

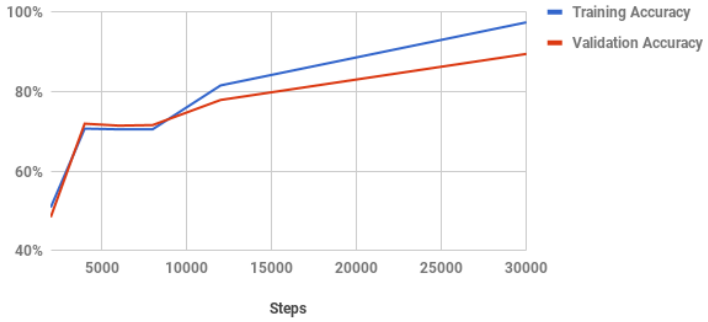


Figure 9: Individual Accuracy of Wind it up' gesture.

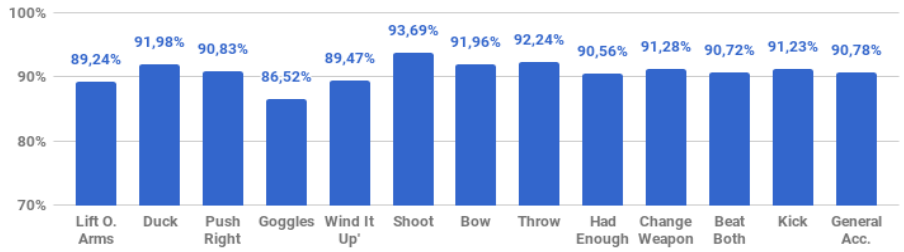


Figure 10: Individual training accuracy of all the Gestures of the evaluation set.

7.2 Experimental Results with Gestures Captured from a Person with the Kinect

After the training and verification of the neural network we passed the execution and classification using real data collected from Microsoft Kinect (v1) with a person, and, we have also used the two strategies, named combined and individual training, for classification. As previously explained, it was necessary to make the communication between the process that collects the data (Python 2.7) and the classification process (Python 3.6), which made classification possible. We used 20 samples of each movement and in each training format of the network, that is, 480 samples of gestures were collected by us.

7.2.1.1 Classification using combined training

Figure 11 shows the results of the accuracy of the system obtained after the training processes and using data captured from the Kinect with a person realizing a gesture. We can observe in the figure that in some movements, such as 'duck' and 'throw', the neural network performs very well, but in others such as 'goggles' and 'wind it up' we find that the performance was well below expectations. We believe this is due to the fact that it is difficult to execute some movements and they are easily confused both by the trained network and by humans, we take the example of the 'googles' movement that the trained network has confused several times with the 'had enough' movement.

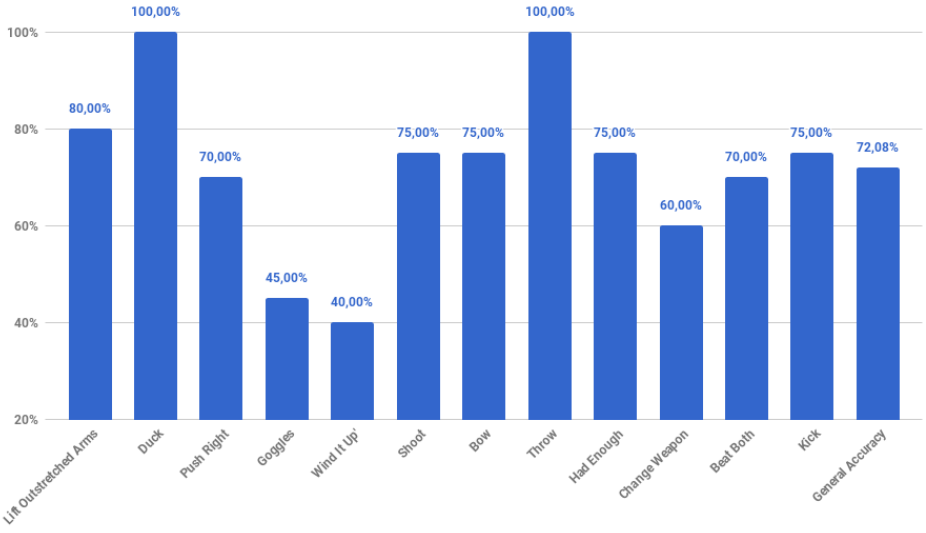


Figure 11: Combined training accuracy on evaluation set using data collected with the Kinect sensor and a person.

7.2.1.2 Classification using individual training

The experiments were repeated using the strategy of the individual training, but this a person will execute a determined movement instead of using the information of the MSRC-12 data set. Figure 12 shows the results accuracy of the individual training accuracy of the network obtained with this strategy, we can observe in the figure, that the accuracy was satisfactory and closer to the performance seen using the evaluation set of MSRC-12.

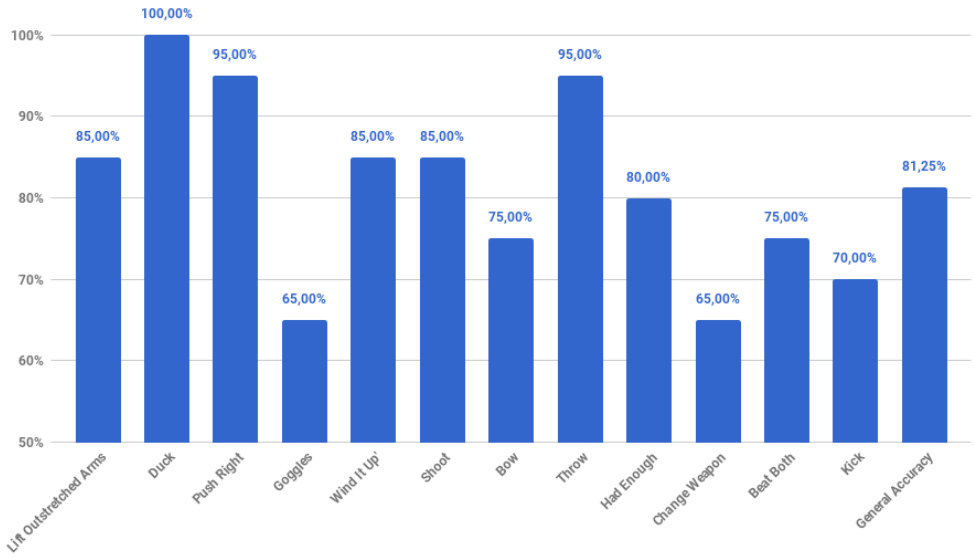


Figure 12: Individual training accuracy on evaluation set using data collected with the Kinect sensor and a person.

7.3 Results Controlling the Mobile Robot

Finally, after the two set of experiments using both training strategies in each of them, we decided that besides including a person it could be possible to include a robot in the control loop, closing the architecture proposed that is shown in Figure 5. The results pointed out that controlling the mobile robot did not cause problems for the execution of the system. Thus, we obtained the same results using the neural network for the gesture classification. Every gesture is related to a predefined robot movement. So, we reached the goal of using the system to control a mobile robot. We can see, in the Figure 13 a sequence of frames from the movement throw, followed for the sequence of movements deployed for the mobile robot in 4 different moments of the experiments.

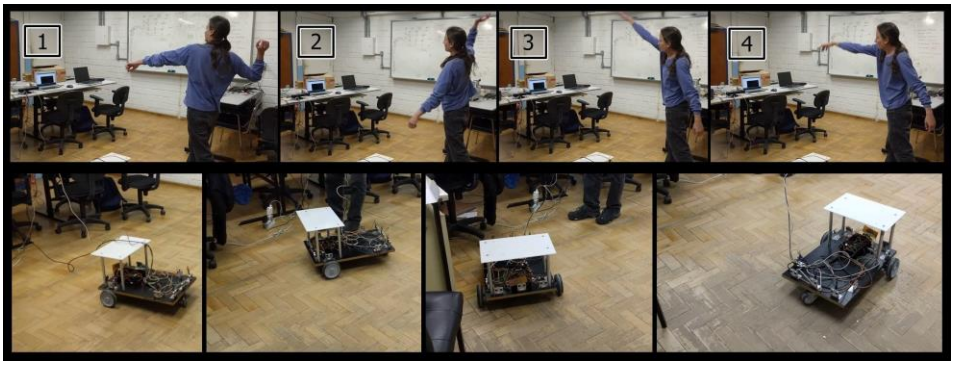


Figure 13: Sequence frames from the movement throw.

After the classification of the movement which was made, according to the gesture predicted by the convolutional network, a command is sent to control the robot. For testing, we used four gestures trained to command the robot: Lift arms, duck, throw and shoot. Any other gesture (provided by the network) is sent as the stop motion command. From these commands, the robot makes one of four simple coded movements: Square anti-clockwise (lift arms); Square clockwise (duck); Counter clockwise circle (shoot); Clockwise circle (throw). Using the robot odometry, it is possible to obtain the position and orientation of the robot. Table 1 shows the cartesian positions captured from square movement counter clockwise developed by the robot. Also, Figure 14 shows the trajectory described by the robot following the predefined trajectory. The performance of the whole system can be seen in this section. Fig. 13 shows the sequence of movements of the robot for the throw gesture, in this case the robot has to make a square following a counter clockwise rotation and the positions of the robot in the four-square corners measured with the odometry of the robot are shown in Table 1, the coordinates captured demonstrates that the robot executed the predefined movement associated to the gesture throw.

Table 1: Positions captured from square movement counter clockwise.

	x coordinate	y coordinate
1	20,24885	0,00000
2	20,22809	-0,21179
3	20,42756	-0,22335
4	20,43920	-0,04351

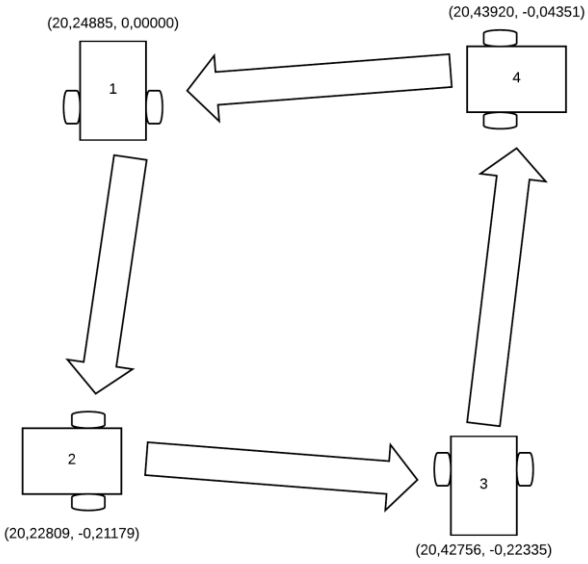


Figure 14: Square movement counter clockwise.

7.4 Results Analysis

In relation with the experiments realizes in the MSRC-12 data set, we can see that individual training achieved an accuracy of 4 percent higher than the combined training. This difference could be explained mainly by the similarity of some images (actions) which makes the distinction in a single network more difficult. On the other

hand, in small networks, this fact could have a minor influence because each network is responsible for recognizing only one gesture. Training the individual networks brings a gain in accuracy, as discussed earlier, but training and execute time is greatly increased and could compromise their use in applications where is required a fast response, for example, a real time application.

Neural network structures based on CNN do not recognize rotated images (if there is no image with the same rotation in the training set). Wu [28] demonstrates that convolutional networks only become efficient and capable in recognizing rotated images if rotation layers as rotated-pooling or flip-rotated-pooling convolution are added to the convolutional model. This CNN characteristic could be a problem for many applications, such as image recognition in general. On the other hand, in our case, this characteristic can be seen as beneficial, because with the movement of articulations other than the originals there is a significant change of the gesture and cannot be confused with the trained and correct action.

We would like to drive the attention now to the classification experiments using data collected by us with persons, we have seen a good performance in some gestures, but below-expected performance in others. We believe that it is a problem caused because some gestures have some similar characteristics. The same results were obtained using the neural network for the gesture classification controlling the robot. The robot was controlled to make simple movements, which, as seen earlier, brought satisfactory results.

Also, we would like to remark that our application shows that it is possible to integrate machine learning methods with robotics as can be explored in other works [29], [30].

8 Conclusion

CNN models are very popular in computer vision problems, but less exploited in other areas. The answer obtained in this paper by a model of a convolutional network demonstrates that it is possible and efficient its use in the recognition of gestures from joints obtained from the Kinect sensor. Furthermore, we proposed and experimented with strategies for the training of the CNN, named the combining and individual training, It was possible to observe that training of 12 smaller networks increased the training time substantially, but it is obtained a higher accuracy that makes this approach more useful and avoid a bottleneck with the training time, it is also possible to parallelize the individual trainings since they are independent of each other. So, these strategies, combined and individual training, is another contribution of the paper. In addition, we collect data from a Kinect sensor to use the system and, thus, get results closer to reality. The use of the Kinect sensor was satisfactory, although, the correctness rate for some gestures did not follow the results seen in the database. The use of the system to control a mobile robot was possible and its performance satisfactory. Some future works could focus on the use of a Recurrent Neural Network instead of a CNN.

9 References

[1] Mo, L., Li, F., Zhu, Y., Huang, A.: Human physical activity recognition based on computer vision with deep learning model. IEEE Int. Instr. and Meas. Technology Conf. Proceedings, Taipei, pp. 1-6 (2016).

[2] MSRC-12 dataset: https://www.microsoft.com/en-us/download/details.aspx?id=52283, last accessed 2018/08/21.

[3] Salvador, S., Chan, P.: FastDTW: Toward accurate dynamic time warping in linear time and space. Intelligent Data Analysis 11.5, pp. 561-580 (2007).

[4] Hou, Y., Li, Z., Wang, P., Li, W.: Skeleton optical spectra based action recognition using convolutional neural networks. IEEE Transactions on Circuits and Systems for Video Technology, vol. 28, no. 3, pp. 807-811 (2018).

[5] Jiang, X., Zhong, F., Peng, Q., Qin, X.: Online robust action recognition based on a hierarchical model. 30: 1021. https://doi.org/10.1007/s00371-014-0923-8 (2014).

[6] Ke, Q., An, S., Bennamoun, M., Sohel, F., Boussaid, F.: SkeletonNet: Mining Deep Part Features for 3-D Action Recognition. In: IEEE Signal Processing Letters, vol. 24, no. 6, pp. 731-735 (2017).

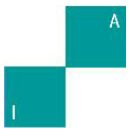
[7] Sharaf, A., Torki, M., Hussein, M. E., El-Saban, M.: Real-time multi-scale action detection from 3D skeleton data. IEEE Winter Conf. on Applications of Computer Vision, Waikoloa, HI, pp. 998-1005 (2015).

[8] Nguyen, D., Le, H.: Kinect Gesture Recognition: SVM vs. RVM. Seventh International Conference on Knowledge and Systems Engineering (KSE), Ho Chi Minh City, pp. 395-400 (2015).

[9] Pan, H., Li, J.: Online human action recognition based on improved dynamic time warping. IEEE International Conference on Big Data Analysis (ICBDA), Hangzhou, pp. 1-5 (2016).

[10] Xia, L., Chen, C. C., Aggarwal, J. K.: View invariant human action recognition using histograms of 3D joints. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, Providence, RI, pp. 20-27 (2012).
[11] Piyathilaka, L., Kodagoda, S.: Gaussian mixture based HMM for human daily activity recognition using 3D skeleton features. IEEE 8th Conference on Industrial Electronics and Applications (ICIEA), Melbourne, VIC, pp. 567-572 (2013).
[12] Althloothi, S., M. Mahoor, H., Zhang, X., Voyles, R. M.: Human activity recognition using multi-features and multiple kernel learning. Pattern Recognition, V. 47, Issue 5, pp. 1800-1812 (2014).
[13] Du, Y., Fu, Y., Wang, L.: Representation Learning of Temporal Dynamics for Skeleton-Based Action Recognition. In: IEEE Transactions on Image Processing, vol. 25, no. 7, pp. 3010-3022 (2016).
[14] Oyedotun, O.K., Khashman, A.: Deep learning in vision-based static hand gesture recognition. Neural Comput and Applic, pp. 28 (2017).
[15] Borja, J. A. T., Alzate, E. B., Lizarazo, D. L. M.: Motion control of a mobile robot using kinect sensor. IEEE 3rd Colombian Conference on Automatic Control (CCAC), Cartagena, pp. 1-6 (2017).
[16] Wang, Y., Song, G., Qiao, G., Zhang, Y., Zhang, J., Wang, W.: Wheeled robot control based on gesture recognition using the Kinect sensor. IEEE International Conference on Robotics and Biomimetics (ROBIO), Shenzhen, pp. 378-383 (2013).
[17] Zhao, X., Naguib, A. M., Lee, S.: Kinect based calling gesture recognition for taking order service of elderly care robot. The 23rd IEEE International Symposium on Robot and Human Interactive Communication, Edinburgh, pp. 525-530 (2014).
[18] Limin, M., Peiyi, Z.: The medical service robot interaction based on kinect. IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS), Srivilliputhur, pp. 1-7 (2017).
[19] Fadli, H., Machbub, C., Hidayat, E.: Human gesture imitation on NAO humanoid robot using kinect based on inverse kinematics method. International Conference on Advanced Mechatronics, Intelligent Manufacture, and Industrial Automation (ICAMIMIA), Surabaya, pp. 116-120 (2017).
[20] Kundu, A. S., Mazumder, O., Chattaraj, R., Bhaumik, S.: Door negotiation of a omni robot platform using depth map based navigation in dynamic environment. Seventh International Conference on Contemporary Computing (IC3), Noida, pp. 176-181 (2014).
[21] Abdelkrim, N., Issam, K., Lyes, K., Khaoula, C.: Fuzzy logic controllers for Mobile robot navigation in unknown environment using Kinect sensor. IWSSIP Proceedings, Dubrovnik, pp. 75-78 (2014).
[22] Kameyama, N., Hidaka, K.: A sensor-based exploration algorithm for autonomous map generation on mobile robot using kinect. 11th Asian Control Conference (ASCC), Gold Coast, QLD, pp. 459-464 (2017)
[23] Bellarbi, A., Kahlouche, S., Achour, N., Ouadah, N.: A social planning and navigation for tour-guide robot in human environment. 8th International Conference on Modelling, Identification and Control (ICMIC), Algiers, pp. 622-627 (2016)..
[24] Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S., Lew, M.l S.: Deep learning for visual understanding: A review. Neurocomputing. Recent Developments on Deep Big Vision, pp. 187:27 – 48 (2016).
[25] Zeiler, M.: Hierarchical Convolutional Deep Learning in Computer Vision. Ph.D. thesis, New York University (2014).
[26] Chu, S., Keogh, E., Hart, D., Pazzani, M.: Iterative Deepening Dynamic Time Warping for Time Series. Proceedings SIAM International Conference on Data Mining, pp. 195-212 (2002).
[27] Tensorflow: Tensorflow Framework documentation. https://www.tensorflow.org, last accessed 2018/10/14.
[28] Wu, F., Hu, P., Kong, D.: Flip-Rotate-Pooling Convolution and Split Dropout on Convolution Neural Networks for Image Classification. arXiv preprint arXiv:1507.08754v1 (2015).
[29] Pinpin, K., Gamarra, D.F.T., Johansson, R., Laschi, C., Dario, P.: Utilizing Gaze Behavior for Inferring Task Transitions Using Abstract Hidden Markov Models. Inteligencia Artificial, v. 19, pp. 1-16, (2016).
[30] Silva, R.M., Cuadros, M.A.S.L., Gamarra, D.F.T.: Comparison of a Backsteping and a Fuzzy Controller for Tracking a Trajectory with a Mobile Robot. 18th International Conference on Intelligent Systems Design and Applications (ISDA), (2018).

Inteligencia Artificial, 22(63) (2019), 135-149
doi: 10.4114/intartif.vol22iss63pp135-149



INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

The importance of context-dependent learning in negotiation agents

Dan Ezequiel Kröhling[1], Omar Chiotti[1], and Ernesto Martínez[1]

[1]INGAR (CONICET/UTN) - Instituto de Desarrollo y Diseño.
Avellaneda 3657, S3002GJC Santa Fe, República Argentina.

Abstract Automated negotiation between artificial agents is essential to deploy Cognitive Computing and Internet of Things. In this sense, the behavior of those negotiation agents depend significantly on the influence of environmental variables, facts, and events, which made up the context of the negotiation game. This context affects not only a given agent preferences and strategies, but also those of his opponents. In spite of this, the existing literature on automated negotiation is scarce about how to properly account for the effect of the context in learning and evolving strategies. In this paper, a novel context-driven representation of the negotiation game is introduced. Also, a simple negotiation agent that queries available information from context variables, internally models them, and learns how to take advantage of this knowledge by playing against himself using reinforcement learning is proposed. Through a set of episodes of our context-aware agent against other negotiation agents in the existing literature, it is shown that it makes no sense to negotiate without taking relevant context variables into account. Our context-aware negotiation agent has been implemented in the GENIUS tool. Results obtained are significant and quite revealing about the role of self-play in learning to negotiate.

Keywords: Agents, Automated Negotiation, Negotiation Intelligence, Internet of Things, Reinforcement Learning.

1 Introduction

Artificial intelligence has definitely entered the mainstream of business innovation [1, 22]. Huge progresses in the existing technology [21], new theories of intelligence [17, 24, 29], and the increasingly refined comprehension of biological brains of humans and animals [13, 28], have lead to the development of new mathematical models that tackle the problem of creating the so-called intelligent agents in our daily life. Some examples of these are [8, 11, 26].

A topic that has gained attention among AI experts in recent years is the implementation of intelligent negotiation agents. First of all, there is a human reason behind this: people is usually reluctant to get involved in negotiations. As Fatima et al. [10], taken from [3], put it: "When engaged in complex negotiations, people become tired, confused, and emotional, making naive, inconsistent, and rash decisions." This is a human condition: we could see it in our everyday life [12]. A second, yet more technological reason behind the creation of negotiation agents is to accomplish the promise of novel technologies such as Internet of Things [1] and Cognitive Computing [22].

Although great efforts are being made to automatize negotiations between artificial agents, some doubts remain about the design aspects of such artificial entities. In this sense, a number of approaches

ISSN: 1137-3601 (print), 1988-3064 (on-line)

to address this problem have been proposed [23, 7, 16, 31]. We consider the works of Fatima et al. [10] and Baarslag [4] a great compendium of the state of the art in this research area. In [10], differences between single and multiple issues, bilateral and multilateral negotiations are shown, with discrete and continue issues' values. Machine-machine and man-machine negotiations are addressed, together with different negotiation protocols, domains, and the selection of proper negotiation agendas. In [4], offering, accepting, and opponent modelling techniques are presented, alongside a framework to develop negotiation agents.

In spite of the progresses made so far, there is an issue in automated negotiation that, from our point of view, has not been properly accounted for in the design of negotiation agents. This is the importance of the context in negotiations, or the existence of key information that could provide a competitive advantage when used to predict and model the opponent by associative learning, including its strategy and perceptions/assumptions from the context. As an example of this, an agent could be i) informed about issues in the environment beyond the opponent himself, and ii) hypothesize about which information the opponent is actually using to make his predictions and learn. The importance of learning in automated negotiations has been previously recognized [32, 33], yet the context-awareness capability is not widely seen as a key issue [2, 19, 30]. Most of previous works circumscribe the agent learning to ad-hoc decision-making policies that may not capture appropriately the influence of the context on the outcome of a negotiation episode.

In order to clarify our point of view, let us discuss briefly some related work. In [2, 30], the context is represented through a fixed model, but any new variable that could change the course of the negotiation is discarded. Another example is given in [19]. Although a novel approach to model the utility functions of the agents is proposed, these functions are still prefixed and they do not take into account changes in relevant contextual variables. In [14], the existence of a number of opponents as outside options is considered, but context remains the same for different negotiations, when the emergence of new opponents will clearly change that context. In [9], agents make offers and take decisions (accept/reject) based in a simple negotiation domain, summarized in reservation value and discount factor, but agents do not consider the dynamics of the context (or domain) to compute this variables. In [20], context obtains a clearer attention from the authors. Still, learning capabilities are used to model and select a forecast method to some contextual variables, but there is no mention to the strategy or policy used by an agent is going to act or concede during a particular negotiation given the context he is in. Moreover, we consider that the use of fixed learning parameters for each context selected after a previous analysis of those same contexts makes this approach hardly extendable. Finally, in the GENIUS negotiation tool [15] (General Environment for Negotiation with Intelligent multi-purpose Usage Simulation), actually one of the most used negotiation simulators and the one we also choose to run our own computational experiments, the negotiation deadline is even of public (common) knowledge, when that is certainly a decision that agents should be able to make on their own, based on their strategies and the information available to them.

Based on the considerations above, the main hypothesis in this work is that negotiation agents that learn to use in their benefit relevant contextual variables to select and evolve their strategies will reap more benefits than those agents that concentrate only on learning their opponents strategies independently of the context. Accordingly, we aim to create a negotiation setting that includes both environmental variables and context-aware agents. We design a novel context-driven negotiation environment and insert therein a learning agent that takes the context into account. This agent will use the available information alongside with reinforcement learning [25, 27] and Self-Play to generate specific knowledge about the context and select the proper actions as negotiations proceed. We will then exploit this knowledge to interact with other negotiation agents defined in the existing literature, agents that do not take into account contextual variables and yet have won the ANAC (Automated Negotiating Agents Competition) in the last years. We use the GENIUS tool [15] to run the simulated negotiation episodes and obtain significant results.

This paper is organized as follows. Firstly, a conceptual representation of the negotiation environment for context-aware negotiation agents is discussed. Next, we present "Strawberry", our own context-aware negotiation agent. We define its internal design and its Self-Play learning strategy alongside the "Oracle", a conceptual entity that is going to answer the information queries made to gather contextual relevant information by our agent. Later on, negotiation experiments are designed and run to generate results that can test our main hypothesis.

2 Negotiation environment

In this section, the main structure and components that constitute the negotiation environment are presented. As in most related works, a group of agents that agree to negotiate over certain issues are considered. In this work, the focus is on bilateral negotiations between two agents negotiating over one single issue with discrete values using a discrete time line. The alternating offers protocol which is, according to [10], the most influential protocol of all, is used throughout.

A colloquial definition of the negotiation context can be found in [20]. According to Rodriguez-Fernandez et al., the negotiation context refers to characteristics or circumstances under which the negotiation process occurs. We extend this definition formally, though. The negotiation environment, from the perspective of a single agent, is divided up in two abstract spaces: the agent's private information and the context (see Fig. 1). The agent's private information is composed by all his internal or private variables, those that other agents can not see but could attempt to model observing the actions the agent performs. The context, on the other hand, is composed by all the external or public variables, those that every agent would consider if relevant. In addition to this abstract spaces, the negotiation environment is populated by agents which resort to different models and strategies.

So, for a given agent A, his private information is defined as follows:

$$PI^A = \{X_1^A,\ X_2^A,\ ...,\ X_i^A,\ ...,\ X_l^A\} \tag{1}$$

where X_i^A in equation 1 is agent A's ith private variable. Except A, no other agent could directly see or query this variables (although anyone could try to estimate them by modelling A's behaviour).

Private variables could refer to the needs or urgency of a factory to buy supplies due to low inventory and high demand, the necessity or rush of a family to buy a bigger car or house, or the urgency of a brand to gain a new market through franchises in order to avoid bankruptcy.

Next, we define the negotiation context that agent A considers relevant as:

$$C^A = \{Y_1,\ Y_2,\ ...,\ Y_k,\ ...\} \tag{2}$$

where Y_k in equation 2 is the kth contextual variable that agent A decides to query. It is noteworthy that the values of these variables are public, so any agent could use them to make predictions upon them: every agent could "see" and query their values, if she has the means to do so and considers them relevant. In this way, C^A is a subset of the whole context C biasing agent A behavior.

These contextual variables may not only refer to changes in global currencies as the dollar or variations in the weather forecast when selling renewable solar energy, but also to particular events as the fall of a government leader, the sinking of a freighter that carry supplies to our factory, or even the risk of losing the possibility to negotiate with another parties due to resentment.

Finally, the group of negotiation agents that agent A considers is defined as:

$$NA^A = \{A,\ B,\ ...,\ J,\ ...,\ M\} \tag{3}$$

where J in equation 3 is the jth negotiation agent. This group set is made up of all the agents A knows or considers relevant to negotiate with. Now again, NA^A is a subset of the whole group of negotiation agents NA.

Summing up, the negotiation environment for a given agent A will be defined by:

$$E^A = PI^A \cup C^A \cup NA^A \tag{4}$$

From the point of view of an external, omniscient observer, the negotiation environment will be constituted by all the agents, contextual and private variables in existence. Formally:

$$E = PI \cup C \cup NA \tag{5}$$

where $PI = \{PI^A,\ PI^B,\ ...,\ PI^J,\ ...,\ PI^M\}$, that is, the conjunction of all agents' private information.

In Fig. 1 is presented a pictorial representation of the proposed negotiation environment.

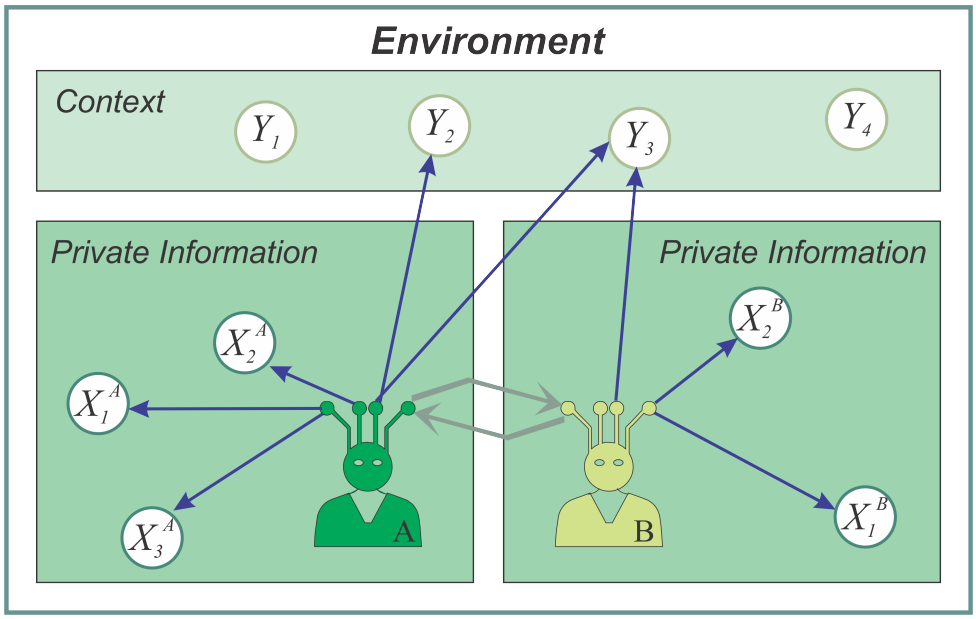


Figure 1: The proposed negotiation environment.

3 Strawberry: a context-aware negotiation agent

In this section, the proposed design for our context-aware agent Strawberry is presented. A component-based architecture proposed by Baarslag in [4], which receives the name of BOA (after Bidding strategy, Opponent model, and Acceptance strategy), is appropriated for implementing Strawberry. However, in order to test our main hypothesis, we incorporate to this architecture the possibility of querying the environment.

