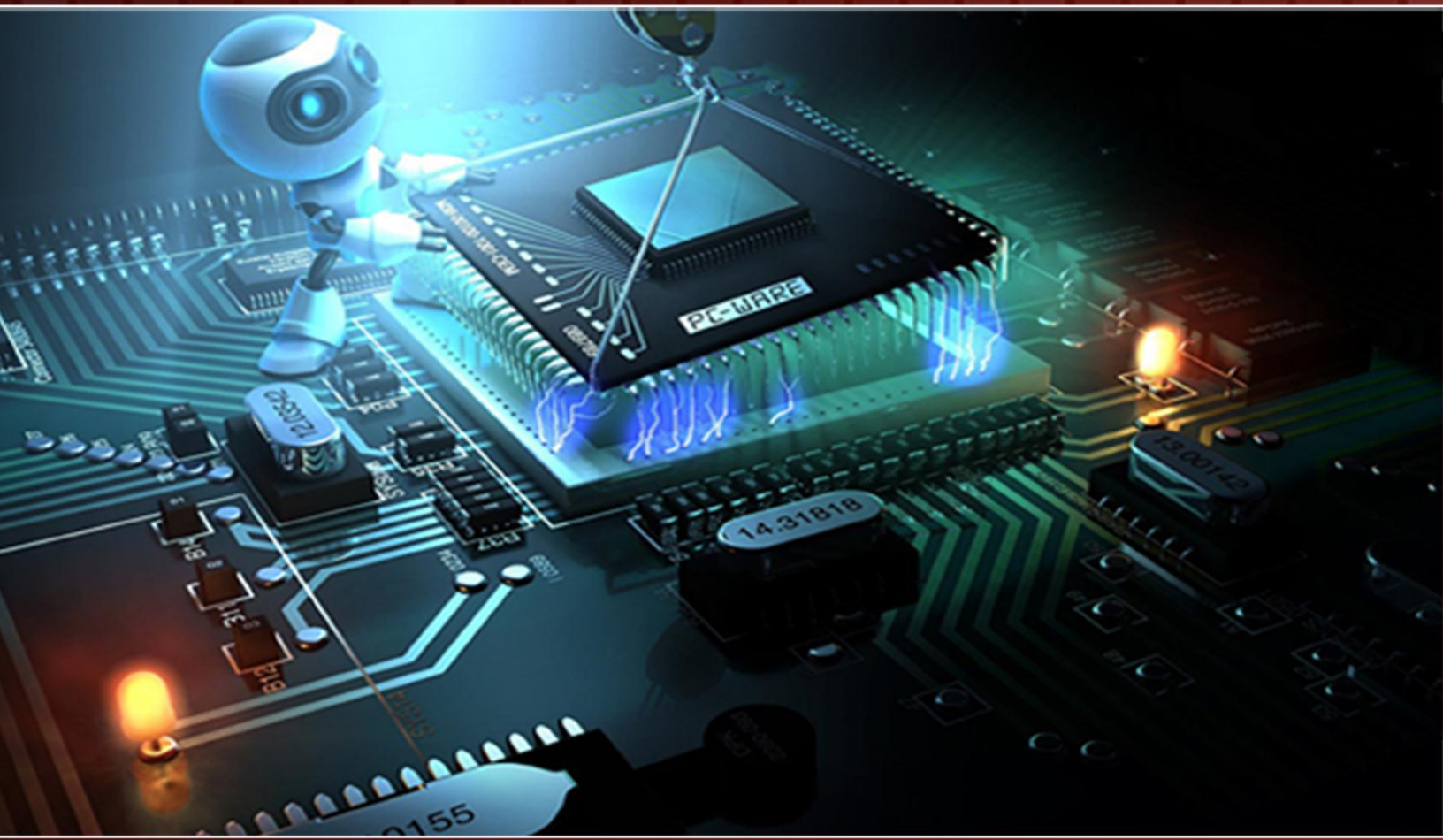
ISSN: 2054-7390

Volume 3 Issue 3, June 2015

Transactions on Machine Learning and Artificial Intelligence



Society for Science and Education
UNITED KINGDOM

TABLE OF CONTENTS

EDITORIAL ADVISORY BOARD

DISCLAIMER

All the contributions are published in good faith and intentions to promote and encourage research activities around the globe. The contributions are property of their respective authors/owners and the journal is not responsible for any content that hurts someone's views or feelings etc.

SOCIETY FOR SCIENCE AND EDUCATION
UNITED KINGDOM
TMLAI TRANSACTIONS ON MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE
VOLUME 3, ISSUE 3
ISSN 2054-7390

Data Mining and Other Data Base Techniques for Ph.D Thesis Preparation

[1]Srinatha Karur, [2]M.V.Ramana Murthy
[1]Orabyte Software Solutions, RTC X Roads, Hyderabad, India;
[2]School of Computer Science & Mathematics, Osmania University, Hyderabad, India;
karurdori@gmail.com; mv.rm50@gmail.com

ABSTRACT

The authors in this paper present the role of Data mining and Data base techniques for estimate the quality of thesis or dissertation at Research level. The Doctorate Research consists of various components which are highly defined by University Research Committee or any other concerned authorities. For all general cases the thesis book consists of different chapters with different aims. Each and every chapter has its own identity and constantly has relation with previous chapters. The authors use different Data mining and Data base techniques for determine the correlation between different entities which are involved in the thesis such as page numbers, references, diagrams, equations , graphs, different concepts covered in the thesis book.

Keywords: Data mining techniques, SQL, thesis preparation, relation between entities.

1 Introduction

Education data mining become more dominant area and most consistent domain. Education Data mining deals about not only deals about the relation and scope between different Education systems and also how we can implement and do research as per needs of enhanced Education system. The enhanced systems generally deal about the higher versions of available systems. For example e-Learning is the enhanced version of Distance Education system. The Distance Education system is enhanced version of traditional or regular system. All enhanced versions are generally convergent as per static needs and once again fired or executes when requirement generated. It is like client server process when client requests server executes and vice versa. The PhD thesis are highest level of code execution or conduct for confer the degree by the University globally. We can use Data mining techniques either Supervised or Unsupervised or Hybrid methods or Semi supervised or Semi Unsupervised (Partial clusters). Semi unsupervised leads partial clusters and more inconsistent results are generated [1]. The author discussed about the semi unsupervised and semi supervised methods in [1] and defines the role and nature of label, unlabeled, and little label and little unlabeled. The authors published different papers on thesis preparation which covers all methods of data mining except Principle Component Analysis (PCA) since they need more knowledge on Mathematical and Statically concepts. Moreover PCA are used to convert orthogonal correlated to un correlated variables which is strictly out of scope of thesis [2]. General real time application or problem deals or gives important to estimate the correlation between available entities but not its negation values. Mainly the authors used Naïve Bayes method, Linear Regression Analysis, Rule based decision trees, Different probability distributions, nonlinear

DOI: 10.14738/tmlai.33.1227
Publication Date: 30th May, 2015
URL: http://dx.doi.org/10.14738/tmlai.33.1227

equations (Log and Exponential Curves), confusion matrix, Geni index, lift, outlier's estimation etc. as a part of supervised methods. The more information about data preparation and implementation details are available in [3, 4, 5, 6, 7, 8]. In these publications the authors are implemented the required phases very successfully and integrated the all phases of thesis for final submission. The authors are used most of the time Hierarchical clusters only on the basis of easy understanding. Since most of the Research community using Agloromative (Bottom to Top) Hierarchical clusters only. K-means also used frequently and EM also used as per context. The authors used very popular Data mining tools such as Tanagra, Weka, R, Orange, Rapid miner, as freeware tools and MS-SQL 2008 R2 as licensed software[8]. The author's main aim of this paper is to estimate the relation between page numbers and number of references in thesis book which is available as final copy for final submission to University as a vital part of course. Meanwhile the authors observed various parameters such as the role of supervised methods and unsupervised methods, results, analysis, tables etc. are available as a part of thesis and how they are related with each other..

2 Data preparation and Experiments

The authors use Weka for main Data mining processing and Tanagra for even Statistics events also. Microsoft Excel is used for find out the linear and other relationship between the defined or available variables. The authors use both continuous and discrete variables for Data preparation and implementation purpose. The below figure shows data is successfully loaded into Weka in .arff form with 7 fields and 301 instances are as follows.

Reference for thesis.arff

Relation: Reference for thesis

No.	1: PageNo Numeric	2: Refno Numeric	3: Graphs Nominal	4: eLearning Nominal	5: Datamining Nominal	6: Pagetype Nominal	7: **Reftype** Nominal
294	154.0	0.0	Y	N	Y	E	Single
295	155.0	0.0	Y	N	Y	O	Single
296	156.0	0.0	Y	N	Y	E	Single
297	157.0	0.0	Y	N	Y	O	Single
298	158.0	0.0	Y	N	Y	E	Single
299	159.0	0.0	Y	N	Y	O	Single
300	160.0	0.0	Y	N	Y	E	Single
301	161.0	0.0	Y	N	Y	O	Single

Figure 1: Shows data is successfully loaded

Page numbers and reference numbers are numeric whereas remaining are character type and we can prepare the data as per needs and method. The weka tool gives various distributions with respect to different field values are as follows.

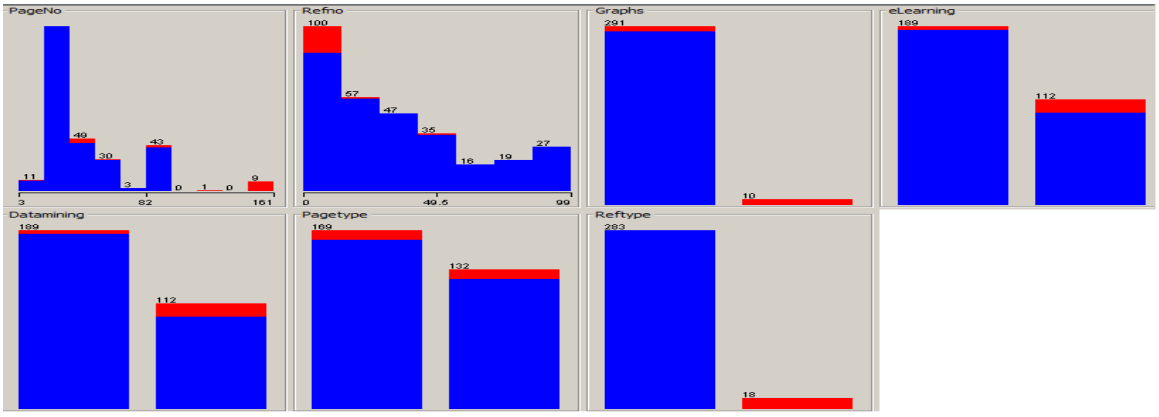


Figure 2: Shows success and failure rates for different classes in data

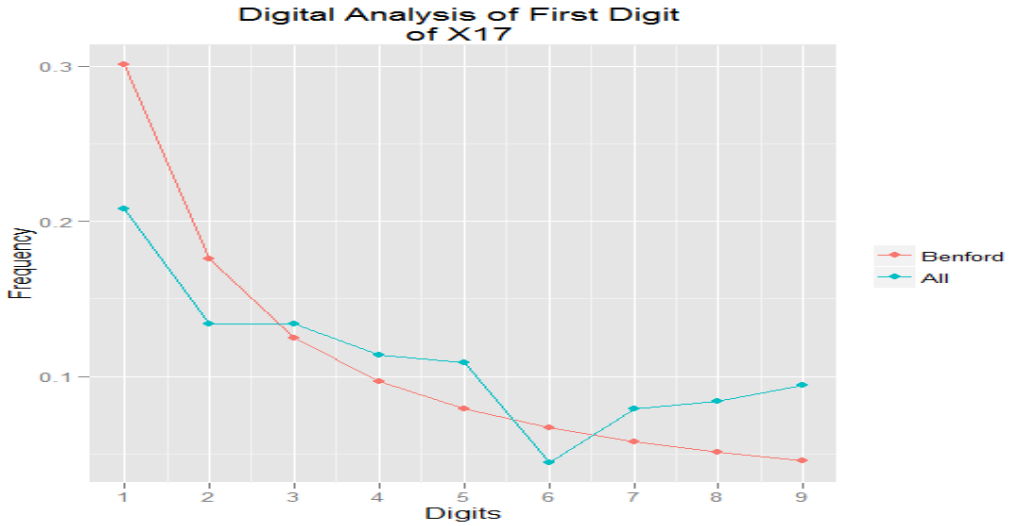


Figure 3 : Shows Benford graph for x and y components

Benford's law, also called the First-Digit Law, refers to the frequency distribution of digits in many (but not all) real-life sources of data. In this distribution, 1 occurs as the leading digit about 30% of the time, while larger digits occur in that position less frequently: 9 as the first digit less than 5% of the time. Benford's law also concerns the expected distribution for digits beyond the first, which approach a uniform distribution. The mathematical equation of Benford law is as follows.

$$P(d) = \log_{10}(d+1) - \log_{10}(d) = \log_{10}\left(\frac{d+1}{d}\right) = \log_{10}\left(1+\frac{1}{d}\right). \quad (1)$$

The quantity P(d) is proportional to the space between d and d + 1 on a logarithmic scale. n extension of Benford's law predicts the distribution of first digits in other bases besides decimal; in fact, any base b ≥ 2. For example the linear regression analysis only first two field values are enough and for Naïve Bayes only classes or attributes are enough. The authors used MS-Excel for estimate the linear and higher degree relations which are as follows and have been shown in below graph form.

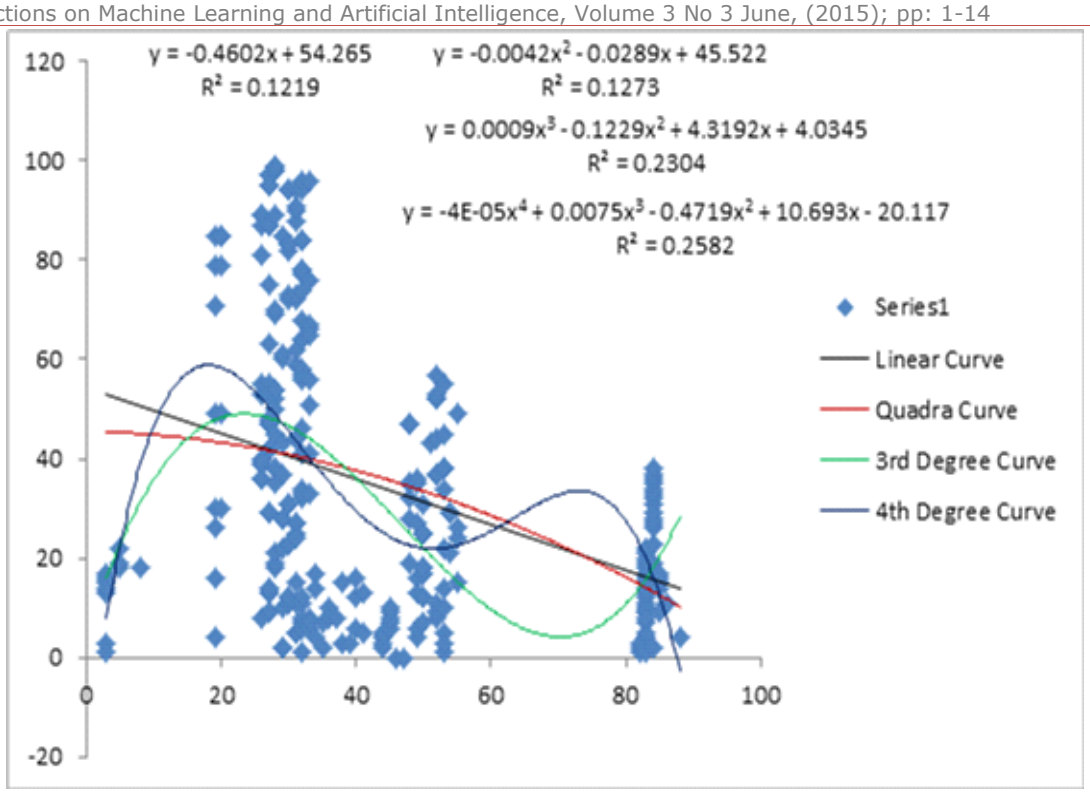


Figure 4: linear and higher degrees for page number and number of references

The authors used to find out the higher degree of relations between page number and number of references which are used during thesis writing. The authors observed that linear and secondary orders are formed almost straight lines and especially linear curve is highly intersects at x and y axis with in first quadrant as shown in the figure. The higher orders > 2 are strictly curve nature is as shown in the figure-3. For residuals and standard error estimation the authors are used curve expert software with numeric values of first and second fields of given data base with 301 instances are shown in the figure-1. The general form of non linear function is given by f(x) = anxn + an-1xn-1 + an-2xn-2 +..... + a1x1 + a0 where a0, a1……. an are stables. In this non linear functions, an is a primary co –efficient and anxn is the principal term. The greatest degree of non – linear function is greater than or similar to 2.A quadratic function is in the form y = ax2 + bx + c, where c ≠ 0 is a non-linear equation. Similarly, a cubic function y = ax3 + bx2 + cx + d, where a≠ 0 is a non-linear equation. Non-linear functions are those which do not form a straight line when graphed. One of the functions which are not a linear function and cannot be a complete linear function by transforming the Y variable.

There are three nonlinear functions normally used in mathematics as follows,

- Exponential function
- Quadratic function
- Logarithmic function

The meaning of the non-linear functions cannot be overstated since without them, thus without graphing it would not be a function. The original testing in the field of simulated equations failed since there was no clear understanding of the importance of non linearity in the output point.