All these components and the resulting design are rather simple but will serve our purpose. As can be imagined, the design complexity could be further improved, but this environmental abstraction will suffice to prove our hypothesis about the role of contextual variables in the negotiation game. We profoundly believe that by keeping things simple (as long as it is possible) helps to maintain a clearer view, as well as highlights the essential aspects of our work.

On top of this architecture, two widely known techniques will be used, namely Self-Play [26] and Reinforcement Learning (or RL) [25, 27]. Then, Strawberry will learn to negotiate by simulating negotiation episodes in different contexts against another instance of himself while using the well-known Q-Learning algorithm.

In Fig. 2[1], we present a graphical representation of our agent Strawberry and the different components that will be explained in subsections below.

3.1 Environmental model

We will begin with the description of our environmental model. As we have shown in Section 2, we believe the environment can be modeled by a group of variables and negotiation agents. All we need is to provide our agent Strawberry (A) the means (abstract sensors) to query context-relevant information as deemed necessary, and combine it with his assumptions to model the negotiation environment including other agents.

To this end, we introduce the Oracle, a conceptual entity that could provide our agent real-time data values from the contextual and private variables, and summarize them in two state variables, *necessity* (ν^A) and *risk* (ρ^A). This two variables, as can be seen in Fig. 2, summarize the state of the environment where our agent is inserted.

[1] Adapted from [6].

Necessity and risk, altogether, stand for the positive and negative effects of closing a particular deal over our agent, given the state in which the environment is. While ν explains the necessity our agent has, according to his private variables, to close the deal, ρ accounts for the contextual variables, representing the risk our agent exposes himself if he does close that deal.

Next, the definition of those variables is given:

$$\nu^A = max\{X_1^A,\ X_2^A,\ ...,\ X_l^A\} \quad ; \quad 0 \leq \nu^A \leq 1 \tag{6}$$

$$\rho^A = max\{Y_1,\ Y_2,\ ...,\ Y_k\} \quad ; \quad 0 \leq \rho^A \leq 1 \tag{7}$$

These variables, ν^A and ρ^A, represent the state for associative learning used by Strawberry.

There are two important aspects to mention here. Firstly, although more variables could have been defined to represent the state of the environment, the use of agent necessity and perceived risk of its decision is a simple representation of each perspective that help us prove our main hypothesis, namely the importance of contextual information in the negotiation game. Secondly, as can be seen, a conservative approach is used in the computation of those state variables: necessity and risk are defined by the greatest necessity and the greatest risk the agent is exposed to. Again, such an approach is only a first approximation to prove our hypothesis about the importance of context-aware negotiation.

Necessity and risk, in this work, have two main attributes. In the first place, these are private variables: opponents can not see if an agent is in great necessity or at a low risk during a negotiation (although they can estimate it given the agent's behaviour, as previously said). In the second place, these variables are fixed during a negotiation episode. This aspect may seem a limitation, but it is not. In automated negotiations, offers are rapidly interchanged between agents, and the whole negotiation episode does not last more than a couple of seconds. In this way, the assumption of a fixed environment during a particular negotiation is not an issue to worry about. Despite this, if the negotiation episode takes place during an extended period of time, it would be desirable that a negotiation agent could perceive relevant changes in the environment to adjust his behavior in the remaining part of the episode.

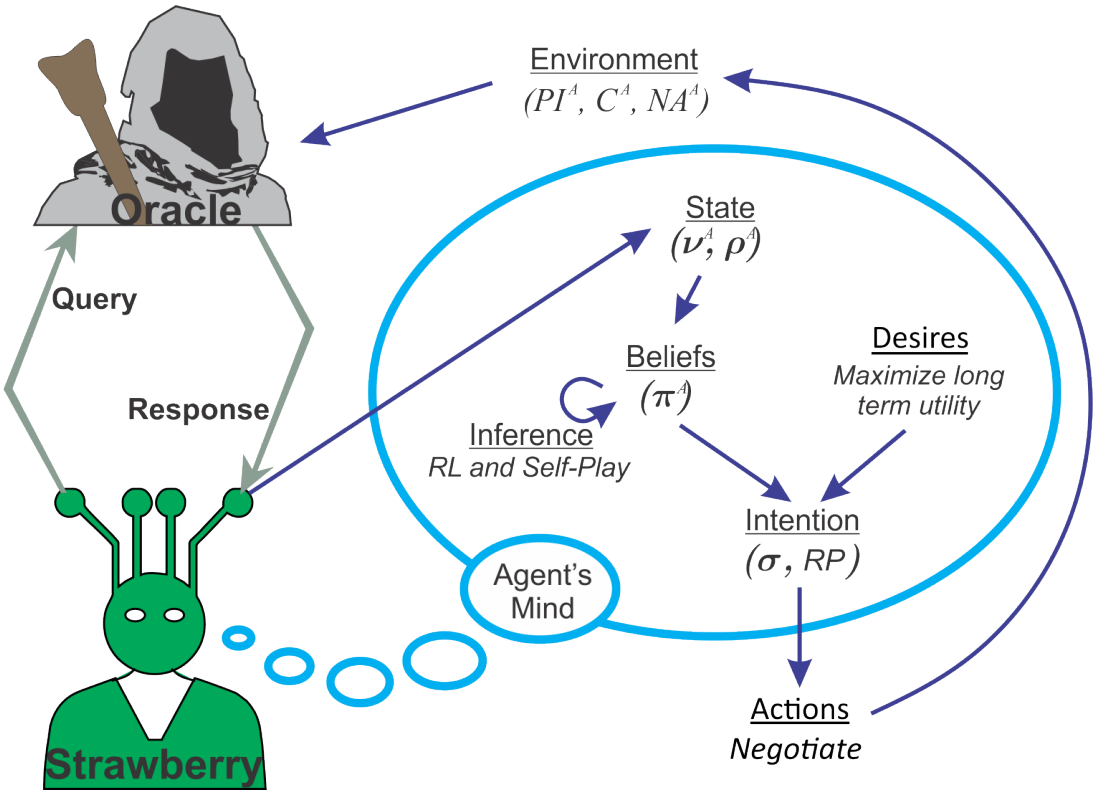


Figure 2: The internal design of Strawberry, our context-aware negotiation agent.

3.2 Bidding strategy

Taken from Fatima [10] and references therein, a heuristic concession strategy for Strawberry is defined as follows:

$$O_t = IP + (RP - IP) * \sigma(\beta, t, n) \tag{8}$$

where O_t is the offer Strawberry will make at time t, IP is the initial price, which is assumed to be the best deal the agent considers it can get from the negotiation, and RP is the reserve price, which is the worst deal the agent can achieve at the end of a negotiation episode. The concession strategy σ is based on:

$$\sigma = \left(\frac{t}{n}\right)^{1/\beta} \tag{9}$$

where t is the time elapsed from the beginning of the negotiation, n is the deadline, and β defines the concession rate.

3.3 Acceptance strategy

The acceptance strategy to be used is AC, taken from Baarslag [4], where its effectiveness was demonstrated. It could be summarized as follows: our agent will accept an offer from his opponent B if and only if this offer supposes a higher utility for our agent than the utility he would obtain from his own next offer. In other words, Strawberry will accept the offer if:

$$u(O_t^B) \geq u(O_{t+1}^A) \tag{10}$$

where $u(O_t^{agent})$ is the long-term utility Strawberry will obtain from the offer O made by the *agent* at time t.

3.4 Self-Play and Reinforcement Learning

Strawberry will use his self-play capacity to acquire some information of the public context by modelling its influence in negotiation game. To this end, he will play with another instance of himself, adapting his policy π^A to select the hyperparameters that define how to properly behave in a negotiation episode.

This does not invalidate the importance of learning from other opponents: the importance of modelling the opponent was clearly stated in previous works [5, 31]. Still, the self-play learning phase could help our agent gain an initial understanding of the context he is in and how does this context affects him.

Strawberry's final desire is not only to maximize his next possible reward r but also to maximize his long term utility, as stated in Fig. 2. This learning strategy is implemented by the Q-Learning algorithm, which consists of a function that iterates over the expected cumulative rewards for future time steps in a negotiation episode (how many will depend on the tuning of the algorithm hyper-parameters) given the perceived state s_t^A of the environment providing that the agent takes a certain action a from the set of possible actions. The action to take is determined by a policy π^A derived from the Q-values, which are the way this algorithm represents the immediate and long-term utility for every state-action pair. At the end of each negotiation episode, Strawberry observes the immediate reward r and the next state s_{t+1}^A that the environment returns, and obtains the Q-value from the best action the agent can take in that situation (indicated by $max_a\ Q(s_{t+1}^A, a)$ in equation 11). Then, the Q-Learning algorithm actualizes the Q-value according to:

$$Q(s_t^A, a_t) = Q(s_t^A, a_t) + \alpha\ \left[r_{t+1} + \gamma\ max_a\ Q(s_{t+1}^A, a) - Q(s_t^A, a_t)\right] \tag{11}$$

We choose each state for Strawberry (A) to be defined by the tuple:

$$s^A = (\nu^A,\ \rho^A) \tag{12}$$

where ν^A and ρ^A are the necessity and risk associated with the perceived state of its environment, as mentioned earlier.

The possible actions for Strawberry are defined by:

$$a = (RP,\ \beta) \tag{13}$$

where RP and β are the variables our agent will choose to vary his negotiation strategy.

A group of hyper-parameters need to be set in this algorithm to work. These are ϵ, α and γ. ϵ defines the exploratory nature of our agent, that is, the probability our agent takes an exploratory move (normally, a random move) rather than exploiting its knowledge (following the current estimation of an optimal policy). α is the learning rate parameter, which gives more weight to recent rewards than to past rewards. γ is the discount rate, with which the agent will try to maximize the sum of the discounted rewards it is going to receive in the future. We could say that by means of these three parameters we have past, present and future into account through the chosen values α, ϵ, and γ, respectively.

3.5 A formal definition of Strawberry's behavior

Here, the different internal conditions Strawberry can be in are discussed. It becomes necessary to explain that these conditions are different to the perceived environmental states that have been defined earlier, since its internal conditions in Fig. 3 correspond to states of a state machine diagram in a given negotiation episode.

In Fig. 3, the two basic activities for the Strawberry agent are shown: it may either learning through Self-Play or actually negotiating against other opponents. Of course, the fact that our agent does not learn while negotiating with other opponents is imposed to avoid exploration when there exist enough knowledge to exploit from. This favor evaluating how good the policy learned through Self-Play actually is.

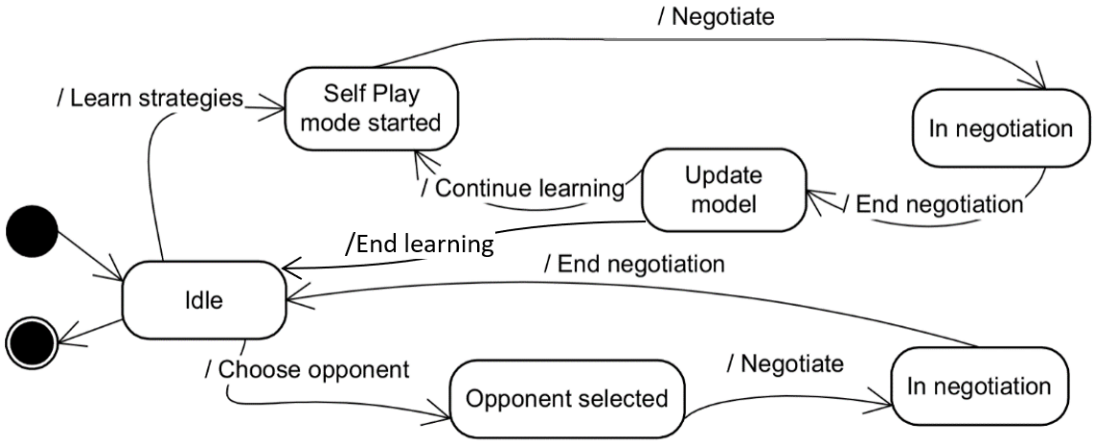


Figure 3: Strawberry's general state machine diagram.

In Fig. 4, an entire negotiation episode is shown. When the negotiation episode starts, Strawberry requests the Oracle some information from the context. In the next step, he selects a negotiation strategy based on the policy he has already learned or is actually learning. To continue with, the negotiation itself is represented: first, the agent makes an offer; then, he waits and evaluates the counteroffer; then he chooses whether to accept or to reject (cases in which the negotiation episode ends), or to make a counteroffer. If the opponent makes the first offer, then the agent must evaluate it and take the proper decision. If the opponent accepts or rejects an offer of the Strawberry agent, then the negotiation episode ends.

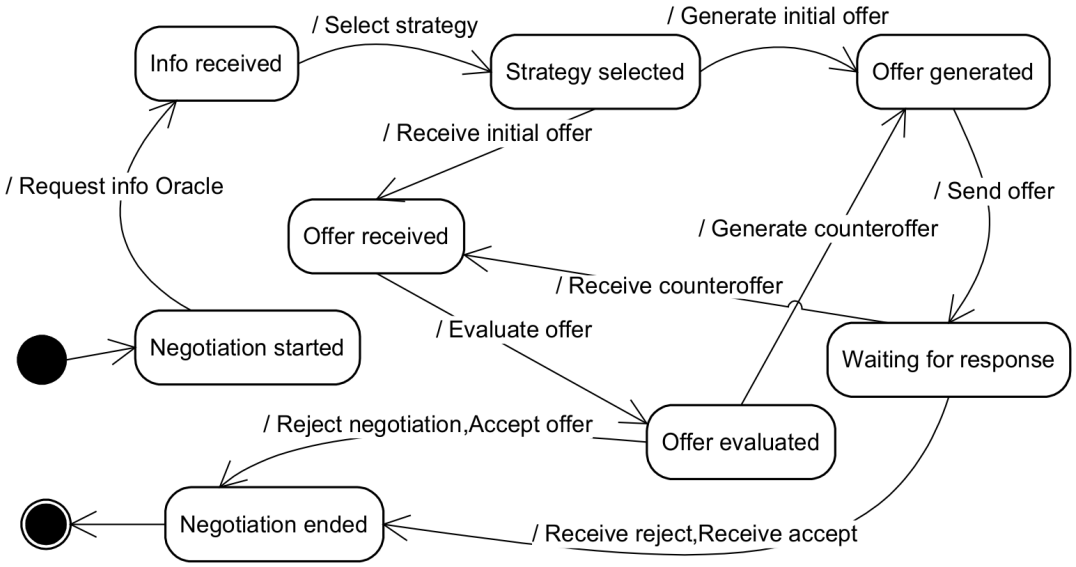


Figure 4: The negotiation process of Strawberry in a state machine diagram.

4 GENIUS

As has been said before, the GENIUS simulation software will be used to test our main hypothesis regarding the key role of contextual variables during a negotiation process. GENIUS is a specialized non-commercial environment for simulating negotiations, where a given agent design can be implemented and then faced against a set of previously available agents. Single negotiations or tournaments could be run, based on deadlines that are common, public knowledge. The GENIUS present the results in a table and a chart, where the Pareto frontier, the Nash equilibrium, and other social welfare metrics could be seen.

The GENIUS simulator is a practical tool to try new agents in the field of automated negotiations. Nevertheless, there is no context in this negotiations and, from our point of view, this makes the negotiation environment too unrealistic. As we would like to prove that agents should take the context into account so as to make more rational decisions, we implemented the negotiation environment represented in Section 2 and included it in GENIUS. We will use these new concepts in the next sections to define and develop the computational experiments.

5 Experiments

In this section, the experimental setting is described, including how negotiation simulations were run, and the results obtained after negotiation episodes.

Firstly, some changes and additions to GENIUS are discussed. New features were developed in Java, using the IntelliJ IDEA environment, a package that gave us the possibility to adapt GENIUS to our needs. Finally, we developed our agent, Strawberry, with the capability to query contextual variables to the Oracle mentioned in section 3.

5.1 Experimental setting

The first step to address our hypothesis was to create Strawberry's private and contextual variables. Without any loss of generality, only two variables in each space are considered (X_1, X_2, Y_1, and Y_2), each one taking random integer values between 0 and 3. These variables will be queried through the Oracle, normalized to 1 when required by Strawberry, and internally modelled using ν^A and ρ^A.

The second step was to select a domain in GENIUS that could gave us simple but revealing results, bearing in mind that the focus here are single-issue negotiations. The domain selected was the "pie domain", a problem usually addressed in game theory [18] and also available in GENIUS. In this domain, two agents negotiate over a pie that is divided into a number of pieces (we choose this number to be a thousand). The aim for each agent is to get as many pieces as it can, taking into account that, if the deadline is reached without a deal, every one gets zero pieces. The utility is given by how many pieces an agent gets by the end of the negotiation episode divided by 1000.

The third step was to define the concrete aspects of the Q-Learning algorithm. We designed a reward function upon which the environment would give Strawberry a reward r at the end of each negotiation episode, depending on the outcome of the negotiation (either an agreement is reached or not) and the environment state s^A at which the negotiation episode takes place. This reward function is defined as follows:

- If the negotiation ended successfully (an agreement is reached):

$$r = u(O_{t=end}) \tag{14}$$

  that is, the utility that the agent obtains from the last offer reported.

- If the negotiation ended unsuccessfully:

$$r = \begin{cases} -1 & \text{if } X_1 = 3 \\ -1/3 & \text{if } X_2 = 3 \\ -2/3 & \text{if } Y_1 = 3 \ \wedge \ Y_2 = 0 \\ 1/3 & \text{if } X_1 \leq 1 \\ 2/3 & \text{if } X_1 \leq 1 \ \wedge \ X_2 \leq 1 \end{cases} \tag{15}$$

The rationale behind this function is that the agent would be not only concerned by the result of the negotiation, but also by the perceived state of his private information, the context, and how these affect him.

A number of hyper-parameters had to be set in order to make Strawberry capable of learning from reinforcements. As a common rule of thumb, α and ϵ are usually set to 0.1 [25], and γ to 0.9, values that contribute to a fast learning by means of a reasonable exploration-exploitation trade-off. These are typical default values used in the reinforcement learning literature.

Given that our agent learns using tabular Q-Learning, states need to be defined in discrete values. For this purpose, we divided the possible values for ν^A and ρ^A, into three intervals $[0; 0,33]$, $(0,33; 0,66]$, and $(0,66; 1]$, as both vary between 0 and 1. In this sense, for our agent, a necessity of 0.1 and a necessity of 0.29 would be taken in the same group, semantically a "low necessity" internal state. Likewise, a perceived risk of 0.8 and other of 1 would be taken in the same group, semantically a "high risk" internal state.

We also define the actions allowed to Strawberry. We set three different values for β: 0.5, 1.0, and 2.0, and five possible values for RP: 0.0, 0.2, 0.4, 0.6, and 0.8, which provide Strawberry with 3 * 5 = 15 different concession strategies. Summing up, we will have a maximum of 3 * 3 * 3 * 5 = 135 Q-values to learn.

5.2 Experimental design

The experiments were conducted in the pie domain, where a deadline of 180 rounds was used for simulating the negotiation episodes. The experiments were divide up in three phases: the learning phase, the negotiation phase, and the Self-Play improvement phase.

In the learning phase, the Strawberry agent is set to negotiate against himself using the tournament setting that GENIUS provides. Self-Play simulations were run for 10, 100, 500, 1000, 2000, 5000, and 10000 negotiation episodes in order to see how much learning may affect the subsequent negotiation phase.

In the negotiation phase, tournaments are played against some of the existent negotiation agents in GENIUS. We have chosen some simple ones and others really difficult to beat, winners of previous ANAC competitions. The chosen opponent types are:

- Random Party (RP)
- BoulwareNegotiationParty (B)
- ConcederNegotiationParty (C)
- CUHKAgent2015 (CUHK)
- AgentFSEGA (FSEGA)
- Agent_K (K)
- IAMcrazyHaggler (Haggler)
- AgentSmith (Smith)
- Gahboninho (G)
- BRAMAgent (BRAMA)

A short description of each opponent will be given. Random Party is a naive agent that makes always a random offer, with the only restriction that he always concedes more than in his previous turn. The BoulwareNegotiationParty and ConcederNegotiationParty use time-dependent negotiation strategies. The Boulware concedes less at the beginning and more at the end of the negotiation if no agreement has been reached yet. The Conceder works the other way around: concedes more at the beginning and less at the end, if no agreement was reached yet. The CUHKAgent2015 estimates the concession degree of the opponent in order to make concessions. AgentFSEGA predicts the opponent's negotiation strategy and then concedes accordingly. Agent_K uses the mean and variance of the utility of all received offers, and then tries to determine the best offer he might get. He then accepts or rejects based on the probability of receiving a better offer. IAMcrazyHaggler makes random proposals among those having a high utility to him. AgentSmith constructs an opponent model and then concedes slowly towards the opponent's offers. Gahboninho uses a meta-learning strategy, trying at first to identify if the opponent is modelling him, and then exploiting this behaviour. Finally, the BRAMAgent uses opponent modeling to propose offers likely to be accepted by the opponent.

Simulations were run in two different settings: Strawberry against all other agents together, and Strawberry against each one of them, separately. Each tournament consisted of 100 negotiations were Strawberry used the knowledge previously gained through Self-Play, but did not learn whilst negotiating with the other opponents.

Finally, the Strawberry agent is put to negotiate against himself, but this time without doing any learning, in order to see if he achieves better agreements in Self-Play compared to the different learning episodes he has previously done.

5.3 Results and analysis

After the learning phase, the negotiation phase has given us some interesting results that are depicted in Fig. 5. At first glimpse, the cumulative utility of Strawberry against the average opponent starts below 300 and reaches 400 as he learns to account for the environment. This behavior was rather expected since the reinforcement learning method resorts to the association of the goodness of an action to the value of contextual variables. On the other hand, it is also possible to see that taking the environment into account could change the perspective of negotiation itself: Strawberry had received, on average, better outcomes than his negotiation counterparts when considering the context of the negotiation. This reasoning tempts us to say that the environment should be taken into account so as to gain a competitive advantage.

Another important observation to make from Fig. 5: if some agent considers environmental variables, there will be an increase in the rest of the agents cumulative utilities, not only in his own. This astonishing result gives rise to two different theories. The first one is that it could be possible that if one of the agents takes some variables into account that the other does not, better agreements are reached, with a tendency to improve social welfare. The second theory, the one we think could explain better this phenomena, is that, as Strawberry learns, he makes more rational decisions and does not take so many actions at random. In this context, the other agents could build a better model out of him and predict better what his moves are going to be. In other words, they model the part of the environment they do not see through the model they made of Strawberry's negotiation behavior. We think these are two key aspects we have discovered through our research.

In Fig. 6, we can see the utilities obtained by Strawberry against each particular agent, and the utilities obtained by his opponents, which do not take environmental variables into account. Again, as expected, the utilities obtained by Strawberry are always better when the rewards of the environment

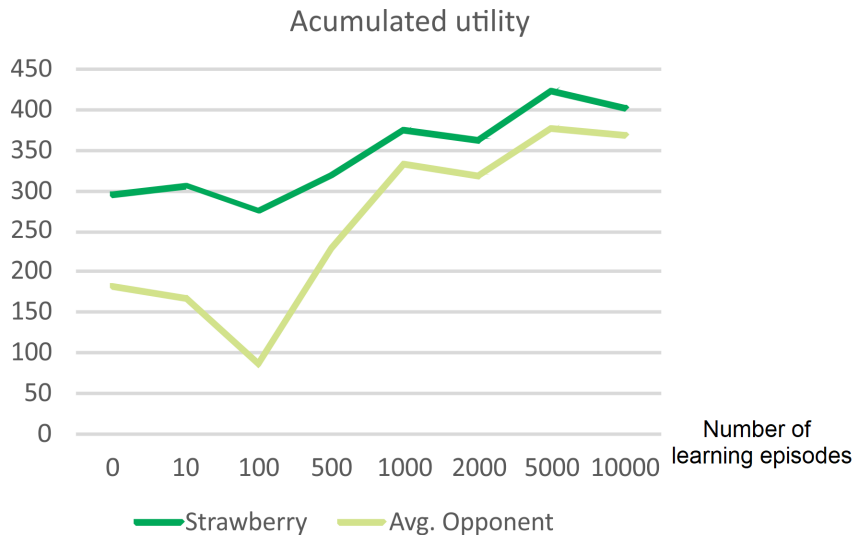


Figure 5: Strawberry's accumulated utility over 100 tournament scenario against the average opponent, that is to say, the average utility the different opponents mentioned in Section 5.2 have obtained. The horizontal axis shows how many learning negotiations with Self Play has Strawberry made. The vertical axis shows the accumulated utility throughout the 100 tournament negotiation session.

are considered than when they are not (the rewards that GENIUS gives). Another thing we could see is that as agents get more complex (e.g., with opponent models and flexible or though strategies, etc.), they make it more difficult for Strawberry to get a good deal. Particularly, the CUHK agent seems to behave really tough, not making too much room for Strawberry to have a competitive edge, but still getting a great deal of accumulated utility. It is worth asking how much an agent like this would gain if he were taking the context into account, as Strawberry does.

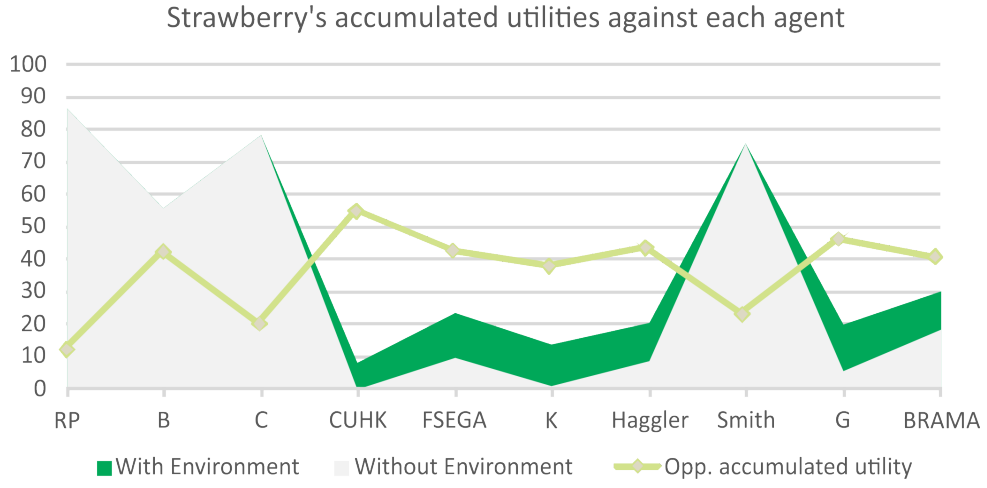


Figure 6: Accumulated utility obtained by Strawberry after 10000 learning negotiation sessions in Self Play against each particular agent, when it is considered the reward from the environment and when it is not, and the utilities obtained by his opponents.

In the Self-Play improvement phase, we have reached other conclusions. As can be seen in Fig. 7, Strawberry achieves better and better agreements as he learns considering the context of the negotiation, thus increasingly maximizing the social welfare over negotiations. Also, in the graph on the right side,

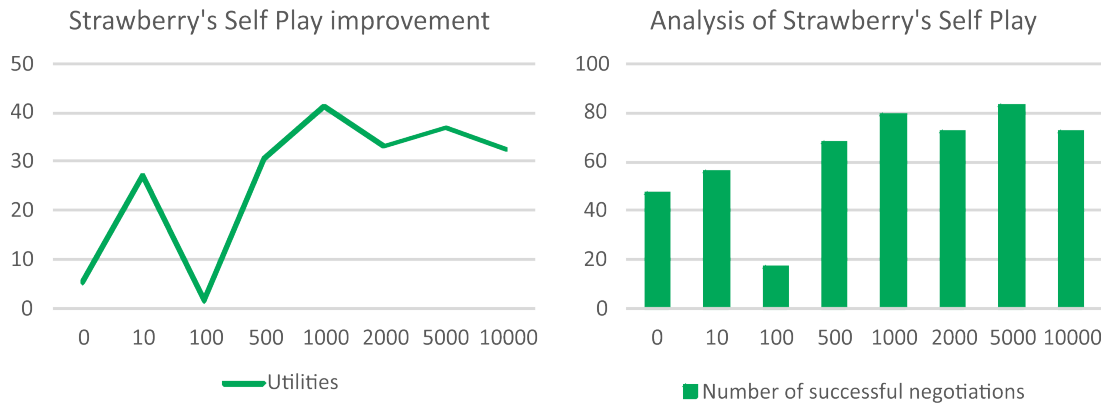


Figure 7: The graphic on the right shows Strawberry's Self Play improvement in accumulated utility as he learns. In the left graphic, we see how many successful negotiations reaches Strawberry's when making 100 negotiation sessions against himself.

we see how Strawberry gets more successful negotiations as he learns from evaluative feedback from negotiation episodes. Fig. 6 and 7 vividly highlight the importance of contextual-learning in negotiations, as can be expected, following the previous results shown in Fig. 5.

A final comment can be made about reaching an equilibrium. We have shown that Strawberry gets better through Self-Play and learning. However, he does not tend to reach the Nash equilibrium that GENIUS proposes, as can be seen in Fig. 8, when playing against himself. In fact, there are no great changes in the mean distance to the Nash point. This has, from our point of view, a simple explanation: GENIUS does not consider the contextual variables to calculate the Nash equilibrium. As the learning advances, mainly after the 500 tournaments learning, there is a difference of approximately 0.3 units, which highlights that the Nash equilibrium is not actually where the GENIUS locates it. In conclusion, it could be stated that is not possible to find the real Nash equilibrium of a negotiation game unless the relevant contextual variables that affect all agents' utilities are taken into account.

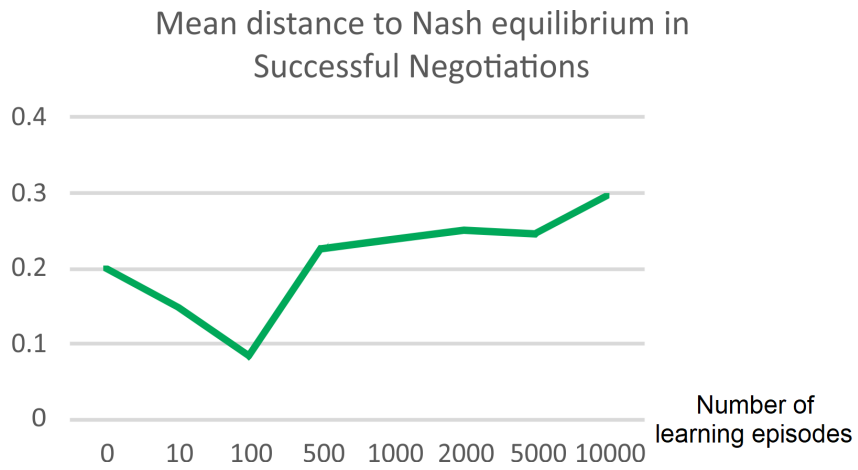


Figure 8: This graphics presents in the vertical axis the mean distance to the Nash equilibrium proposed by GENIUS in 100 Self-Play episodes once learning has ended. The horizontal axis shows the number of learning episodes previously made by Strawberry.

6 Concluding remarks

The importance of taking the context into account when two agents are negotiating over a certain issue has been addressed. A novel way of modeling the negotiation environment based on characterizing both the agent's private variables, which consists of the agent's strategies and preferences, and the context, where, besides other negotiation agents, a group of external variables that influence the utilities and values of concerned agents, are used to learn negotiation strategies.

The proposed Strawberry agent resorts to contextual variables to gain a competitive advantage. We have presented the way our agent account for contextual variables, based on which his bidding and acceptance strategies are built up. Then, we have explained how Strawberry would obtain some negotiation knowledge using the Q-Learning algorithm and Self-Play.

Computational experiments were designed to assess the validity of our central hypothesis: the advantage of including contextual variables in a negotiation agent. Results obtained confirm our earlier thoughts whereas other results are rather unexpected. Strawberry is quite competitive in a heterogeneous environment composed of a number of agents, even though he makes no explicit model of his opponents. We have proven that the utilities agents perceive are different whether we take or not the variables of the context and the agent's private variables into account. These results sustain our main hypothesis, but more learning experiments are needed. We have seen, along with Strawberry's improvement, his opponent improvements as he learns. We theorize that the models other agents make out of Strawberry help them discover implicitly the variables Strawberry takes into account, although they do not know of their existence. It can also be stated, from the results shown in Fig. 6, that the Strawberry agent reap higher utilities when he takes these variables into account. The importance of Self-Play for cheap Learning is highlighted through the results obtained. Hence, social welfare can be increased as agents learns collectively through inexpensive simulation-based on Self-Play learning.

As a final word, it can be said that our hypothesis seems correct from the point of view of the Nash equilibrium. If we take the contextual variables into account, it makes no sense to find an equilibrium between the strategies of the negotiation agents considered in isolation. If contextual variables are not perceived by any of the agents, then they would attempt to reach an equilibrium that is nonexistent. Our agent Strawberry, simple as it is, when playing against himself shows us that the equilibrium is somewhere else, not just "in the middle". In other words, should we assume that, when two people claim for a piece of pie, the whole is to be always divided exactly in two? We thing we should not, and the results support our earlier thoughts that this division does not only depend on the agents and the pie itself, but also on external variables agents should not leave aside.

7 Future work

There are a number of research avenues that can be taken to further this research. For example, we suspect these results could be extended to multiple-parties negotiations, different protocols, multiple issues, various domains, and so on. We have run our simulations in GENIUS, but other tools could also be used to obtain more comprehensive results and to finally test our central hypothesis. Strawberry could be also improved. It was found that it is not so good against top agents like CUHK or Ghaboninho. To this aim, the environmental model it builds can be more complex and new bidding and acceptance strategies could be added. Self-Play could be just a component of the learning process; learning to negotiate must be extended to many other agents in order to get better outcomes, modeling not just the external variables, but also the opponent model of the environment and the other opponents using Theory of Mind and evolving strategies using multi-agent Q-learning. More experiments should be made so as to ratify these new approaches. Different external variables can be provided by the Oracle: more complex variables which may give rise to unexpected behavior that will make it more difficult not only for Strawberry to model, but also to the rest of the interacting agents that may be disconcerted by Strawberry's actions. A new intrinsically-motivated reward function should be designed, that could give Strawberry a competitive edge to identify sooner what is going on in the environment by pinpointing key variables. Deadlines could vary between negotiation episodes. We see this as a great restriction imposed by GENIUS: that the deadline is indeed public knowledge. More sophisticated opponents could be addressed, and querying strategies to the Oracle could be also part of the learning process.

References

[1] Ajay Agrawal, Joshua Gans, and Avi Goldfarb. *Prediction machines: The simple economics of artificial intelligence*. Harvard Business Review Press, Boston, Massachusetts, 2018.

[2] Bedour Alrayes, Özgür Kafalı, and Kostas Stathis. Concurrent bilateral negotiation for open e-markets: The conan strategy. *Knowledge and Information Systems*, 2017.

[3] Dan Ariely. *Predictably irrational: The hidden forces that shape our decisions*. Harper Perennial, New York, revised and expanded ed. edition, 2010.

[4] Tim Baarslag. *Exploring the strategy space of negotiating agents: A framework for bidding, learning and accepting in automated negotiation*. Springer theses. Springer, Switzerland, 2016.

[5] Tim Baarslag, Mark J.C. Hendrikx, Koen V. Hindriks, and Catholijn M. Jonker. *Learning about the opponent in automated bilateral negotiation: a comprehensive survey of opponent modeling techniques*, volume 30. Springer US, 2016.

[6] Chris L. Baker, Julian Jara-Ettinger, Rebecca Saxe, and Joshua B. Tenenbaum. Rational quantitative attribution of beliefs, desires and percepts in human mentalizing. *Nature Human Behaviour*, 1(4):1–10, 2017.

[7] Natalia Criado Pacheco, Carlos Carrascosa, Nardine Osman, and Vicente Julián Inglada. *Multi-Agent Systems and Agreement Technologies*, volume 10207. Springer International Publishing, Cham, 2017.

[8] Daisuke Wakabayashi. Waymo's autonomous cars cut out human drivers in road tests, 2017.

[9] Eden S. Erez, Inon Zuckerman, and Dror Hermel. Automatic negotiation: Playing the domain instead of the opponent. *Journal of Experimental and Theoretical Artificial Intelligence*, 29(3):597–616, 2017.

[10] S. Fatima, S. Kraus, and M. Wooldridge. *Principles of Automated Negotiation*. Cambridge University Press, 2014.

[11] David Ferrucci, Anthony Levas, Sugato Bagchi, David Gondek, and Erik T. Mueller. Watson: Beyond jeopardy! *Artificial Intelligence*, 199-200:93–105, 2013.

[12] Roger Fisher and William Ury. *Sí, ¡de acuerdo! Como negociar sin ceder*. Libros universitarios y profesionales. Serie Norma de desarrollo gerencial. Norma, [Colombia], 1985.

[13] Paul W. Glimcher, editor. *Neuroeconomics: Decision making and the brain*. Academic Press, London and San Diego, CA, 1st ed. edition, 2009.

[14] Cuihong Li, Joseph Giampapa, and Katia Sycara. Bilateral negotiation decisions with uncertain dynamic outside options. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, 36(1):31–44, 2006.

[15] Raz Lin, Sarit Kraus, Tim Baarslag, Dmytro Tykhonov, Koen Hindriks, and Catholijn M. Jonker. Genius: An integrated environment for supporting the design of generic automated negotiators. *Computational Intelligence*, 30(1):48–70, 2014.

[16] Ariel Monteserin and Analía Amandi. Argumentation–based negotiation planning for autonomous agents. *Decision Support Systems*, 51(3):532–548, 2011.

[17] Philip Ball. How life (and death) spring from disorder. *Quanta Magazine*, 2017.

[18] Ariel D. Procaccia. Cake cutting: Not just child ' s play. *Communications of the ACM*, 56(7):78–87, 2013.

[19] Fenghui Ren and Minjie Zhang. A single issue negotiation model for agents bargaining in dynamic electronic markets. *Decision Support Systems*, 60(1):55–67, 2014.

[20] J. Rodriguez-Fernandez, T. Pinto, F. Silva, I. Praça, Z. Vale, and J. M. Corchado. Context aware q-learning-based model for decision support in the negotiation of energy contracts. *International Journal of Electrical Power and Energy Systems*, 104(October 2017):489–501, 2019.

[21] Stuart Russell. Artificial intelligence: The future is superintelligent. *Nature*, 548(7669):520–521, 2017.

[22] Arvind Sathi. *Cognitive (internet of) things: Collaboration to optimize action.* Nature America Inc, New York NY, 1st edition, 2016.

[23] Seventh International Conference on Learning Representations. *Emergent Communication through Negotiation*, 2018.

[24] Shane Legg. *Machine Super Intelligence: PhD Thesis.* Shane Legg, 2008.

[25] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: An introduction.* Adaptive computation and machine learning. MIT Press, Cambridge Mass., 2nd edition, 2018.

[26] Gerald Tesauro. Temporal difference learning and td-gammon. *Communications of the ACM*, 38(3):58–68, 1995.

[27] Christopher John Cornish Hellaby Watkins. *Learning from Delayed Rewards.* PhD thesis, King's College, Cambridge, UK, 1989.

[28] Hermanes Albertus de Weerd and Rineke Verbrugge. *If you know what I mean: Agent-based models for understanding the function of higher-order theory of mind.* University of Groningen and Rijksuniversiteit Groningen, Groningen, 2015.

[29] A. D. Wissner-Gross and C. E. Freer. Causal entropic forces. *Physical Review Letters*, 110(16):1–5, 2013.

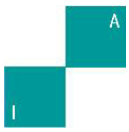
[30] G. Yang, Y. Chen, and J. P. Huang. The highly intelligent virtual agents for modeling financial markets. *Physica A: Statistical Mechanics and its Applications*, 443:98–108, 2016.

[31] Farhad Zafari and Faria Nassiri-Mofakham. Popponent: Highly accurate, individually and socially efficient opponent preference model in bilateral multi issue negotiations. *IJCAI International Joint Conference on Artificial Intelligence*, 2016(April):5100–5104, 2017.

[32] Daniel Dajun Zeng and Katia Sycara. Benefits of learning in negotiation. In *Proceedings of the fourteenth National Conference on Artificial Intelligence and ninth Innovative Applications of Artificial Intelligence Conference*, volume July 27-31 of *AAAI '97/IAAI '97*, Menlo Park and London, 1997. The AAAI Press, Menlo Park, California.

[33] Yi Zou, Wenjie Zhan, and Yuan Shao. Evolution with reinforcement learning in negotiation. *PLoS ONE*, 9(7), 2014.

Inteligencia Artificial, 22(63) (2019), 150-161
doi: 10.4114/intartif.vol22iss63pp150-161



INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

Genetic Algorithms for Satellite Launcher Attitude Controller Design

Paulo R. Silva[1,A], Ivanildo S. Abreu[1,B], Paulo A. Forte[1,C], Henrique M. C. do Amaral[1,D]

[1] Computer Engineering Department, at State University of Maranhao, Brazil.
[A]paulosilva14@aluno.uema.br
[B]ivanildoabreu@professor.uema.br
[C]pauloforte@aluno.uema.br
[D]henriqueamaral@professor.uema.br

Abstract For proper attitude control of space-crafts conventional optimal Linear Quadratic (LQ) controllers are designed via trial-and-error selection of the weighting matrices. This time consuming method is inefficient and usually results in a high order complex controller. Therefore, this work proposes a genetic algorithm (GA) for the search problem of the attitude controller gains of a satellite launcher. The GA's fitness function considers some control features as eigenstructure, control goals and constraints. According to simulation results, the search problem of controller parameters with evolutionary algorithms was faster than usual approaches and the designed controller reached all the specifications with satisfactory time responses. These results could improve engineering tasks by speeding up the design process and reducing costs.

Keywords: Genetic Algorithm, Optimal control, Attitude control

1 Introduction

Rockets and space-crafts engineering have led substantially improvements in communications, navigation, space and earth observation, bringing progress to society. In order to travel safely through space, these non-linear vehicles need good navigation and guidance modules with embedded digital controllers to control attitude angles and velocities [10, 14, 13, 7].

Modern optimal control strategies design controllers for linear time invariant systems to satisfy desired specifications by minimizing a quadratic performance index [4]. This index directly affects the control outcome and includes states and control vectors that must be weighted by user defined matrices. Selection of these matrices is not straightforward as it requires familiarity with the subject and massive simulations for refinement [16, 3].

Usage of Evolutionary approaches to tune optimal controllers has been stated in the literature in a range of areas producing exceptional results and reducing the time spent in the design process. In [15], the authors successfully tuned a Linear Quadratic Regulator (LQR) and Proportional-Integral-Derivative (PID) controllers with a Genetic Algorithm (GA) for the aircraft pitch control problem and demonstrated that the LQR is better than PID. [8] concluded that a GA-tuned LQR controller for the magnetically actuated attitude control of CubeSats is better than a simple LQR and a Proportional-Derivative, resulting in smaller steady state error and faster time response. [11] obtained optimal control gains via Genetic Algorithms, the GA-based controller is superior since the conventional tuning techniques

ISSN: 1137-3601 (print), 1988-3064 (on-line)

is not effective due to unseen non-linearities of the tracker robot. [5] proposed a neural-genetic controller for the attitude control problem of a nonlinear satellite in chaotic motion due to large external motions without any previous knowledge of the system dynamics.

This work proposes a Genetic Algorithm approach for gain computing to the attitude control system of a satellite launcher, given that [2] proposed an analytical method for computing the controller gains of a satellite launcher by tuning the weighting matrices empirically.

This paper is organized as follows. Section 2 presents the satellite launcher longitudinal model and the Linear Quadratic method. Section 3 demonstrates the methodology used in this work. The Genetic Algorithm simulation results are presented in Section 4. Finally, conclusions are given.

2 Background

2.1 Model Description

The equations of motion are derived for a simplified model considering the forces acting on the body. As can be seen in Figure 1, the simplified space-craft is subject to aerodynamic forces, $\vec{F_{aero}}$, gravitational force, $\vec{W}$, thrust force, $\vec{T}$, due to the propellant burn and wind velocity, $\vec{V_{wind}}$. Also in this model, the angle of attack, α, is the angle between the body reference line, u, and the oncoming wind velocity vector, $\vec{V_{wind}}$. The attitude pitch angle is controlled by β_z, that represents the angular deflection of the boosters.

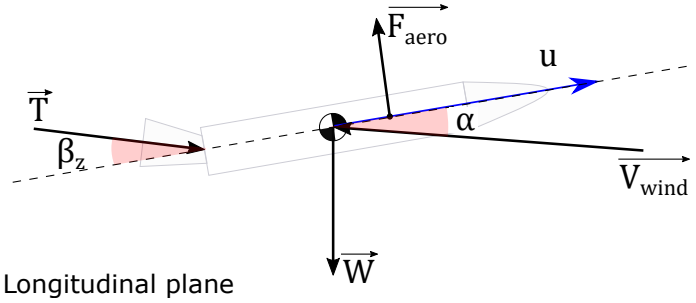


Figure 1: Simplified model of satellite launcher (longitudinal plane).

The simplified state-space model for the rigid-body dynamics of a launcher is given in [2] as

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \mu_\alpha & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ \mu_{\beta z} \end{bmatrix} \beta_z \tag{1}$$

where

$\theta \rightarrow$ pitch attitude angle;

$\beta_z \rightarrow$ Booster deflection angle due to actuator deflection;

μ_α and $\mu_{\beta z} \rightarrow$ angular acceleration coefficients.

2.2 The Linear Quadratic Method

As stated by [17], the Linear Quadratic (LQ) method for a system with state space representation given by

$$\dot{x} = Ax + Bu \tag{2}$$

focus on finding a state-feedback control input

$$u = -Kx \tag{3}$$

that minimizes a quadratic cost function,

$$J = \frac{1}{2}\int_{t_0}^{T} [x^T Q x + u^T R u] dt \tag{4}$$

where Q is a positive semi-definite weighting matrix of the state vector, x, and R is a positive definite weighting matrix of the control input vector, u.

The optimal feedback gain matrix, K, is given by

$$K = R^{-1} B^T P \tag{5}$$

where P is a symmetric matrix solution of the algebraic Ricatti equation (ARE),

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{6}$$

3 Methodology

The control structure, GA model and operators used in this work are discussed in the following subsections.

3.1 Control Structure

Figure 2 illustrates the control structure used in this work. In this structure the control input, β_z, is computed according to a PI controller that computes its output based on the attitude error $(\theta_{ref} - \theta)$ and an angular velocity feedback $(d\theta/dt)$. This structure ensures better tracking to reference commands, good robustness and temporal performance, [2].

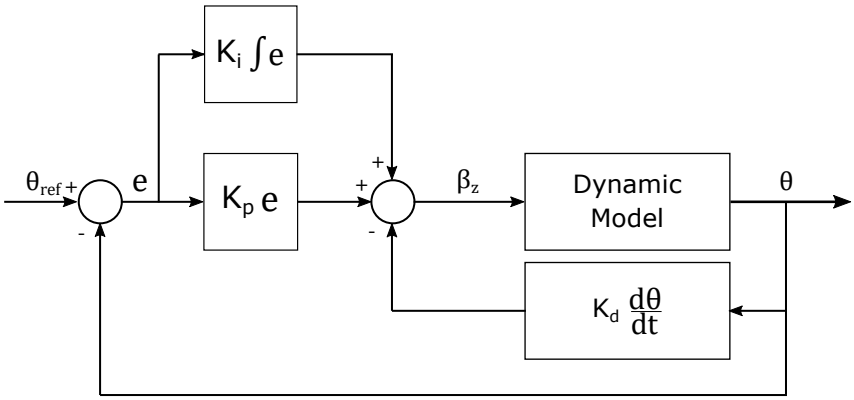


Figure 2: Control structure for the controller design.

Closed-loop model The closed loop state space model for the block diagram presented in Figure 2 is given as follows

$$\begin{bmatrix} \dot{x}_{2\times1} \\ \dot{\tau} \end{bmatrix} = \left[\begin{array}{cc|c} A_{2\times2} & & 0_{2\times1} \\ -1 & 0 & 0 \end{array}\right] \begin{bmatrix} x_{2\times1} \\ \tau \end{bmatrix} + \begin{bmatrix} B_{2\times1} \\ 0 \end{bmatrix} \beta_z + \begin{bmatrix} 0_{2\times1} \\ 1 \end{bmatrix} \theta_{ref} \tag{7}$$

where τ is the error integral

$$\tau = \int \theta_{ref}(t) - \theta(t) dt \tag{8}$$

and the control input, β_z, is given by

$$\beta_z = \begin{bmatrix} -Kp & -Kd & Ki \end{bmatrix} \begin{bmatrix} x_{2\times1} \\ \tau \end{bmatrix} + K_p \theta_{ref} \tag{9}$$

Once the system is in the form $\dot{x} = Ax + Bu$, the linear quadratic method can be used to find control gains in Equation 9. The concern now is how the control problem will be encapsulated in Genetic Algorithms and how the GA will converge to good weighting matrices Q and R that lead optimal control gains.

3.2 Genetic Algorithm Models and Operators

This part presents the genetic models and operators proposed for this work.

Chromosome Model Since $Q_{n\times n}$ and $R_{m\times m}$ are symmetric positive-definite matrices satisfying the linear quadratic specifications, the chromosome model can be given as a diagonal of Q e R, [3]. The total genes is

$$g = n + m \tag{10}$$

The resulting chromosome is then

$$QR_z = [q_{11}\ q_{22}\ \ldots\ q_{nn}\ \ r_{11}\ r_{12}\ \ldots\ r_{nn}] \tag{11}$$

Population Model A population is a set of chromosomes. If a chromosome with g genes contains Q e R, then a population is represented by $QR_{n_{indiv}\times g}$, where n_{indiv} is the number of individuals in the population.

Fitness Model The fitness function evaluates each individual in a population and ensures the GA will find a optimal solution. The function is given by [3] as:

$$\begin{array}{lcl} K_z & = & LQR_z(A, B, Q_z, R_z) \\ A_z & = & (A - BK_z) \\ S_z & = & \dfrac{||V_z||^2||W_z||^2}{< V_z, W_z >} \\ F_{S_z} & = & \sum S_z \\ R_{S_z} & = & rank(S_z, F_{S_z}) \end{array} \tag{12}$$

where $z = 1, ..., n_{indiv}$, A_z is the closed-loop matrix for the gain vector K_z. S_z is the sensibility, V_z and W_z are eigenvectors of A_z. F_{S_z} is the fitness and R_{S_z} represent each individual fitness. Additionally, each individual is graded according to user-defined control goals.

The fitness model in Equation 12 scores each individual based on its current location in the s-plan. If the closed-loop poles from A_z are located inside the user defined eigenstructure (red zone) of Figure 3 the fitness model will ensure a high score for this chromosome.

Elite Selection The elite selection ensures that the best individuals (highest fitness) of a given population will survive in the next generation. This operator avoids the fittest individuals being lost in crossover and mutation operations, [1].

The algorithm for the Elite operator is given by

Algorithm 1 Elite

```
mean = sum_{j=1}^{m} f_i/m
for i = 1 → m do
    if f_i > mean then
        Select individual i
    end if
end for
```

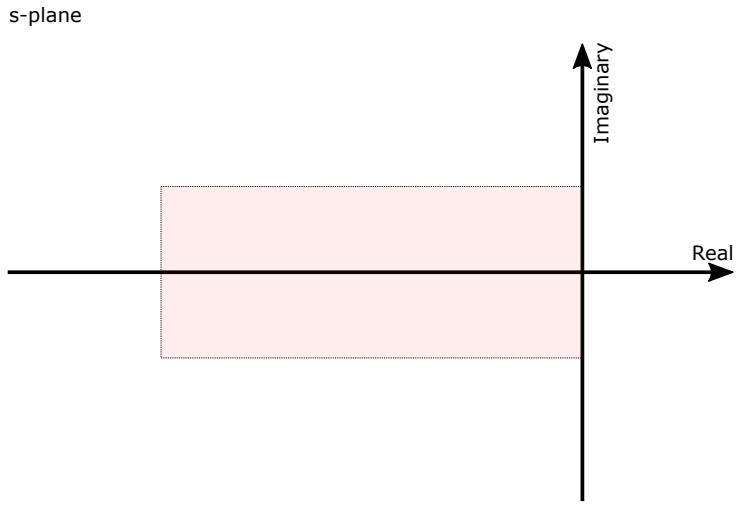


Figure 3: Fitness evaluation according to eigenstructure.

Roulette Selection This operator selects individuals based on the fitness. The operator can be given as a random experiment where

$$P[b_{j,t}\ selection] = \frac{f(b_{j,t})}{\sum_{k=1}^{m} f(b_{k,t})} \tag{13}$$

The algorithm for the Roulette selection is given by [6] as

Algorithm 2 Roulette

$p_i = f_i / \sum_{j=1}^{n} f_j$
$q_i = \sum_{j=1}^{i} p_j$
while $i < m$ **do**
  $x = random(0, 1)$;
  if $r < q_i$ **then**
    Select individual i
  end if
end while

Crossover Operator The crossover operator combines two individuals randomly in order to generate another two chromosomes.

The operator is given by [12] as

Algorithm 3 Crossover

$pos = random(1, ..., g)$
for $i = 1 \rightarrow pos$ **do**
  $Child_1[i] = Parent_1[i]$;
  $Child_2[i] = Parent_2[i]$;
end for
for $i = pos + 1 \rightarrow n$ **do**
  $Child_1[i] = Parent_2[i]$;
  $Child_2[i] = Parent_1[i]$;
end for

Mutation Operator This operator is essential as it avoids premature convergence, [12]. This operator randomly changes a gene of a given individual based on the probability of mutation, p_m.

The algorithm is given by

Algorithm 4 Mutation

```
for i = 1 → n do
    if random(0, 1) < p_m then
        Mutate gene
    end if
end for
```

4 Simulation Results

This section aims to present performance results of the Genetic Algorithm and a time-domain analysis of the control gains found by the proposed method.

4.1 Simplified Model

The simplified model of the space-craft that will be used is given by [2] as:

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 4.16 & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ 7.21 \end{bmatrix} \beta_z \tag{14}$$

4.2 GA Parameters and Results

For the proposed work, the parameters used for the GA simulation are presented in Table 1. The table also presents the control goals and constraints used in the controller design process, the sensitivity values that were include in order to enable the user to weight which parameter would be more valuable.

Table 1: Genetic Algorithm parameters.

Parameters	
Population size	120
Number of individuals produced by Elite	20
Number of individuals produced by Roulette	20
Number of individuals produced by Crossover	20
Number of individuals produced by Mutation	40
Number of new individuals	20
Mutation probability	5
Mutation factor	0.1
Goals and constraints	
Eigenstructure	$-8 \pm 1.5j$ to $0 \pm 1.5j$
Settling time	$t_s < 3$ sec.
Rise time	$t_r < 1$ sec.
Overshoot	$\%OS < 40\%$
Sensitivities	
Eigenstructure	1
Settling time	1.2
Rise time	1.2
Overshoot	1.5

Initial population Initial population was randomly created with 120 individuals. Mean fitness of this population was 1.625 and the best individual presented fitness of 3.9. As can be noted in Figure 4, initial population presented good diversity that is essential to avoid local maxima or minima.

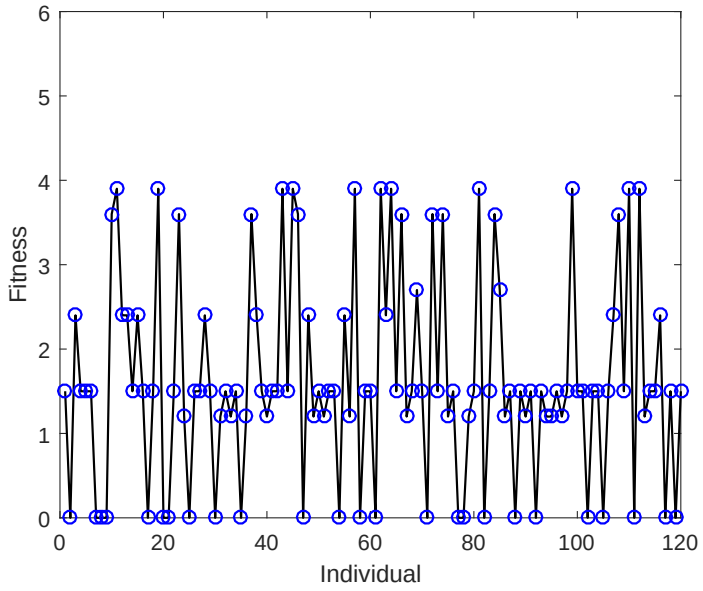


Figure 4: Initial population fitness.

Final population Final population mean fitness was 4. As can be noted in Figure 5, diversity was reduced suggesting that GA is close to the stop criterion. Last individuals of this population presented poor fitness as they were recently created.

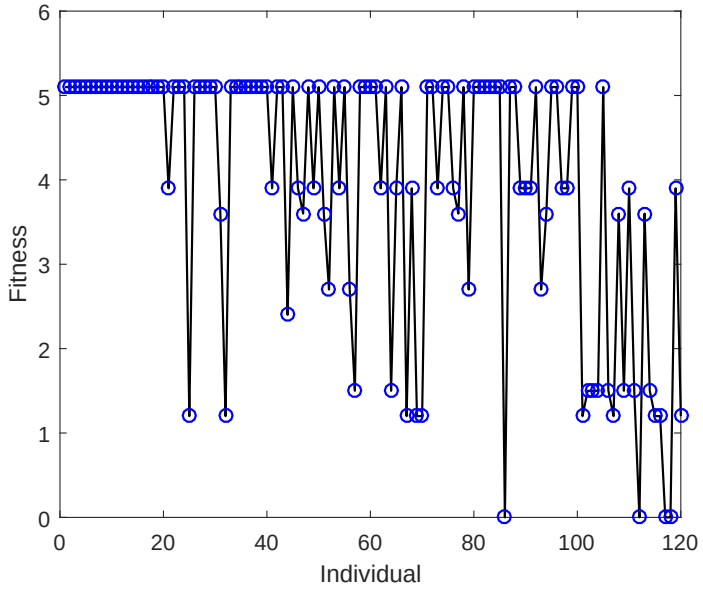


Figure 5: Final population fitness.

Fitness Evolution Figure 6 highlights the evolution of the mean fitness of populations. It can be noted that GA met the stop criteria with 50 iterations.

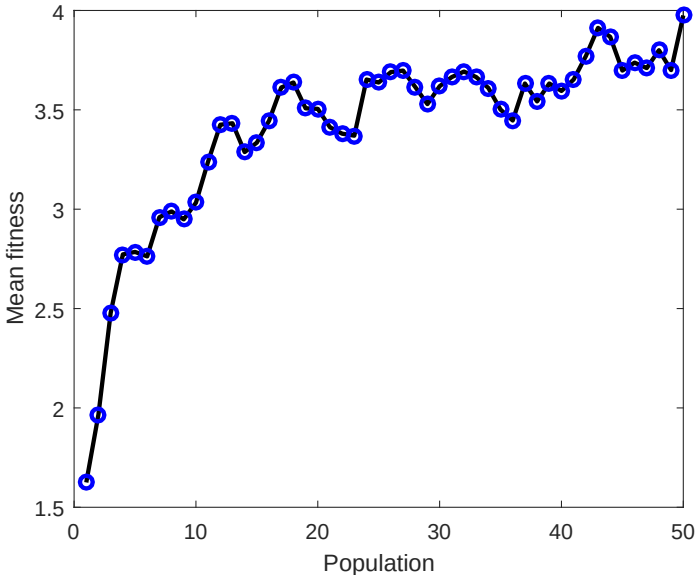


Figure 6: Fitness evolution.

Statistical analysis A Monte-Carlo simulation suggests that the algorithm converges with 30 populations as depicted by Figure 7.

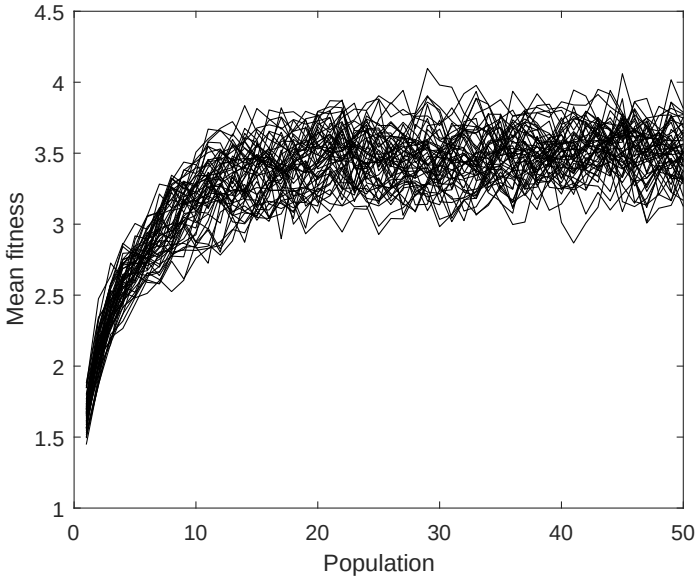


Figure 7: Monte Carlo simulation.

Additionally, for the proposed eigenstructure, goals and constraints, given in Table 1, the algorithm suggests that $0.1 < Q_{11} < 0.2$, $Q_{22} < 0.1$, $Q_{33} \approx 0.5$ and $R < 0.1$ often result in optimal gains, as can be seen in Figure 8.

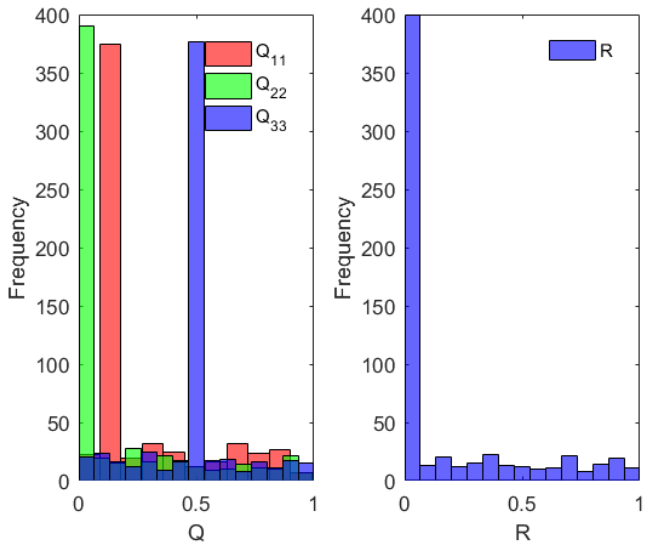


Figure 8: Histogram of Q and R.