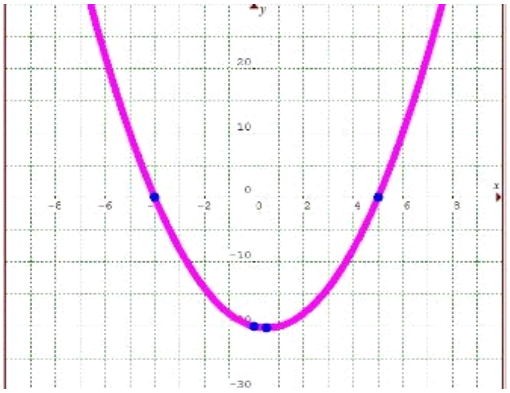


Figure 5: Non linear form for degree 2

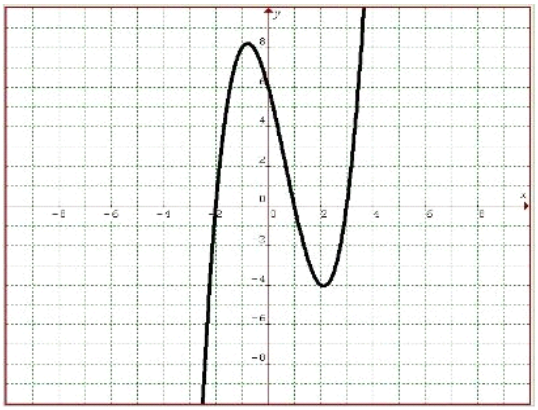


Figure6: Non linear form of degree>2

The authors observed the same data for NavieBayes Networks and NavieBayes the experiment is repeated for different values and dimensions. All values are recorded and mentioned in Results section. The authors observed the tree in both Naive layout and priority layout. The figures are as follows.

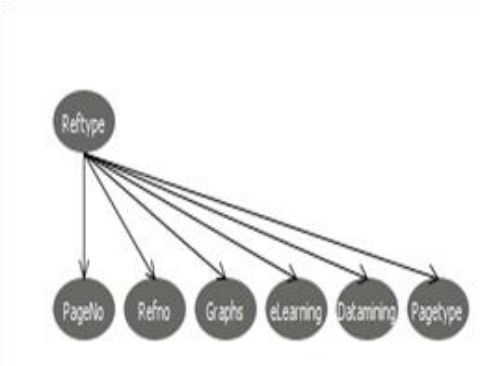


Figure 7: Shows Navies layout and each node has prob>0.2

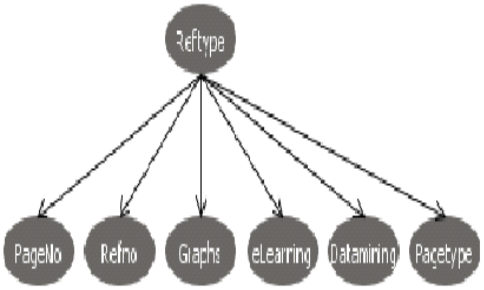


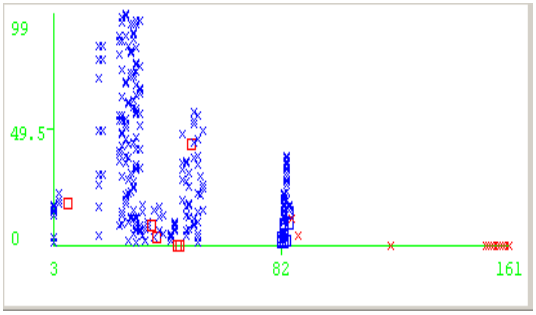
Figure 8: Shows priority model and minimum probability is 0.3

All confusion matrices are available in Results and analysis section and the authors are observed that there is little variation in confusion matrices of Navie and priority models. Due to the format of the paper the authors did not present the confusion matrix values in this section and it is available along with other experiment results. The authors test the data with Naive bayes model consists of the following things.

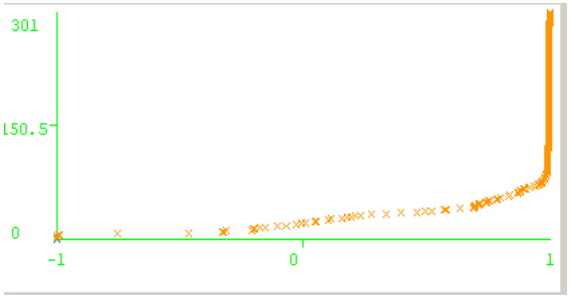
- Visualization Margin curve
- Visualize threshold curve
- Cost/Benefit analysis
- Cost curve analysis.

The authors observed the following things during the data testing and analysis for Naive Bayes. The authors test the data with this constraint

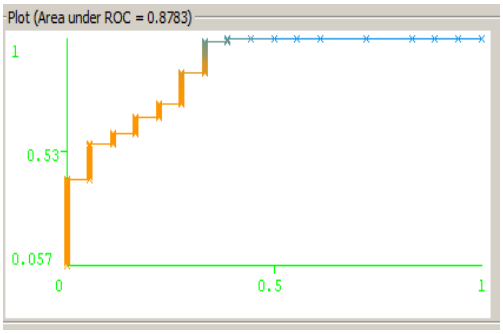
- Along the x-axis False positive rate
- Along the y-axis True positive rate



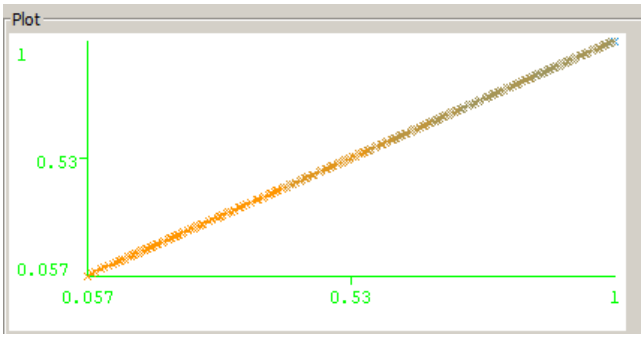
Visualize the errors for Pageno and references classes



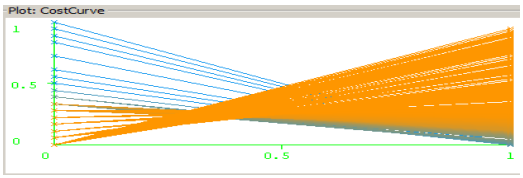
Marginal Curve for Pageno and references curve



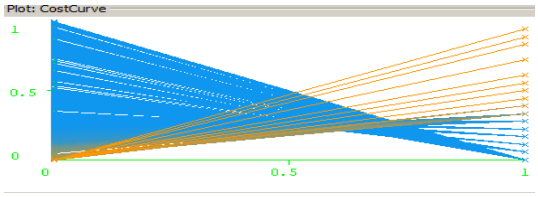
Visualize threshold value for Multiple class Under ROC



Visulaize the threshold curve under plot



For multiple class cost curve



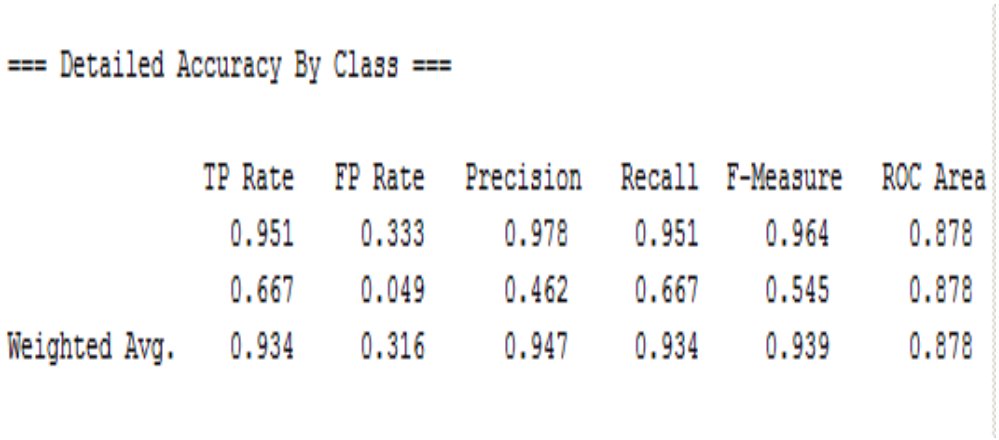
For single class Cost curve

Figure 9: Shows Naive Bayes modeling with factors

The authors tested the data with various properties are shown in the figure as follows. The mathematical statistics for the given data is as follows.

Incorrectly Classified Instances	20	6.6445 %
Kappa statistic	0.5109	
Mean absolute error	0.0813	
Root mean squared error	0.2136	
Relative absolute error	70.6934 %	
Root relative squared error	90.0848 %	
Coverage of cases (0.95 level)	98.3389 %	
Mean rel. region size (0.95 level)	59.9668 %	
Total Number of Instances	301	

Figure 10: Statistics for Page no and reference



=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area
	0.951	0.333	0.978	0.951	0.964	0.878
	0.667	0.049	0.462	0.667	0.545	0.878
Weighted Avg.	0.934	0.316	0.947	0.934	0.939	0.878

Figure 11: Different parameters for Naïve Bayes.

Table 1: Shows nature of single and multiple, classes graph nature for given data

S. No	Property	Single class	Multiple class
1	Precision	Depend	depend
2	Recall	Strictly diagonal	Diagonal
3	Fall out	Depend	depend
4	F-Measure	Curve on y-axis	Almost diagonal
5	Sample size	depend	diagonal
6	Lift	Zigzag line on y-axis	depend
7	Curve nature	depend	Generally diagonal

The authors observed the relation between various intervals for Single and multiple classes are as follows. Empty cells indicate any type of nature even straight line passing through origin. For linear and non linear relations along with R^2 values are as follows (all r2 values are >0)

$$y=1.103x+0.717 \quad (2)$$

$$R^2 = 0.621 \quad (3)$$

$$y = -2.150x^2 + 3.324x + 0.628 \quad (4)$$

$$R^2 = 0.644 \quad (5)$$

$$y = 121.3x^3 - 135.4x^2 + 15.39x + 0.542 \quad (6)$$

$$R^2 = 0.730 \quad (7)$$

The authors observed the following points are as their observation

- All r2 values are >0
- All x-coefficients are >0
- All constants are >0
- For quadratic equation factors are x1>0 and x2=1.7162

The authors used online tool for solve the quadratic equation [9].

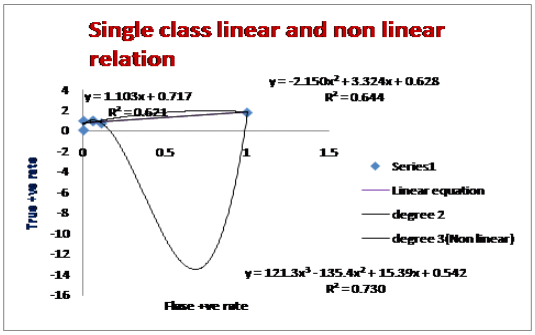


Figure 12: Shows linear and non linear for Single class for Naive Bayes

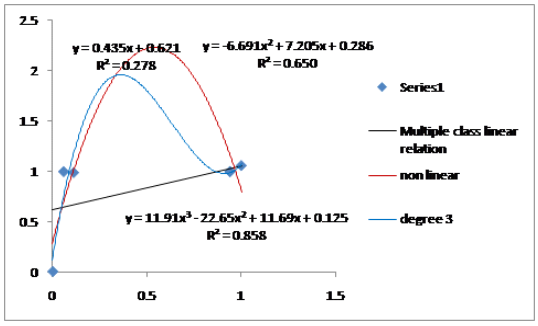


Figure 13: Shows linear and non linear for Multiple classes Naïve Bayes

Tree

- Refno < 43.5000
  - Refno < 20.5000 then cluster n°1, with 130 examples (43.19%)
  - Refno >= 20.5000 then cluster n°2, with 76 examples (25.25%)
- Refno >= 43.5000
  - Refno < 69.5000 then cluster n°3, with 48 examples (15.95%)
  - Refno >= 69.5000 then cluster n°4, with 47 examples (15.61%)

Computation time : 0 ms.
Created at 3/22/2015 12:00:11 PM

Figure 13A: Tree rules for given data

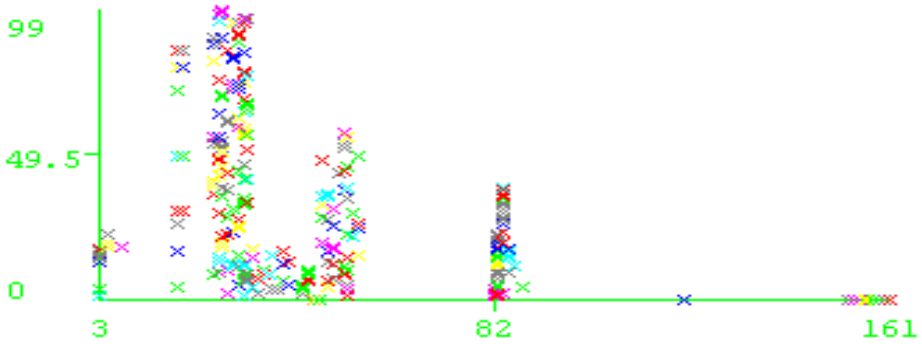
The authors tested the data for Unsupervised and semi unsupervised methods such as EM method and the diagrams are as follows.

```
Time taken to build model (full training data) : 4.12 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       102  ( 34%)
1        50  ( 17%)
2        10  (  3%)
3       139  ( 46%)


Log likelihood: -9.69074
```

Figure 14: EM implementation with Weka with log likely hood -9.7

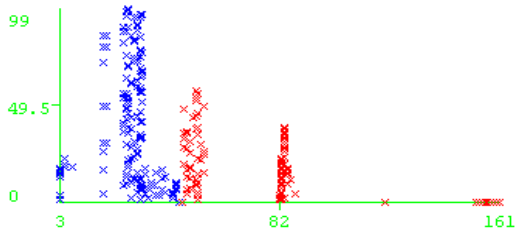
From the above figure it is observed that four clusters are formed and second cluster consists of only 3% of instances which forms very small cluster and cluster 3 is a big cluster and has 46% of instances. Both are at extreme values. For many applications, the natural logarithm of the likelihood function, called the log-likelihood, is more convenient to work with. Because the logarithm is a monotonically increasing function, the logarithm of a function achieves its maximum value at the same points as the function itself, and hence the log-likelihood can be used in place of the likelihood in maximum likelihood estimation and related techniques. Finding the maximum of a function often involves taking

the derivative of a function and solving for the parameter being maximized, and this is often easier when the function being maximized is a log-likelihood rather than the original likelihood function. It is observed that from the above figure minimum and maximum occurrences are one by one and intermediate values are formed randomly. The four clusters formation is as shown in the figure

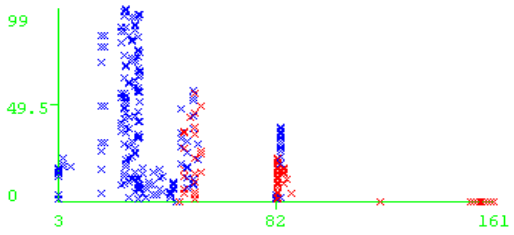


Cobweb method shows 4 clusters are formed with 0.016 seconds

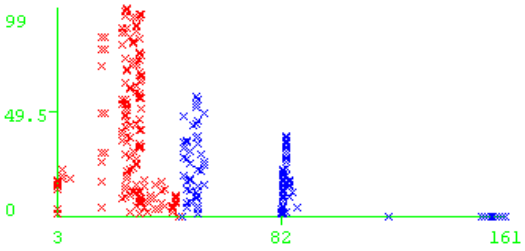
It is observed that from the figure between [3,82] interval maximum events are occurred and after this interval almost negligible events are occurred and these events or objects are called idle objects and independent on clusters and we can say deviate from clusters almost. The authors repeat the experiment for all remaining unsupervised methods and the noted the contents is as follows.



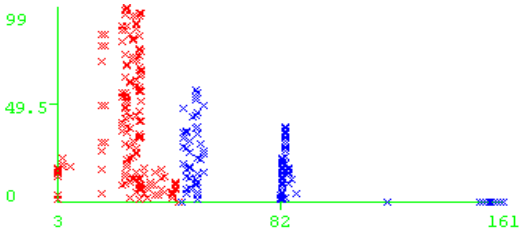
Hierarchical clusters with 4.23 seconds



Farthest first clusters within 0.01 seconds

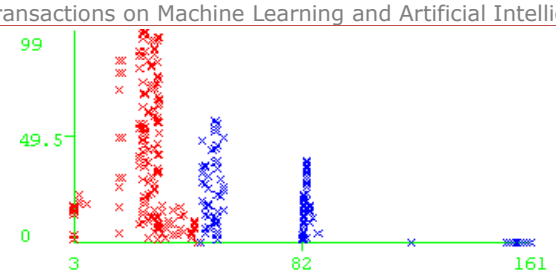


Filtered cluster with in 0.01 sec

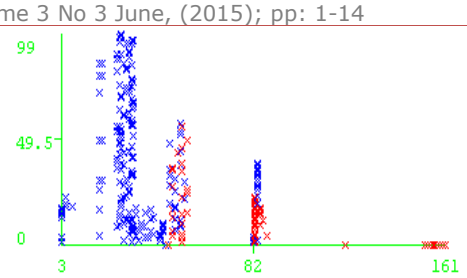


Density based clusters with -10.25311 likely hood functions with 0.04 seconds

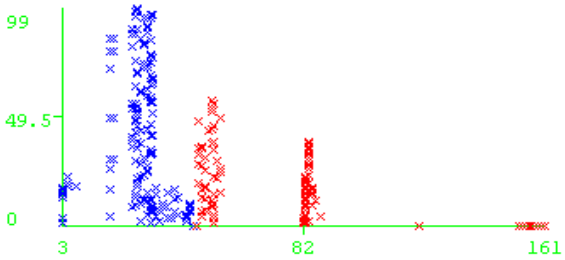
Figure 15: shows different methods of Clusters



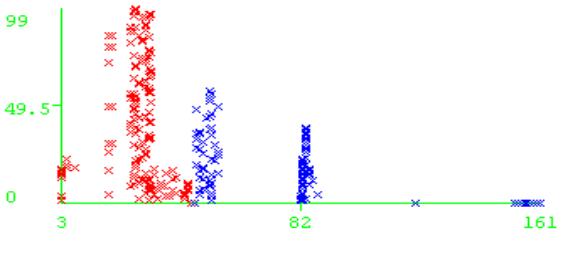
K-Means with 2 clusters 0.01 seconds



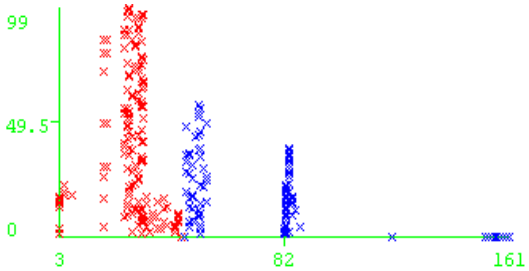
Farthest cluster assignments



Hierarchical clusters with 4.23 seconds



Filtered cluster with in 0.01 sec



Density based clusters with -10.25311 likely hood function with 0.04 seconds

Figure 16: Shows different clusters assignments

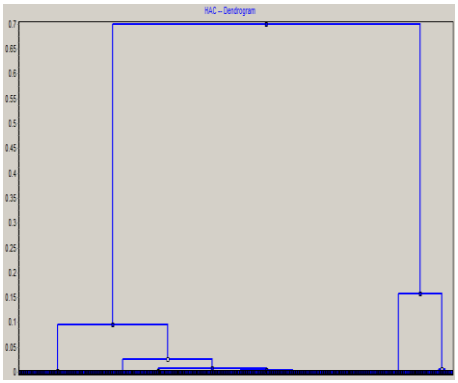
Classifier performances

Error rate			0.1860			
Values prediction			Confusion matrix			
Value	Recall	1-Precision		Y	N	Sum
Y	1.0000	0.2286	Y	189	0	189
N	0.5000	0.0000	N	56	56	112
			Sum	245	56	301

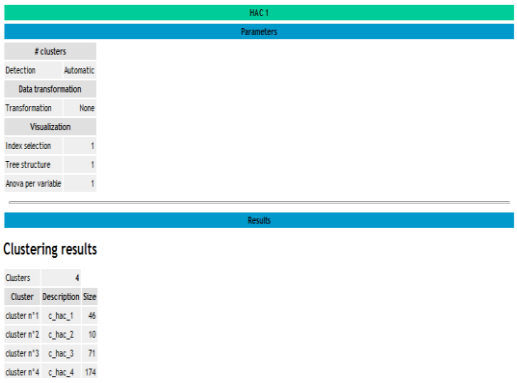
Figure 17B: Shows cluster assignments and SVM parameters

SVM Parameters	
Exponent	1
Filter type	NORMALIZE
Use polynom space normalization	0
Use RBF kernel	0
Gamma for RBF kernel	0.0100
Complexity	1.0000
Calculation parameter	
Epsilon for rounding	1.0E-012
Tolerance for accuracy	1.0E-003

Figure 17A: Shows SVM parameters



Hierachical clusters with TANAGRA with height=0.7



Formed 4 clusters with maximum and minimum instances

Figure 18: Shows HR with 4 clusters

3 Results and Analysis

The authors tested and observed the data between no of pages and number of references is as shown in figure-1. The authors repeat the HC repeatedly for different types of distances and links then the results are as follows. The authors used R software with Rattle GUI for this purpose and the tool snapshot is not available in this paper.

Table-2 shows HC for different Distance methods and links

S. No	Distances	Wards	Complete	Single	Average	Mequitty	Median	Centroid
1	Euclidian	2000	90	20	60	60	40	40
2	Maximum	2300	42	25	50	60	40	42
3	Manhattan	4000	150	30	80	80	65	50
4	Canberra	40	2.0	1.0	1.5	1.75	2	1.5
5	Binary	0	0	0	0	0	0	0
6	Pearson	30	0.8	0.02	0.3	0.3	0.3	0.3
7	Correlation	190	2.0	0	1.75	1.75	2	1.8
8	Spearman	200	2.0	2.0	2.0	2.0	2.0	2.0

The authors used supervised methods also for estimate the classifiers of a given problem. More details are available in [11]. Different data mining tools are available for applying these evaluation methods. For all popular tools such as Weka, Tanagra, Oranage, Rapid miner, R with Rattle are used CONFUSION MATRIX as common evaluation methods. For more details of this implementation by these tools are available in their respective documentation. The tool R consists of not only command prompt and also lot of GUI tools

for implementation as highly user friendly. For more details of tools of R is available in http://www.linuxlinks.com/article/20110306113701179/GUIsforR.html. The authors applied different evaluation methods for Naïve Bayes are as follows.

- Visualization Margin curve
- Visualize threshold curve
- Cost/Benefit analysis
- Cost curve analysis.

All the nature of graphs are tabulated in table-1. Confusion matrix for Naïve net and Naïve Bayes , and Naïve Bayes Update is as follows.

```
=== Confusion Matrix ===

   a   b   <-- classified as
 283   0 |   a = Multiple
   5  13 |   b = Single
```

```
=== Confusion Matrix ===

   a   b   <-- classified as
 269  14 |   a = Multiple
   6  12 |   b = Single
```

Naive Bayes update table

```
=== Confusion Matrix ===

   a   b   <-- classified as
 269  14 |   a = Multiple
   6  12 |   b = Single
```

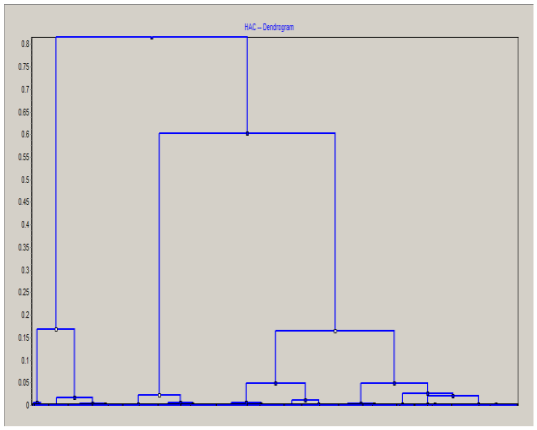
Naive Bayes confusion matrix

Figure 19: Shows confusion matrix for Naïve net, Naïve Bayes and Update Naïve models.

The authors note the generated output of Naïve Bayes Evaluation methods are as follows. The evaluation methods are mentioned in Table-3

S. No	Name	Single Class IV	Multi Class IV
1	Precision	[0.06,1]	[0.94,1]
2	Recall	[0.056,1]	[0.057,1]
3	Fall out	[0,0.94]	[0,0.06]
4	FMeasure	[0.11,0.76]	[0.11,0.99]
5	Sample size	[0.0033,1]	[0.94,1]
6	Lift	[1,11.2]	[1,1.06]

The cost benefit analysis for Naïve Bayes method is -11.2957 and 97.6744 with respect to Max cost and min cost where along the x axis Sample size and along the y axis cost/benefit is available. More details and mathematical model of Supervised Vector Machine are available in [11]. SVM mathematical modeling is like Linear Programming problem model and its details are out of scope. The authors finally tested for clustering and tree rules. The output is as follows. Finally the authors used to find out the outliers estimation for given or prepared data. All the above aims are available as follows. For this the authors used TANAGRA software. It is observed that 0 outliers are found for Univariate from below figure-21.



Parameters		Filtering Parameters	
Grubbs' p-value	0.0500	Remove outliers	1
Multiple of sigma	3.0	Use sigma	1
Use Fences	1	Use inner fence	1
Show outliers	0	Use outer fence	1

Computation time : 0 ms.
Created at 3/22/2015 11:40:47 AM

Univariate Outliers Detection

Detailed results for each variable

Removed outliers (if option activated) : 0

Figure 20: For both tree rules and HC

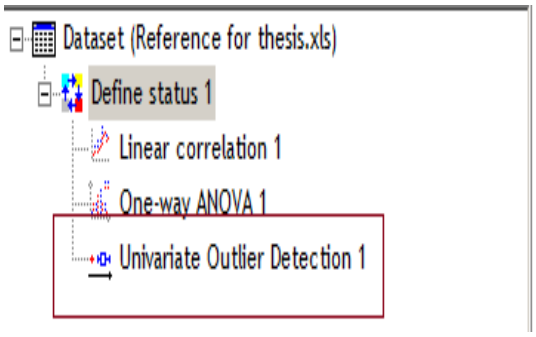


Figure 21: Shows outliers estimation for given thesis

4 Conclusions

Using Data mining techniques it is possible to estimate the relation between to estimate the relation between various factors such as relation between number of pages and references used, how the chapters are distributed and how the topics are distributed, Statistical view of entire nature of distribution of information, how the diagrams are correlated and how many diagrams and tables are arranged in odd and even pages and their relation etc. we can find easily. We can also find the role of mathematical equations and its distribution throughout the thesis book. The authors are used only interpretation concept and did not test for orthogonal trajectories which mainly deals the Principal component analysis. The authors also estimated the outliers for given thesis and found that almost zero errors are available. The authors used only TANAGRTA software for outlier's estimation. We can examine these outliers nature and estimation using different free Data mining tools such as Weka, Orange, Rapid miner and R.

REFERENCES

[1] www.umiacs.umd.edu/~hal/docs/daume09sslnlp.pdf

[2] http://en.wikipedia.org/wiki/Principal_component_analysis.

[3] www.airccse.org/journal/ijaia/papers/4513ijaia02.pdf

[4] www.airccse.org/journal/ijaia/papers/4413ijaia12.pdf

[5] www.gssrr.org/index.php?journal=JournalOfBasicAndApplied..,

[6] www.ijecs.in/issue/v2-i10/16%20ijecs.pdf

[7] http://www.ijettcs.org/Volume2Issue6/IJETTCS-2013-12-10-061.pdf

[8] www.ijaiem.org/volume3issue5/IJAIEM-2014-05-29-093.pdf

[9] http://www.math.com/students/calculators/source/quadratic.htm

[10] http://www.bth.se/fou/forskinfo.nsf/0/c655a0b1f9f88d16c125714c00355e5d/$file/Lavesson_lic.pdf

[11] http://www.cs.columbia.edu/~kathy/cs4701/documents/jason_svm_tutorial.pdf

SOCIETY FOR SCIENCE AND EDUCATION
UNITED KINGDOM
TMLAI TRANSACTIONS ON MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE
VOLUME 3, ISSUE 3
ISSN 2054-7390

Accuracy of the Java Simulation for the Charge Motion in Electric and Magnetic Fields

Masami Morooka[1] and Midori Morooka[2]
[1]*Department of Electrical Engineering, Fukuoka Inst. Tech, higashi-ku, Fukuoka 811-0295, Japan;*
[2]*Flash Design Center, Micron Japan, Kamata 5-37-1, Ota-ku, Tokyo144-8721, Japan*
morooka@ee.fit.ac.jp

ABSTRACT

The accuracy of the Java simulation by the Runge-Kutta method for the charge motion in electric and magnetic fields has been investigated in comparison with the analytical solution. The error of the simulation depends on the time increment, h, used for the numerical calculation. If we use an increment that is larger than the boundary value, the simulation results in a non-accurate image of the charge motion. In this case, the simulation almost results in an underestimation, that is, a motion that is smaller than the real motion. The boundary increment is proportional to the mass of the charge, m, and is inversely proportional to the charge, q, and the magnetic field, B0. The empirical results conclude that the image of the charge motion can be obtained accurately by Java simulation using $h < 0.2m/qB0$.

Keywords: Image learning, charge motion in electric and magnetic fields, Java programming, accuracy of Java simulation by Runge-Kutta method.

1 Introduction

The authors proposed a Java simulation for the rapid and accurate image learning of the charge motion in electric and magnetic fields [1] and for those of the electric characteristics of RCL circuits [2]. In these simulations, the text fields of the selected parameters, such as the electric field and magnetic field for the charge motion or the values of R, C, and L for the electric characteristics, are set on the display, and the calculation by the Runge-Kutta method is initiated by clicking the start button after inputting values into the text fields. Immediately following the completion of the calculation, the results are plotted as a figure on the display, e.g. a charge locus for the charge motion or the change of the current and voltage with time for the electric circuit. By changing the values in the text fields, new results can be represented immediately and a simulation under the new condition can be easily obtained. The value of the time increment, h, used in the numerical calculation by the Runge-Kutta method is limited to obtain an accurate simulation in spite of the useful simulation.

In this paper, the accuracy of the Java simulation for the charge motion in electric and magnetic fields has been investigated in comparison with the analytical solution, and the boundary value of the time increment to obtain an accurate simulation is shown empirically.

DOI: 10.14738/tmlai.33.1180
Publication Date: 04th June, 2015
URL: http://dx.doi.org/10.14738/tmlai.33.1180

2 Numerical Method and Analytical Solutions for the Charge Motion

2.1 Equations for the Charge Motion in Electric and Magnetic Fields

The charge motion in an applied electric field E = (Ex, Ey, Ez) and an applied magnetic field B = (Bx, By, Bz) is given as

$$m\frac{d\boldsymbol{v}}{dt} = -a\boldsymbol{v} + q\boldsymbol{E} + q\boldsymbol{v} \times \boldsymbol{B} - b(\boldsymbol{r} - \boldsymbol{r}_0), \tag{1}$$

$$\frac{d\boldsymbol{r}}{dt} = \boldsymbol{v}. \tag{2}$$

Here, t is time and m, q, $\boldsymbol{v} = (v_x, v_y, v_z)$, and $\boldsymbol{r} = (x, y, z)$ are the mass, charge, velocity, and displacement of the charge, respectively. a and b are coefficients of the resistance and restoring forces. $\boldsymbol{r}_0$ is the restoring centre. We consider only the electric and magnetic forces, such as the electron motion in a vacuum, to facilitate an easy comparison between the numerical simulation and the analytical solution. We use $\boldsymbol{E} = (E_x, 0, 0)$, $\boldsymbol{B} = (0, 0, B_z)$, $E_x = E_0\sin(2\pi ft)$, and $B_z = B_0$ to easily obtain the analytical solution. Here, E_0 and B_0 are constants, and f is the frequency of the electric field. In this case, we have four ordinary differential equations from Equations (1) and (2) for the charge motion in the $x - y$ plane.

$$m\frac{dv_x}{dt} = qE_0\sin(2\pi ft) + qv_yB_0, \tag{3}$$

$$m\frac{dv_y}{dt} = -qv_xB_0, \tag{4}$$

$$\frac{dx}{dt} = v_x, \tag{5}$$

$$\frac{dy}{dt} = v_y, \tag{6}$$

2.2 Numerical Method and Analytical Solutions

The numerical method for obtaining the solutions of Equations (3) – (6) by Java programming is described in Reference [1] using the fourth-order Runge-Kutta method.

We have the linear differential equation from Equations (3) and (4), as shown below.

$$\frac{d^2v_x}{dt} = 2\pi fcE_0\cos(2\pi ft) - c^2B_0^2v_x. \tag{7}$$

Here, $c = q/m$. The general solution of Eq. (7) is obtained as the sum of the general solution of the homogeneous equation, $d^2v_x/dt^2 + c^2B_0^2v_x = 0$, and the particular solution.

$$v_x = d_1\exp(jcB_0t) + d_2\exp(-jcB_0t) + \frac{2\pi fcE_0}{(cB_0)^2 - (2\pi f)^2}\cos(2\pi ft). \tag{8}$$

Here, d_1 and d_2 are unfixed constants and the last term is the particular solution. We use the initial conditions $v_x = v_0$ and $v_y = 0$ at $t = 0$, that is, the charge is injected into the x-direction of the fields at the initial velocity v_0. We obtain the other initial condition, $dv_x/dt = 0$ at $t = 0$, from Equation (3) using $v_y = 0$ at $t = 0$. We have the analytical solution, Equation (8), under these initial conditions,

$$v_x = \left[v_0 - \frac{2\pi f c E_0}{(cB_0)^2 - (2\pi f)^2} \right] \cos(cB_0 t) + \frac{2\pi f c E_0}{(cB_0)^2 - (2\pi f)^2} \cos(2\pi f t) \tag{9}$$

We obtain the analytical solution for v_y from Equation (3),

$$v_y = \left[\frac{2\pi f c E_0}{(cB_0)^2 - (2\pi f)^2} - v_0 \right] \sin(cB_0 t) - \frac{cB_0 cE_0}{(cB_0)^2 - (2\pi f)^2} \sin(2\pi f t \tag{10}$$

The analytical solutions for x and y under the initial conditions $(x,y) = (0,0)$ at $t = 0$ are obtained from Equations (5) and (6),

$$x = \frac{1}{cB_0} \left[v_0 - \frac{2\pi f c E_0}{(cB_0)^2 - (2\pi f)^2} \right] \sin(cB_0 t) + \frac{cE_0}{(cB_0)^2 - (2\pi f)^2} \sin(2\pi f t), \tag{11}$$

$$y = \frac{1}{cB_0} \left[v_0 - \frac{2\pi f c E_0}{(cB_0)^2 - (2\pi f)^2} \right] \cos(cB_0 t) + \frac{1}{2\pi f} \frac{cB_0 cE_0}{(cB_0)^2 - (2\pi f)^2} \cos(2\pi f t)$$

$$- \frac{1}{cB_0} \left[v_0 - \frac{2\pi f c E_0}{(cB_0)^2 - (2\pi f)^2} \right] - \frac{1}{2\pi f} \frac{cB_0 cE_0}{(cB_0)^2 - (2\pi f)^2}. \tag{12}$$

3 Comparison of the Numerical Simulations with the Analytical Solutions

Typical numerical simulations and analytical solutions for an electron motion accelerated by the electric frequency that is synchronized with the Larmor frequency using E_0 = 30 V/m, f = 28.55 MHz, and B_0 = 0.00102 T are shown in Figure 1. The simulations and solutions are shown in the left and right regions, respectively. The loci of the electron are shown in the upper regions and the changes of v_x and v_y with time are shown in the lower regions. The numerical calculation is performed using the time increment h = 0.5 ns by the double precision method. The final position of the calculation at t = 2000 ns, corresponding to 4000 total calculations, is (x,y) = (0.023912495, 0.01770018) in meters for the numerical simulation and (x,y) = (0.023911081, 0.017702656) for the analytical solution. The differences between the two results are $(\Delta x, \Delta y) = (-1.414 \times 10^{-6}, 2.476 \times 10^{-6})$, and their rates are 5.91×10^{-5} for x and 1.40×10^{-4} for y. The differences increase with the increase of the calculation time, such as (x,y)=(-0.6840279, -0.23154235) and (x,y) = (-0.6834052, -0.23318635) after 100,000 calculations in which $(\Delta x, \Delta y)$ = (0.0006227, -0.001644) and their rates are 9.11×10^{-4} for x and 7.05×10^{-3} for y. The global error of this method is $O(h^4)$ [3]. The position at t = 2000 ns by the simulation using h = 0.25 ns is (x,y) = (0.023911174, 0.017702503), and their differences from the analytical results are $(\Delta x, \Delta y) = (-9.3 \times 10^{-8}, 1.53 \times 10^{-7})$, which are approximately 1/16 of the values from the simulation obtained with h = 0.5 ns.