4.3 Time domain analysis

The fittest chromosome for the 35-th population represents the following weighting matrices

$$Q = \begin{bmatrix} 0.1523 & 0 & 0 \\ 0 & 0.0892 & 0 \\ 0 & 0 & 0.5 \end{bmatrix} \text{ and } R = 0.089 \text{ that leads to } K = [3.5156 \quad 1.4023 \quad -2.3570]$$

The closed-loop step response for these gains is represented in Figure 9. The time domains specifications for this closed-loop step response follows: $t_s = 3.655$ seconds, $t_r = 0.429$ seconds and $\%OS = 34\%$.

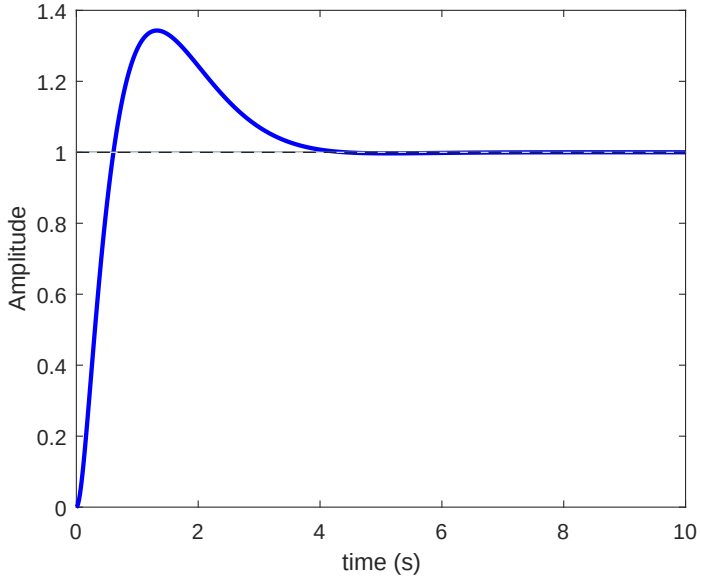


Figure 9: Step response for the controlled plant with matrices produced by the GA.

These specifications are due to the location of the closed loop poles inside the desired eigenstructure, as can be noted in Figure 10.

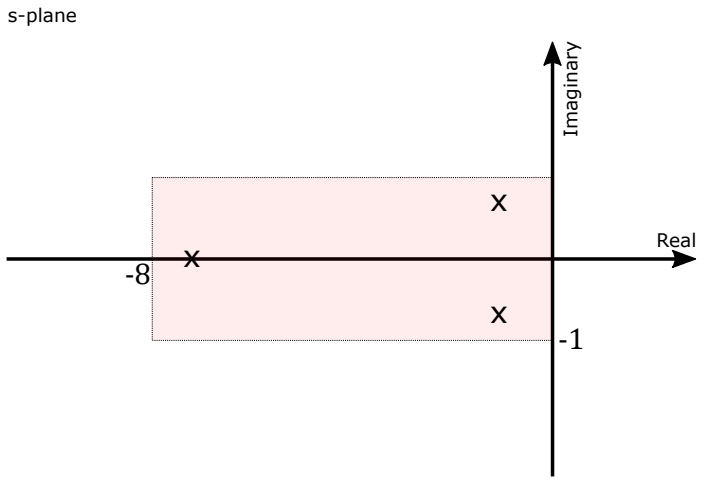


Figure 10: Poles in the desired eigenstructure.

Algebraic Method Comparison

The corresponding weighting matrices obtained via the methodology proposed in [9] are:

$$Q = \begin{bmatrix} 0.0048 & 0 & 0 \\ 0 & 0.0179 & 0 \\ 0 & 0 & -0.0049 \end{bmatrix} \text{ and } R = 0.001 \text{ that leads to } K = [0.6958 \quad 2.1909 \quad 0.3477]$$

As can be seen in Figure 11 the closed loop step response produced by the algebraic approach is not acceptable as it does not follow the reference line (dashed black) in a finite time. Also, the weighting matrix Q presents a negative term that is not in accordance with the theory presented in Chapter 2. Furthermore, algebraic or analytical methods often requires expertise on the dynamic behaviour of a system. Usage of evolutionary approaches in this case is very welcomed by engineers since they often bring better results in a short time.

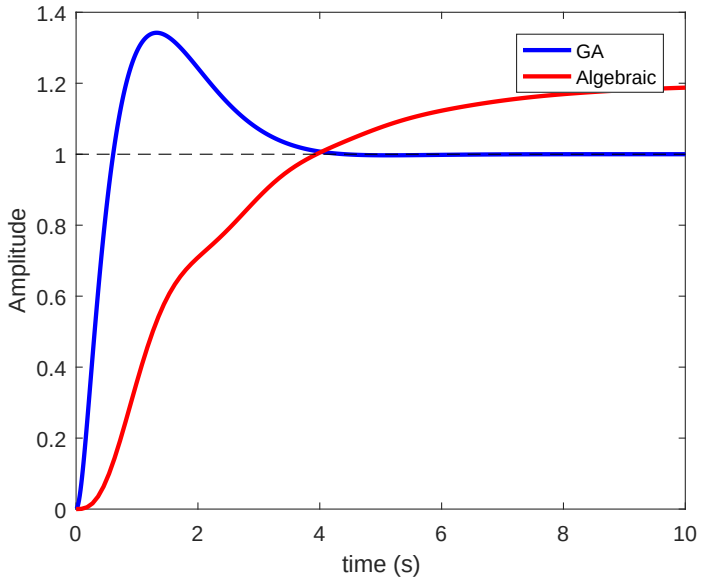


Figure 11: Step response comparison.

Comments on GA approach The algorithm converges in a finite number of iterations. Although, the time domain analysis shows that the settling time goal is not met, this is due to sensitivity values in Table 1. For this simulation, as the overshoot sensitivity is greater, the algorithm prioritize this parameter. Thus, in any other simulation with settling time sensitivity greater than overshoot the algorithm would find a satisfying solution.

In order to avoid the excitation of undesired dynamics mentioned in [2], the rise time goal must be made greater, however this effect was neglected here as this is not the focus of the research.

The effect of each parameter was evaluated. Mutation probability and mutation factor are the key parameters as they directly affect the speed of convergence. Very high or very low values of these parameters make the GA diverge and not to find a solution. Eigenstructure size can also smash the convergence, in this case the parameter was made smaller over time.

5 Conclusion

In this paper, a genetic algorithm for the control gains search problem of the satellite launcher attitude controller gains was proposed since currently techniques require prior experience about the problem and often result in inefficient controller.

Overall results show that the proposed method reaches the design specifications with 30 iterations with a population of 120 elements. The study also suggested the values of the weighting matrices ($0.1 < Q_{11} < 0.2$, $Q_{22} < 0.1$, $Q_{33} \approx 0.5$ and $R < 0.1$) to reach the design specifications. Indeed, usage of evolutionary techniques speeds up the search process and reduce design costs. Consequently, it is believed that the proposed approach can be used instead analytical methods.

For future work the authors will propose new fitness model approaches, usage of other evolutionary algorithms - neural networks or fuzzy logic - in the search problem and refine the control problem to a more realistic one.

Acknowledgements

The authors would like to thank Professor Alain Giacobini, Instituto Tecnológico de Aeronática, for his valuable comments on this work and Fundação de Amparo à Pesquisa e ao Desenvolvimento Científico e Tecnológico do Maranhão for the great opportunity to obtain a Master Degree.

References

[1] Ulrich Bodenhofer. Genetic algorithms: Theory and applications. 2004.

[2] D Carmona and WC Leite Filho. Analytical method for computing controller gains of a satellite launcher. *IFAC Proceedings Volumes*, 37(6):1019–1023, 2004. doi:10.1016/S1474-6670(17)32313-3.

[3] J. V. da Fonseca Neto, I. S. Abreu, and F. N. da Silva. Neural-genetic synthesis for state-space controllers based on linear quadratic regulator design for eigenstructure assignment. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 40(2):266–285, April 2010. doi:10.1109/TSMCB.2009.2013722.

[4] S. Das, I. Pan, K. Halder, S. Das, and A. Gupta. Optimum weight selection based lqr formulation for the design of fractional order piλdμ controllers to handle a class of fractional order systems. In *2013 International Conference on Computer Communication and Informatics*, pages 1–6, Jan 2013. doi:10.1109/ICCCI.2013.6466137.

[5] D. C. Dracopoulos and A. J. Jones. Adaptive neuro-genetic control of chaos applied to the attitude control problem. *Neural Computing & Applications*, 6(2):102–115, Jun 1997. doi:10.1007/BF01414007.

[6] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1989.

[7] Yu Han, James D Biggs, and Naigang Cui. Adaptive fault-tolerant control of spacecraft attitude dynamics with actuator failures. *Journal of Guidance, Control, and Dynamics*, 38(10):2033–2042, 2015. doi:10.2514/1.G000921.

[8] Sarthak Kukreti, Alex Walker, Phil Putman, and Kelly Cohen. Genetic algorithm based lqr for attitude control of a magnetically actuated cubesat. In *AIAA Infotech @ Aerospace*, page 0886. 2015. doi:10.2514/6.2015-0886.

[9] E. V. Kumar, J. Jerome, and K. Srikanth. Algebraic approach for selecting the weighting matrices of linear quadratic regulator. In *2014 International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE)*, pages 1–6, March 2014. doi:10.1109/ICGCCEE.2014.6922382.

[10] F Landis Markley and John L Crassidis. *Fundamentals of spacecraft attitude determination and control*, volume 33. Springer, 2014. doi:10.1007/978-1-4939-0802-8.

[11] M.H. Sangdani, A.R. Tavakolpour-Saleh, and A. Lotfavar. Genetic algorithm-based optimal computed torque control of a vision-based tracker robot: Simulation and experiment. *Engineering Applications of Artificial Intelligence*, 67:24 – 38, 2018. doi:10.1016/j.engappai.2017.09.014.

[12] Kumara Sastry, David E. Goldberg, and Graham Kendall. *Genetic Algorithms*, pages 93–117. Springer US, Boston, MA, 2014. doi:10.1007/978-1-4614-6940-7_4.

[13] Pyare Mohan Tiwari, S. Janardhanan, and Mashuq un Nabi. Rigid spacecraft attitude control using adaptive integral second order sliding mode. *Aerospace Science and Technology*, 42:50 – 57, 2015. URL: http://www.sciencedirect.com/science/article/pii/S1270963815000024, doi:https://doi.org/10.1016/j.ast.2014.11.017.

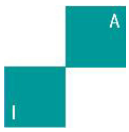
[14] Pyare Mohan Tiwari, S. Janardhanan, and Mashuq un Nabi. Attitude control using higher order sliding mode. *Aerospace Science and Technology*, 54:108 – 113, 2016. URL: http://www.sciencedirect.com/science/article/pii/S1270963816301420, doi:https://doi.org/10.1016/j.ast.2016.04.012.

[15] Vishal and J. Ohri. Ga tuned lqr and pid controller for aircraft pitch control. In *2014 IEEE 6th India International Conference on Power Electronics (IICPE)*, pages 1–6, Dec 2014. doi:10.1109/IICPE.2014.7115839.

[16] C. Wongsathan and C. Sirima. Application of ga to design lqr controller for an inverted pendulum system. In *2008 IEEE International Conference on Robotics and Biomimetics*, pages 951–954, Feb 2009. doi:10.1109/ROBIO.2009.4913127.

[17] D. Xue, Y. Chen, and D. Atherton. *Linear Feedback Control*. Society for Industrial and Applied Mathematics, 2007. doi:10.1137/1.9780898718621.

Inteligencia Artificial, 22(63) (2019), 162-195
doi: 10.4114/intartif.vol22iss63pp162-195



INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

Hybrid Adaptive Computational Intelligence-based Multisensor Data Fusion applied to real-time UAV autonomous navigation

Ângelo de Carvalho Paulino[1,2,A], Lamartine Nogueira Frutuoso Guimarães[2,B], Elcio Hideiti Shiguemori[2,C]

[1]ITA - Aeronautics Institute of Technology, São José dos Campos – 12.228-900, Brazil.

[2]IEAv - Institute for Advanced Studies, São José dos Campos – 12.228-001, Brazil.

[A]angeloacp@ieav.cta.br, [B]guimarae@ieav.cta.br, [C]elcio@ieav.cta.br

Abstract Nowadays, there is a remarkable world trend in employing UAVs and drones for diverse applications. The main reasons are that they may cost fractions of manned aircraft and avoid the exposure of human lives to risks. Nevertheless, they depend on positioning systems that may be vulnerable. Therefore, it is necessary to ensure that these systems are as accurate as possible, aiming to improve navigation. In pursuit of this end, conventional Data Fusion techniques can be employed. However, its computational cost may be prohibitive due to the low payload of some UAVs. This paper proposes a low-cost Multisensor Data Fusion application based on Hybrid Adaptive Computational Intelligence (HACI) - the cascaded use of Fuzzy C-Means Clustering (FCM) and Adaptive-Network-Based Fuzzy Inference System (ANFIS) algorithms - that have been shown able to improve considerably the accuracy of current positioning estimation systems for real-time UAV autonomous navigation, reducing the error in approximately $300cm^2$. In addition, the proposed methodology outperformed two other Computational Intelligence methodologies – Artificial Neural Networks and Regression Models, with 18 different tested approaches – in estimating an UAV position considering the Root-Mean-Square Error against the real trajectory. The generated Fuzzy Inference System has proved to be effective in providing an improved positioning estimation with a low computational burden, about 600 times faster than the fastest embedded sensor refresh rate.

Keywords: Data Fusion, Computational Intelligence, Unmanned Aerial Vehicles, Autonomous Navigation, Inertial Sensors, Positioning Estimation, ANFIS.

1 Introduction

Several applications are possible for aerospace assets. Among them, one can mention the integration and logistics for different sectors of the society, the assessment of natural or public calamities, and the monitoring of areas of government interest [1]–[3]. Manned aircraft with onboard remote sensing equipment, such as cameras, sensors, and radars, have been responsible for detailed tracking, data collecting and help in detecting threats, aiming at protecting and upholding the law [4]–[9].

However, for many applications, the use of manned aircraft is not feasible or desirable since the crew lives may be exposed to risk scenarios, like situations involving disasters or radiation. In many cases, this could be avoided. Therefore, an alternative to the use of manned aircraft is the use of Unmanned

ISSN: 1137-3601 (print), 1988-3064 (on-line)

Aerial Vehicles (UAVs) [9], [10]. Given some factors like their low flying altitude and slow speed, they are more difficult to be detected by conventional sensors [9]. In addition, some models cost only a fraction of manned aircraft. According to Cook [9], the UAVs, which historically were thought of as merely complementary to manned aircraft, have proven to be an excellent power asymmetry tool.

The world trend in the use of UAVs is undeniable. The use of such systems in the military plays an important role in surveillance, reconnaissance, target prediction, and even combat missions by operating in a wider range of conditions and scenarios than conventional aircrafts [9]–[12]. Regarding the civilian scope of aerospace robotics employment, there can be mentioned topographic surveys, mapping of areas of interest, precision agriculture, surveillance, and access control, remote sensing for geological and climatic research and various utilities [2], [3], [8], [12]. State-of-the-art applications include detection and monitoring of sea wildlife [13], thermographic inspection of photovoltaic plants [15], radiation detection [14] and identification of helicopter landing zones and airdrop zones in calamity situations [16].

Though, it is known that UAVs require a structured navigation control protocol, such as via Radio Frequency or via satellite, which makes it vulnerable to interference such as jamming or spoofing [10]. One solution to this problem is the use of autonomous navigation. However, even the use of Global Navigation Satellite Systems (GNSS) or Inertial Navigation System (INS) - systems widely covered in the literature - are not immune to faults or interference [17]. Ignoring the attitude and trajectory of an aircraft can cause the loss of the aircraft altogether, so it is essential to ensure the operational reliability, accuracy, and robustness of their navigation [8].

In this sense, the use of Data Fusion to enable more secure and robust navigation is a good alternative to the exclusive reliance on the usual navigation systems. The technology of Data Fusion plays an invaluable role in the search for increased navigational safety [23, 18]. Therefore, given the reality of the current applications of aerospace robotics, promoting research and development in the area of Data Fusion is imperative [1], [8], [12].

Thus, the main objective of this paper is to evaluate a Multisensor Data Fusion application, based on Hybrid Adaptive Computational Intelligence (HACI), through the fusion and integration of sensed data obtained by embedded sensors of the UAVs. The proposed approach aims to reduce the imprecision of current methods of positioning estimation, like GPS and INS, and to deliver solutions in a feasible time to the problem of real-time navigation of low-performance UAVs. The proposed methodology offers an improvement in the quality of the positioning estimation information provided as input to its In-flight Navigation and Attitude Control System – a system responsible for controlling the UAV's frame attitude and trajectory and for the Decision-Making process – or simply "control loop". It is hoped, therefore, to provide UAVs with safer and more robust navigation given more precise information is being used for the flight. It should be noted that the control loop modeling is not in the scope of this article.

The referred Fusion has been performed using two Computational Intelligence techniques in a cascaded way, i.e., Fuzzy C-Means Clustering (FCM) [60] and Adaptive-Network-Based Fuzzy Inference System (ANFIS) [61]. These techniques have shown to be promising to effectively increase the accuracy of the positioning estimation of the UAV held by the Global Positioning System (GPS), considering the Root-Mean-Square Error (RMSE) between the position taken as real and the estimated one. The methodology proposed in this work was compared with two other Computational Intelligence methodologies – Artificial Neural Networks and Regression Models – with 18 different tested approaches, outperforming them in the capability of enhancing the positioning estimation precision considering the RMSE against the real trajectory, as detailed in the next sections.

2 Data Fusion and related work

The systems most widely employed for the navigation of UAVs are the GNSS. The most known are the GPS and the Globalnaya Navigatsionnaya Sputnikovaya Sistema (GLONASS) [8], [17], [19]. Nonetheless, several studies show that these systems may present vulnerabilities, both from natural factors, such as the South Atlantic Magnetic Anomaly (SAMA), and human factors, such as jamming, spoofing and malicious blockages and interference [21]–[26].

Another widely used system in the literature is the INS. The problems involving its use is that low-cost INS can accumulate drift error that, if not corrected, could result in a substantial divergence between the estimated position and the real position of the UAV [8], [17], [24]. Thus, the use of Data Fusion of

multiple sensors for the positioning estimation has shown to be a feasible option to the dependence on one or the other system alone [27]–[29].

Along with technological development, the number of available sensors and data has been increasing. In general, it is seen that there is a need to use different sources of data to obtain more accurate estimates [8, 24]. In this context, Data Fusion techniques can be used to fuse the data originated from various sensors and thereby generate improved information [23, 24]. Figure 1 shows possible combinations of differences between data sources.

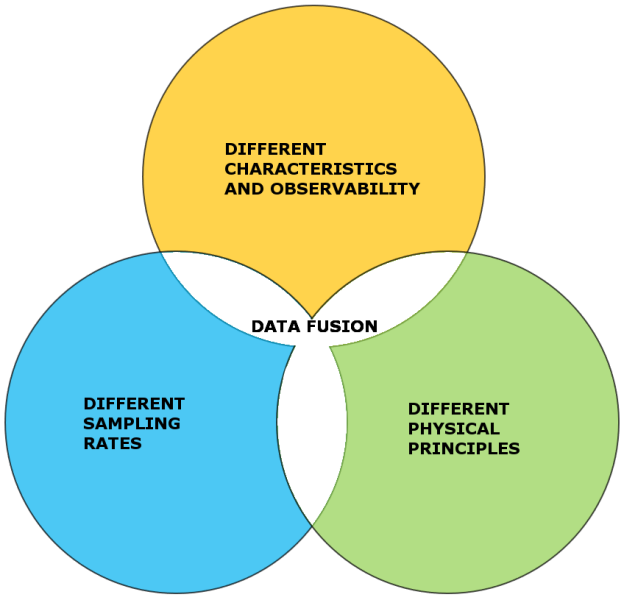


Figure 1: Possible combinations of differences between data sources addressed by Data Fusion.

In the literature, several recent works deal with the improvement of the positioning estimation using Data Fusion of GPS and INS. The positioning estimation is done through different methods, as seen, for instance, in [30]-[34], where is shown the fusion of GPS and INS to bridge the period of GPS outages for vehicular navigation. The authors in [35] use random forest regression to enhance the positioning estimation of an airplane, whereas [40] employs Kalman filtering to a high-performance ultra-tightly coupled GPS/INS. In [36]-[39] is seen the use of adaptive networks to perform the Multisensor Data Fusion. Additionally, [41, 42] apply Fuzzy Inference Systems to perform the fusion. Finally, it is seen in [43] the use of ANFIS as a sub-remedy system to temporarily replace the GPS positioning estimation used as input to a Kalman filter during GPS outages. Although, according to the authors, the methodology implies an average performance improvement of not more than 40%, the great counterpoint to this work is that the employed methodology is subject to all the problems related to the use of the Kalman filter. Problems include imprecision about noise information that could lead to position and velocity deviations and filter divergences due to system linearization, which implies in no guarantee of optimality [45, 48, 44].

The Data Fusion approach has shown to be effective in reducing the imprecision of the positioning estimation process. Nevertheless, it is worth mentioning that the only papers found regarding the improvement of the autonomous navigation of a UAV using ANFIS as the main estimator was published by the authors in [46] and [47].

With that said, Data Fusion is a process that acts through the detection, association, correlation, estimation and combination of data of different sensors [18], [48], [49]. This may be invaluable for the collection of useful information aiming at the analysis of the various scenarios for Decision-Making [50]. Regarding UAVs, the higher the exposure to an unfamiliar and unstructured scenario, the higher the risk in its operation due to eventual obstacles that may arise in its trajectory [10].

The use of Data Fusion from several sensors has many advantages, such as improved estimates of information about the target, such as position and speed, when several identical sensors are used in com-

bination in an optimized way, acquiring statistical advantages by increasing the number of observations of the same event; improvement of the observation process using relative data between multiple sensors, which opens up new possibilities for combining and extracting information; and greater observability, since a sensor can capture data that another cannot, either by divergences in their physical nature, position or refresh rate, for example, reducing errors and complementing information [12], [48].

However, the use of several sensors can imply large volumes of data, sometimes noisy, with different characteristics, unconjugated sampling rates or even nature and physical principles that are de-correlated with each other. Therefore, considering the complexity of such a data set, conventional Data Fusion techniques may not present timely solutions to the navigation problem, given, in general, its high computational cost [48], [50], and also considering the need for real-time processing in the case of an ongoing flight.

Hence, a Data Fusion application based on HACI is proposed, aiming at making feasible its employment in real-time flight applied to the problem of navigation, given the mathematical simplicity of the tool used to estimate the position [51] and its inherent low computational cost.

3 FCM and ANFIS

The tool used to perform the Data Fusion was the Fuzzy logic, which is the representation of uncertainties in the mathematical form [53]. Regarding aerospace systems, the uncertainties arise mainly from their complexity and non-linearity, both as a result of inaccuracies in the mathematical model and the number of variables that interact epistatically in obtaining the results [70]. In the field of multivalued logic, it is important to emphasize that this logic, considered as of "infinite" values, aims to formalize the representation of vagueness using linguistic terms [54]. Thus, it is an efficient solution to the fusion of vague, noisy or partial data [50].

Focusing on the applications, Fuzzy logic is extremely flexible for the most diverse uses since it uses "membership functions" (MFs) that model and quantify the meaning of the symbols [50, 70] for their deductive apparatus in an intuitive and close to the natural language way [54], through well-defined "if-then" relationship rules [70]. A good example of the application of Fuzzy logic is its use to improve the accuracy of the landing procedure of an UAV [38]. Another is the joint use with Computational Vision tools for estimating the position of an UAV in real-time flight, based on landmark recognition, as seen in [52].

Fuzzy logic fits into the Soft Computing area, which refers to the use of computational tools to find approximate solutions to approximately formulated problems, creating some tolerance for imprecision and uncertainties aiming at modeling systems in a more treatable, robust and cost-effective way, with greater computational power saving and communication bandwidth [70]. Thus, Soft Computing is a good choice for the implementation of intelligent control of complex systems, such as aerospace systems. Other areas of Artificial Intelligence are also covered by Soft-Computing, among them: Artificial Neural Networks, Probabilistic Reasoning, Expert Systems, and Genetic Algorithms [70].

In the case of neural networks, Kothari and Bhattacharjee [56] state that it is a stochastic and heuristic tool that learns the relation between the parameters and their responses when trained with a finite number of input data and predicts the values of a new set of independent variables based on their training (learning) experience. It should be noted that, from the optimization point of view, "learning" is equivalent to minimize the global error function [58]. With that said, neural networks are interesting because they can handle some complex problems through a powerful and flexible framework [59].

According to Teodorović [57], the chosen structure of a neural network model can influence the convergence rate of a training algorithm and even determine the type of learning to be used. Also according to the author, the training algorithms are very simple mechanisms that adapt the weights of the branches of a network, requiring for each node only locally available data. For this reason, he concludes that its implementation generally does not involve complex calculations and, thereby, powerful computational configurations are not necessary. For this work, the chosen architecture was the Multi-Layer Perceptron (PMC), one of the most used in the literature [55].

In this paper, two techniques of Computational Intelligence were applied sequentially: Fuzzy C-Means Clustering (FCM) and Adaptive-Network-Based Fuzzy Inference System (ANFIS). The first one aims to group a data set in centroids according to a non-crystalline similarity from Fuzzy sets [60], while the

second one is an Adaptive Network – a class of feed-forward Neural Networks with supervised learning capability – that emulates the behavior of a Fuzzy Inference System (FIS) System [61].

In the case of a flight data (log) containing real data captured from embedded sensors, such as the one used in this work, there is a significant amount available for extracting information. This is a task that is not trivial [62]. Consequently, techniques that use methods that combine knowledge areas such as machine learning, pattern recognition, and information retrieval can be employed towards a better analysis of the data [62].

3.1 The Fuzzy C-Means Clustering (FCM) algorithm

Like any clustering technique, the FCM assists in obtaining an intuition about data whose volume makes the analysis by humans difficult. For this reason, algorithms such as FCM are widely used for exploratory data analysis in the sense that they act as meta-learning tools [62].

According to Saxena et al. [63], clustering techniques can be divided into Hierarchical, such as BIRCH, CURE, ROCK, and CHAMELEON, or Partitional, such as k-means, PAM, CLARA, CLARANS, FCM, EM, DBSCAN, and CLICK. As stated in [64], clustering techniques can be divided, according to the statistical point of view, into a probabilistic approach, based on a model that uses probability distributions, or in a non-parametric approach, which essentially employs methods based on objective functions of similarity or dissimilarity. Also according to Yang and Nataliani [64], the best-known probabilistic algorithm is the Expectation-Maximization (EM), while the k-means is the most famous among the non-parametric. Another example of a non-parametric algorithm is the FCM, used in this work.

Crisp clustering algorithms consider that each sample may belong to one, and only one centroid. This prevails in the fact that such algorithms present difficulties in modeling systems that exhibit uncertainties of non-statistical nature, such as noises [60]. Thus, the Fuzzy Set Theory proves to be substantially useful since it formalizes the mathematical representation of vagueness/uncertainty [65], which can be applied to the clustering problem.

Unlike crisp clustering techniques, such as k-means and x-means, the FCM allows one sample to belong to a greater or lesser degree to more than one centroid, determined by a Fuzzy membership function [63]. This expands both the ability of differentiation and of agglomeration between samples and allows, for example, the identification of noises in regions of overlapping membership [60]. A simple illustration of the fact is depicted in Figure 2.

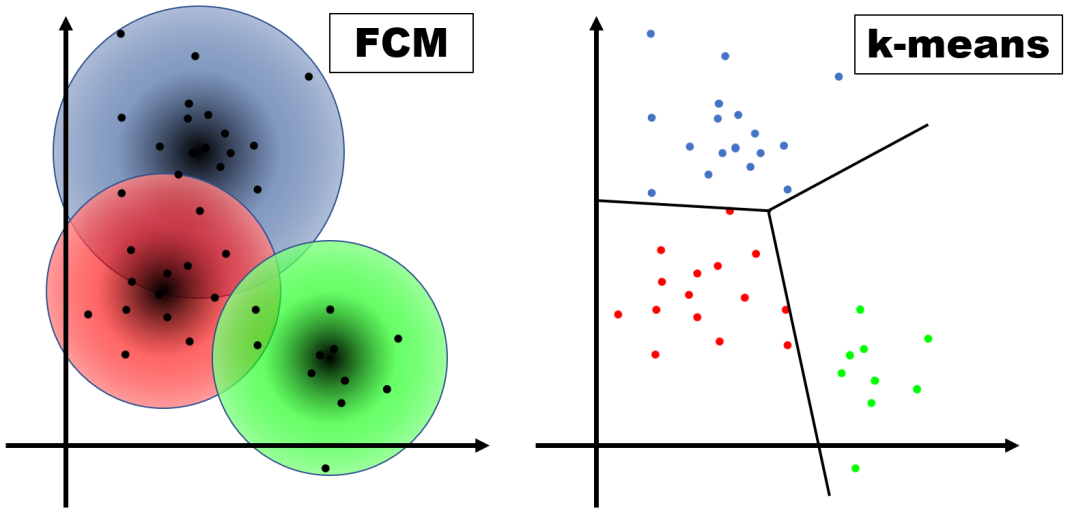


Figure 2: Simplified representation on how FCM and k-means algorithms cluster the data.

Furthermore, the FCM algorithm is based on the minimization of the following objective function [63]:

$$J_m = \sum_{i=1}^{N}\sum_{j=1}^{c} u_{ij}^m \|x_i - v_j\|^2; 1 < m < \infty \quad (1)$$

where N is the number of samples, c the number of centroids, m a real number that controls the degree of overlap between the centroids, u_{ij} (Fuzzy Partition Matrix) the level at which an observation x_i belongs to a cluster j, v_j the j-th cluster's centroid, and $\| * \|$ represents any norm that expresses the similarity between the measured data and the centroids, such as the Euclidian norm in (1).

Thus, the FCM algorithm is comprised of the following steps [60], [63], [66]:

1. The initial values of the u_{ij} membership functions of the centroids (c) are randomly adjusted;
2. Calculate the prototype centroids (j = 1, ... ,c):

$$V_j = \frac{\sum_{i=1}^{N} u_{ij}^m x_i}{\sum_{i=1}^{N} u_{ij}^m} \quad (2)$$

3. Calculate u_{ij} according to:

$$u_{ij} = \left[\sum_{k=1}^{c} \left(\frac{\|x_i - v_j\|}{\|x_i - v_k\|}\right)^{\frac{2}{m-1}}\right]^{-1} \quad (3)$$

4. Calculate Objective Function J_m (1); and
5. Repeat steps 2 to 4 until J_m reaches the desired convergence or the iteration number exceeds a stipulated limit.