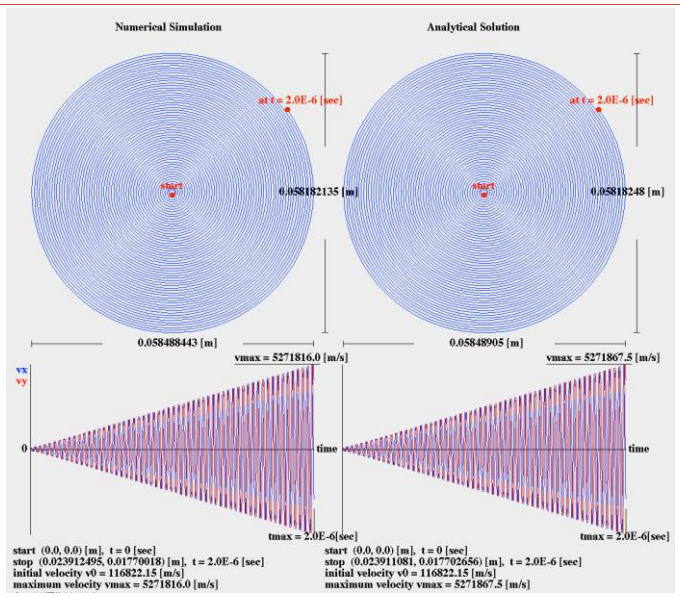


Figure 1: Comparison of a typical simulation with the analytical solution for an electron motion after 4000 calculations, that is, t = 2000 ns, using h = 0.5 ns at E0 = 30 V/m, f = 28.55 MHz, and B0 = 0.00102 T

The error of the Runge-Kutta simulation increases with the increase of h, and if we use a large h, the Java simulation results in a non-accurate image of the electron motion in comparison with that of the analytical solution, as shown in Figure 2. In this simulation, the range of the electron motion and the velocity at t = 2000 ns are approximately half that of the accurate analytical values.

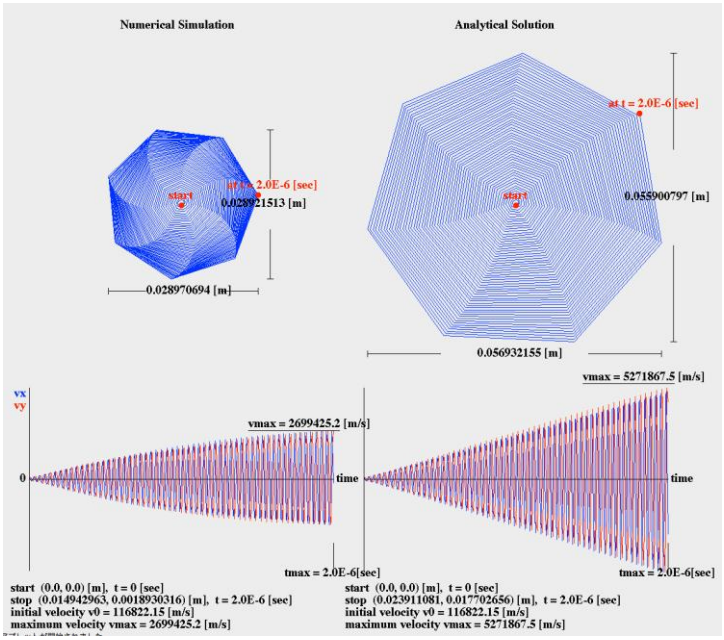


Figure 1: A non-accurate Java simulation for the electron motion using h = 5 ns. The values except h are the same as those used for the calculations in Figure1. The calculations are performed 400 times, corresponding to the same amount of time, 2000 ns, as in Figure 1.

The accuracy of the Runge-Kutta simulation depends on hqB_0/m from Equation (3), and the simulation at B_0 = 0.0102 T, which is ten times larger than that used in Figure 1, is not accurate, even with the use of

h = 0.5 ns, similar to the result from using h = 5 ns at B_0 = 0.00102 T, as shown in Figure 3. We use f = 285.5 MHz for the calculation in Figure 3 to synchronize with the Larmor frequency.

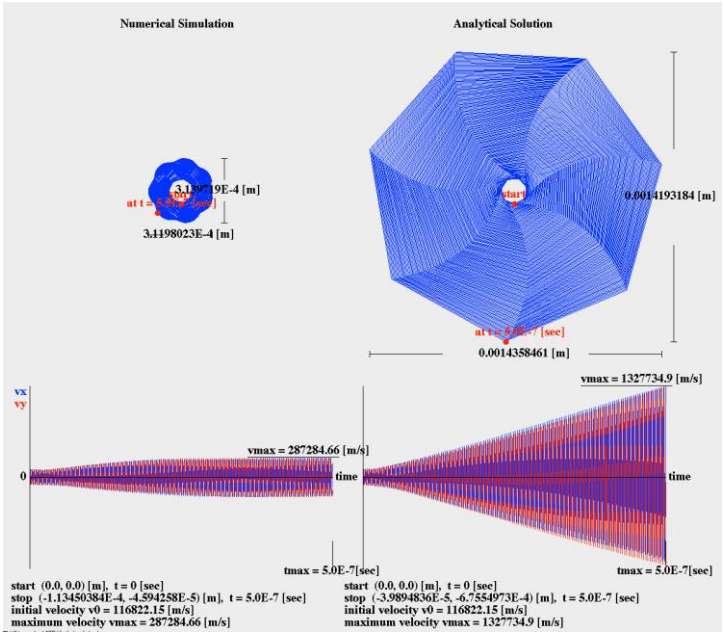


Figure 2: A non-accurate Java simulation for the electron motion even with the use of h = 0.5 ns due to the use of the ten times larger B0, 0.0102 T, relative to that in Figure 1. f = 285.5 MHz is used to synchronize with the Larmor frequency in this calculation, and the calculations are performed 1000 times until t = 500 ns.

In the case of an ion motion, the value of h used to obtain an accurate image of the motion is not that small, which is in contrast to the electron motion because the accuracy of the Runge-Kutta simulation is proportional to h/m from Equation (3). The Java simulations and analytical solutions for an He^+ motion accelerated by the synchronized electric frequency to the Larmor frequency are shown in Figure 4 using h = 50 ns, E_0 = 30 V/m, f = 391300 Hz, and B_0 = 0.102 T. An accurate image for the ion motion can be obtained by the Java simulation even with the use of h = 50 ns.

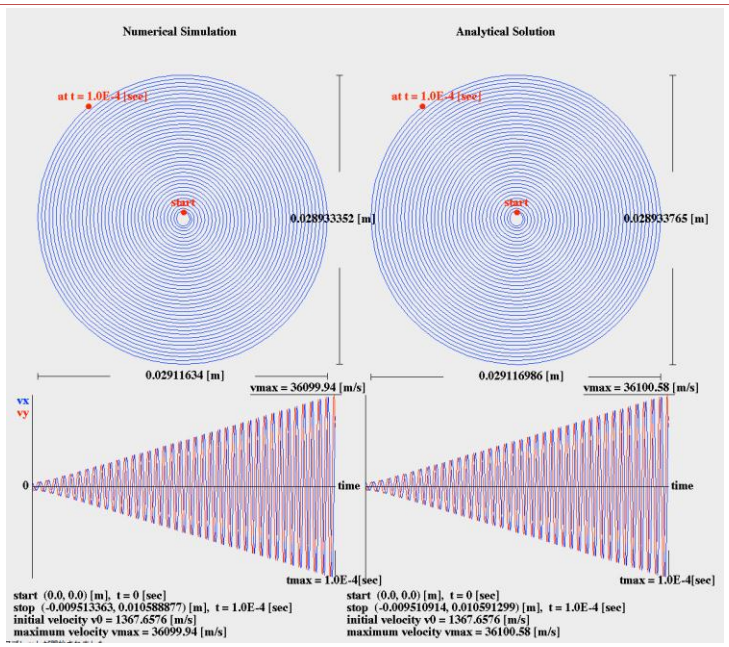


Figure 3: A typical He+ motion accelerated by the electric frequency synchronized with the Larmor frequency using h = 50 ns at E0 = 30 V/m, f = 391300 Hz, and B0 = 0.102 T. The calculations are performed 2000 times until t = 0.0001 sec.

4 Discussion

The accuracy of the numerical calculation used for the Java simulation depends on hqB_0/m from Equation (3). If we use an h that is too large, the calculation is not performed accurately and an appropriate image of the motion is not obtained, as shown in Figures 2 and 3. The accuracy of the simulation is better for evaluating the difference of the motion width, which is $\Delta x_{max} = x_{max} - x_{min}$ or $\Delta y_{max} = y_{max} - y_{min}$, compared to that of the analytical solution. Here, the subscripts $_{max}$ and $_{min}$ are used to represent the maximum and minimum values, respectively. The values of Δx_{max} at t = 500, 1000, 2000, 5000, and 10000 ns obtained by the simulations and the solutions for the electron motion using h = 0.5, 1.0, 2.0, 3.0, 4.0, and 5.0 ns at E_0 = 30 V/m, f = 28.55 MHz, and B_0 = 0.00102 T are shown in Table 1. The ratio of Δx_{max} obtained by the simulation to that obtained by the solution and the value of hqB_0/m corresponding to each h are also shown in the table. The ratio represents the accuracy of the Java simulation, that is, accuracy = 1 means that the Java simulation is equal to the analytical solution, and an accuracy > 1 and < 1 indicate the overestimation and underestimation of the simulations, respectively. The dependence of the accuracy for the electron motion on the value of hqB_0/m is shown in Figure 5.

Table 1: The motion width, xmax – xmin, obtained by the Java simulation and the analytical solution for the electron motion at t = 500, 1000, 2000, 5000, and 10000 ns using h = 0.5, 1.0, 2.0, 3.0, 4.0, and 5.0 ns at E0 = 30 V/m, f = 28.55 MHz, and B0 = 0.00102 T

Time (ns)	Methods	h = 0.5 ns hqB_0/m = 0.0897	h = 1.0 ns hqB_0/m = 0.1794	h = 2.0 ns hqB_0/m = 0.3588	h = 3.0 ns hqB_0/m = 0.5381	h = 4.0 ns hqB_0/m = 0.7175	h = 5.0 ns hqB_0/m = 0.8969
500	Java Simulation	0.014223397	0.014209879	0.014060617	0.01385032	0.012969641	0.011567365
	Analytical Solution	0.014223453	0.014211405	0.014097456	0.014027963	0.013868473	0.013841441
	Ratio (Accuracy)	0.99999607	0.9998926	0.9973869	0.9873365	0.9351888	0.8357053
1000	Java Simulation	0.029137922	0.029083265	0.028724693	0.028237505	0.025068847	0.01968627
	Analytical Solution	0.029138096	0.0290887	0.028879715	0.029069254	0.028370567	0.028370567
	Ratio (Accuracy)	0.99999404	0.99981314	0.9946321	0.9713873	0.8836217	0.69389766
2000	Java Simulation	0.058488443	0.058380943	0.05749023	0.054800544	0.045379903	0.028970694
	Analytical Solution	0.05848905	0.05839335	0.057914756	0.057436377	0.057914756	0.056932155
	Ratio (Accuracy)	0.9999896	0.9997875	0.9926698	0.95410866	0.78356373	0.50886345
5000	Java Simulation	0.14653207	0.14628382	0.14346443	0.12560898	0.079387136	0.03248504
	Analytical Solution	0.14653541	0.14635521	0.14584598	0.14584598	0.14476995	0.14280663
	Ratio (Accuracy)	0.99997723	0.9995122	0.9836708	0.861244	0.5483675	0.22747572
10000	Java Simulation	0.2936057	0.29312417	0.282346	0.21564397	0.086871445	0.03248504
	Analytical Solution	0.29361787	0.2933971	0.2933971	0.29165736	0.29239708	0.282726
	Ratio (Accuracy)	0.9999585	0.99906975	0.96233404	0.73937434	0.29710093	0.11489938

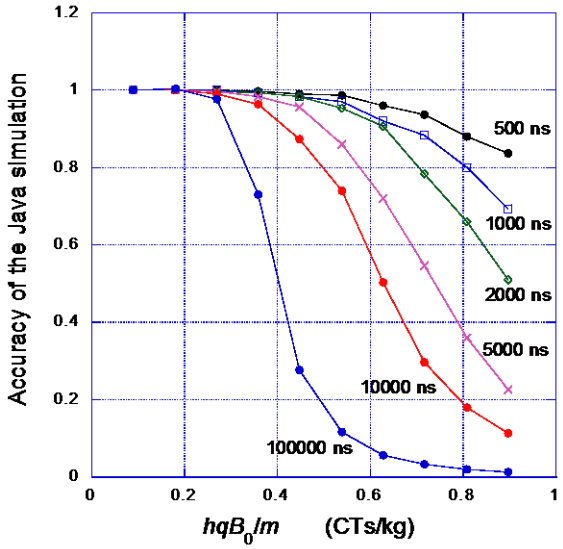


Figure 4: Dependence of the accuracy of the Java simulation on the hqB0/m used for the numerical calculation. Accuracy = 1 means that the Java simulation is equal to the analytical solution. Accuracy > 1 and < 1 indicate the overestimation and underestimation of the simulations, respectively.

The values of Δx_{max} at t = 20000, 50000, 100000, 200000, and 500000 ns obtained by the simulations and the solutions for the He^+ motion using h = 25, 50, 100, 200, 300, and 400 ns at E_0 = 30 V/m, f = 391300 Hz, and B_0 = 0.102 T are shown in Table 2. The ratio of Δx_{max} and the value of hqB_0/m corresponding to each h are also shown in the table. The dependence of the accuracy for the Java simulation of the He^+ motion on the value of hqB_0/m is shown in Figure 6.

Table 2: The motion width, xmax – xmin, obtained by the Java simulation and the analytical solution for the He+ motion at t = 20000, 50000, 100000, 200000, and 500000 ns using h =25, 50, 100, 200, 300, and 400 ns at E0 = 30 V/m, f = 391300 Hz, and B0 = 0.102 T

Time (ns)	Methods	h = 25 ns hqB_0/m = 0.0615	h = 50 ns hqB_0/m = 0.1229	h = 100 ns hqB_0/m = 0.2459	h = 200 ns hqB_0/m = 0.4917	h = 300 ns hqB_0/m = 0.7376	h = 400 ns hqB_0/m = 0.9834
20000	Java Simulation	0.0055589615	0.0055585667	0.0055583343	0.005472003	0.005226011	0.0047790827
	Analytical Solution	0.0055589606	0.0055585555	0.0055585555	0.005496019	0.005364648	0.0051418506
	Ratio (Accuracy)	1.0000001	1.000002	0.9999602	0.9956304	0.9741573	0.929448
50000	Java Simulation	0.014506064	0.014495008	0.014490361	0.014259509	0.013312547	0.010083964
	Analytical Solution	0.014506063	0.014495134	0.014495134	0.014377031	0.014377031	0.013768999
	Ratio (Accuracy)	1.0000001	0.99999124	0.99967074	0.99182564	0.9259594	0.7323673
100000	Java Simulation	0.029134799	0.02911634	0.028972711	0.02827198	0.024131613	0.014651205
	Analytical Solution	0.029134804	0.029116986	0.028993592	0.028993592	0.028993592	0.028993592
	Ratio (Accuracy)	0.9999998	0.9999778	0.99927986	0.97511137	0.83230853	0.5053256
200000	Java Simulation	0.05841807	0.05838068	0.058114525	0.055021226	0.04045283	0.01634748
	Analytical Solution	0.0584181	0.05838337	0.058229085	0.05778163	0.05778163	0.05778163
	Ratio (Accuracy)	0.99999946	0.99995387	0.99803257	0.952227	0.70009845	0.28291827
500000	Java Simulation	0.14663371	0.1465667	0.14601156	0.12891208	0.056520145	0.01634748
	Analytical Solution	0.14663388	0.14658329	0.14658329	0.14594747	0.14540245	0.1458892
	Ratio (Accuracy)	0.99999887	0.9998869	0.99609965	0.88327724	0.38871524	0.11205408

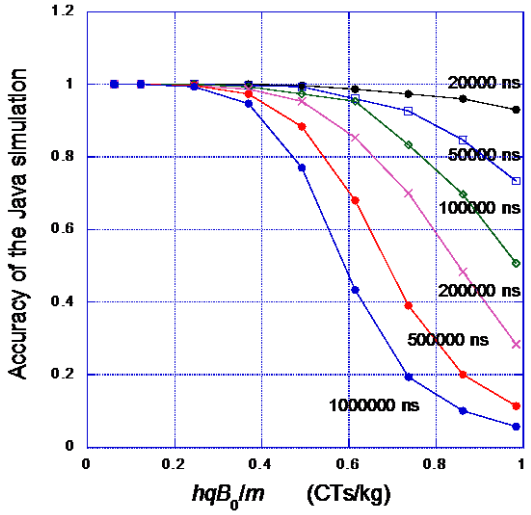


Figure 5: Dependence of the accuracy of the Java simulation for He+ motion on hqB0/m. Accuracy = 1 means that the Java simulation is equal to the analytical solution. Accuracy >1 and <1 indicate the over estimation and under estimation of the simulations, respectively.

5 Conclusion

The accuracy of the Java simulation by the Runge-Kutta method for the charge motion in electric and magnetic fields has been investigated in comparison with the analytical solution. The results are summarized as follows:

1. The accuracy depends on the value of hqB_0/m used for the numerical calculation, which also depends on the numerical calculation time.
2. In the case of non-accurate simulation, the simulation almost results in an underestimation, that is, a motion that is smaller than the real motion.
3. An accurate image for the charge motion in electric and magnetic fields is able to be obtained by the Java simulation using a less value of hqB_0/m than 0.2 sCT/kg, that is, using $h < 0.2m/qB_0$, as shown in Figures 5 and 6.

REFERENCES

[1]. M. Morooka, S. Qian, and M. Morooka, *Image Learnig of Charge Motion in Electric and Magnetic Fields by Java Programming*, Transactions on Machine Learnig and Artificial Intelligence, 2014. **2**(2): p.1- 19.