In this paper, the centroids calculated by the FCM algorithm are then used as the starting point for the generation of initial membership functions that will be optimized by the ANFIS algorithm.

3.2 The Adaptive-Network-Based Fuzzy Inference System (ANFIS) algorithm

The ANFIS algorithm, developed by Jang [61], uses a hybrid learning scheme and can construct an input-output type mapping based on human knowledge, represented by if-then rules and associated pairs of inputs and outputs. In this sense, it is able to learn to map highly nonlinear functions [61], [67]. Briefly, this algorithm can be seen as a flexible mathematical structure that can address the approximation of a large class of complex nonlinear systems with a desirable level of accuracy [67], without, however, losing the mathematical rigor [51], [65].

Concerning the Fuzzy Logic, it should be emphasized that there is no standard methodology for transforming human knowledge or experience into a rule base for a FIS. In addition, there is a need to make adjustments to the established membership functions to both improving performance and minimizing systems' errors [61]. Hence, finding appropriate rules and applying appropriate adjustment methods are key issues [68], and the adequate representation of these rules is crucial.

Thus, the form of treatment of if-then rules proposed by Takagi and Sugeno (T-S) is particularly interesting since it employs parameters of mathematical functions in place of the classic linguistic expressions [51], [68] and has fuzzy sets involved only in the part of the premises [61]. Here is an example of T-S type fuzzy reasoning:

$$Rule\ 1: IF\ x\ is\ A_1\ AND\ y\ is\ B_1,\ THEN\ f_1 = p_1 x + \ q_1 y + r_1$$

$$Rule\ 2: IF\ x\ is\ A_2\ AND\ y\ is\ B_2,\ THEN\ f_2 = p_2 x + \ q_2 y + r_2$$

being x and y the inputs, A_i and B_i fuzzy sets, f_i the outputs of same Universe of Discourse as the input variables and p_i , q_i and r_i calculated parameters during the learning.

In this approach, the consequent part is described by non-fuzzy equations of fuzzy input variables (antecedents) [61]. This particular fact allows its application in several machine learning algorithms, among which is the ANFIS algorithm.

By using a hybrid scheme of Fuzzy Logic and Neural Networks such as Adaptive Networks, an ANFIS Network harnesses advantages of these two large areas of Computational Intelligence [69]. While Fuzzy Logic can handle uncertainty in inputs and outputs, Adaptive Networks can address the uncertainty of the model and thus, when employed synergistically, can be applied to a wide variety of complex problems [70]. Figure 3 illustrates the complementarity of these two paradigms.

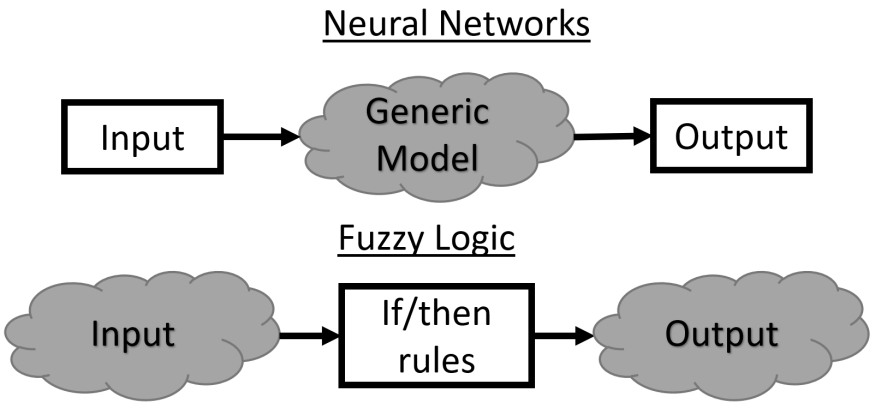


Figure 3: Neural Networks and Fuzzy Logic and how they deal with uncertainty (clouds).

The complementarity depicted in Figure 3 is achieved by means of an Adaptive Network structure, according to Figure 4 [61], [71]:

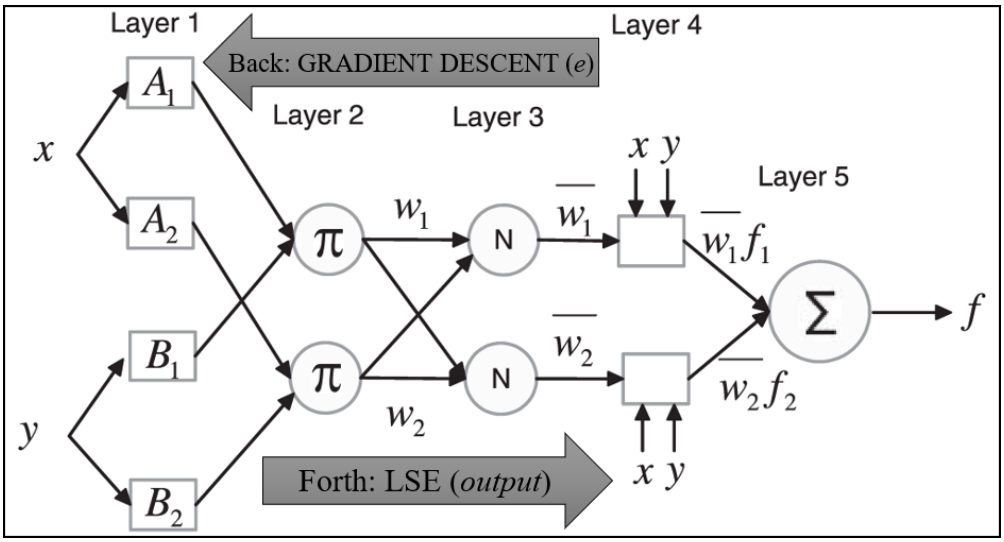


Figure 4: Structural model of an ANFIS Network. The arrows indicate the direction of the hybrid learning scheme. Source: Adapted from [71].

Each layer of the structure has the following constitution:

1. Fuzzy sets that determine the degree of membership of inputs x and y : $\mu_{A_i}(x)$ and $\mu_{B_i}(y)$.
2. T-norm operator, such as the product $w_i = \mu_{A_i}(x) \cdot \mu_{B_i}(y)$, these being the membership functions of each fuzzy variable as a function of the inputs.

3. Normalization of activations, according to:

$$w_i = \frac{w_i}{w_1+w_2} \quad (4)$$

4. Defuzzification according to consequent's parameters p_i , q_i and r_i .

5. Output as a function of the sum of activation of all the rules (composition):

$$f = \sum_i w_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \quad (5)$$

Once the structure of the Adaptive Network was detailed, the steps of the ANFIS algorithm applied to the context of this work are as follows:

1. Load the *training* and *validation* data and determine the input parameters and number of epochs;
2. Generate initial membership functions (antecedents);
3. Perform the training of the ANFIS System by means of a hybrid optimization of the FIS model (Figure 4) until a predetermined stop criterion is reached (maximum error, for example):
   (a) Antecedents: Fixed in the forward pass (forth) of the algorithm and optimized in the backward pass (back) by the Gradient Descent method, based on the outputs of the model; and
   (b) Consequents: Optimized on the forward pass (forth) by the Least Square Optimization (LSE), based on error rates (e), and fixed at the backward pass (back)
4. Validate training results - trained ANFIS systems - according to *validation* data; and
5. Verify the efficiency of the trained model by measuring the degree of accuracy of the model in predicting the output, given certain inputs, through RMSE.

The use of the ANFIS Networks proposed in this work is performed with real data obtained in flight as explained below. Several configurations were tested for four dimensions, namely: number of inputs (sensors used); number of membership functions for each input; number of training epochs; and data division for training. The results are discussed in section 5.

4 Materials and methods

The methodology chosen for this scientific work - which is now detailed - promotes the Data Fusion through HACI, using real measured data of several embedded sensors from flight logs. These logs were generated in real-time. Once the data is presented, the chosen algorithms perform a non-linear mapping from the input data (coordinate and chosen sensors) to the output data (real position), generating trained Fuzzy Inference Systems. The trained FIS are then used to perform the positioning estimation of the UAV, as shown in Figure 5. Several tests presented at section 5 show the evaluation of the proposed methodology effectiveness.

The Real-Time Kinetic GPS sensor data, treated by a Kalman filter (KF-GPS-RTK), was used as ground truth. Given its notorious precision, this data makes it possible to measure the errors of the positioning estimates generated by the Data Fusion System. It should be noted that the dataset of a flight log refers to all samples taken between takeoff and landing with a given sampling rate.

According to Al-Hmouz *et al* [71], the choice of training data is of paramount importance. However, given the complexity of data such as those recorded in flight logs of an UAV, choosing the *training*, *validation*, and *generalization* data sets that provide the best results is not trivial. Thus, Cross-Validation (CV) techniques can be employed to minimize the risk - or error - in the choice of data groups, bearing in mind that the data groups are chosen in order to be independent of each other [73].

This popular algorithm selection strategy [73] works with a data division that aims to measure the generalization power of a given model, that is, how good is its ability to make predictions on newly

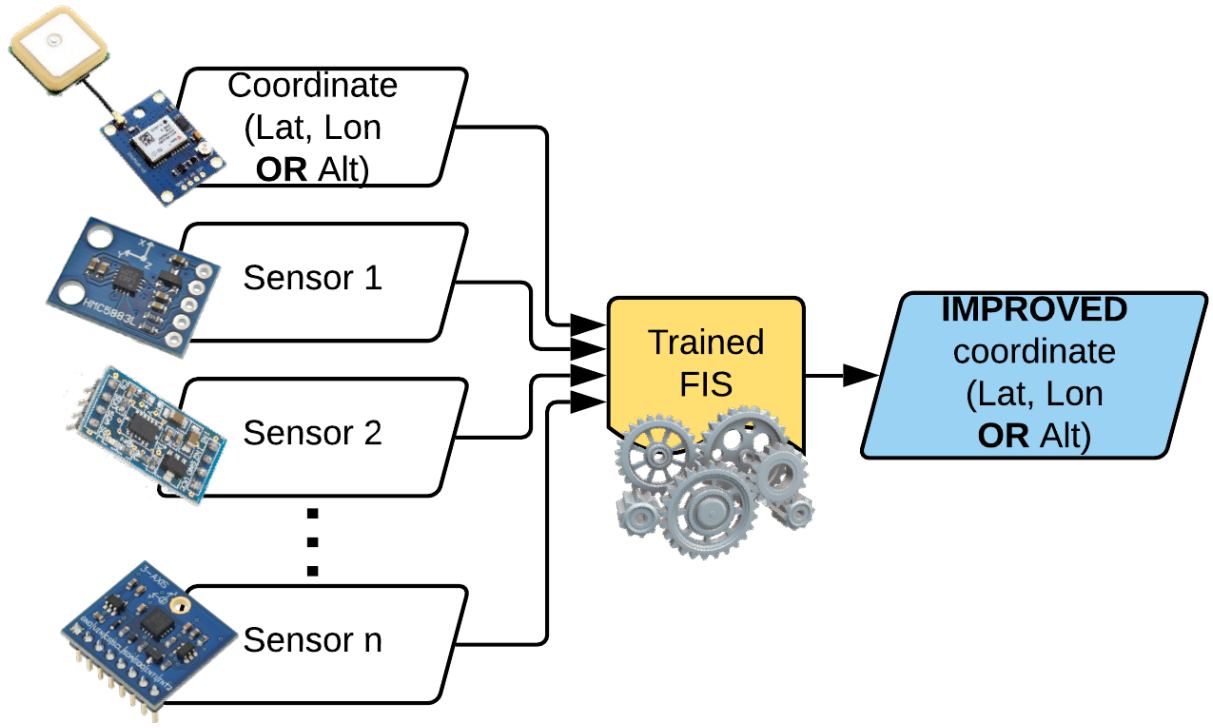


Figure 5: Illustration of the methodology applied to the improvement of the positioning estimation of a UAV using Hybrid Adaptive Computational Intelligence.

presented data. According to Varoquaux [74], CV is the best tool available in this assignment, since it is the only nonparametric method for the generalization capability test. Therefore, once the ANFIS Networks trained in this work are aimed at the real-time flights, it is hoped that such capacity will be as high as possible.

The chosen technique was a modified version of the K-fold Cross-Validation (KFCV), which, in addition to presenting a mild computational cost, divides the data into K sub-samples of approximately equal cardinality [76, 73]. This way, each sub-sample successively exerts the role of the validation set [73] and generalization set, when present. The common use of cross-validation techniques considers a set of $K-1$ sub-samples for training and 1 for validation. The modification proposed in this work consists that now $K-2$ sub-samples are used for the training set, 1 is used for the validation set, and another 1 sub-sample is used for the generalization set.

In the literature, optimal values for K are commonly chosen between 5 and 10 [74], [75]. However, a small number of samples may imply bad results [74]. Considering that in the literature it is commonly seen that something about 70% to 80% of data is used for training, the number 7 was chosen for the K value since the training set will have 5/7 of the sub-samples, or about 71,4% of the data being used as training set as well. With that said, the chosen technique will be henceforth called 7FCV in this work.

The whole dataset was permutated, and the data samples assigned to each sub-sample randomly taken. Then 5 of the 7 sub-samples are chosen to compose the training set; 1 to the validation set; and 1 to the generalization set. This way, $C_5^7 * C_1^2$ (42) independent combinations (groups) of the 7 sub-samples were generated for the application of the proposed methodology, numbered from 1 to 42. The 7FCV data division scheme is used only from the fourth series of experiments and on, once the first three series of experiments have their own peculiarities regarding the data division, as demonstrated in section 5.

4.1 Methodology phases

The proposed methodology is implemented in two phases: the first generates a Fuzzy Inference System through Data Fusion while the second uses the trained FIS to fuse new data presented to it. For the

latter, the data can be presented in real-time flight or simulated flight time, in the case of previous flights (logs). Regarding previous flights, it is assumed that the next position to be provided for the Data Fusion System has already been duly corrected by the In-flight Navigation and Attitude Control System. The flowcharts of each phase are shown in Figures 6 and 7. It is stated, however, that the red items in the flowcharts are outside the scope of this study and, therefore, will not be detailed.

4.1.1 Training phase

The training phase (Figure 6) aims to generate a FIS that can be used for the Data Fusion positioning estimation once new data is presented to it. Thus, the flight log will provide the dataset to be used for the training, which includes the data of all embedded sensors. Hence, sensor data such as GPS estimates are taken to provide the estimated positions at each instant of time. KF-GPS-RTK sensor data supplies the true position and other sensors data that will be used for the fusion are taken as well. The data is loaded in batches that correspond to the entire flight time, from takeoff to landing.

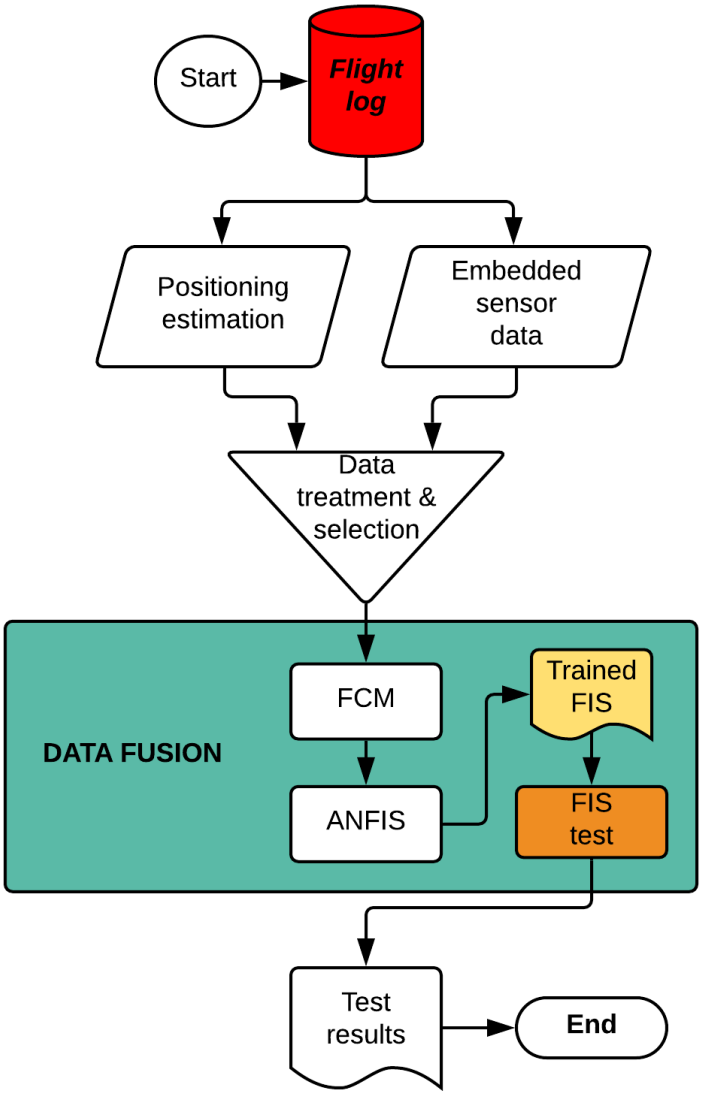


Figure 6: Flowchart of the training phase of the proposed methodology.

Then the cross-validation scheme 7FCV is responsible - when present - for generating the 42 combinations of the 7 different sub-samples at this time. After that, the data is presented to the FCM algorithm for the initial FIS creation. For the FCM stage, the degree of centroid overlapping (m) was empirically adjusted at about 1.2 for the significant majority of the tests. The initial FIS generated by the FCM algorithm is then optimized by the ANFIS algorithm, beaconed by the validation sub-sample. Once trained, the FIS is subjected to efficiency and efficacy tests in the positioning estimation of an UAV - generally with the generalization sub-sample data. The results are then recorded for performance control and eventual re-use. This ends the training phase.

4.1.2 Employment phase

The employment phase (Figure 7) intends to apply the FIS trained in the previous phase in a real or simulated flight, in order to improve the accuracy of positioning estimation of an UAV. Thus, from the point at which a flying UAV generates positioning and embedded sensors data, such data is processed and selected. It is worth mentioning that, unlike the training phase, where the data is loaded in batches, in the case of real-time use the data is processed timely at each given instant of time. Then, the chosen data is provided as input to the trained FIS to perform the Data Fusion, as seen in Figure 5, where a more accurate positioning estimate is obtained.

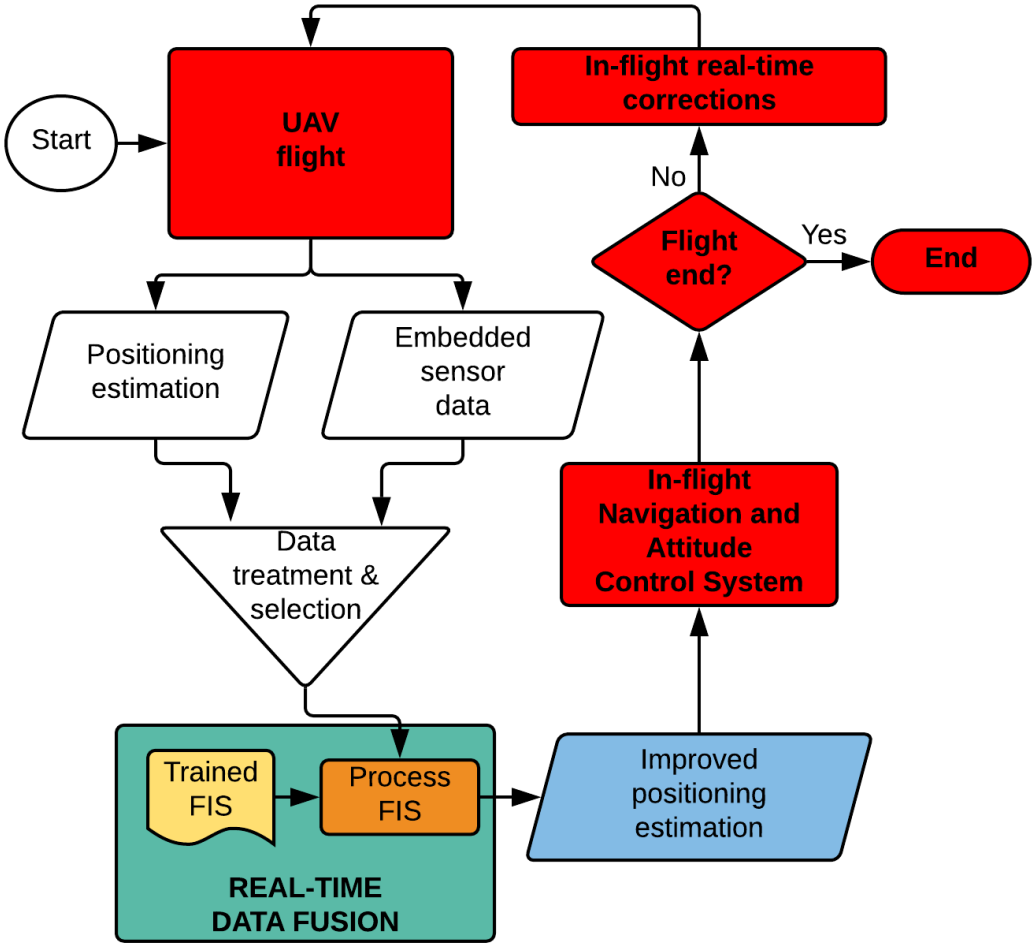


Figure 7: Flowchart of the employment phase in real or simulated time of the proposed methodology.

The improved positioning estimation is then used to feed in the In-flight Navigation and Attitude Control System. Such a system - which will not be treated within the scope of this Work - will carry out the necessary corrections in real flight time and will feedback the shown flowchart given the flight end was not reached. Thereafter, the UAV flight follows normally with a corrected positioning estimation at each new iteration until landing.

Once the proposed methodology is shown, the details of the flight data used for practical experimentation will be explored.

4.2 WITAS Dataset

A flight was performed by the WITAS Project [72] in an area centered at the coordinates 58° 29' 41.58.6" N and 15° 06' 09.0" E, in Sweden. The UAV used is based on the commercial model "Yamaha Rmax UAV", equipped with avionics developed at the Department of Computer & Information Science - Linköping University [72].

In Figure 8 the flight path of the UAV is presented, recorded with a Real-Time Kinetic GPS and refined by a Kalman Filter (KF-GPS-RTK), with sub-meter accuracy. The path was recorded in three dimensions: Latitude, Longitude, and Altitude. The scale to the right of the graph reflects the Altitude values, in order to make it easier to understand the graph. The starting and endpoints are shown as well.

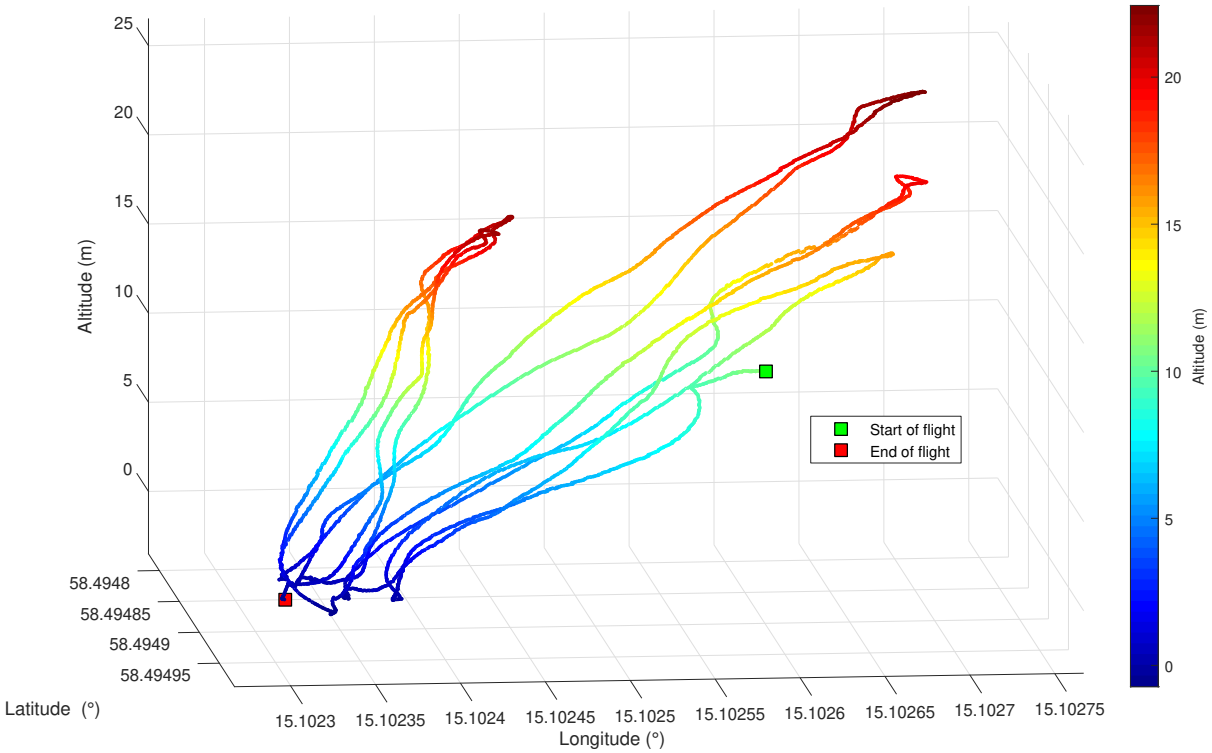


Figure 8: Trajectory of the flight performed by the WITAS Project UAV.

The UAV avionics has captured 139 different types of real-time flight data, including data from embedded sensors such as compass, barometer, GPS, and INS (Accelerometer and Gyroscope), among others. For this work, the data listed in Table 1 were extracted.

The data, arranged in 66483 lines (interpolated temporal samples by the maximum update rate between all sensors) by 11 columns (measured variables), were organized as files, and those specified in Table 1 were accessed directly from these files. Only for the two last series, this work used a dataset of 3319 non-interpolated temporal samples (cropped at the least update rate between all sensors), also with 11 columns. For all experiment series, the samples were randomly permutated.

Table 1: Data used in this work.

SENSOR	REFRESH RATE	EXTRACTED DATA	NO. OF VARIABLES
Embedded GPS	10 Hz	Clock, Latitude (GPS-Lat) and Longitude (GPS-Lon)	4
Embedded Accelerometer (ACCEL)	66 Hz	Acceleration fields - X, Y and Z axis	4
Embedded Gyroscope (GYRO)	200 Hz	Angular rates relative to Euler angles (Roll, Pitch and Yaw rates)	4
KF-GPS-RTK	50 Hz	Reference/truth for both Latitude and Longitude	3

Table 2: Detail of data use according to each series of experiment.

SERIES NO.	QTY. OF SAMPLES	INTERPOLATED DATA	7FCV	NO. OF TESTS
1	66,483	Yes	No	4
2	66,483	Yes	No	6
3	66,483	Yes	No	100
4	66,483	Yes	Yes	336
5	3,319	No	Yes	672
6	3,319	No	Yes	225

Table 2 shows how the data has been used in the experiments series, indicating whether the 7FCV scheme was employed or not. In addition, the table indicates the number of tests of each series, the number of samples considered for training and if the data was interpolated or not. It's worth mentioning that the 7FCV scheme was not employed at the 3 first series because the main purpose of these series was to gather information about the general functionalities of the FCM+ANFIS methodology and its interactions with the flight log data.

5 Experiments and results

In this work, experiments were carried out with different architectures of the ANFIS Network to assess the effectiveness of the methods studied in this work. Five series were carried out, totaling more than 1,115 different tests. These experiments initially aimed at a better understanding of the data set and also the way how ANFIS work, to then evaluate the possible improvement in the accuracy of the positioning estimation by embedded sensors. In the experiments, tests were performed by varying the four dimensions mentioned before, to gather intuitions about the behavior of the ANFIS Network in each configuration.

A final series was held aiming at the testing of two other Data Fusion methodologies, namely Artificial Neural Networks (ANNs) and Regression Models (RM), totaling more than 255 tests. Different approaches for both methodologies were tested. For the ANNs, the chosen training approaches were Levenberg-Marquardt [81], Bayesian Regularization [80] and Scaled Conjugate Gradient [79]. As for the RMs, 15 training approaches of four different classes were tested [82]-[87]: for Linear Regressions class, Linear, Interactions Linear, Robust Linear, and Stepwise Linear; for the Trees class, Fine Tree, Medium Tree and Coarse Tree; for the Support Vector Machines class, Linear SVM, Quadratic SVM, Cubic SVM, Fine Gaussian SVM, Medium Gaussian SVM and Coarse Gaussian SVM; and finally, for the Ensembles class, Boosted Trees and Bagged Trees. The sixth series was performed for the sake of comparison with the results obtained by the FIS trained by the ANFIS algorithm.

For all experiments, different combinations of fused sensors were tested. The truth used for both training and evaluating results was the set of data obtained by the KF-GPS-RTK sensor, given its

notorious accuracy, either for Latitude or for Longitude. Therefore, in relation to KF-GPS-RTK, it was measured that embedded GPS presents an RMSE of approximately 21.10cm in Latitude and 30.81cm in Longitude.

Regarding the number of training epochs, some considerations have to be made. The algorithm used is instructed to generate two FIS: one at the end of the set number of epochs; and another which produced the least error among all epochs, or best fitness, if not equal to the first FIS. Although the number of epochs is fixed, the shown results correspond to the least error value found during the training process, that not necessarily corresponds to the last epoch of training. This fact is not a problem since the fuzzy inference system that yielded the best results of the entire training process will ever be available for the next experiments. The information about which epoch corresponds to the best result of a given test is not relevant to this work.

The experiments were performed on a notebook PC with a Core i7-6500U@2.50GHz processor, 16GB RAM, and a 64 bit Windows 10 Operating System.

5.1 First experiment series - the influence of the number of training epochs

A first approach was performed to preliminarily assess the impact of the number of training epochs on the performance of the technique. It is well known that an insufficient number of epochs may not promote the necessary convergence, whereas a large number of epochs may lead to overfitting [71].

Having fixed the quantities of 2 input membership functions and a training set of 80% of the data, the number of training epochs was varied for GPS-Lat and ACCEL sensors in 4 tests. The results are shown in Table 3.

Table 3: Results for the first series – 2 sensors.

Test	Epochs	Time (s)	RMSE (cm)
1	5	27.00	17.89
2	60	269.43	17.89
3	60	273.10	17.89
4	200	908.77	17.89

From the analysis of Table 3 it is seen that increasing the number of training epochs did not imply a significant improvement in the performance of the trained network. The computational time, in turn, was severely affected, indicating that increasing the number of training epochs is not always advantageous.

5.2 Second experiment series - the influence of training percentage

Given the preliminary results regarding the influence of the number of epochs on the achieved result, a series was performed to understand the influence of the percentage of the data used for the training. It is commonly seen in the literature the use of high percentages of the data for the training of Adaptive Networks.

Then, having fixed the quantities of 2 membership functions per input and 50 training epochs, the training percentage was varied in 6 tests, this time for three sensors: GPS-Lat, ACCEL, and GYRO. Table 4 shows the results.

Table 4: Results for the second series – 3 sensors.