[2]. M. Morooka, S. Qian, and M. Morooka, *Image Learnig of Electric characteristics of Resistance, Capacitance, Inductance, and their Circuits by Java Programming*, Transactions on Machine Learnig and Artificial Intelligence, 2014. **2**(3): p.1- 19.

[3]. A. Ralston and C. L. Meek, *ENCYCROPEDIA OF COMPUTER SCIENCE*, Van Nostrand Rainhold Company (1976), p. 984.

SOCIETY FOR SCIENCE AND EDUCATION UNITED KINGDOM
TMLAI TRANSACTIONS ON MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE
VOLUME 3, ISSUE 3
ISSN 2054-7390

Difficulty-Level Classification for English Writings

[1]Hiromi Ban, [2]Rei Oguri and [3]Haruhiko Kimura
[1]Graduate School of Engineering, Nagaoka University of Technology, Niigata, Japan;
[2,3]Graduate School of Natural Science and Technology, Kanazawa University, Ishikawa, Japan;
je9xvp@yahoo.co.jp; oguri@blitz.ec.t.kanazawa-u.ac.jp; kimura@blitz.ec.t.kanazawa-u.ac.jp

ABSTRACT

The popularity of e-books has grown recently. As the number of e-books continues to increase, the task of categorizing all books manually requires a significant amount of time. If English sentences can be categorized according to their level of difficulty, it becomes possible to recommend a foreign-language book compatible with the reader's level of competency in English. This study extracted eleven types of attribute from English text data, with the aim of classifying English text according to level of difficulty by learning and categorization. Using the method of "leave-one-out cross-validation," text was subjected to machine learning and categorization. In order to improve accuracy, furthermore, an experiment was carried out in which the size of text data was varied, and the attribute selection method was implemented. As a result, accuracy was improved to 77.04%, and F-measure to 63.96%.

Keywords: Accuracy; Difficulty-level; F-measure; Machine learning.

1 Introduction

The popularity of e-books has grown recently, with the number of books and magazines distributed within Japan in 2014 growing by 18.3% compared with the previous year to 720,000, as shown in Figure 1. Furthermore, it is predicted that in 2016, this number will reach 1.2 million [1].

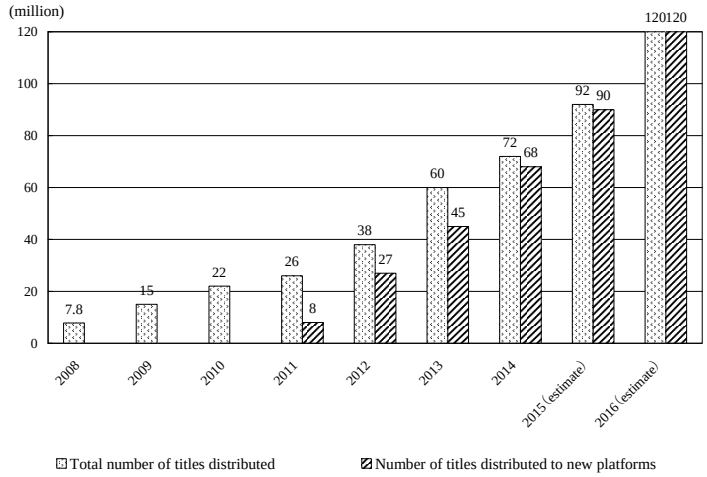


Figure 1. Number of titles of digital books and magazines distributed in Japan.

DOI: 10.14738/tmlai.33.1245
Publication Date: 16th June, 2015
URL: http://dx.doi.org/10.14738/tmlai.33.1245

The number of books listed in the Kindle store as of 28th January 2015 is shown in Table 1, broken down by genre [2]. Compared with the 23 genres of domestically published e-book, all non-Japanese books (of which 3 million are available) are categorized in a single genre.

Table 1. Number of books per genre at Kindle store on Jan. 28, 2015.

Genre	Number	Genre	Number	Genre	Number
Literature & commentary	60,912	Medicine & pharmacology	2,094	Language study, dictionary, cyclopedia & yearbook	1,849
Humanities & thought	17,663	Computer & IT	3,959	Education, study-aid book & examination	3,239
Society & politics	9,118	Art, construction & design	3,209	Picture book & children's book	3,228
Nonfiction	2,611	Hobby & practical use	9,441	Comic	99,187
History & geography	7,854	Sports & outdoor amusement	2,237	Light novel & BL	24,629
Business & economic	11,329	Qualification & authorization	640	Entertainment	2,317
Investment, finance & company management	3,593	Living, health & child-rearing	9,654	Adult	16,912
Science & technology	8,757	Travel guide & map	2,890	Kindle foreign book	3,071,739

As the number of e-books continues to increase, the task of categorizing all books manually requires a significant amount of time; this time requirement becomes even greater if the genre of the book is not clear from its title or the name of its author. In addition to categorization by genre, books may also be categorized according to their level of difficulty. Readers who are studying English may wish to read a simple foreign-language book, while those wishing to extend their language abilities may wish to read a slightly more difficult book. In such cases, analysis is simple, because e-books are a form of electronic data. If English sentences can be categorized according to their level of difficulty, it becomes possible to recommend a foreign-language book compatible with the reader's level of competency in English. For this reason, this research aims to identify the difficulty level of English text.

2 Related Research

In a prior report, the authors implemented quantitative linguistic analysis on English language textbooks used in Finland, which is considered to have the highest level of reading comprehension, mathematical and scientific literacy according to the Organization for Economic Cooperation and Development (OECD)'s Program for International Student Assessment (PISA), and English language textbooks used in Japan, and compared their difficulty level based on the words occurring therein [3]. We also extracted attributes such as the average word length and number of words per sentence.

In this study, the text data and attributes from our previous report were used with the aim of identifying level of difficulty within English sentences.

3 Method

3.1 Data Used

In this paper, the text data used was the same as that used in other related studies, in other words, the textbook used in in third and fourth grade elementary school English lessons in Finland [3][4].

- *Wow! 3* (2002, WSOY)
- *Wow! 4* (2003, WSOY)
- *Wow! 5* (2005, WSOY)
- *Wow! 6* (2006, WSOY)

3.2 Proposed Method

Attributes are extracted from the text data to create data sets. The data sets thus created are subjected to machine learning and categorized.

3.2.1 Attribute Extraction/Data Set Creation

The attributes used for data set creation in this study are the eleven types shown in Table 2.

Table 2. Attributes to be educed.

Total number of characters	Mean word length
Total number of character-type	Words/sentence
Total number of words	Sentences/paragraph
Total number of word-type	Words/word-type
Total number of sentences	Commas/sentence
Total number of paragraphs	

There are a total of 12 objective variables, consisting of grades three through six divided into the three categories of preliminary, intermediate and final phases. This takes into account the fact that even within the same school year, the sentences in the first pages of the textbook have a different difficulty level to those in the final pages.

The eleven attributes were extracted from each text file, and defined as one instance. Table 3 depicts the data sets where as an example, the quantity of text per instance was defined as one page of the textbook.

Table 3. Data set in the case of 1 page per instance.

Total num. of characters	Total num. of character-type	Total num. of words	・・・	Sentences/ paragraph	Words/ word-type	Commas/ sentence	Class
207	36	40	・・・	1.25	1.429	0.10	a
252	40	44	・・・	1.00	1.257	1.17	a
213	37	38	・・・	1.60	1.226	0.75	a
252	37	52	・・・	2.00	1.529	0.60	a
261	36	60	・・・	2.60	1.429	0.08	a
・	・	・	・・・	・	・	・	・
・	・	・	・・・	・	・	・	・
・	・	・	・・・	・	・	・	・
1040	50	181	・・・	2.57	1.361	0.44	l
1315	58	241	・・・	2.33	1.461	0.54	l
1526	52	288	・・・	2.25	1.834	0.44	l
2099	58	396	・・・	2.04	2.052	0.38	l
2132	54	416	・・・	1.96	2.286	0.44	l

3.2.2 Machine Learning

The data sets were subjected to machine learning and categorization. Leave-one-out cross-validation was used in learning. Leave-one-out cross-validation is a learning method involving taking one piece of data from the whole as test data, and defining the rest as learning data, and repeatedly validating so that each piece of data becomes the test data once.

The classifier used was a Random Committee.

The classifier used the open source data mining tool Weka in learning and identification [5].

4 Experimentation

In this study, two experiments were carried out using the following evaluation methods during machine learning.

4.1 Evaluation Methods

The evaluation procedure used in this study is as shown in Figure 2.

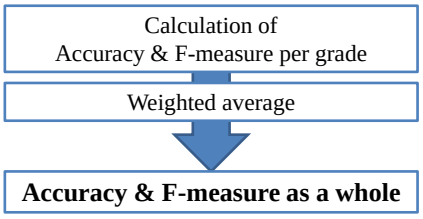


Figure 2. Evaluation procedure.

For example, among data predicted by the classifier to be in the fourth-grade textbook, data that actually was in the fourth-grade textbook was defined as a TruePositive, while that not in the fourth-grade textbook was a FalsePositive. Among data predicted by the classifier to not be in the fourth-grade textbook, data that was in fact in the fourth-grade textbook was defined as a FalseNegative, while that not actually in the fourth-grade textbook was defined as a TrueNegative. The threat scores of these categories are compiled in a categorization table such as that in Table 4.

Table 4. Contingency table.

	Correct answer +	Correct answer -
Estimate +	TruePositive	FalsePositive
Estimate -	FalseNegative	TrueNegative

All data was categorized, as in Figure 3, using the 12 objective variables.

			Correct answer											
			3rd grade			4th grade			5th grade			6th grade		
			a	b	c	d	e	f	g	h	i	j	k	l
Estimate	3rd grade	a	8	2	3	2	1	2	0	1	0	0	0	0
		b	3	2	2	3	4	2	0	0	0	0	2	1
		c	1	3	1	4	2	3	2	0	2	1	0	0
	4th grade	d	2	5	3	4	7	3	0	1	0	1	0	0
		e	1	2	3	3	2	1	1	2	0	0	0	1
		f	0	1	1	3	3	6	2	0	1	4	1	0
	5th grade	g	0	1	2	0	0	2	4	1	1	2	2	4
		h	0	0	0	1	1	0	3	2	3	5	7	3
		i	0	1	1	1	0	0	1	5	6	3	2	2
	6th grade	j	2	0	0	1	0	1	4	4	1	4	3	6
		k	0	0	0	0	0	0	3	5	5	4	4	8
		l	0	0	0	0	1	1	3	2	3	6	9	4

	Correct answer +	Correct answer -
Estimate +	TruePositive	FalsePositive
Estimate -	FalseNegative	TrueNegative

Figure 3. Evaluation Method

In addition to the categorization of each academic year into preliminary, intermediate and final phases, the final phase of the previous academic year and preliminary phase of the year above were also counted as correct, giving a total of five correct categorizations for data. In other words, as shown in the example of Figure 3, in the case of the fourth grade textbook, data categorized into either the preliminary, intermediate or final phase of the fourth grade, the final phase of the third grade or the preliminary phase of the fifth grade was considered a correct answer.

The categorization results obtained using the evaluation method shown in Figure 3 were summarized by academic year, as shown in Table 5.

Table 5. Threat score for each grade.

3rd grade	Correct answer +	Correct answer -
Estimate +	35	48
Estimate -	15	173
4th grade	Correct answer +	Correct answer -
Estimate +	43	50
Estimate -	21	148
5th grade	Correct answer +	Correct answer -
Estimate +	38	76
Estimate -	30	127
6th grade	Correct answer +	Correct answer -
Estimate +	55	51
Estimate -	34	131

Next, the rate of accuracy, that is, Accuracy and F-measure were calculated for each academic year.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \quad (1)$$

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

$$F - measure = \frac{2 * precision * recall}{precision + recall} \quad (4)$$

Finally, the weighted average was obtained from the calculated accuracy and number of data sets, to calculate overall accuracy and F-measure. This was defined as the evaluation value in this case.

4.2 Experiment 1

4.2.1 Details of Experiment

An experiment was carried out to establish the relationship between changes in the volume of text data used to extract attributes, accuracy and F-measure.

Three types of data set – taking one page, two pages and three pages of text as a single instance of text – were subjected to machine learning and categorization under the conditions shown in Table 6.

Table 6. Experiment environment

Number of characteristics	11
Classifier	Randomcommitte
Technique	leave-one-out cross-validation

The method used to create data sets with two pages of text per instance is as shown in Figure 4.

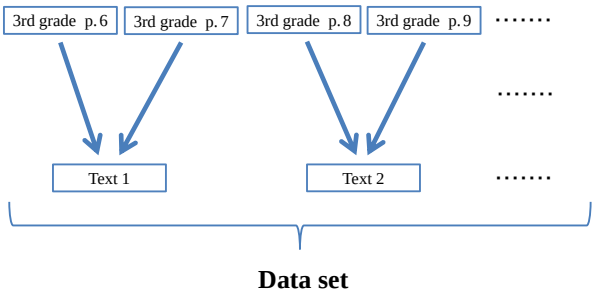


Figure 4. Method of making a data set in the case of 2 pages per instance.

Similarly, with three pages of text per instance, the text data was created in order, so as not to overlap, three pages at a time. The number of instances was 271, 136 and 92, respectively, depending on whether the quantity of text was one, two or three pages.

4.2.2 Results

Results of Experiment 1 are shown in Table 7.

Table 7. Accuracy and F-measure in Experiment 1.

	Accuracy	F-measure
1 page	68.62%	50.95%
2 pages	70.36%	53.48%
3 pages	74.24%	58.87%

From Table 7 we see that the greater the number of pages, the higher the accuracy and F-measure achieved. Given this, it is considered that using larger quantities of text data for extracting attributes is effective in categorization.

Hereafter, three pages of the textbook will be used per instance when creating data sets for this study.

4.3 Experiment 2

4.3.1 Details of Experiment

The attribute selection method was implemented using the attribute selection function of Weka. The attribute selection method involves searching for items with a low contribution in regard to the objective variable, or attributes that are difficult to predict. These are output as in Figure 5, using attribute selection. The smaller the numerical value, the lower the contribution. A threshold is defined, and attributes below the threshold are deleted, after which attributes are selected once again. Each time attribute selection is implemented, accuracy and F-measure are recorded. This is repeated until all attributes are above the threshold value.

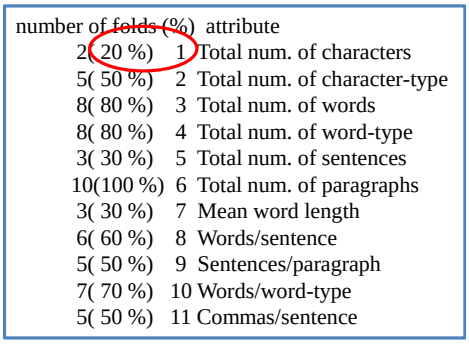


Figure 5. Output of feature selection.

4.3.2 Results

After three repeats at threshold value 40%, accuracy and F-measure both demonstrated maximum values. These results are shown in Figure 6.

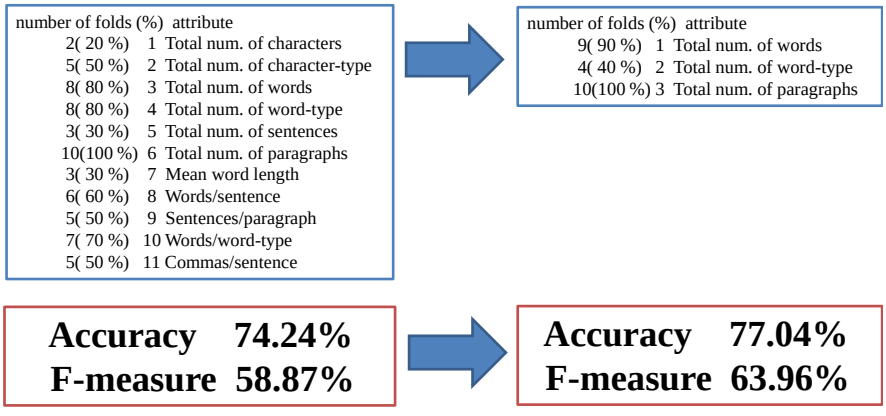


Figure 6. Result of Experiment 2.

As a result, the attribute selection method was implemented, and when the number of attributes was reduced to the following three: “total number of words,” “total number of word types” and “total number of paragraphs,” accuracy increased to 77.04% and the F-measure to 63.9%.

5 Considerations

Accuracy and F-measure were both highest when three pages of text were used per instance. From this, it is believed that the attributes extracted from three pages of text are effective in categorization.

Next, the use of the attribute selection method allowed a reduction in the number of attributes from 11 to 3, and increased accuracy to 77.04% and the F-measure to 63.9%. The remaining three attributes, in other words “total number of words,” “total number of word types” and “total number of paragraphs,” are believed to be those that have the most impact on the difficulty level of English text.

Using these two experiments and reducing the number of attributes improved accuracy, but as shown in Table 8, some data was categorized in significantly erroneous categories.

Table 8. Estimate and correct answer in Experiment 2.

			Correct answer											
			3rd grade			4th grade			5th grade			6th grade		
			a	b	c	d	e	f	g	h	i	j	k	l
Estimate	3rd grade	a	2	0	0	1	2	0	0	0	0	0	0	0
		b	2	3	0	2	0	1	0	0	0	0	0	0
		c	1	0	1	0	1	1	0	0	0	0	0	0
	4th grade	d	1	3	2	4	1	1	0	0	0	0	0	0
		e	0	0	0	1	1	1	1	0	0	0	0	0
		f	0	0	1	0	1	2	2	0	1	0	0	1
	5th grade	g	0	0	0	0	1	0	5	1	1	0	0	0
		h	0	0	0	0	0	0	0	3	1	2	1	2
		i	0	0	0	0	0	1	0	2	0	1	1	2
	6th grade	j	0	0	1	0	0	0	0	1	2	0	2	2
		k	0	0	0	0	0	0	0	0	0	2	4	1
		l	0	0	0	0	0	0	0	1	2	5	2	2

When the pages that were significantly mis-categorized were examined, it was found that they all contained columns. In other words, it is believed that the mistaken identification was caused by the impact of the columns between sentences. As a result, it is considered that removing columns from the scope of investigation is likely to improve accuracy.

6 Conclusions

This study extracted eleven types of attribute from English text data, with the aim of classifying English text according to level of difficulty by learning and categorization. Using the method of “leave-one-out cross-validation,” text was subjected to machine learning and categorization. In order to improve accuracy, furthermore, an experiment was carried out in which the size of text data was varied, and the attribute selection method was implemented. As a result, accuracy was improved to 77.04%, and F-measure to 63.96%. At the same time, we noted erroneous identification resulting from the impact of columns between sentences.

In the future, when identifying the difficulty level in English text, we intend to consider new attributes that allow more accurate categorization, and more effective combinations of attribute quantity.

REFERENCES

(1) ITmedia eBook USER | What is the total number of titles of e-books and e-magazines distributed within Japan? http://ebook.itmedia.co.jp/ebook/articles/1412/19/news033.html

(2) Kindle Store, http://www.amazon.co.jp/Kindle-%E3%82%AD%E3%83%B3%E3%83%89%E3%83%AB-%E9%9B%BB%E5%AD%90%E6%9B%B8%E7%B1%8D/b?node=2250738051

(3) Hiromi Ban and Takashi Oyabu, Text Mining of English Textbooks in Finland, "Proceedings of the Asia Pacific Industrial Engineering & Management Systems Conference 2012", V. Kachitvichyanukul, H.T. Luong and R. Pitakaso eds., pp.1674-1679.

(4) Wow! 3 (2002, WSOY) Wow! 4 (2003, WSOY) Wow! 5 (2005, WSOY) Wow! 6 (2006, WSOY), http://www.kknews.co.jp/developer/finland/

(5) Weka: Data Mining Software in Java, http://www.cs.waikato.ac.nz/ml/weka/

SOCIETY FOR SCIENCE AND EDUCATION
UNITED KINGDOM
TMLAI TRANSACTIONS ON MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE
VOLUME 3, ISSUE 3
ISSN 2054-7390

Bilingual Information Hiding System: A Formalized Approach

[1]Riad Jabri and [2]Boran Ibrahim
Faculty of Information Technology, Computer Science Department,
University of Jordan, Amman, Jordan
jabri@ju.eu.jo; boranabed@yahoo.com

ABSTRACT

Steganography and cryptography are used to maintain privacy and security over communication channels. Due to their complexity and diversity, there is a need for their continuous improvements. In this paper, we consider such improvements and propose a new bilingual information hiding system. The proposed system is based on a formal approach that incorporates text-based steganography and cryptography in a way that permits multilayered levels of security and privacy and improves the quality of information hiding in terms of perceptual transparency, robustness and hiding capacity. Considering an Arabic text as a cover one, the inter-word spaces and word features have been used as places for information hiding. As a result, 3.676 have been achieved as an improved average of capacity ratio.

Keywords: Information hiding; Steganography; Cryptography; Encoding; Decoding.

1 Introduction

Steganography is an information hiding technique [1, 2, 3]. For the purpose of our research, we consider steganography as a method of hiding the existence of a bilingual secret message in Arabic text [4]. Text-steganography proceeds according to the following scheme [5]:

- A secret message is concealed in a cover- text using an embedding algorithm to produce a stego- text.
- The stego – text is then transmitted over a communication channel (Internet).
- Upon its delivery, the secret message is recovered using an extracting algorithm.
- The embedding and the extracting algorithms are augmented by the so called a stego- key to encrypt and decrypt the hidden data respectively.

Based on the presented scheme, the secret message is concealed using one of the following methods [5, 6]:

- Modification of the cover-text, such as insertion of spaces, misspelling , modifying the features (name, shape, position, color, size) of the individual characters
- Substitution, such as replacement of insignificant data within the cover text by hidden ones.
- Generation, such as creation of a fake cover.

The most recent efforts, techniques and tools related to our work can be classified as follows:

DOI: 10.14738/tmlai.33.1261
Publication Date: 17th June, 2015
URL: http://dx.doi.org/10.14738/tmlai.33.1261

- Approaches based on inter-word and inter-paragraph spacing to generate dynamic stego-text, as suggested by Por et. al. [7] and Jabri et. al. [5].
- Techniques based on combining the following methods: Open space; syntactic (punctuation) and semantic encoding (synonym words), as suggested by Bender et al. [2].
- Tools based on open space concealing method combined with compression and encryption, such as SNOW as suggested by Kwan [8].
- Technique based on natural language processing and using either the sentence structures or linguistic coding scheme, as suggested by Bergmair [9].
- Techniques specialized for Arabic text, classified into four categories [10]:

1. Dot steganography defined as a vertical displacement of dots in Arabic characters [11].
2. La steganography uses special forms of '' La" word for hiding information [12]
3. Techniques that use letters with extension (kashida) and un-extended ones to hold the secret bits ''one" and ''zero'' respectively [13, 14].
4. Techniques that use a diacritic Arabic text for information hiding [15].

The above-mentioned efforts aim at improving one or more of the following quality indictors: perceptual transparency, robustness and hiding capacity [15]. For example, the hiding algorithms suggested in [13, 14] aim at improving hiding capacity.

Our proposed approach constitutes further improvements of such indictors. In addition, it is characterized by the following:

- The system hides bilingual (Arabic, English) secret messages. It combines inter-word spaces and letter extension (kashida) to hide secret bits. Such combination contributes to perceptual transparency, robustness and increases the hiding capacity.
- The system is based on a formal approach that combines steganography and cryptography as function compositions and introduces multilayered security levels.

The remainder of this paper is organized as follows. Section 2 presents formalization and implementation of the proposed system. Section3 presents analysis and results. A conclusion is given in section 4.

2 The Proposed Method

Based on our set objectives, we define a steganography system in terms of two main stages. The first one is an encoding stage to embed a secret message in a cover text and produce a stego-text. The second one is a decoding stage to extract the secret message from the stego text. The encoding stage transforms the secret message into a sequence of encrypted bits. Such transformation improves the efficiency, the privacy and the security of the hiding process. Then, it considers the cover text as composing of sequence of characters and the uses these characters as positions to hide the secret bits. Such a use is based on the fact that most of Arabic letters can be extended and such extension is considered as one of its writing styles. Thus, the hiding positions are defined as spaces and letters suitable for extension. The spaces are extended by additional space and the letters are extended by kashida (-) Hence, the Arabic characters are classified into two sets. The first one includes the Arabic characters that are suitable for extension by kashida. For example, the letters ""ص and ""س are

members of such a set. They can be extended by kashida as صـ" " and سـ" " respectively. The second set includes the special characters (?, : , ", %), the digits (0,..,9) and the non-extendable Arabic characters . For example, the letters"ز" , "و"and" ر" .

Based on such classification, suitability criteria are defined and used to determine the hiding positions in the cover text. The hiding process then proceeds as follows. The secret bits and the cover text are scanned bit by bit and character by character respectively. Upon capturing a hiding position, the respective character (blank or extendable letter) is extended, if the secret bit is "1". Otherwise, no extension is performed. On the other hand, the decoding stage includes an extracting process to retrieve the hidden bits from their respective positions in the stego-text.

Based on the above-mentioned definition, the encoding and the decoding stages are formalized as a composition of respective functions. The implementation of the proposed system is then reduced to the implementation of an interaction context and the functions from which these stages are composed. They have been implemented according to the algorithms given below using C#.NET as a programming tool. As a result, the proposed system has been constructed with following functionality:

- The sender interacts with the proposed system through an interaction context to facilitate: User authentication; browsing the secret message (SM) and the cover text (CT) from their respective text files; testing the suitability of CT to hide SM and initiating the encoding process.
- In addition to authentication, the receiver interaction context facilitates: browsing the stego-text from its respective text file; and initiating the decoding process.
- The system responds to the sender-initiated request by activating the functions respective to the encoding stage.
- The system responds to the receiver-initiated request by activating the functions respective to the encoding stage.

2.1 The encoding stage

The encoding stage consists of two major steps. First, it transforms the secret message into a sequence of encrypted bits. Second, it considers the cover text as composing of sequence of characters and the uses these characters as positions to hide the secret bits. These steps are formalized and implemented as follows.

Let SM = SM_1 SM_2 … SM_n be the sequence of characters representing the secret message.

The transformation step is defined as a composition of the functions:

$$\text{Bit (Encrypt (Compress (Byte (SM[ ])) } \rightarrow \text{ BSM[ ], where}$$

- Byte (SM[ ]) $\rightarrow$ SM[ ] is a function that converts the secret message (SM) to a stream of bytes, using Unicode encoding.
- Compress (SM[ ]) $\rightarrow$ SM[ ] is a compression function, for example, eliminating extra zero-bytes in the Unicode for Arabic.
- Encrypt (SM[ ]) $\rightarrow$ ESM[ ] is an encryption function. It uses two passwords of type string to generate a fixed random stream of keys (Key [ ]). The keys are then XORed with SM [ ] to obtain an encrypted secret message.
- Bit(ESM[ ]) BSM[ ] is function that converts the encrypted SM into a stream of bits represented as BSM[ ] = BSM_1 BSM_2 … BSM_n.

Let EC = { ت ,…, ب} denotes the set of extendable Arabic characters.

Let NEC = {9…,0,1 ,…. , ر ,ز } denotes the set of non-extendable Arabic characters.

Let PEC = {(ACi, ACj) denotes the set of pair-wise Arabic characters (AC_i, AC_j) such that AC_i EC, AC_j (EC or NEC) and their occurrence as sequence in a text enables the extension of AC_i.

A hiding position is defined as an Arabic character (AC) subject to the following suitability criteria

- Suitability (AC) = True, if (AC (NEC PEC) or AC= " ")
- Suitability (AC) = False, otherwise.

Let CT = CT_1 CT_2 … CT_n is the sequence of characters representing the cover text.

Let HidingPosition (CT[ ]) → HCT[ ] be a function to determine the hiding positions in the cover text.

The embedding process is then defined by the function Embed (BSM, HCT) → ST. However, such embedding is subject to satisfied quality indicators computed by the following functions:

- HidingRatio to express conceivability as (Length (BSM)/Length (hiding positions))%
- CapacityRatio to express conceivability as (Bytes(SM)/Bytes(CT))%

Thus, the encoding stage proceeds according to following steps:

Step1: SM = Input (Secret-message);

Step2: BSM[ ] = Bit (Encrypt (Compress (Byte (SM)))

Step3: CT = Select-cover-text

Step4: HCT[ ] = HidingPosition (CT[ ])

Step4: Compute quality indicators

Step3: If (Satisfied (quality indicators){ Embed (BSM, HCT)}

Else if { repeat from step3}

The implementation of the encoding stage is reduced to the implementation of its functions according to respective algorithms. Representative ones are given below.

Algorithm 1: Encryption and Decryption

Input: compressed stream of bytes respective to SM

Output: encrypted or decrypted stream of bytes respective to SM

Method:

```
encryptedMessage [ ] = encrypt (newMessage, password1) ;
decryptedMessage [ ] = decrypt (encryptedMessage, password2) ;
 encrypt (message [ ] , password1)
 {
   return EncryptDecrypt (message, password1) ;
 }
```

```
decrypt (message [ ] , password2)
{
  return EncryptDecrypt (message, password1) ;
}
  EncryptDecrypt (message [ ] , password)
  {
    randomNumbers [ ] = { 08, 06, 02 };
    DerivePasswordBytes = dpb (password, randomNumbers) ;
    key [ ] =dpb.GetBytes(128) ;  // Return 128 random numbers
    returnMessage [ message. length ] ;
    for i = 1 To message. length        step 1
     {
       index = i mod key. length ;
       returnMessage [ i ] = key [index]  XOR  message [ i ] ;
     }
    return returnMessage ;
  }
```

Algorithm 1 performs encryption and decryption. It uses two passwords of type string to generate a fixed random stream of keys The keys are then XORed with SM (or CT)to obtain an encrypted (or decrypted) secret message. By using passwords and randomly generated keys, Algorithm 1 introduces two levels of security.

Algorithm 2: Hiding positions

Input: Covertext (CT)

Output: Hiding positions in CT.

Method:

```
Hiding positions [ ] = Suitable (CT[ ] )
Suitable (CT[ ] )
{  for i =1  To  CT. length     step 1
  {
    if ( Suitability (CT [i] = True )  Then
     { HidingPositions  [i ] = " 1"
     Else
      {  HidingPositions  [i ] = " 0"
    }
```

Algorithm 3: Embedding

Input: The stream of bits BSM respective to the secret message SM.

The over text (CT)

The hiding positions (HidingPositions) in CT

Output: stego_text (secret bits inside cover text).

Method:

```
Embed  (BSM , CT, HidingPositions )
{
 cover = CT ;
 result = "  " ;
 coverIndex = 1 ;
 for i =1  To  message. length     step 1
  {
    while ( not HidingPositions [coverIndex ] = "1 ")
     Do
     { result = result + cover [ coverIndex ] ;
       coverIndex = coverIndex + 1
     }
     if CT [coverIndex ] = "  " )   Then
        {
          result = result + cover [ coverIndex ] + "  " ;
          coverIndex = coverIndex + 1 ;
        }
       else
        {
          result = result + cover [ coverIndex ] + " - " ;
          coverIndex = coverIndex + 1 ;
  }
  Insert an end marker for BSM at cover [ coverIndex ]
  Complete the stego text by remaining cover text:
   for i = coverIndex + 1  To  cover. length   step 1
     result = result + cover [ i ] ;
   return result ;
}.
```

2.2 The Decoding Stage

The decoding stage extracts the secret message from the stego-text at receiver's side according to the following steps:

Step 1: ST = Browse (stego-text)

Step 2: If (authentication = True) {Decompress (Decrypt (Byte (Bit-extract (ST)))}, where

- Bit-extract (ST) BSM[ ] is function that extracts the encrypted SM as the stream of bits BSM[ ] = BSM_1 BSM_2 … BSM_n
- Byte (BSM[ ]) SM[ ] is a function that converts extracted message (BSM) to a stream of bytes, represented as SM[ ]
- Decrypt (SM[ ]) SM[ ] XOR Key [ ] is an decryption function. As in Encrypt, the function Decrypt uses two passwords of type string to generate a fixed random stream of keys. However, the keys are XORed with SM [ ] to obtain a decrypted secret message.
- Decompress (SM[ ]) SM[ ] is a decompression function to recover SM as represented by Unicode for Arabic.

The implementation of the decoding stage is reduced to an interaction context for authentication, browsing the stego text and then displaying the secret message. This is achieved by applying the functions respective to the decoding stage. These functions have been implemented according to respective algorithms. Representative ones are given below.

Algorithm 4: Bit-Extract

Input: stego_text

Output: stream of bits

Method:

```
lastPosition = message.LastIndexOf ("   ") ;    // 3 spaces
if ( lastPosition < 0 )  Then
  {
     lastPosition = getLastPosition ( message ) ;
  }
temp = message ;
bits = extractBits ( lastPosition , temp ) ;
getLastPosition ( message )
  {
    for i = 1  to  message. length    step 1
        if ( message[ i ] = ' - ' AND  message [ i +1] = ' - ' )  Then
            return i +1 ;
    return i ;
  }
```

```
extractBits ( lastPosition , temp )
{
bits = "  " ;
for i = 0   To   lastPosition    step 1
   if ( temp [ i ] = ' ' )    Then
    {
       if ( temp [ i +1] = ' ' )    Then
         {
            bits = bits + " 1 " ;
            i = i + 1 ;
         }
       else
            bits = bits + " 0 " ;
    }
   else if ( temp [ i ] = ' - ' )   Then
    {
       bits = bits + " 1 " ;
    }
   else
    {
       if ( temp [ i + 1 ] <> ' - ' )    Then
          if checkTwoCharacters ( temp [ i ] , temp [ i + 1 ] ) )   Then
           {
              bits = bits + " 0 " ;
           }
    }
    return bits ;
}.
```

Algorithm 5: Conversion to bytes

Input: stream of bits

Output: stream of bytes

Method:

```
message [ ] = decode (bits) ;
  decode (bits)
```

```
{
   hidden [bits. length / 8] ;
   c = 0 ;
   for i =1 To hidden. length      step 1
     for j = 7 To 0     step -1
      {
         if (bits [c++] = '1' )  Then
         hidden [ i ] = (hidden [ i ] OR (1<< j)) ;
      }
   return hidden ;
}.
```

3 Experiments and Analysis

The proposed system has been developed with an interface represented by two forms. The first form is denoted facilitates interaction with the presented encoding stage and its respective concealing functions. Furthermore, it displays quality indicators such as hiding and capacity ratios respective to the secret message and the browsed cover text. The second form facilitates interaction with the presented decoding stage and its respective functions. Through its interface, the proposed steganography system has been tested using several multilingual texts (Arabic and English) as demonstrated by Figure 1, Figure 2, Figure 3 and Figure 4 respectively.