Test	Training percentage (%)	RMSE (cm)
1	10	258.56
2	30	31.92
3	50	31.66
4	60	22.03
5	70	18.20
6	80	17.24

From these experiments, it was confirmed that the higher the percentage of data used for training, the better the overall performance of the trained network. The different tested data percentages were 10% , 30% , 50% , 60% , 70% and 80% . Furthermore, it was observed that using data from three sensors, GPS-Lat, ACCEL, and GYRO, showed better results than using only GPS-Lat and ACCEL, when comparing the last result of Table 4 with those of Table 3.

5.3 Third experiment series - the influence of the number of membership functions

Knowing minimally satisfactory parameters for conducting new tests, the third series was performed. The purpose of this study was to evaluate the impact of the number of membership functions per input to be optimized by the ANFIS algorithm in the performance of the proposed technique. For this purpose, the percentage of 70% of the data for the training was adopted, considering the good performance presented in the previous series.

In this series of 100 tests, the fused sensors were GPS-Lat and ACCEL, an association that presented relevant results. Both the number of MFs per input (2, 3, 4 e 5 MFs) and the number of epochs (5, 10, 50, 100 and 500 epochs) were combined and tested. For each of the 20 combinations of MFs per input and epochs, the generalization capability was evaluated by the average of 10 repetitions of the evaluation of 10,000 points randomly taken from the whole dataset.

Unlike the previous series, here the coordinates are not considered as inputs. That is, the input variables of the FIS refer only to the other sensors, which are mapped to the difference between the position estimated by the KF-GPS-RTK and that provided by the embedded GPS. The motivation arose from the perception that including the coordinates as inputs, besides increasing the processing time, demand that the direct employment (with no offset) of the FIS is restricted to the window of the coordinates used for training. This is due to the fact that the maximum and minimum values of each input variable are used to establish the FIS Universes of Discourse.

Therefore, the strategy of not treating coordinates as inputs — but as a difference — is interesting in the sense that performing non-linear mapping for a difference, rather than for an absolute value, makes its application feasible in regions of flight not used for training. It should be noted that, since the proposed approach concerns only the coordinate, its effectiveness is likely to be greater on flights with similar conditions, albeit in different regions. That is, issues such as trajectory, velocity, and similar climatic characteristics are factors that should bring positive contributions to the performance of the Data Fusion. Thus, $\vec{\Delta}$ corresponds to the horizontal inaccuracy of the UAV in degrees. A mathematical representation for determining the difference to be used for training can be found in the Equation 4.

$$\vec{\Delta} = p\vec{o}s_{out} - p\vec{o}s_{in} \tag{4}$$

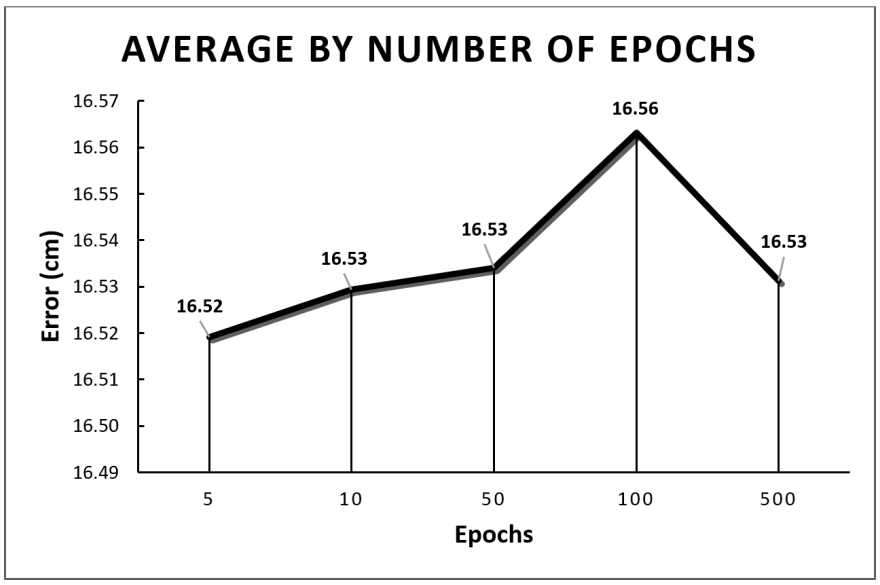
where $p\vec{o}s_{out}$ represents the vector containing the output coordinate (truth from RTK) and $p\vec{o}s_{in}$ represents the vector containing the input coordinate (measured from GPS). Thus, at a later stage, as for

example in a new flight, a position estimated by a GPS could be improved by a trained FIS with the calculation that appears in Equation 5:

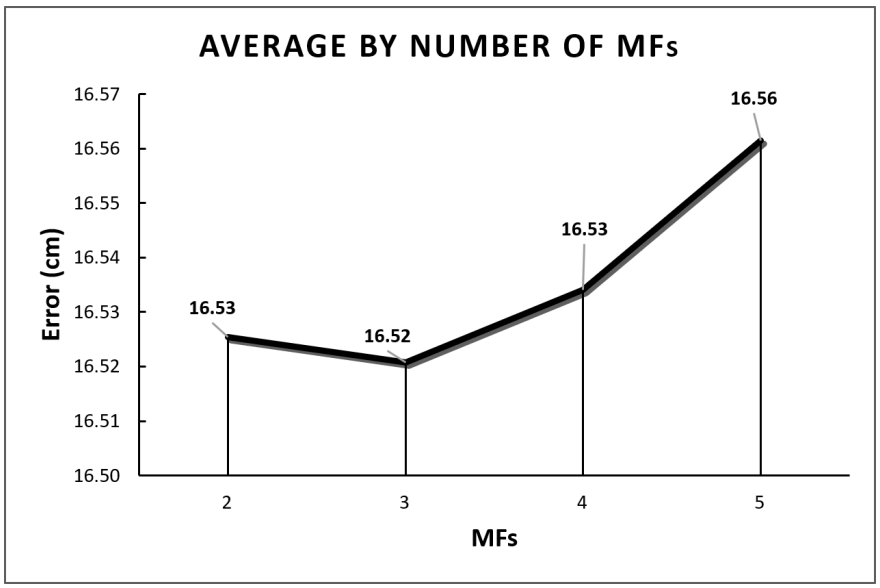
$$p\vec{o}s'_{out} = p\vec{o}s'_{in} + \vec{\Delta} + \vec{e}_{\Delta} \tag{5}$$

with the vector $p\vec{o}s'_{out}$ representing the improved position and $p\vec{o}s'_{in}$ the input coordinate estimated by the GPS, the $\vec{\Delta}$ is obtained by the Data Fusion promoted by the trained FIS and the vector $\vec{e}_{\Delta}$ represents the model error. Such an error is unknown a priori and composes the positioning estimation.

In short, the significant advantage of the strategy used is that both the location of the flight and the dimensions of the coordinate window used for training are no longer limiting factors for the employment of the methodology. However, the study of the impact of all other factors besides the coordinates, such as changes in speed, trajectory, and climatic characteristics, is suggested as future work. Thus, the results obtained are shown in Figure 9.



(a) Average by number of epochs



(b) Average by number of rules

Figure 9: Average error of 10 repetitions of the combinations of the Second series.

Confirming the hypothesis about the results obtained previously for this dataset, it was seen that increasing the number of training epochs in many cases proved to be ineffective in the generalization results, in relation to the RMSE mean. Also, increasing the number of MFs per input indistinctly has led to a worsening of the results, except for the leap from 2 to 3 MFs per input. Finally, the best results were achieved with a 5 epoch and 3 MFs per input setting, as seen in Figure 9. However, the use of 2 MFs per input enables a performance comparable to 3 MFs per input, but with shorter processing time. This fact prevailed in the choice of 2 MFs per input for the next series.

5.4 Fourth experiment series - the influence of the data organization

The fourth series was undertaken with the main purpose of evaluating the impact of data organization on the performance of the proposed methodology. The values of 5 epochs and 2^n rules were adopted, where 2 is the number of MFs per input and n is the number of input sensors. This variable amount of rules aims to provide more structural complexity to networks that will handle increased sets of data, from the addition of data inputs from other sensors. In other words, more sensors used imply more rules appearing in the structures of the ANFIS networks. Thus, 168 tests were performed for GPS-Lat associated with other sensors, and other 168 similar tests for GPS-Lon. Also, from this series onwards, the dataset employed was structured according to the 7FCV Cross-Validation technique.

In this series was observed that, indeed, the choice of CV groups combinations exerted great influence on the performance of ANFIS Networks. As an example, in the fusion of 3 sensors (GPS-Lon, ACCEL, and GYRO), the worst combination (nº 18) showed an error 73.40% greater than the best combination (nº 10). The RMSE comparison of the performance of each CV *training* groups in predicting the corresponding *validation* groups is showed in Figure 10. Additionally, it was seen that the average computational time went from 8.52s (2 rules) to 9.225s (8 rules), until remarkable 644.12s for 64 rules, with worse results, worth mentioning.

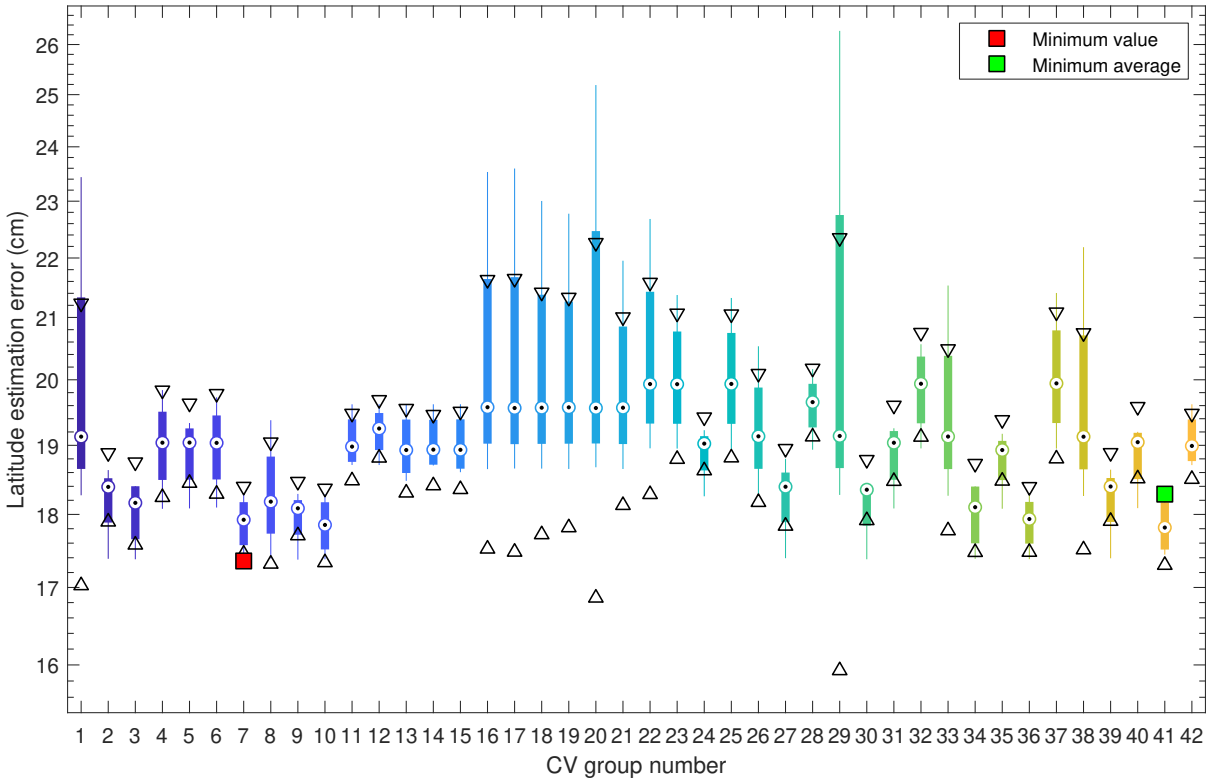
Concerning Figure 10, the bars indicate the concentration of the second quartile results for each CV group, with the upper and lower limits indicating, respectively, the delimitation of the third and first quartiles. The "whiskers" indicate the extremes that are not considered outliers. Red and green squares indicate respectively the best performance in terms of minimum and average of the whole set of tests, for a given coordinate. Finally, the "target" present in each bar indicates the median of the results for each CV group and the triangles down and up an indication about the significance of the differences between the medians.

Through an analysis of the tables of results and a visual inspection of these graphs, the choice of the training data has shown to have a great impact on the performance of the methodology. Furthermore, it was seen that there is a certain tendency that groups that perform well for GPS-Lat perform as good as for GPS-Lon. The best results were achieved with the CV combination nº 10, that yielded the following results: 17.38cm for GPS-Lat + ACCEL fusion and 24.86cm for GPS-Lon + ACCEL + GYRO fusion, both results better than the embedded GPS estimation alone. As for the worst results, they are 26.27cm for GPS-Lat + ACCEL + GYRO and 43.11cm for GPS-Lon + ACCEL + GYRO.

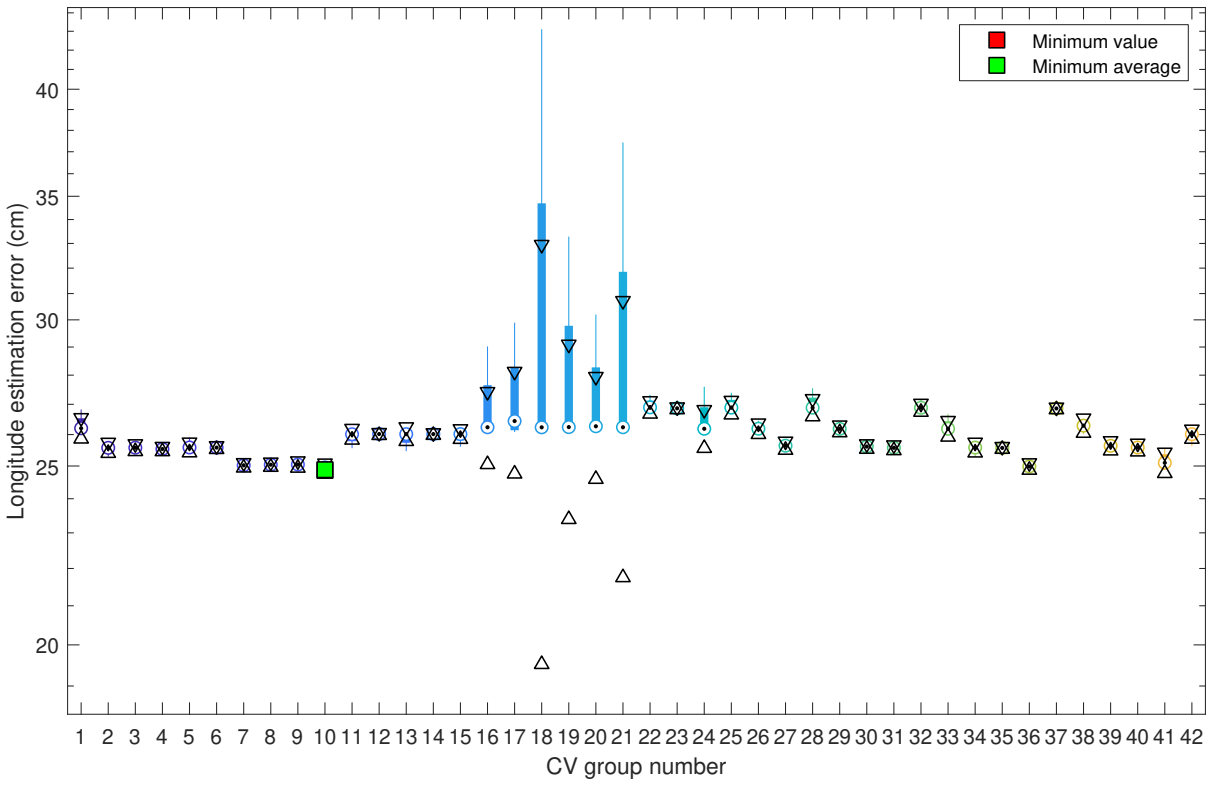
5.5 Fifth experiment series - the impact of using coordinates as inputs

In order to obtain better results and, primarily, to compare the performance of strategies that consider or not the coordinate as input, this fifth series was undertaken. Thanks to good results obtained previously, the values of 50 epochs and 16 rules were adopted for all the 672 tests.

Four main experiments were carried out in which the coordinate associated to the other sensors was varied, namely GPS-Lat; GPS-Lon; Δ-Lat; and Δ-Lon. In the "Δ-coordinate" cases, the FIS input variables concern only sensors other than GPS, which are mapped to the difference between KF-GPS-RTK and the embedded GPS data, that is, coordinates are not input. The comparison between the performance of the various sensors combinations of CV *training* groups in predicting – now the *generalization* group – is demonstrated in Figures 11 and 12.

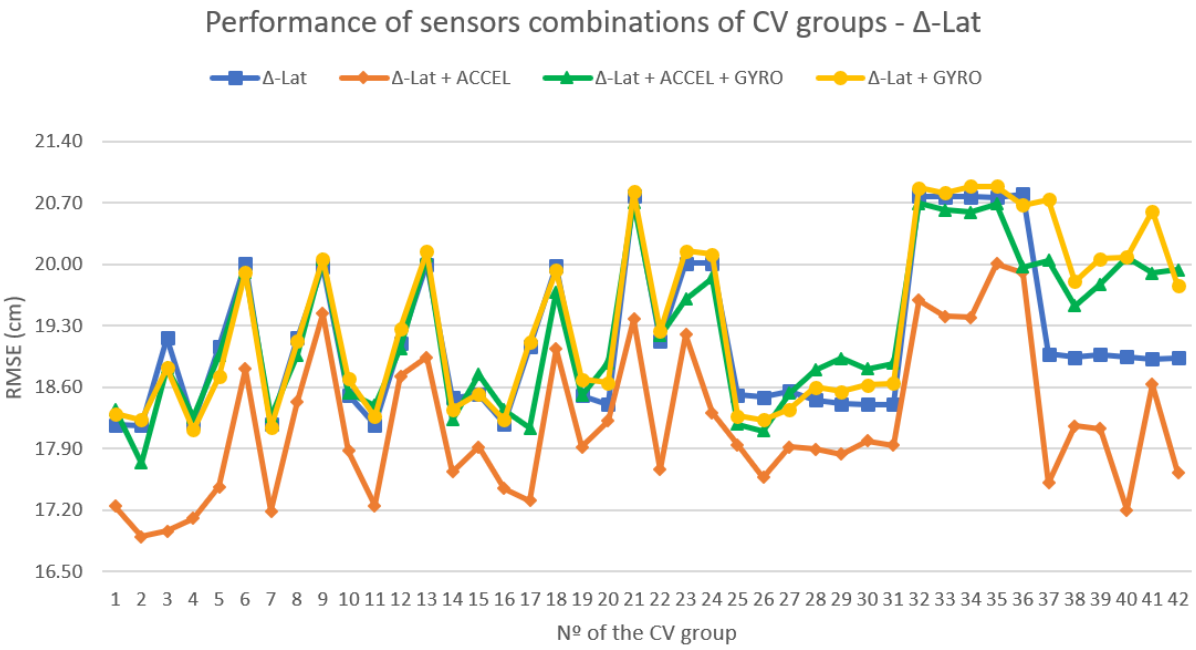


(a) Latitude

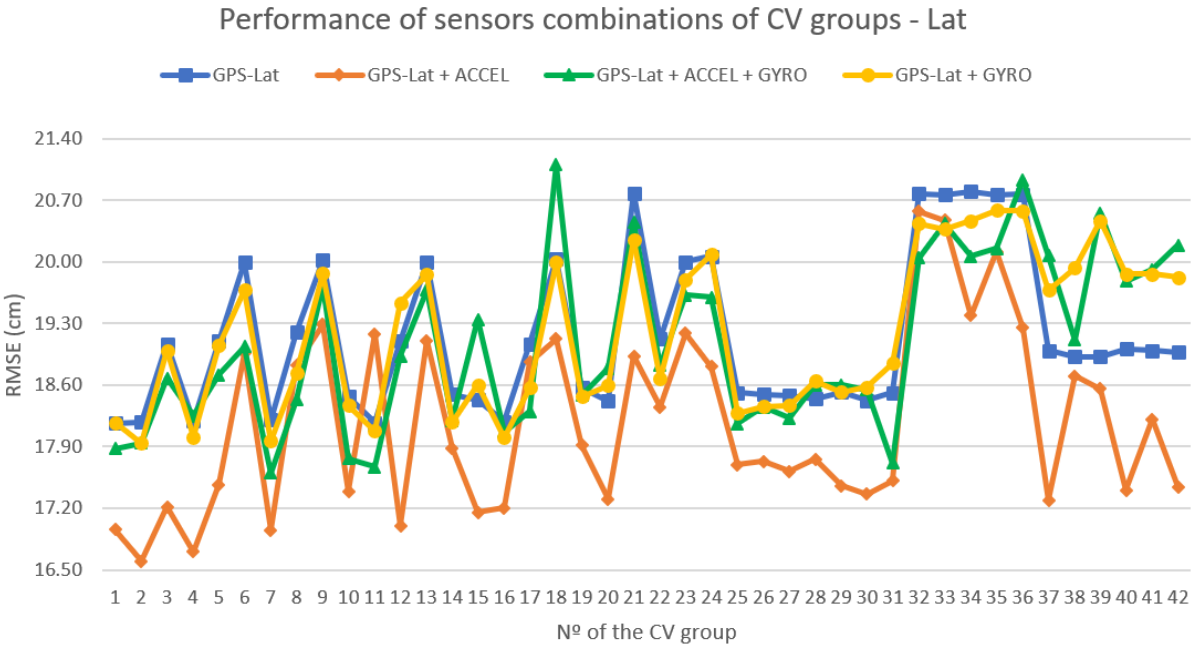


(b) Longitude

Figure 10: Average performance measured in terms of RMSE (cm) of each CV *training* group based on the prediction capability for the *validation* group - fourth series.

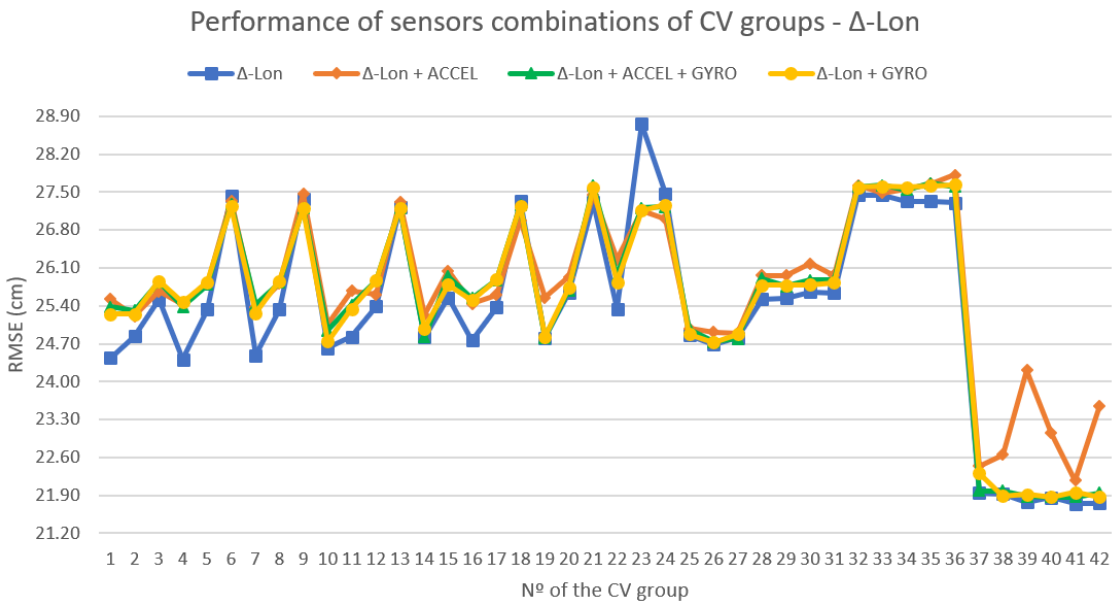


(a) Δ-Latitude

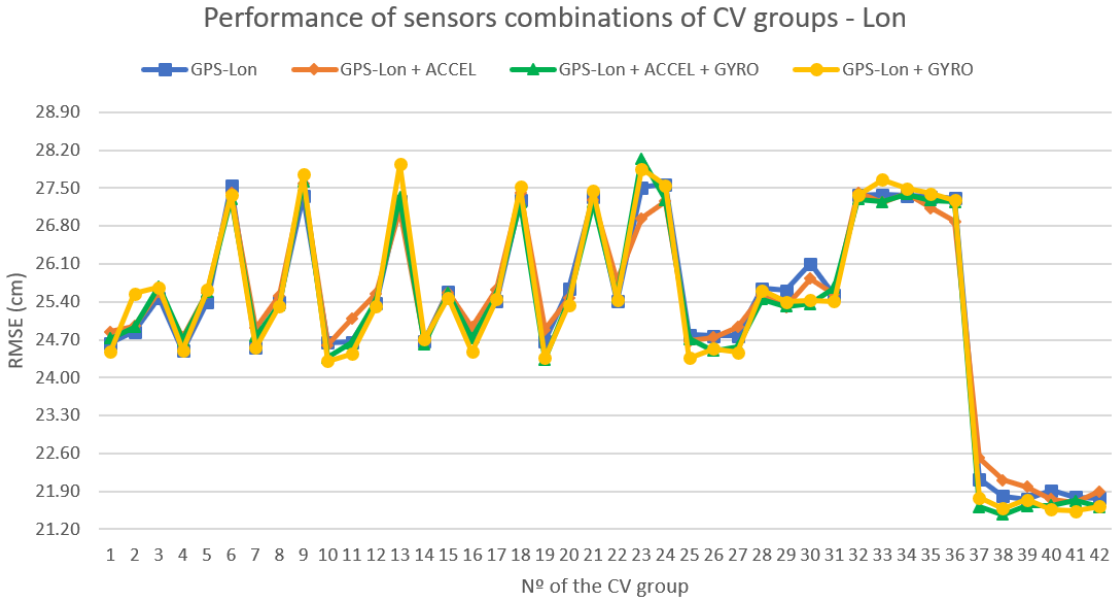


(b) Latitude

Figure 11: Performance measured in terms of RMSE (cm) of each CV *training* group based on the prediction capability for the *generalization* group, for each combination of sensors - Latitude and Δ-Latitude - fifth series.



(a) Δ-Longitude

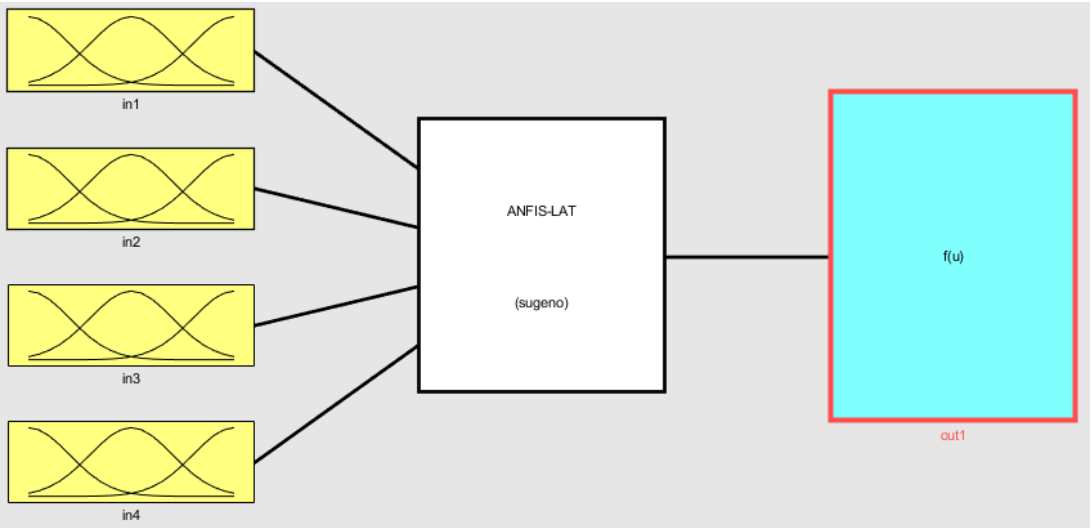


(b) Longitude

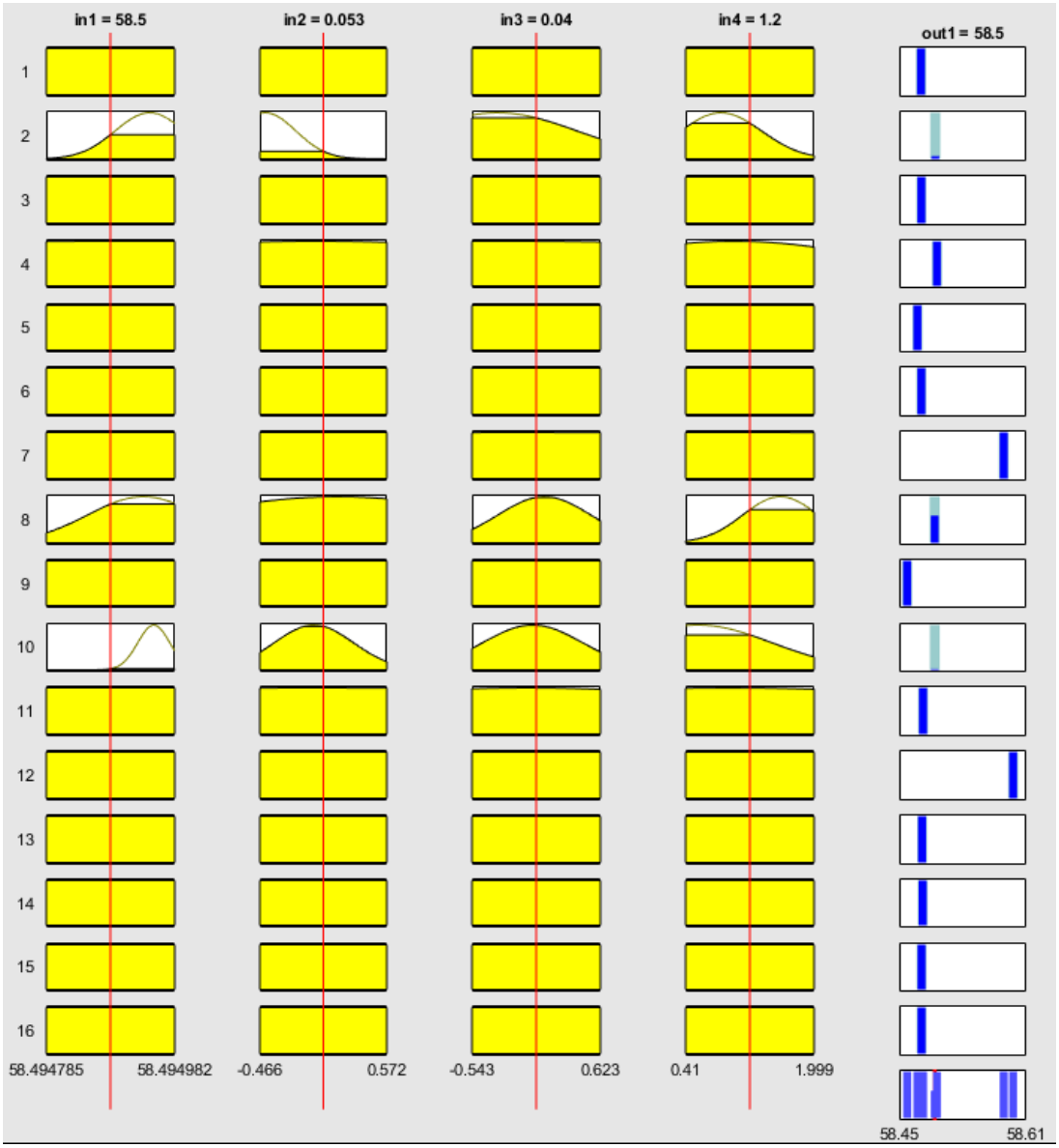
Figure 12: Performance measured in terms of RMSE (cm) of each CV *training* group based on the prediction capability for the *generalization* group, for each combination of sensors - Longitude and Δ-Longitude - fifth series.