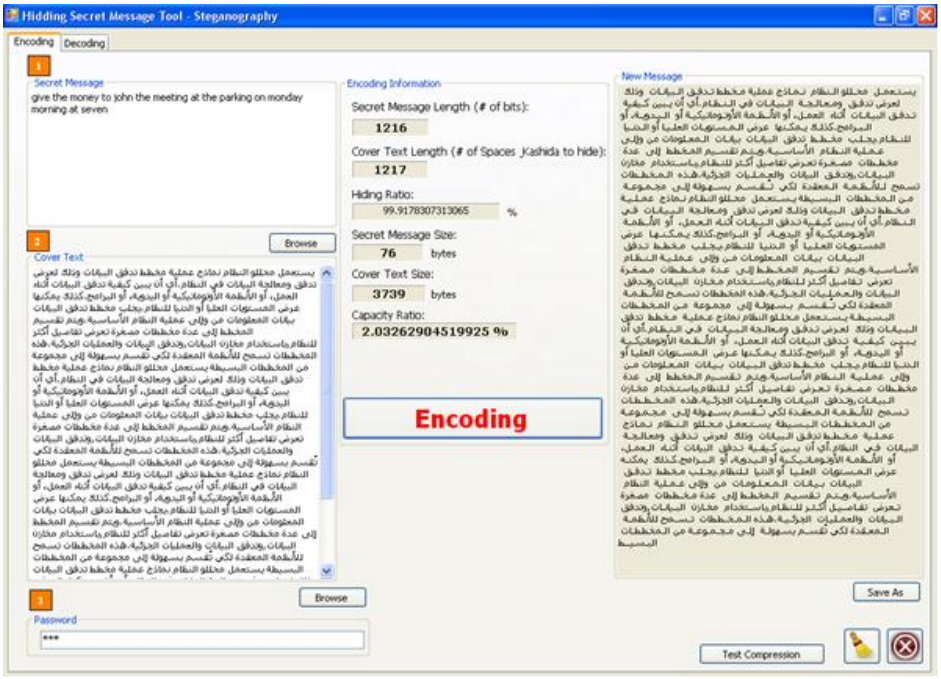


Figure 1: Encoding of English text

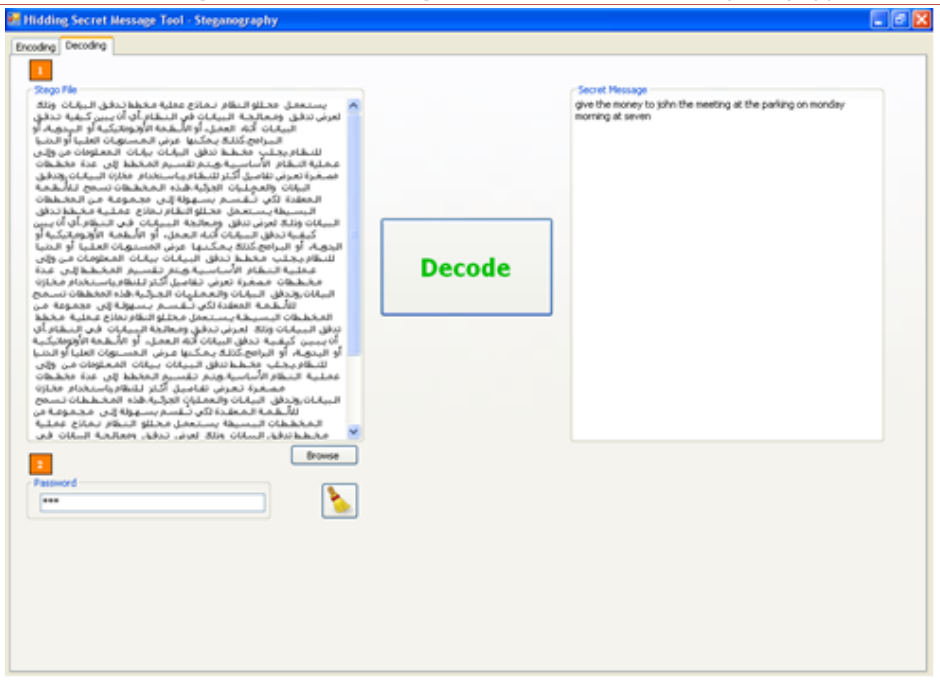


Figure 2: Decoding of English text

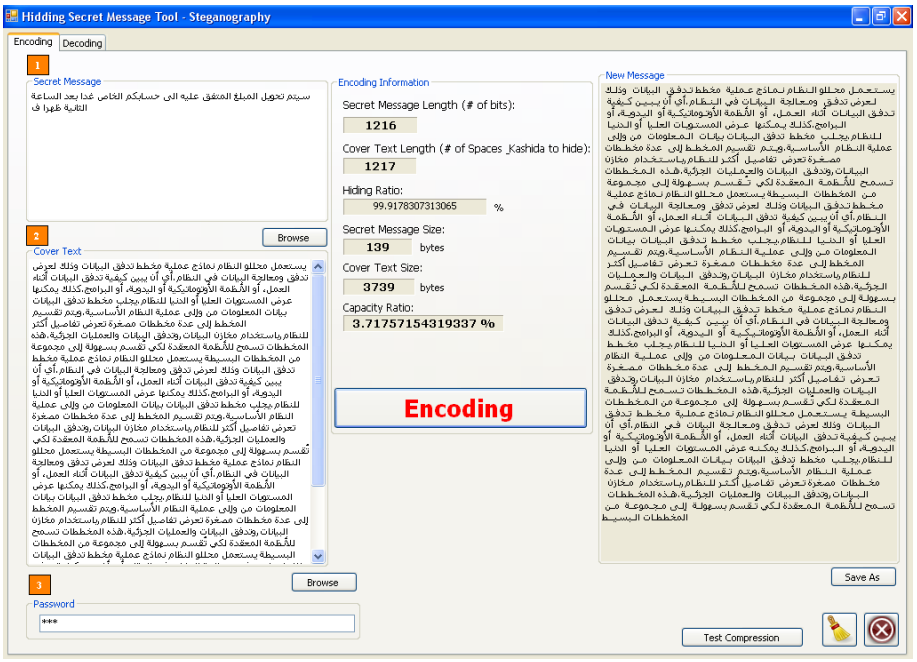


Figure 3: Decoding of an Arabic text

Table 1 shows the achieved capacity ratios for secrete messages and cover texts of different size, where 3.67 is indicated as the average capacity ratio. This ratio is higher than the ones reported by other approaches as demonstrated by table 2. In [1], 2.46 and3.73 are reported as the average capacity ratio using one and three kashidas respectively. However, our approach is based on using one kashida per letter. Using the same text as a cover one, Table 3 shows the size of the stego_text for Arabic and

English secrete messages. In both cases the size of cover text has been increased by 27.6% and 29% respectively.

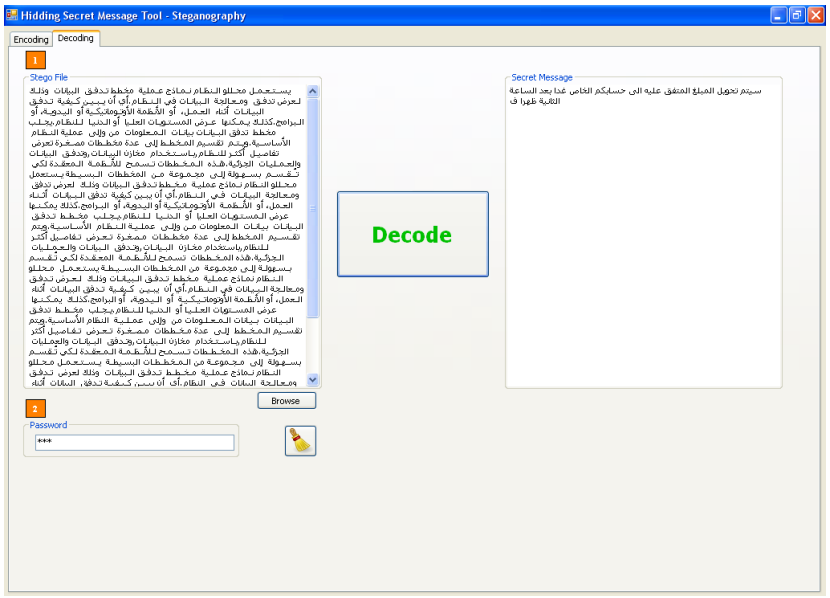


Figure 4: Decoding of an Arabic text

Table 1: The achieved capacity ratio

Cover size (byte)	Number of hidden bits	Capacity ratio (%)	Average capacity (%)
1714	560	3.676	3.657
6976	2288	3.655	
19264	6320	3.649	
31552	10352	3.648	

Table 2: The capacity ratios achieved by other approaches

Approach	Capacity ratio (%)
Dots[15]	1.37
Kashida [13]	2.46-3.73
Kashida [14]	3.09
Diactrics [15]	3.27

Table 3: The size of the secret message

Type of the secret message	Size of the cover text	Size of the stego_text
Arabic	3.65 kb	4.66 kb
English	3.65 kb	4.71 kb

In addition to its efficiency and flexibility, the proposed text steganography system has demonstrated improvements in its quality indicators as follows:

- The adopted encoding methods are characterized by their flexibility and enables hiding dynamic secret messages. The binary conversion and the encryption constitute two level of security
- The adopted embedding criteria combine suitability and randomness to ensure robustness of the stego-text, perceptual transparency and improved hiding capacity.

Compared to similar systems, our system has a higher increase in the size of the cover text. For example, the average size according to our system is 25% and the one according to the system proposed in [5] is 2%. However, the proposed system has better results in terms of the following:

- A higher utilization percentage of the cover- text. This demonstrated by the ratio between the size of the secret message and the one of the cover text. For example, our approach has achieved 1:3 ratio while in [7] a cover text with a size < 16KB is required for a secret message with a length < 4 KB.
- A higher hiding capacity as demonstrated by Table 1 and Table 2.

4 Conclusion

In this research, an information hiding model has been suggested. Based on such model, a text steganography system has been implemented. The system is characterized by its scalability and flexibility. Although the proposed system has better quality indicators than the ones for similar system, more improvements are needed for such indicators. Mainly, the capacity of the cover- text and the robustness of the stego- text. Hence, efforts in this direction constitute a future work.

REFERENCES

[1]. Bennett, K., *Linguistic steganography: Survey, analysis and robustness concerns for information hiding in text*. Purdue University, CERIAS Technical Report, 2004.

[2]. Bender, S., Gruhl, D., Morimoto, N., *Techniques for data hiding.* IBM System Journal, 1996. 35(4): p. 313-336.

[3]. Komal, P., Sumit, V., Hitesh, C., *A Survey of Information Hiding Techniques*. International Journal of Emerging Technology and Advanced Engineering, 2013. 3(1): p.347-349

[4]. Cachin, C., *An information –theoretic model for steganography*. Lecture Notes in Computer Science, 1998. 1225: p. 306-318.

[5]. Jabri, R., Ibrahim, B., Al-Zoubi, H., *Information Hiding: A Generic Approach*. Journal of Computer Science, 2009. 5(12): 930-936.

[6]. Sumathi, G., Santanam, T., Umamaheswari, G., *A study of Various steganogrrarphic Techniques Used for Information Hiding.* International Journal of Emerging Technology and Advanced Engineering, 2013. 4(6): P. 9-25.

[7]. Por, L., Delina, B., *Information hiding: A new approach in text steganography*. 7th WSEAS Int. Conf. on Applied Computer & Applied Computational Science, Hanghou, Chiana, 2008. P. 689-695.

[8]. Kwan, M., The SNOW homepage, http://www.darkside.com.au/snow/, 1998.

[9]. Haddouch Bergmair, R., *Towards linguistics stganogrphy: A systematic investigation of approaches, Systems and Issues*. Technical Report. 2004.

[10]. Shirali-Shareza, M.,H., Shirali-Shareza, M. *Steganography in Persian and Arabic Unicode Texts Using Pseudo-Space and Pseudo-Connection Characters*. Journal of Theoretical and Applied Information, 2008. 4(8): p. 682-687.

[11]. Shirali-Shareza, M., H., Shirali-Shareza, M. *A New Approach to Persian/Arabic Text Steganography*. 5th IEEE/ACIS on Computer and Information Science (ICIS- COMSAR 06), 2006. P. 310-315.

[12]. Shirali-Shareza, M. *A New Persian/Arabic Text Stegonography Using " La Word"*. Proceedings of the International Joint Conference on Computer**,** Information and Systems Sciences and Engineering**,** Bridgeport, CT, USA, 2007.

[13]. Al Haidari, F., Gutub, A., Al-Kahsah, K., Hamodi, J. 2009. *Improved Security and Capacity for Arabic Text Steganography Using 'Kashida ' Extensions*. IEEE/ACS International Conference on Computer Systems and Applications, Rabat, 2009. P. 396-399.

[14]. Al-Azawi, A.F., Fadhil, M.,A., *Arabic Text Steganography Using Kashida Extensions with Huffman Code*. Journal of Applied Sciences 2010. 10(5): P. 436-439.

[15]. Gutub, A., Elarian, Y., Awaideh, S., Alvi, A., *Arabic Text Steganography Using Multiple Diacritics*, WoSPA 5th IEEE International Workshop on Signal Processing and its Applications, Sharjah, UAE, 2008.

SOCIETY FOR SCIENCE AND EDUCATION UNITED KINGDOM
TMLAI TRANSACTIONS ON MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE
VOLUME 3, ISSUE 3
ISSN 2054-7390

Combining Overall and Target Oriented Sentiment Analysis over Portuguese Text from Social Media

[1]José Saias, [2]Ruben Silva, [3]Eduardo Oliveira and [3]Ruben Ruiz
[1]Universidade de Évora, Portugal;
[2]Cortex Intelligence, Portugal;
[3]BizDirect, Portugal
jsaias@uevora.pt; ruben.silva@cortex-intelligence.com; eduardo.oliveira@bizdirect.pt; ruben.ruiz@bizdirect.pt

ABSTRACT

This document describes an approach to perform sentiment analysis on social media Portuguese content. In a single system, we perform polarity classification for both the overall sentiment, and target oriented sentiment. In both modes we train a Maximum Entropy classifier. The overall model is based on BoW type features, and also features derived from POS tagging and from sentiment lexicons. Target oriented analysis begins with named entity recognition, followed by the classification of sentiment polarity on these entities. This classifier model uses features dedicated to the entity mention textual zone, including negation detection, and the syntactic function of the target occurrence segment. Our experiments have achieved an accuracy of 75% for target oriented polarity classification, and 97% in overall polarity.

Keywords: Sentiment Analysis; Opinion Mining; Text classification; Machine Learning; Natural Language Processing.

1 Introduction

Microblogging and social networks are used by people of all ages. These platforms offer a new form of Web based socialization, simplifying communication to restricted groups or to crowds. They aggregate user-generated content, such as opinions that people write and publish online, and are now valued for market research and trend analysis.

In this paper we describe the use of Natural Language Processing (NLP) techniques for Sentiment Analysis (SA) of social media texts, in Portuguese, sensing the overall and the target oriented sentiment polarity. To automatically extract such information from text, it is necessary to deal with the challenges of natural language, plus the present-day web writing style, full of symbols, tags, abbreviations and misspellings. Given a post or tweet text, the system must find the overall polarity and also the polarity of any specific entity mention. Figure 1 has three examples, of increasing complexity, with the original text in Portuguese, and the respective translation. In the first case we must detect negative polarity in the overall sentiment. For 1(b), beyond the overall sentiment it is necessary to detect the reference to benfica, and that it is positive. The third example is more complicated, because the second and third

DOI: 10.14738/tmlai.33.1297
Publication Date: 26th June, 2015
URL: http://dx.doi.org/10.14738/tmlai.33.1297

sentences both mention two entities, and with opposite polarities. In each sentence, Belenenses has a negative polarity, while Benfica is referred with positive polarity.

The system being described here is a continuation of previous work on overall polarity classification [1] and aspect based sentiment analysis [2] with English texts. We use a supervised machine learning classifier for both overall and target oriented sentiment polarity, with different tuning in feature extraction. Opinion target entities are automatically detected with named entity recognition, complemented with an entity catalog. The analysis pipeline is explained in detail in section 3.

Fiquei aborrecido com o exame nacional.
(in English: "*I was disgusted with the national examination.*")
(a)

Finalizada a época , o benfica é vencedor
(in English: "*Finished the season, the winner is benfica*")
(b)

Adorei o jogo. Benfica 3-1 Belenenses... O Benfica arrasou o Belenenses.
(in English: "*I loved the game. Benfica 3-1 Belenenses... Benfica smashed Belenenses.*")
(c)

Figure 1: Different examples of text to be analyzed: without entities (a); with a single entity (b); and having more than one entity (c).

2 Related Work

There are many works in the SA field, from industry and from academia, which differ in the specific objectives, in their subdomain of expertise, on the language in which they operate, or in the approach taken for polarity classification.

Twitómetro [3] is a system to gauge sentiment towards five political leaders, via Twitter, during the campaign for the 2011 portuguese elections. His politics domain predecessor was OPTIMISM [4], an opinion mining system using an ontology of political entities to assist in entity recognition, and whose polarity classifier combines rules on lexical-syntactic patterns with machine learning. POPSTAR is a research initiative on public opinion and sentiment tracking, producing indicators on entity mentions frequency, and their polarity, over time. Several sub-projects emerged from there, particularly on reputation as described in [5]. Besides Twitter, news feeds are also used for opinion mining, as in [6], through an UIMA-based pipeline for SA, in Portuguese.

Modern systems and SA's state of the art can be seen in NLP Workshops and competitive challenges, such as RepLab [7], SemEval Twitter SA [8], and aspect based SA tasks [9,10]. These events attract participants around the world, but most focused in the English language. One of the top participating systems in SemEval aspect based SA task, in 2014, was the NRC-Canada system [11], which extended the base training corpus with additional customer reviews corpora, and used Stanford CoreNLP to perform tokenization, POS tagging, and dependency analysis. Polarity classification was dealt with a linear SVM classifier, having features for the target and its surrounding words, POS based features, dependency tree based features, unigrams and bigrams, and lexicon based features.

3 Method

In this paper, we assume the existence of a posts and tweets collector module. Our system has a REST API, where the content of those publications can be sent for analysis.

3.1 Underlying Platform and Preprocessing

This work reuses the basic framework of a recent real-time SA system [2] for English texts. Our system is developed in Java, using the tool MALLET [12], a package for statistical natural language processing and machine learning applications to text. Jersey[1] RESTful Web Services framework was used in the system frontend, for making the core functionality available as a service.

The received input is preprocessed through noise removal, tokenization, POS tagging and lemmatization. Data representing social media content is rich in metadata tags and hyperlinks. These noisy parts may hinder the automatic understanding of sentences. Thus, in preprocessing we remove certain elements such as URL addresses and retweet prefixes. However, other elements cannot be removed at this point, because they are potential indicators for polarity, as in the case of hashtags. Instead of using MALLET's default tokenization pipe, we implemented a new tokenization and POS tagging module, based on Apache OpenNLP[2] library, with Portuguese trained models. The lemmatizer is a proprietary software that depends on the POS tag and the textual context of words.