The results show that the performance of the networks that consider the coordinates as inputs was slightly better than those considered "Δ-coordinate". However, the latter can be applied quickly on a broader range of scenarios because they do not use absolute values. Also, was seen that using data from all available sensors does not always bring improvements to the performance of networks, but results indicate that, for the given data set, fusing data from two or more sensors is advantageous. The summary of the best results obtained in the Fifth series is in Table 5.

The trained FIS that yielded the best results can be seen in Figures 13 and 14, as well as the ANFIS structure used for the training at Figure 15.

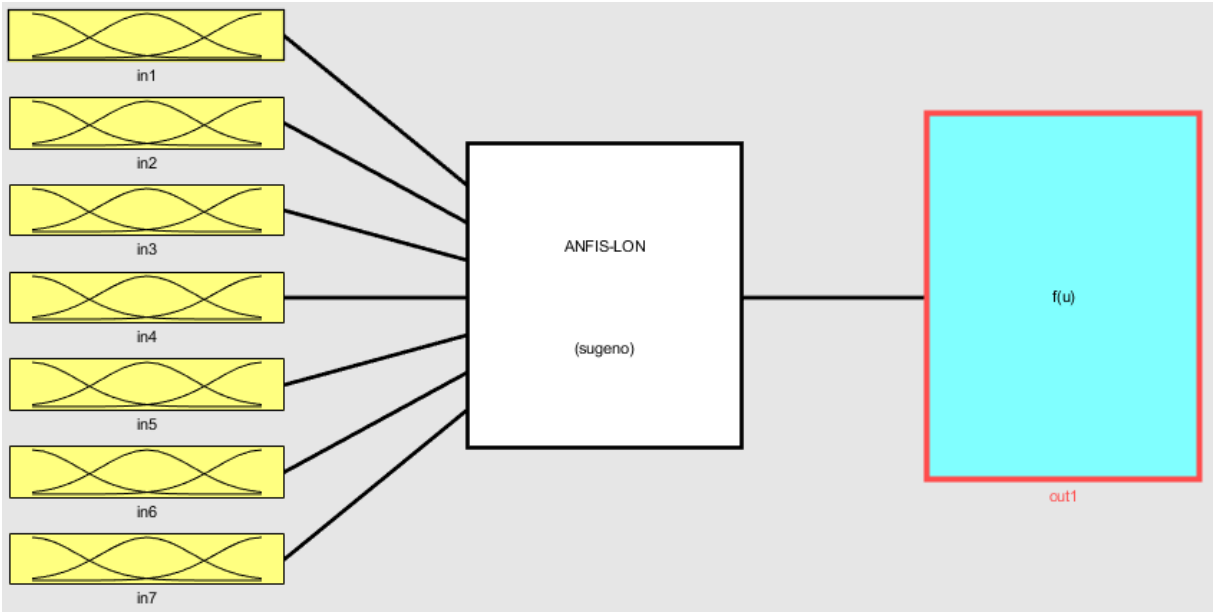


(a) FIS model

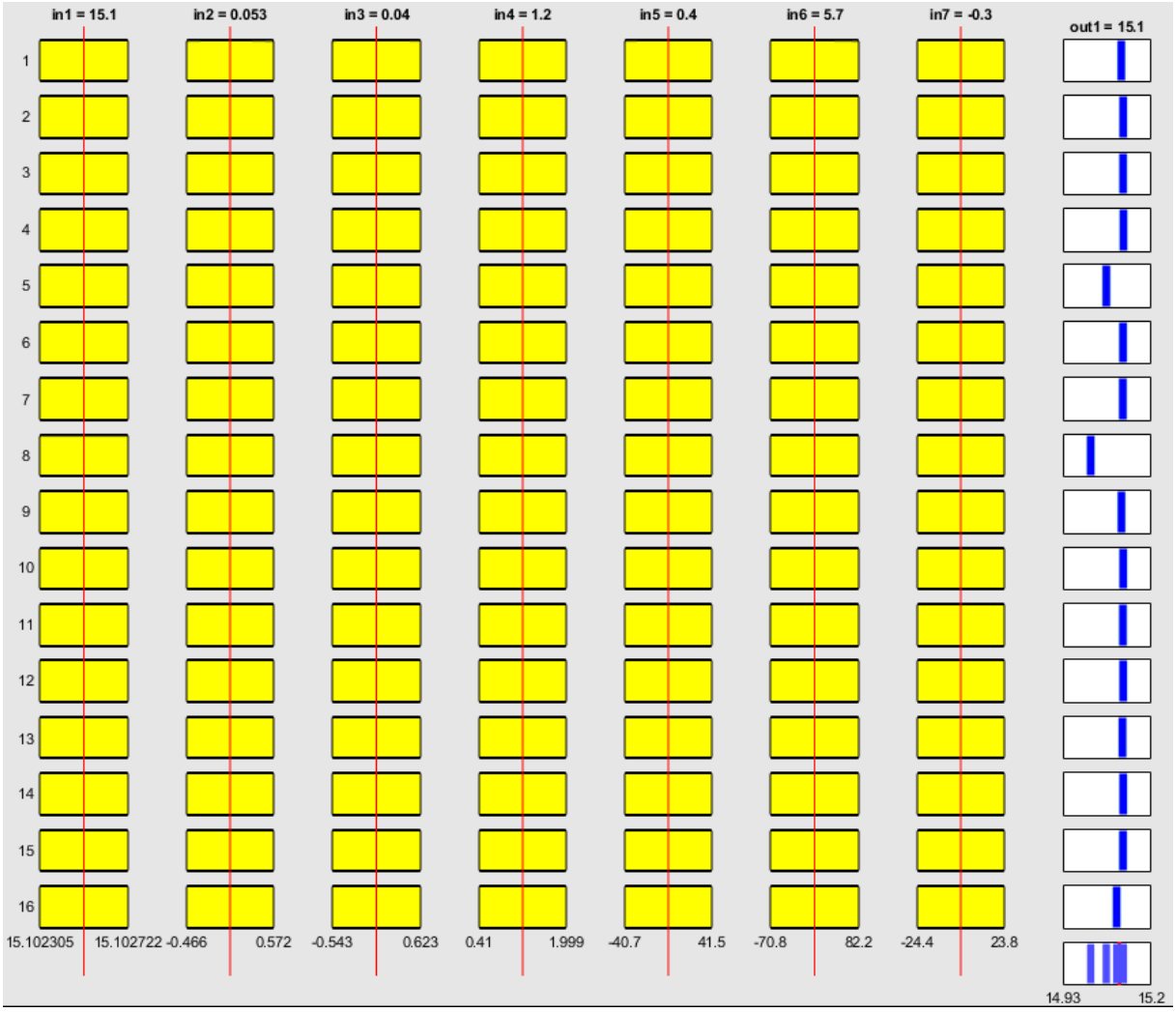


(b) FIS rules

Figure 13: The trained FIS model and FIS rules of the best result - Latitude.

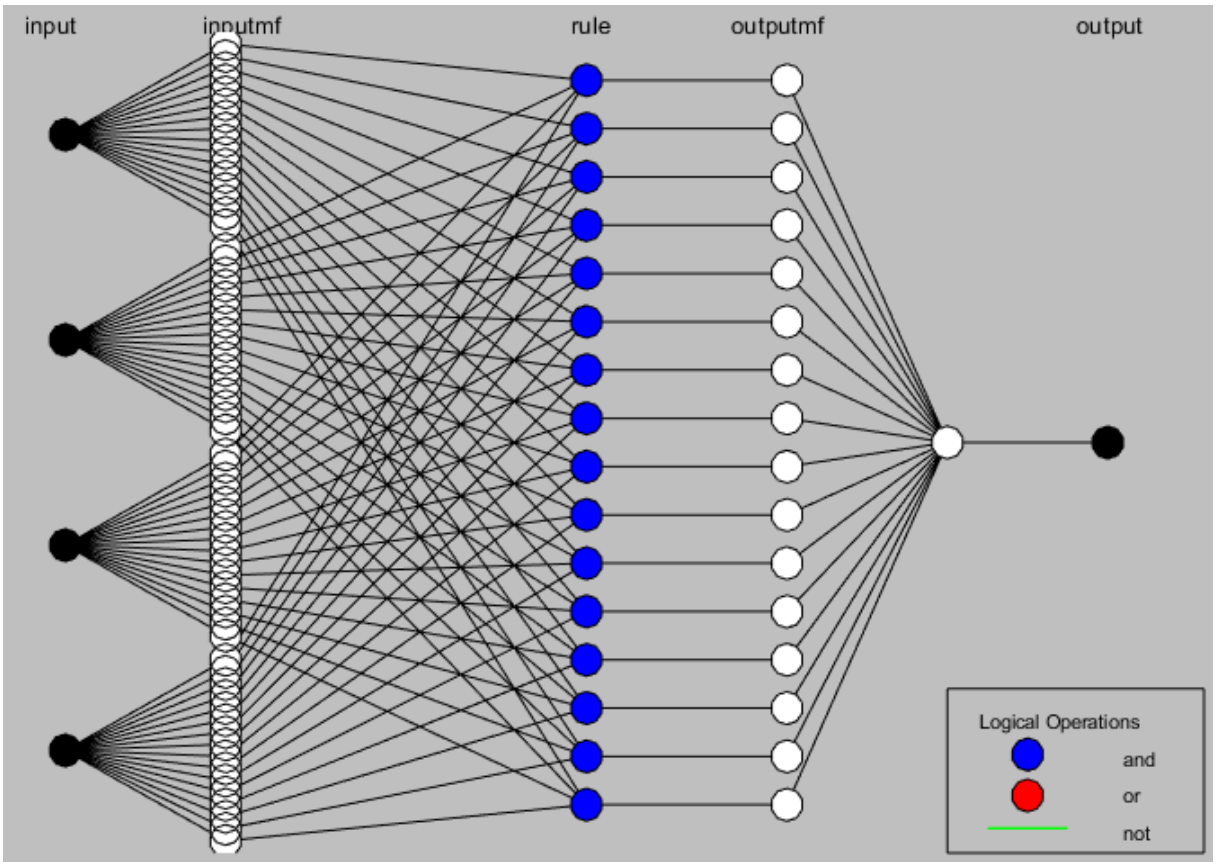


(a) FIS model

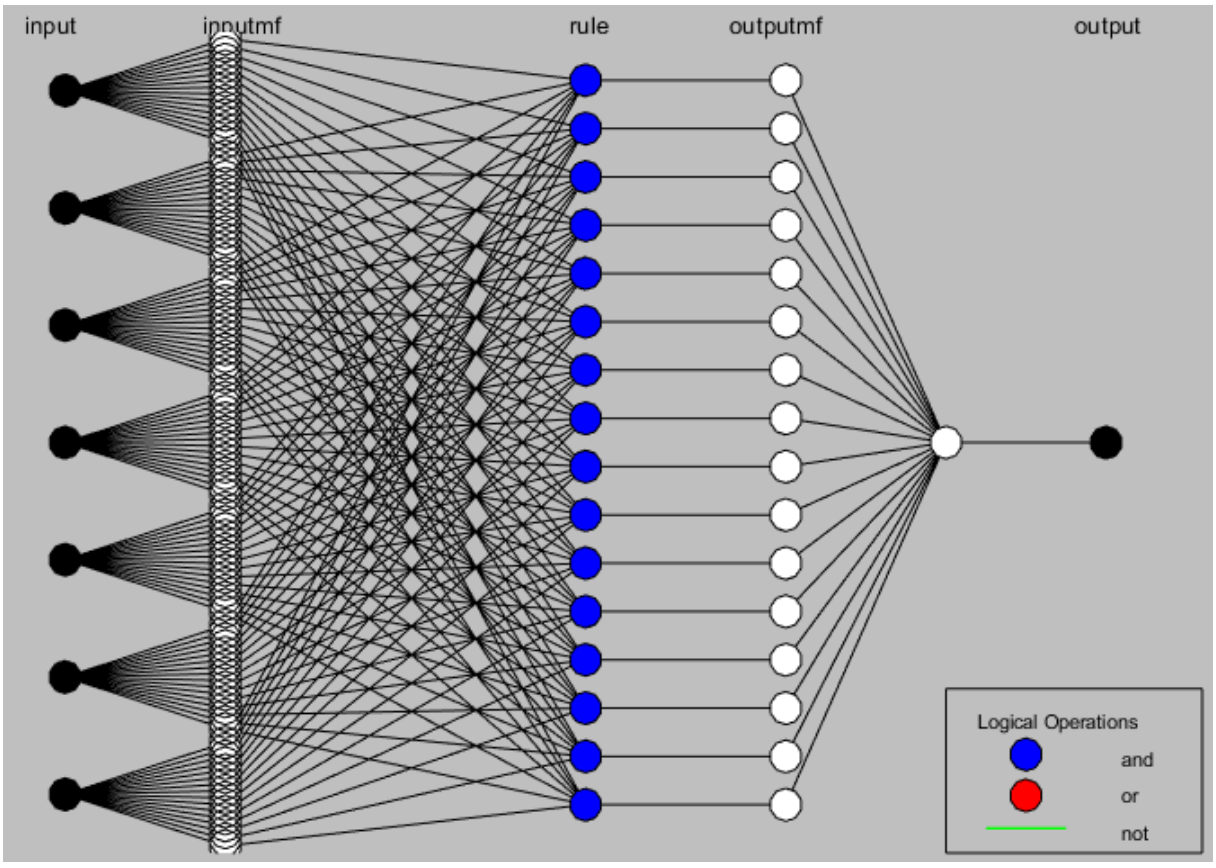


(b) FIS rules

Figure 14: The trained FIS model and FIS rules of the best result - Longitude.



(a) Latitude



(b) Longitude

Figure 15: The ANFIS Network of the best result.

Table 5: Comparison of best positioning estimation RMSE according to each combination of the different sensors, regarding each of the base coordinates.

Base coordinate	Sensors used at training process - RMSE in cm			
	Coordinate only	Coord. + ACCEL	Coord. + ACCEL + GYRO	Coord. + GYRO
GPS-Lat	18.17	**16.60**	17.60	17.94
Δ-Lat	-	16.90	17.74	18.11
GPS-Lon	21.76	21.69	**21.47**	21.54
Δ-Lon	-	22.17	21.87	21.87

5.6 Sixth experiment series - comparison with other methods and final results

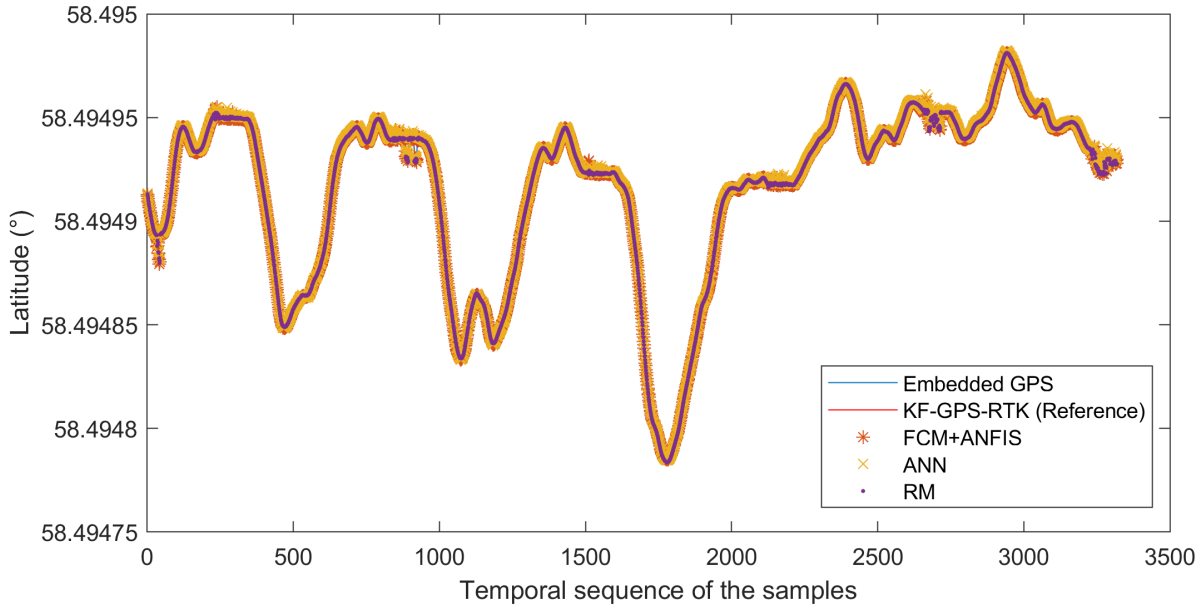
In this last series, the fusion was performed using ANNs and RMs, tuned with well-known parameters, in 225 tests, aiming at a comparison with the now fine-tuned FCM+ANFIS. The ANNs were trained with a two-layer feed-forward architecture with sigmoid hidden neurons and linear output neurons. The input layer is composed of 4 neurons for both Latitude and Longitude since the best results were obtained with 2 sensors (GPS-Lat/Lon + ACCEL, with ACCEL having 3 variables). The hidden layer has 2,000 neurons and the output just one, the improved coordinate estimate. A data division was performed, where 70% of the samples were used for training, 15% for validation and 15% for testing, with samples taken randomly. The training process had no fixed epoch number, being halted when the generalization capability stopped improving. Finally, the best results were 17.02cm for Latitude and 24.70cm for Longitude. For the ANNs, the training algorithm that yielded the best results was the Bayesian Regularization. Table 7 at the appendix lists the best results for each training method.

Regarding the RMs, the best results were obtained with 3 sensors for Latitude (GPS-Lat + ACCEL + GYRO) and 2 for Longitude (GPS-Lon + ACCEL), trained with Interactions Linear and Fine Tree techniques respectively. All tests were performed with a 7-fold Cross-Validation scheme over the sample data. It is worth mentioning that the Fine Tree was employed with a minimum leaf size of 4 and maximum surrogates per node of 10. The best obtained results were 19.15cm for *Lat* and 24.34cm for *Lon*. Table 8 at the appendix presents the best results for each approach.

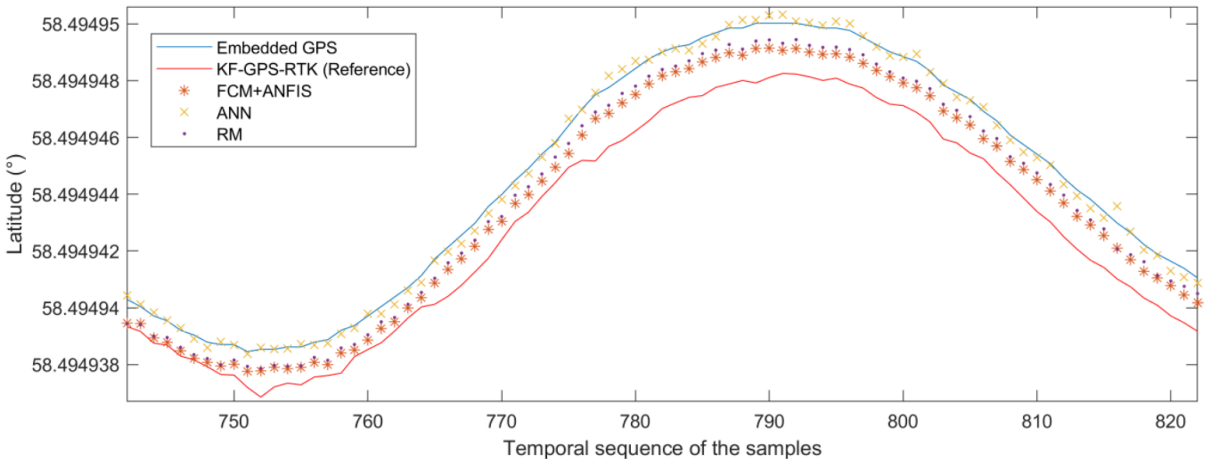
Based on the estimates of Latitude and Longitude performed by the FIS trained by ANFIS Networks, a comparison can be made with the estimates made by the GPS, the ANNs and the RMs through the evaluation of their respective RMSE, which is shown in Table 6. An excerpt of the trajectories estimated by each Fusion method, the real trajectory provided by the KF-GPS-RTK and the trajectory estimated by the embedded GPS is shown in Figures 16 and 17, as well as the comparison of error between the methods' estimates and the truth trajectory for the given coordinate. It is appropriate to mention that the figures include only short excerpts of such trajectories and that at other regions of the complete plot the other methods were eventually more precise than the trained ANFIS algorithm.

Table 6: Comparison of positioning estimation systems RMSE and accuracy.

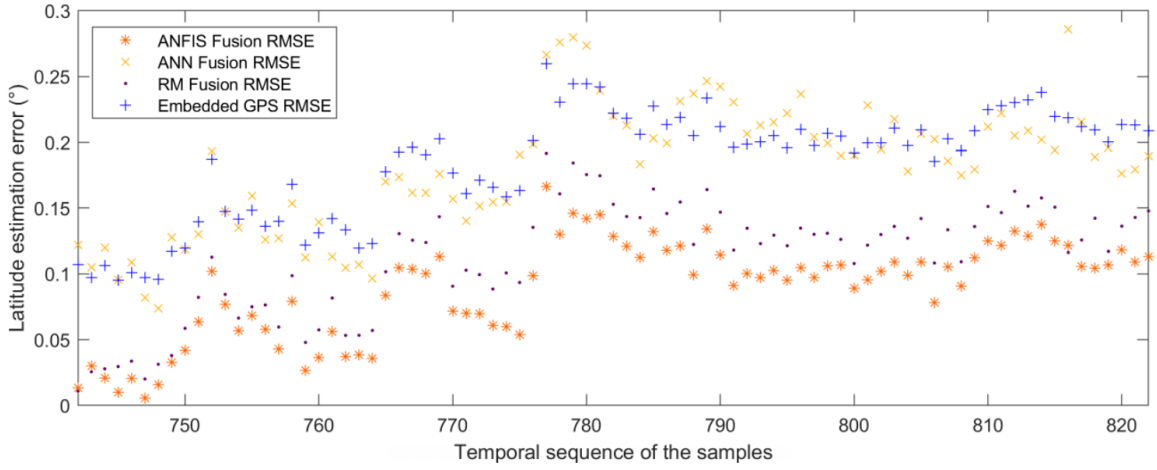
Estimation System	Coordinate	Fusioned sensors	Training method	Lat error (cm)	Lon error (cm)	Imprecision (area in cm²)	Imprecision reduction
GPS	**LAT, LON**	-	-	21,10	30,81	650,10	-
Regression Models	**LAT**	GPS-Lat + ACCEL + GYRO	Interactions Linear	19,15	24,34	465,92	**28,33%**
	LON	GPS-Lon+ACCEL	Fine Tree				
Neural Networks	**LAT**	GPS-Lat+ACCEL	Bayesian Regularization	17,02	24,70	420,35	**35,34%**
	LON	GPS-Lon+ACCEL					
FCM + ANFIS	**LAT**	GPS-Lat+ACCEL	Hybrid	16,60	21,47	356,31	**45,19%**
	LON	GPS-Lon + ACCEL + GYRO					



(a) Complete trajectory - Latitude

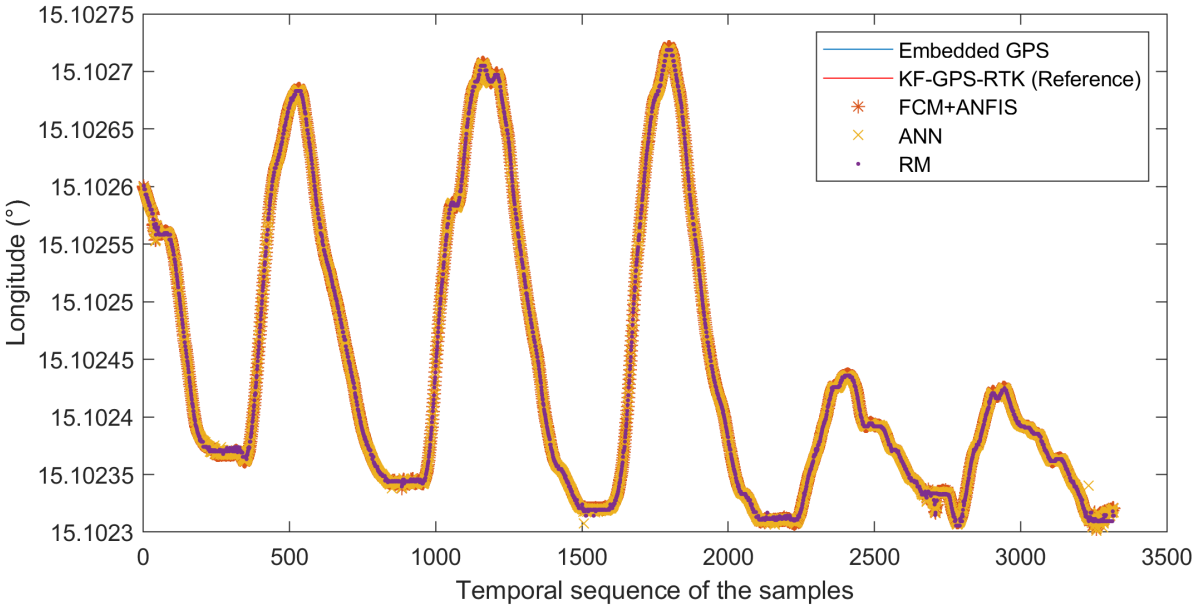


(b) Detailed trajectory - instants 742 to 805

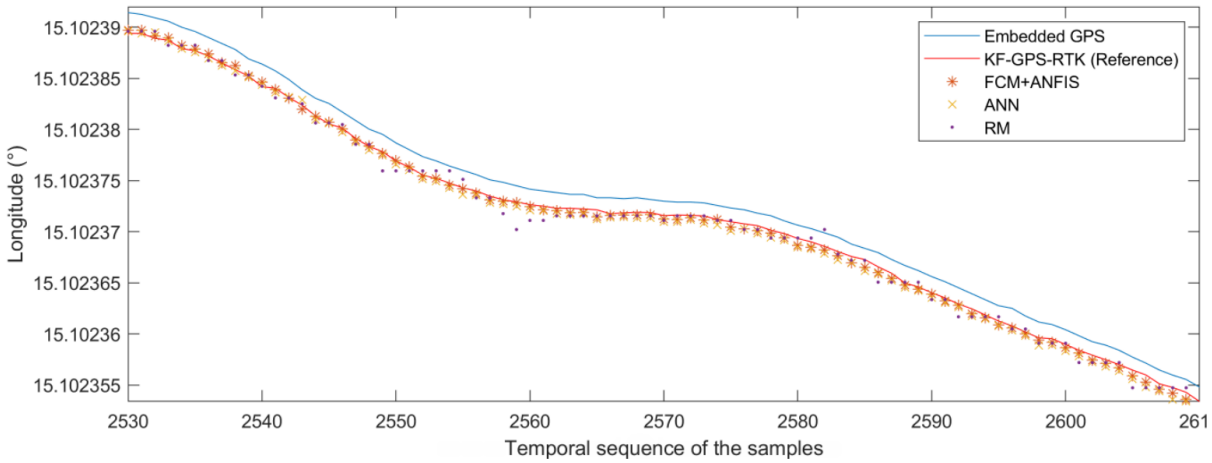


(c) Detailed trajectory error - instants 742 to 805

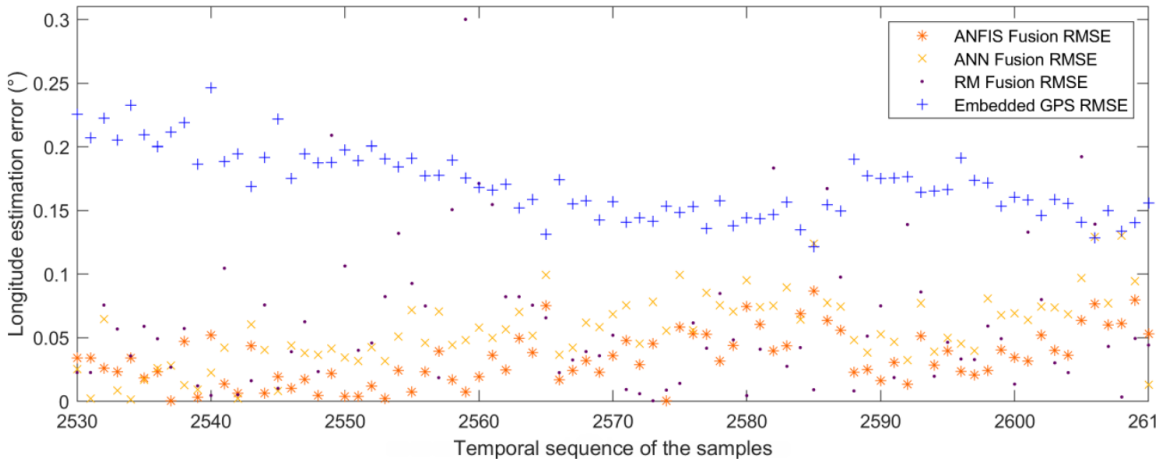
Figure 16: Comparison of a excerpt of the estimated trajectories and respective errors yielded by the best results of each Fusion method - Latitude.



(a) Complete trajectory - Longitude



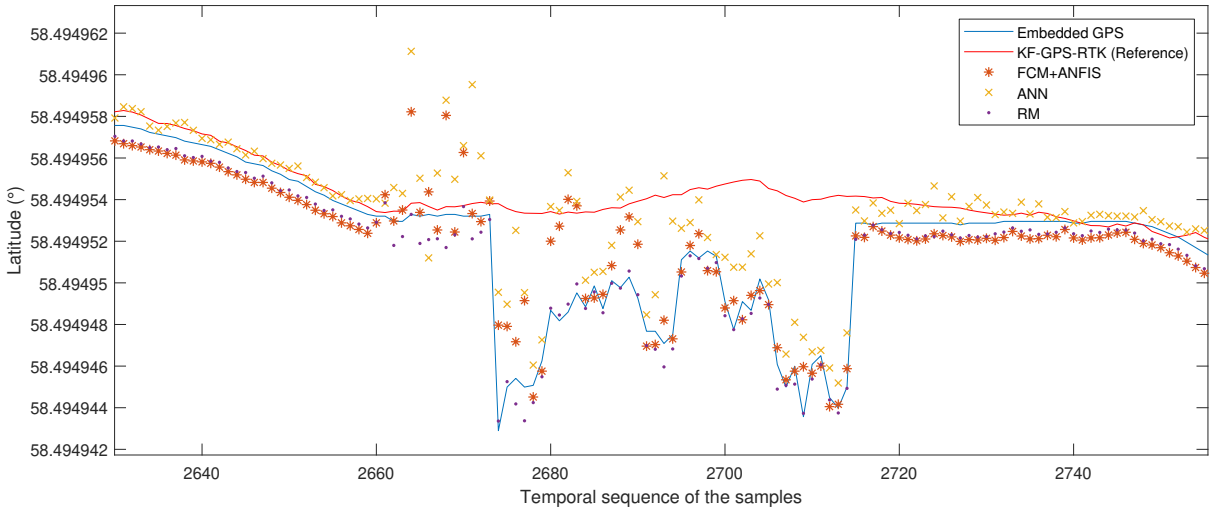
(b) Detailed trajectory - instants 2530 to 2610



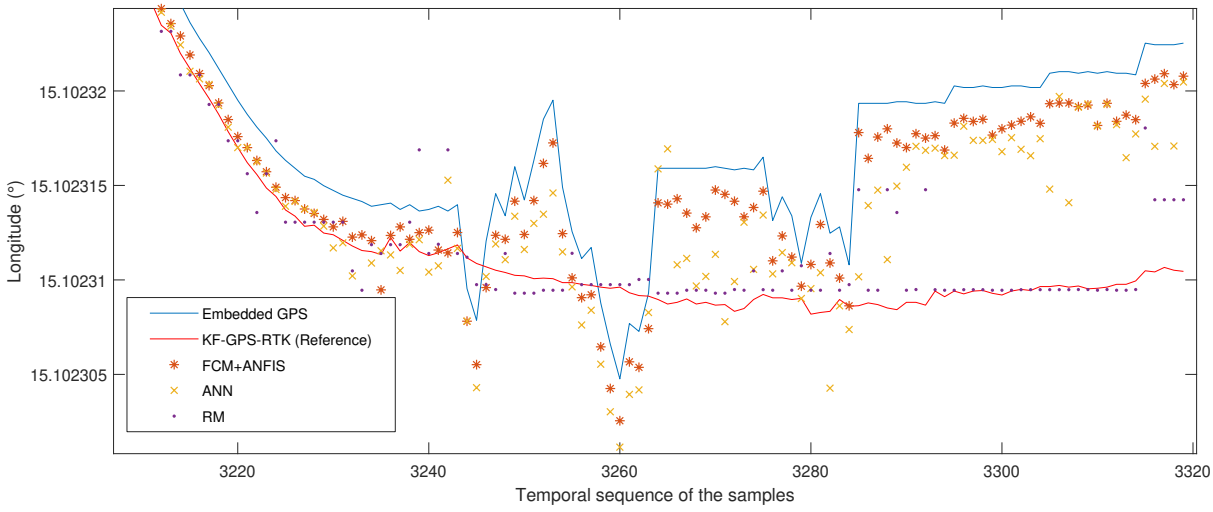
(c) Detailed trajectory error - instants 2530 to 2610

Figure 17: Comparison of an excerpt of the estimated trajectories and respective errors yielded by the best results of each Fusion method - Longitude.

In general, all approaches prevailed in an imprecision reduction of the positioning estimation. After a thorough visual inspection of the trajectories (Figures 16 and 17) estimated by the algorithms, it was perceived that the estimates made by all the fusion methods have some sensitivity to input signal noise. Such sensitivity can be easily seen in Figure 18, which allows immediate identification of the tendency of the points estimated by the various methods to follow the original noisy signal (blue line). Due to the complexity of the data of a real flight log, in the excerpts where noises are present, the estimates diverge significantly from the actual trajectory. In other words, the proposed methodology showed some sensitivity to noises tending to follow the GPS's quick variation noises, indicating that prior treatment of GPS deviations would bring improvements to the methodology employment. In addition, noises regarding other sensors might influence the outliers present in excerpts where the blue line shows low variation. The analysis of all these facts is suggested as future work. Finally, it was observed that the overall performance of the ANFIS Systems exceeded by 45.19% the accuracy of the estimation performed by the GPS.



(a) Latitude



(b) Longitude

Figure 18: Excerpt of the trajectories estimated by each method evidencing the influence of noise on the quality of the estimation.

5.7 Final remarks

Although no tests of the FIS fusion were conducted on actual UAVs – but on a notebook PC – it was found that, for a total of 3,319 estimates by fusing the sensors corresponding to the best trained FIS for each coordinate, the average time of 100 repetitions of the 3,319 estimates was 2.0317s for *Lat* and 2.7081s for *Lon*, or about 6.12 x 10^{-6} and 8.16 x 10^{6} seconds per estimation, respectively. This fact reveals that, indeed, a FIS is of high-speed processing, considering the computational power of the used notebook PC, with updates occurring at more than 120,000 estimates per second ($\cong$ 163kHz - *Lat* e $\cong$ 122kHz - *Lon*). A feature like this is of great interest in the application of the developed methodology to real-time navigation. Tests on real-time flights with the FIS fusion being performed by the embedded avionics of an UAV are suggested as future work.

6 Conclusions

As demonstrated in the results of the carried-out experiments, the methodology employed in this Multisensor Data Fusion application, using a HACI methodology, has shown to be effective in increasing the positioning estimation precision. In the tests, the technique presented a reduction of approximately $300cm^2$ in the area error, compared to the estimation of the position provided by the embedded GPS. How useful this reduction is, in practice, depends on the selected application. Applications that need high precision could be undoubtedly benefited. In addition, the proposed methodology outperformed two other Computational Intelligence methodologies, with 18 different approaches. The obtained results show that the use of multiple data sources has brought benefits to the solution of the navigation problem, considering the safety in the operation of the UAV, by offering less uncertainty to the Decision-Making System.

The main contribution of this work is the development of a Data Fusion application which presents a desirable level of accuracy and a low computational cost. This can make feasible its use in real-time autonomous navigation by low-performance UAVs, considering they usually lack computational power and payload. Once trained, the T-S rules-based FIS system has shown to be indeed agile, considering the computer used in the tests, to perform positioning estimation based on multidimensional data, about 600 times faster than the fastest sensor refresh rate. However, given the complexity of a flight log, further analysis of the trained fuzzy rules didn't bring any new useful knowledge, as expected.

Although the good performance, the proposed methodology still shows some sensitivity to noise in some parts of the trajectory. An assessment of the impact of noise treatment or reduction is suggested as future work since it could, theoretically, improve the performance of the employed techniques. In addition, at this point of the research, there is no guarantee that the proposed methodology will present comparable performance in flights with completely different trajectories or weather conditions. Studies in this sense are proposed as future work as well.

Other suggestions for future work include: perform the fusion with sensors other than GPS and INS; a study of why the other methods were more precise than ANFIS in some particular excerpts of the trajectory; perform the fusion using other Computational Intelligence Data Fusion methodology, and the conduction of tests on new flights data.

References

[1] BRASIL. Ministério da Defesa. Comando da Aeronáutica. Planejamento., DCA 11-45 - Concepção Estratégica Força Aérea 100. 2014.

[2] M. G. Lacerda, Â. C. Paulino, Á. J. Damião, E. H. Shiguemori, and Lamartine N. F. Guimarães, "O Emprego de Árvore de Decisão para a Identificação e Classificação de ZLs e ZPHs em Imagens Obtidas por ARPs de Pequeno Porte", in XIX Simpósio de Aplicações Operacionais em Áreas de Defesa, 2017.

[3] M. Lacerda, Â. C. Paulino, E. Shiguemori, A. Damiao, L. Guimaraes, and C. Anjos, "The Employment of Unmanned Aircraft Systems for Identification and Classification of Helicopter Landing

Zones and Airdrop Zones in Calamity Situations", in ICUAS 2018: International Conference on Unmanned Aircraft Systems, 2018, vol. 12, no. 5, p. 2190.

[4] Agência Força Aérea, "Veja o trabalho dos esquadrões especializados em reconhecimento aéreo", 2017. [Online]. Available: www.fab.mil.br/noticias/mostra/30328. [Accessed: 21-Feb-2018].

[5] Agência Força Aérea, "Hermes 900 participa de treinamento em Campo Grande", 2014. [Online]. Available: http://fab.mil.br/noticias/mostra/19863. [Accessed: 17-Feb-2018].

[6] Agência Força Aérea, "Esquadrão Hórus participa da vigilância aérea nos Jogos Olímpicos", 2016. [Online]. Available: http://fab.mil.br/noticias/mostra/26951. [Accessed: 17-Feb-2018].

[7] Agência Força Aérea, "FAB intercepta aeronave que transportava 500 kg de droga", 2017. [Online]. Available: www.fab.mil.br/noticias/mostra/30439. [Accessed: 21-Feb-2018].

[8] A. Al-Kaff, D. Martín, F. García, A. de la Escalera, and J. María Armingol, "Survey of computer vision algorithms and applications for unmanned aerial vehicles", Expert Syst. Appl., vol. 92, pp. 447–463, Feb. 2018. doi: 10.1016/j.eswa.2017.09.033.

[9] K. L. B. Cook, "The Silent Force Multiplier: The History and Role of UAVs in Warfare", in 2007 IEEE Aerospace Conference, 2007, pp. 1–7. doi: 10.1109/AERO.2007.352737.

[10] L. Davis, M. J. McNerney, J. Chow, T. Hamilton, S. Harting, and D. Byman, "Armed and Dangerous? UAVs and U.S. Security", Santa Monica, CA, 2014.

[11] D. Glade, "Unmanned Aerial Vehicles: Implications for Military Operations", Occas. Pap. Cent. Strateg. Technol. Air War Coll., no. 16, p. 38, 2000.

[12] M. E. Liggins, D. L. Hall, and J. Llinas, Handbook of multisensor data fusion: theory and practice, 2nd ed. Boca Raton, FL: CRC Press, 2009.

[13] U. K. Verfuss et al., "A review of unmanned vehicles for the detection and monitoring of marine fauna," Mar. Pollut. Bull., vol. 140, pp. 17–29, Mar. 2019. doi: 10.1016/J.MARPOLBUL.2019.01.009.

[14] C. M. Chen, L. E. Sinclair, R. Fortin, M. Coyle, and C. Samson, "In-flight performance of the Advanced Radiation Detector for UAV Operations (ARDUO)," Nucl. Instruments Methods Phys. Res. Sect. A Accel. Spectrometers, Detect. Assoc. Equip., Nov. 2018. doi: 10.1016/J.NIMA.2018.11.068.

[15] S. Gallardo-Saavedra, L. Hernández-Callejo, and O. Duque-Perez, "Technological review of the instrumentation used in aerial thermographic inspection of photovoltaic plants," Renew. Sustain. Energy Rev., vol. 93, pp. 566–579, Oct. 2018. doi: 10.1016/J.RSER.2018.05.027.

[16] M. G. Lacerda, A. De Carvalho Paulino, E. H. Shiguemori, A. J. Damiao, L. N. F. Guimaraes, and C. S. dos Anjos, "Identification and Classification of Drop Zones and Helicopter Landing Zones in Images Obtained by Small Size Remotely Piloted Aircraft Systems," in IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium, 2018, pp. 7906–7909. doi: 10.1109/IGARSS.2018.8517753.

[17] W. da Silva, "Navegação Autônoma de VANT em período noturno com imagens Infravermelho Termal", INPE, 2016.

[18] R. Cardoso, "Integração de Sensores Via Filtro de Kalman", ITA, 2003.

[19] W. da Silva, E. H. Shiguemori, N. L. Vijaykumar, and H. F. de C. Velho, "Estimation of UAV position with use of thermal infrared images", in 2015 9th International Conference on Sensing Technology (ICST), 2015, pp. 828–833. doi: 10.1109/ICSensT.2015.7438511.

[20] J. R. G. Braga, H. F. d. C. Velho, and E. H. Shiguemori, "Estimation of UAV position using LiDAR images for autonomous navigation over the ocean", in 2015 9th International Conference on Sensing Technology (ICST), 2015, pp. 811–816. doi: 10.1109/ICSensT.2015.7438508.

[21] G. Conte and P. Doherty, "An Integrated UAV Navigation System Based on Aerial Image Matching", in 2008 IEEE Aerospace Conference, 2008, pp. 1–10. doi: 10.1109/AERO.2008.4526556.

[22] L. Faria, C. M. Silvestre, and M. A. F. Correia, "GPS-Dependent Systems: Vulnerabilities to Electromagnetic Attacks", J. Aerosp. Technol. Manag., vol. 8, pp. 423–430, 2016.

[23] J. R. G. Braga, "Navegação autônoma de VANT por imagens LiDAR," Instituto Nacional de Pesquisas Espaciais, 2018.

[24] P. F. F. Silva Filho, "Automatic Landmark Recognition in aerial images for the autonomous navigation system of Unmanned Aerial Vehicles", ITA, 2016.

[25] L. D. A. Faria, C. A. de M. Silvestre, M. A. F. Correia, and N. A. Roso, "GPS Jamming Signals Propagation in Free-Space, Urban and Suburban Environments", J. Aerosp. Technol. Manag., vol. 10, pp. 1–8, Feb. 2018. doi: 10.5028/jatm.v10.870.

[26] L. D. A. Faria, C. A. de M. Silvestre, M. A. F. Correia, and N. A. Roso, "Susceptibility of GPS-Dependent Complex Systems to Spoofing", J. Aerosp. Technol. Manag., vol. 10, pp. 1–11, Jan. 2018. doi: 10.5028/jatm.v10.839.

[27] G. Stamatescu, I. Stamatescu, D. Popescu, and C. Mateescu, "Sensor fusion method for altitude estimation in mini-UAV applications", in 2015 7th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), 2015, p. SSS-39-SSS-42. doi: 10.1109/ECAI.2015.7301203.

[28] S.-M. Oh, "Multisensor fusion for autonomous UAV navigation based on the Unscented Kalman Filter with Sequential Measurement Updates", in 2010 IEEE Conference on Multisensor Fusion and Integration, 2010, pp. 217–222. doi: 10.1109/MFI.2010.5604461.

[29] C. Luo, S. I. McClean, G. Parr, L. Teacy, and R. De Nardi, "UAV Position Estimation and Collision Avoidance Using the Extended Kalman Filter", IEEE Trans. Veh. Technol., vol. 62, no. 6, pp. 2749–2762, Jul. 2013. doi: 10.1109/TVT.2013.2243480.

[30] J. Li, N. Song, G. Yang, M. Li, and Q. Cai, "Improving positioning accuracy of vehicular navigation system during GPS outages utilizing ensemble learning algorithm", Inf. Fusion, vol. 35, pp. 1–10, 2017. doi: 10.1016/j.inffus.2016.08.001.

[31] S. Adusumilli, D. Bhatt, H. Wang, V. Devabhaktuni, and P. Bhattacharya, "A novel hybrid approach utilizing principal component regression and random forest regression to bridge the period of GPS outages", Neurocomputing, vol. 166, pp. 185–192, Oct. 2015. doi: 10.1016/J.NEUCOM.2015.03.080.

[32] A. Noureldin, A. El-Shafie, and M. Reda Taha, "Optimizing neuro-fuzzy modules for data fusion of vehicular navigation systems using temporal cross-validation", Eng. Appl. Artif. Intell., vol. 20, no. 1, pp. 49–61, Feb. 2007. doi: 10.1016/J.ENGAPPAI.2006.03.002.

[33] D. Bhatt, P. Aggarwal, V. Devabhaktuni, and P. Bhattacharya, "A novel hybrid fusion algorithm to bridge the period of GPS outages using low-cost INS", Expert Syst. Appl., vol. 41, no. 5, pp. 2166–2173, 2014. doi: 10.1016/j.eswa.2013.09.015.

[34] X. Li, W. Chen, C. Chan, B. Li, and X. Song, "Multi-sensor fusion methodology for enhanced land vehicle positioning", Inf. Fusion, vol. 46, pp. 51–62, Mar. 2019. doi: 10.1016/J.INFFUS.2018.04.006.

[35] S. Adusumilli, D. Bhatt, H. Wang, P. Bhattacharya, and V. Devabhaktuni, "A low-cost INS/GPS integration methodology based on random forest regression", Expert Syst. Appl., vol. 40, no. 11, pp. 4653–4659, 2013. doi: 10.1016/j.eswa.2013.02.002.

[36] A. Noureldin, A. El-Shafie, and M. Bayoumi, "GPS/INS integration utilizing dynamic neural networks for vehicular navigation", Inf. Fusion, vol. 12, no. 1, pp. 48–57, 2011. doi: 10.1016/j.inffus.2010.01.003.

[37] N. Musavi and J. Keighobadi, "Adaptive fuzzy neuro-observer applied to low cost INS/GPS", Appl. Soft Comput., vol. 29, pp. 82–94, Apr. 2015. doi: 10.1016/J.ASOC.2014.12.024.

[38] M. K. Al-Sharman, B. J. Emran, M. A. Jaradat, H. Najjaran, R. Al-Husari, and Y. Zweiri, "Precision landing using an adaptive fuzzy multi-sensor data fusion architecture", Appl. Soft Comput., vol. 69, pp. 149–164, Aug. 2018. doi: 10.1016/J.ASOC.2018.04.025.

[39] A. Noureldin, A. El-Shafie, and N. El-Sheimy, "Adaptive neuro-fuzzy module for inertial navigation system/global positioning system integration utilising position and velocity updates with real-time cross-validatio",. doi: 10.1049/iet-rsn:20070001.

[40] S. H. Oh and D.-H. Hwang, "Low-cost and high performance ultra-tightly coupled GPS/INS integrated navigation method", Adv. Sp. Res., vol. 60, no. 12, pp. 2691–2706, Dec. 2017. doi: 10.1016/J.ASR.2017.06.007.

[41] H. Nourmohammadi and J. Keighobadi, "Fuzzy adaptive integration scheme for low-cost SINS/GPS navigation system", Mech. Syst. Signal Process., vol. 99, pp. 434–449, Jan. 2018. doi: 10.1016/J.YMSSP.2017.06.030.

[42] V. Havyarimana, D. Hanyurwimfura, P. Nsengiyumva, and Z. Xiao, "A novel hybrid approach based-SRG model for vehicle position prediction in multi-GPS outage conditions", Inf. Fusion, vol. 41, pp. 1–8, May 2018. doi: 10.1016/J.INFFUS.2017.07.002.

[43] C. H. Lim, T. S. Lim, and V. C. Koo, "Implementation of ANFIS for GPS-aided INS UAV motion sensing at short term GPS outage," J. Comput. Sci., vol. 10, no. 12, pp. 2564–2575, Dec. 2014. doi: 10.3844/jcssp.2014.2564.2575.

[44] K. Saadeddin, M. F. Abdel-Hafez, M. A. Jaradat, and M. A. Jarrah, "Optimization of Intelligent Approach for Low-Cost INS/GPS Navigation System," J. Intell. Robot. Syst., vol. 73, no. 1–4, pp. 325–348, Jan. 2014. doi: 10.1007/s10846-013-9943-2.

[45] X. Wang and W. Wang, "Nonlinear Signal-Correction Observer and Application to UAV Navigation," IEEE Trans. Ind. Electron., vol. 66, no. 6, pp. 4600–4607, Jun. 2019. doi: 10.1109/TIE.2018.2860540.

[46] Â. de C. Paulino, L. N. F. Guimarães, and E. H. Shiguemori, "An Adaptive Neuro-Fuzzy-based Multisensor Data Fusion applied to real-time UAV autonomous navigation," in ENIAC 2018 - Encontro Nacional de Inteligência Artificial e Computacional, 2018, pp. 847–858. doi: 10.5753/eniac.2018.4472.

[47] G. da P. Neto, Â. de C. Paulino, E. H. Shiguemori, H. F. de C. Velho, and L. N. F. Guimarães, "Computational Intelligence-based Multisensor Data Fusion applied to positioning estimation and autonomous navigation of an UAV," in Conference of Computational Interdisciplinary Science 2019, 2019, pp. 1–13.

[48] F. Castanedo, "A Review of Data Fusion Techniques", Sci. World J., vol. 2013, pp. 1–19, 2013. doi: 10.1155/2013/704504.

[49] F. E. White, "Data Fusion Lexicon", Jt. Dir. Lab. Tech. Panel C3, Data Fusion Sub-Panel, Nav. Ocean Syst. Cent., p. 16, 1991.

[50] B. Khaleghi, A. Khamis, F. O. Karray, and S. N. Razavi, "Multisensor data fusion: A review of the state-of-the-art", Information Fusion, vol. 14, no. 1. Elsevier, pp. 28–44, 01-Jan-2013. doi: 10.1016/j.inffus.2011.08.001.

[51] T. Takagi and M. Sugeno, "Fuzzy Identification of Systems and Its Applications to Modeling and Control", IEEE Trans. Syst. Man Cybern., vol. SMC-15, no. 1, pp. 116–132, 1985. doi: 10.1109/TSMC.1985.6313399.

[52] A. R. Kuroswiski, N. M. F. de Oliveira, and E. H. Shiguemori, "Autonomous long-range navigation in GNSS-denied environment with low-cost UAV platform," in 2018 Annual IEEE International Systems Conference (SysCon), 2018, pp. 1–6. doi: 10.1109/SYSCON.2018.8369592.

[53] T. J. Ross, Fuzzy logic with engineering applications, 3rd ed., vol. 761. West Sussex: John Wiley & Sons Ltd, 2010.

[54] Y. Tanaka, "An overview of fuzzy logic," in WESCON/'93. Conference Record, 1993, pp. 446–450. doi: 10.1109/WESCON.1993.488475.

[55] W. Mao and F.-Y. Wang, "Cultural Modeling for Behavior Analysis and Prediction," in Advances in Intelligence and Security Informatics, Elsevier, 2012, pp. 91–102. doi: 10.1016/B978-0-12-397200-2.00008-7.

[56] V. K. Kothari and D. Bhattacharjee, "Artificial neural network modelling for prediction of thermal transmission properties of woven fabrics," in Soft Computing in Textile Engineering, Elsevier, 2011, pp. 403–423. doi: 10.1533/9780857090812.5.403.

[57] D. Teodorović, "Traffic and Transportation Analysis Techniques," in Transportation Engineering, Elsevier, 2017, pp. 63–162. doi: 10.1016/B978-0-12-803818-5.00003-2.

[58] M. F. Møller, "A scaled conjugate gradient algorithm for fast supervised learning," Neural Networks, vol. 6, no. 4, pp. 525–533, Jan. 1993. doi: 10.1016/S0893-6080(05)80056-5.

[59] S. K. Das, "Artificial Neural Networks in Geotechnical Engineering," in Metaheuristics in Water, Geotechnical and Transport Engineering, Elsevier, 2013, pp. 231–270. doi: 10.1016/B978-0-12-398296-4.00010-6.

[60] J. C. Bezdek, R. Ehrlich, and W. Full, "FCM: The fuzzy c-means clustering algorithm", Comput. Geosci., vol. 10, no. 2, pp. 191–203, 1984. doi: 10.1016/0098-3004(84)90020-7.

[61] J.-S. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system", IEEE Trans. Syst. Man. Cybern., vol. 23, no. 3, pp. 665–685, 1993. doi: 10.1109/21.256541.

[62] P. Nerurkar, A. Shirke, M. Chandane, and S. Bhirud, "Empirical Analysis of Data Clustering Algorithms", Procedia Comput. Sci., vol. 125, pp. 770–779, Jan. 2018. doi: 10.1016/J.PROCS.2017.12.099.

[63] A. Saxena et al., "A review of clustering techniques and developments", Neurocomputing, vol. 267, pp. 664–681, Dec. 2017. doi: 10.1016/J.NEUCOM.2017.06.053.

[64] M.-S. Yang and Y. Nataliani, "Robust-learning fuzzy c-means clustering algorithm with unknown number of clusters", Pattern Recognit., vol. 71, pp. 45–59, Nov. 2017. doi: 10.1016/J.PATCOG.2017.05.017.

[65] L. A. Zadeh, "Fuzzy sets", Inf. Control, vol. 8, no. 3, pp. 338–353, 1965. doi: 10.1016/S0019-9958(65)90241-X.

[66] The MathWorks Inc., "FCM", 2018. [Online]. Available: https://www.mathworks.com/help/fuzzy/fcm.html. [Accessed: 24-May-2018].

[67] S. Kamarian, M. H. Yas, A. Pourasghar, and M. Daghagh, "Application of firefly algorithm and ANFIS for optimisation of functionally graded beams", J. Exp. Theor. Artif. Intell., vol. 26, no. 2, pp. 197–209, 2014. doi: 10.1080/0952813X.2013.813978.

[68] H. Zamani Sabzi, D. Humberson, S. Abudu, and J. P. King, "Optimization of adaptive fuzzy logic controller using novel combined evolutionary algorithms, and its application in Diez Lagos flood controlling system, Southern New Mexico", Expert Syst. Appl., vol. 43, pp. 154–164, Jan. 2016. doi: 10.1016/J.ESWA.2015.08.043.

[69] S. Amr and S. Qin, "Robust adaptive flight controller for UAV systems", in Proceedings - 2017 4th International Conference on Information Science and Control Engineering, ICISCE 2017, 2017, pp. 1214–1219. doi: 10.1109/ICISCE.2017.252.

[70] L. H. Tsoukalas and R. E. Uhrig, Fuzzy and Neural Approaches in Engineering, 1st ed. New York, NY, USA: John Wiley & Sons, Inc., 1997.

[71] A. Al-Hmouz, Jun Shen, R. Al-Hmouz, and Jun Yan, "Modeling and Simulation of an Adaptive Neuro-Fuzzy Inference System (ANFIS) for Mobile Learning", IEEE Trans. Learn. Technol., vol. 5, no. 3, pp. 226–237, 2012. doi: 10.1109/TLT.2011.36.

[72] G. Conte and P. Doherty, "Vision-Based Unmanned Aerial Vehicle Navigation Using Geo-Referenced Information", EURASIP J. Adv. Signal Process., vol. 2009, no. 1, p. 387308, Dec. 2009. doi: 10.1155/2009/387308.

[73] S. Arlot and A. Celisse, "A survey of cross-validation procedures for model selection", Stat. Surv., vol. 4, pp. 40–79, 2010. doi: 10.1214/09-SS054.

[74] G. Varoquaux, "Cross-validation failure: Small sample sizes lead to large error bars", Neuroimage, Jun. 2017. doi: 10.1016/J.NEUROIMAGE.2017.06.061.

[75] Y. Zhang and Y. Yang, "Cross-validation for selecting a model selection procedure", J. Econom., vol. 187, no. 1, pp. 95–112, 2015. doi: 10.1016/j.jeconom.2015.02.006.

[76] C. Bergmeir, R. J. Hyndman, and B. Koo, "A note on the validity of cross-validation for evaluating autoregressive time series prediction," Comput. Stat. Data Anal., vol. 120, pp. 70–83, 2018. doi: 10.1016/j.csda.2017.11.003.

[77] L. Xu et al., "Stochastic cross validation," Chemom. Intell. Lab. Syst., vol. 175, pp. 74–81, Apr. 2018. doi: 10.1016/J.CHEMOLAB.2018.02.008.

[78] T. T. Wong, "Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation," Pattern Recognit., vol. 48, no. 9, pp. 2839–2846, Sep. 2015. doi: 10.1016/j.patcog.2015.03.009.

[79] S. Berisha and J. G. Nagy, "Iterative Methods for Image Restoration," in Academic Press Library in Signal Processing, vol. 4, Elsevier, 2014, pp. 193–247. DOI: 10.1016/B978-0-12-396501-1.00007-8.

[80] Z. Sun, Y. Chen, X. Li, X. Qin, and H. Wang, "A Bayesian regularized artificial neural network for adaptive optics forecasting," Opt. Commun., vol. 382, pp. 519–527, Jan. 2017. DOI: 10.1016/J.OPTCOM.2016.08.035.

[81] A. Sharif Ahmadian, "Numerical Methods and Procedures," in Numerical Models for Submerged Breakwaters, Elsevier, 2016, pp. 93–108. DOI: 10.1016/B978-0-12-802413-3.00006-7.

[82] G. M. Fitzmaurice, "Regression," Diagnostic Histopathol., vol. 22, no. 7, pp. 271–278, Jul. 2016. DOI: 10.1016/j.mpdhp.2016.06.004.

[83] M. Fernández-Delgado, M. S. Sirsat, E. Cernadas, S. Alawadi, S. Barro, and M. Febrero-Bande, "An extensive experimental survey of regression methods," Neural Networks, vol. 111, pp. 11–34, Mar. 2019. DOI: 10.1016/J.NEUNET.2018.12.010.

[84] B. Choubin, E. Moradi, M. Golshan, J. Adamowski, F. Sajedi-Hosseini, and A. Mosavi, "An ensemble prediction of flood susceptibility using multivariate discriminant analysis, classification and regression trees, and support vector machines," Sci. Total Environ., vol. 651, pp. 2087–2096, Feb. 2019. DOI: 10.1016/J.SCITOTENV.2018.10.064.

[85] A. Nazarpour, G. R. Paydar, and E. J. M. Carranza, "Stepwise regression for recognition of geochemical anomalies: Case study in Takab area, NW Iran," J. Geochemical Explor., vol. 168, pp. 150–162, Sep. 2016. DOI: 10.1016/J.GEXPLO.2016.07.003.

[86] I. The MathWorks, "Choose Regression Model Options," 2019. [Online]. Available: https://www.mathworks.com/help/stats/choose-regression-model-options.html. [Accessed: 11-Mar-2019].

[87] R. N. Forthofer, E. S. Lee, and M. Hernandez, "Linear Regression," in Biostatistics, Elsevier, 2007, pp. 349–386. DOI: 10.1016/B978-0-12-369492-8.50018-2.

Appendix

See tables 7 and 8.

Table 7: Best results for each training method - ANNs.

Supervised Learning Models		**Best results (cm)**	
Class	Training method	Latitude	Longitude
Artificial Neural Networks	Levenberg-Marquardt	17,02	24,70
Artificial Neural Networks	Bayesian Regularization	640,43	1840,78
Artificial Neural Networks	Scaled Conjugate Gradient	1518,32	5894,84

Table 8: Best results for each approach - RMs.

Unsupervised Learning Models		**Best results (cm)**	
Class	Approach	Latitude	Longitude
Linear Regression	Linear	19,18	25,46
Linear Regression	Interactions Linear	19,15	25,45
Linear Regression	Robust Linear	19,46	25,44
Linear Regression	Stepwise Linear	19,24	25,46
Tree	Fine Tree	19,34	24,34
Tree	Medium Tree	20,53	25,94
Tree	Coarse Tree	28,41	39,30
Support Vector Machine	Linear SVM	32,95	55,99
Support Vector Machine	Quadratic SVM	61,91	51,75
Support Vector Machine	Cubic SVM	119,24	58,98
Support Vector Machine	Fine Gaussian SVM	300,97	664,14
Support Vector Machine	Medium Gaussian SVM	52,48	119,63
Support Vector Machine	Coarse Gaussian SVM	34,49	58,37
Ensemble	Boosted Trees	2,76E+07	7,11E+06
Ensemble	Bagged Trees	28,96	62,39