3.2 Overall SA

For this phase, the system must search for sentiment clues on the message transmitted globally by the text. To determine the general sentiment polarity we used a machine learning solution, with a supervised approach and Maximum Entropy classifier, through MALLET classification libraries. To build the model, the classifier was trained from a set of 59000 labeled instances with texts on popular expressions, and online comments on music, festivals, television, sports and politics domains. Two sentiment lexicons are used to assist in feature extraction. The first is SentiLex-PT [13], a sentiment lexicon for Portuguese, made up of 7014 lemmas, and 82347 inflected forms of verbs, nouns, adjectives and idiomatic expressions. The other lexicon is a complementary polarity table, contained in a linguistic knowledge base from a previous work [14]. It has evolved with the gradual introduction of new Portuguese expressions, including idioms, but also popular English expressions, Internet jargon, and common symbols and abbreviations. Our classifier model is then trained with the following features:

- Bag-of-Words (BoW) on lemmas: instead of counting the occurrences of the original text on each token, we consider the frequency of their respective lemmas. To illustrate, let's consider the sentence from the example in Figure 1(a). Some of its features would be: `ficar=1, aborrecer=1, com=1, o=1, exame=1, nacional=1`.
- A pair (POS tag, simple polarity), for each token, and the counting of these values. The second token in the example would have the feature `(v-fin.negative)=1`.
- Bigrams of pairs (POS tag, simple polarity). Like the previous feature, but on each consecutive token pair. The first value in the given example would be `(v-fin.neutral,v-fin.negative)=1`.

[1] https://jersey.java.net
[2] http://opennlp.apache.org

- Trigrams of pairs (POS tag, simple polarity), as before but over three consecutive tokens.
- Bigram before/after polarized terms (positive or negative), according to each sentiment lexicon. If a sentiment lexicon identifies a token T as positive or negative, we generate two features: one with the previous word and T, and another with T and the next token's text, all in lemma form. In the example, it would be `ficar. aborrecer` and `aborrecer. com`.
- Subject/object polarity, if a sentiment lexicon determines the polarity that some expression originates, on the subject and on the object, inside that sentence. As example, the verb *defeated* is positive for the subject, but negative for the entity in the object.
- Presence of terms with positive/negative polarity within the last five tokens. Because sometimes the last words summarize the main idea or polarity.
- Balance of polarity, according to each sentiment lexicon, calculated by the total of positive expressions minus the total of negative expressions, considering also denial detection.
- Bigrams after verbs, and after negation terms, using the lemmatized forms.

Figure 2 shows the result of processing the text in Figure 1(a), with the negative polarity shown in the `overallPolarity` field. If this field is zero, the polarity is neutral; if the value is less than zero, we have negative polarity; and a value greater than zero corresponds to a positive polarity. The remaining two fields denote the absence of entities in the text, and will be explained in the next section. The service output can be in JSON or XML format.

```
{
  "overallPolarity" : -0.9822897,
  "targetCount" : 0,
  "targetPolarityList" : [ ]
}
```

Figure 2: Overall result for sentence in Fig. 1(a).

3.3 Target Oriented SA

The system starts by identifying entity mentions on the text. These mentions can be opinion targets, and when they are not, the system should give them neutral sentiment classification. Our process to detect target entities comprises a named entity recognition (NER) module, complemented by the use of an entity catalog. For NER, we use an OpenNLP classifier with a model trained for Portuguese. Entities whose categories are currency, time, numeric or abstract are discarded. The most plausible, in categories person, organization, brand, and location, are selected. The entity catalog is an inventory whose records contain the entity canonical name, possible name aliases, and the entity type. This resourse allows the system to realize that, for example, SCP is an alias or alternative designation for Sporting Clube de Portugal, an organization.

In the next step, the system must assign a sentiment polarity (positive, negative or neutral) to every detected entity mention, according to the text content. For such, we prepared a second Maximum Entropy classifier, now based on different features. In this supervised learning, the training labeled instances do not include only the text content, as before. Each instance also has the target entity, and there is an indication of where it is mentioned. Due to the added complexity in corpus annotation, this

training set is smaller than the used for the overall sentiment classifier. Here we have only 13100 instances. We seek the sentiment for each specific entity mention. So for these instances we want features related to that same mention, probably more confined to a short text area. The features for the target oriented classifier are:

- Bag-of-Words for the mention's text area. We define a feature for the original text of each token inside the short sentence that includes the entity mention. The entity name is replaced by TARGET , to facilitate harmonization of cases of sentences with similar structure but in which the opinion focuses on different names. Taking the example shown in Figure 1(b), these first features are [o=1, TARGET=1, é=1, vencedor=1] .
- Lemma bigrams for tokens within the metion's text area. For the same mention, it would be: [m.bigram_o.TARGET=1, m.bigram_TARGET.ser=1, m.bigram_ser.vencedor=1].
- Syntactic function associated with the target. When possible, indicate whether the target appears in the subject or object, according to sentence structure.
- Subject/object polarity. As before, if a sentiment lexicon determines the polarity that some expression originates, on the subject or on the object part, we create a feature for it. Returning to the example, *vencedor* is an adjective with positive polarity to the subject, which in this case is *benfica*.
- Lemma bigrams, after the target, and before the target mention.
- Pairs and bigrams and trigrams of (POS tag, simple polarity) pairs, as before, for the full text.
- Bigrams before/after polarized expressions, as before, across the full text.


```
{
  "overallPolarity" : 0.98907655,
  "targetCount" : 1,
  "targetPolarityList" : [
  { "target" : "benfica",
    "polarity" : 0.8114217,
    "countPositiveRefs" : 1,
    "countNegativeRefs" : 0,
    "countNeutralRefs" : 0,
    "targetReferencesOverDoc" : [
       { "referencePolarity" : 0.8114217,
         "from" : 23, "to" : 30,
         "sentenceNumber" : 0 } ]
  } ]
}
```


Figure 3: Analysis result for the example in Fig. 1(b)

In Figure 3 we can see the detailed output returned by the system, result of analyzing the text in Figure 1(b). In targetCount field we have the number of entities mentioned in the text. Then we have a list with the sentiment polarity for each target entity. In this case we have benfica, an entity with positive polarity, and one (positive) reference, which takes place in the first sentence (by sentenceNumber field), more precisely in the text between the position 23 and the position 30.

Having all this detail in the answer, we can provide a friendly visual output. With the polarity and the precise location of the target entity, we can assign colors to facilitate the interpretation of the results, as shown in Figure 4.

Figure 4: Visual output of the analysis for Fig. 1(b)

Sometimes we may have different opinions on the same document, or even in the same sentence, on the same entity or not. This is the case of the example shown in Figure 1(c). The JSON code with our system's output for such example is listed on Figure 5. The targetCount field shows us that there are references to two entities. Belenenses is referred to twice, in the second and in the third sentences, and both times having negative polarity, resulting in a target polarity value of -1.47. Benfica is also referred to twice, but with positive polarity. With four entity mentions, this is an example where the visual output is clearly easier to read. The result for this case is shown in Figure 6.


```
{
  "overallPolarity" : 0.28949898,
  "targetCount" : 2,
  "targetPolarityList" : [
  { "target" : "Belenenses",
    "polarity" : -1.4753892,
    "countPositiveRefs" : 0,
    "countNegativeRefs" : 2,
    "countNeutralRefs" : 0,
    "targetReferencesOverDoc" : [
       { "referencePolarity" : -0.5799957,
         "from" : 12, "to" : 22,
         "sentenceNumber" : 1 },
       { "referencePolarity" : -0.89539355,
         "from" : 20, "to" : 30,
         "sentenceNumber" : 2 } ]
  },
  { "target" : "Benfica",
    "polarity" : 0.92901266,
    "countPositiveRefs" : 2,
    "countNegativeRefs" : 0,
    "countNeutralRefs" : 0,
    "targetReferencesOverDoc" : [
       { "referencePolarity" : 0.09627137,
         "from" : 0, "to" : 7,
         "sentenceNumber" : 1 },
       { "referencePolarity" : 0.83274126,
         "from" : 2, "to" : 9,
         "sentenceNumber" : 2 } ]
  } ]
}
```


Figure 5: Analysis result for the example in Fig. 1(c)

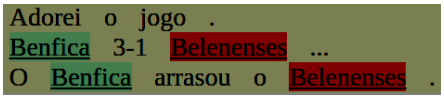


Figure 6: Visual output of the analysis for Fig. 1(c).

3.4 Model Improvement

The system can evolve, by adjustments in the framework that contribute to a faster running, and improvements in the classification model, for better performance in terms of accuracy and precision. The priority is the second case, on system output quality. For the detection of entities, the opinion

targets, the main NER tool may be replaced, if necessary. And minor flaws can be overcomed by the introduction of new entries in the supplementary entities catalog.

For this system's most important component, the sentiment polarity classification, we chose two aspects to monitor: the adequacy of the model features, and the training set. By studying the output of the system, we seek clues to distinguish the characterization of mislabeled instances. Observing their respective text, we can add features on the sentences structure, about new symbols, or context aspects based. On the other hand, and in parallel, the training set will grow, being added new annotated instances, both for overall sentiment and target oriented model tuning. For this purpose, we set up an interface for collecting feedback, which facilitates marking the sentiment polarity. This way, on the next model iteration, these new instances are already used to train the classifier. Figure 7 illustrates the collection of feedback on the third sentence of the last example. The user may indicate the sentiment polarity regarding Belenenses, leading to a new labeled instance. Also, when collecting this feedback, if a marked target entity was not yet known, it will be added to the system catalog.

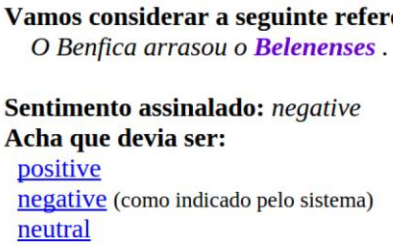


Figure 7: Web interface for gathering feedback and corpus annotation

4 Results

The perception of the performance in the main components of a system is critical, because only then we can realistically improve the service. To evaluate our classification model, on both analysis types, we have used a k-fold cross-validation method. The labeled instances are partitioned into k subsets. Then there are k rounds of evaluation, in which, and in turn, each of the k instance sets is used to test the classifier trained with the other k-1 sets. At the end of the process, all subsets were used only once for testing, and their results are combined. We used the typical 10-fold evaluation, which means that each training round has 90% of the instances. Table 1 has the accuracy values, for both overall and target oriented SA. It is a general measure for success on the predicted polarity. The underlying set of labeled instances is not equal in both cases. In overall SA we have more instances, but fundamentally with short texts, often with a single short sentence, for which the classifier worked fine.

Increasing the evaluation detail, we calculated the precision, recall and F-measure, for each polarity class, and these metrics' results are shown in Table 2. The overall polarity classifier has the top performance, reaching 98% precision and recall for the negative class, and slightly less in the positive class. The target oriented classifier achieved poorer results, particularly on the effectively polarized classes (positive and negative). Its best precision and recall were obtained in neutral class, being noticed, on the other side, a weak coverage for positive class, with only 64%.

Table 3 shows the weight of each polarity class in training, for both the classifiers. Considering also the information from the previous table, we notice, as expected, that in classes with more training instances

the evaluation results are better. An important part in target oriented SA is the detection of the sentiment target entities. When evaluating our entity recognition method with the annotated corpus we used for polarity training, the accuracy is 88%.

Table 1: Accuracy for sentiment polarity.

Analysis	accuracy
overall	0.97
target oriented	0.75

Table 2: Detailed evaluation result.

Analysis	class	precision	recall	f-measure
overall	positive	0.96	0.96	0.96
	negative	0.98	0.98	0.98
	neutral	0.78	0.77	0.78
target oriented	positive	0.67	0.64	0.65
	negative	0.75	0.75	0.75
	neutral	0.77	0.80	0.78

Table 3: Polarity class weight in the training instances.

Analysis	positive	negative	neutral
overall	19%	72%	9%
target oriented	21%	40%	39%

5 Conclusion

We described a sentiment analysis system for social media content, thought to classify both overall sentiment, and sentiment towards specific targets mentioned in the text. Despite the good accuracy, in development time, of our overall sentiment classifier, its use in post-development period reveals more errors. We think this is due to the differences between training instances and these recent input texts, which have greater length and a more complex structure than the former. Portuguese corpus for overall sentiment usually do not have many long texts.

As in other NLP tasks, small errors in the modules that carry the first part of the analysis can compromise the quality of the final classification result. Establishing a comparison with our previous experiences in English language target oriented SA [2], in this work we got 3% less accuracy. But in English we have more tools to work the text, and more labeled data resources, than those available for Portuguese.

As future work, we plan to increase the size of the corpus annotated for training, as a measure to mitigate the difference in results between classes. At the same time, we will continue to experiment with new features, looking to improve precision and recall. Apart from the performance of our current system, there is a feature that could be introduced: aspect classification. The analysis result would be richer, pointing out a particular aspect of the target that is affected by some sentiment polarity. In the case of comments on some restaurant, possible aspects could be the price or the food. And for each of them we could then examine the sentiment polarity, independently.

Although there still is a significant error rate, of approximately 25% for target oriented SA, this tool can greatly facilitate the analyst work. Let us consider a collection of posts or tweets, where only 40% are not neutral. On average, to find 40 positive or negative opinions, a human would have to read 100 documents, whereas using a system like the one presented in this paper, the human analyst would only have to filter the classification results. Even on the worst case, the gain is that the work will be halved.

ACKNOWLEDGMENTS

This project was approved under the "SI ID&T – projeto individual", and according to AAC nº 07/SI/2012, project number 38601.

REFERENCES

[1] J. Saias, Senti.ue: Tweet overall sentiment classification approach for SemEval-2014 task 9. In Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), Dublin, Ireland, August 2014. ISBN 978-1-941643-24-2, p. 546–550.

[2] J. Saias, Sentiue: Target and Aspect based Sentiment Analysis in SemEval-2015 Task 12. In Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), Colorado, USA, 2015. ACL.

[3] M. J. Silva et al., Notas sobre a Realização e Qualidade do Twitómetro. Technical Report. University of Lisbon, LASIGE. 2011.

[4] M.J. Silva et al., The Design of OPTIMISM, an Opinion Mining System for Portuguese Politics. In New Trends in Artificial Intelligence: Proceedings of EPIA 2009 - Fourteenth Portuguese Conference on Artificial Intelligence, 2009, p. 565-576.

[5] J. Filgueiras and S. Amir, POPSTAR at RepLab 2013: Polarity for Reputation Classification. In Proceedings of the 4th International Conference of the CLEF initiative, CLEF 2013, Valencia, Spain.

[6] P. Lambert and C. Rodriguez-Penagos, Adapting Freely Available Resources to Build an Opinion Mining Pipeline in Portuguese. In Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), Iceland, 2014.

[7] E. Amigó et al., Overview of RepLab 2014: Author Profiling and Reputation Dimensions for Online Reputation Management. In Information Access Evaluation. Multilinguality, Multimodality, and Interaction. Lecture Notes in Computer Science, Volume 8685, 2014, p. 307-322.

[8] S. Rosenthal et al., SemEval-2014 Task 9: Sentiment Analysis in Twitter. In Proceedings of the Eighth International Workshop on Semantic Evaluation (SemEval'14). August 23-24, 2014, Dublin, Ireland.

[9] M. Pontiki et al., SemEval-2014 Task 4: Aspect Based Sentiment Analysis. Proceedings of the 8th SemEval, Dublin, Ireland. 2014.

[10] M. Pontiki et al., SemEval-2015 Task 12: Aspect Based Sentiment Analysis. In Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), Denver, Colorado, USA. 2015.

[11] S. Kiritchenko et al., NRC-Canada-2014: Detecting Aspects and Sentiment in Customer Reviews. In Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014). Dublin, Ireland, 2014, p. 437–442.

[12] A. K. McCallum, MALLET: A Machine Learning for Language Toolkit. 2002. http://mallet.cs.umass.edu

[13] M. J. Silva et al., Building a Sentiment Lexicon for Social Judgement Mining. In Lecture Notes in Computer Science (LNCS) / Lecture Notes in Artificial Intelligence (LNAI), International Conference on Computational Processing of Portuguese (PROPOR), Coimbra, 2012.

[14] M. Mourão and J. Saias, BCLaaS: implementação de uma base de conhecimento linguístico as-a-service. In L. Ferreira and V. Pedro, editors, Actas das 3as Jornadas de Informática da Universidade de Évora. ECT, Universidade de Évora, Portugal, 2013.

SOCIETY FOR SCIENCE AND EDUCATION
UNITED KINGDOM
TMLAI TRANSACTIONS ON MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE
VOLUME 3, ISSUE 3
ISSN 2054-7390

Probabilistic Search and Pursuit Evasion on a Graph

[1]E. Ehsan and [2]F. Kunwar
National University of Sciences and Technology Islamabad Pakistan
[1]ehsan74@mts.ceme.edu.pk; [2]kfaraz@gmail.com

ABSTRACT

This paper presents an approach to locate an adversarial, mobile evader in an indoor environment using motion planning of mobile pursuers. The approach presented in this paper uses motion planning of mobile robots to search a target in a graph and clear the workspace. The algorithm used is Partially Observable Markov Decision Process (POMDP), a probabilistic search method to clear the indoor workspace in a pursuit evasion domain. In this paper, the indoor environment is assumed to be known beforehand and the mobile evader to be adversarial with no motion model given. The workspace is first discretized and then converted to a graph, whose nodes represent the rooms and corridors and edges represent connection between them. The task of pursuer is to clear the whole graph with no contaminated node left in minimum possible steps. Such path planning problems are NP-hard and the problem scales exponentially with increased number of pursuers and complex graph.

1 Introduction

Pursuit-evasion (PE) has been extensively explored in the fields of software engineering, mathematics and robotics. Many different approaches have been proposed to capture one or more evaders throughout the course of history after Parsons [1] proposed the pursuit-evasion games in a graph. The approaches include search in graph, search in a polygon, adversarial search, non-adversarial search, probabilistic search, and search with only one pursuer and search with multiple pursuers etc. Many proposed strategies focused on finding a target in minimum time or minimum steps, others proposed guaranteed search algorithms irrespective of the cost incurred, time elapsed or steps taken.

This paper presents an approach to find an adversarial target using probabilistic search algorithm. The task is to capture the evader by clearing a graph with guarantee and keeping the cost, and capture time as low as possible.

2 Related Work

One of the basic games that take place on graph is the cops and robber game. In this game cops (pursuers) try to capture a robber (evader) along the vertices of a graph[2][3][4]. The question that arises in this type of problem is (1) what is the search number of graph? Irrespective of the pursuer's initial position and (2) what is the class of graph? Another problem with the above given approaches is that, during the play, both pursuer and evader know each other's position all the time. An approach with local visibility of evader was proposed in [5]. In this approach evader is considered to be having a local visibility or i-visibility of the environment. If the evader has 1-visibility, two pursuers with 1-visibility

DOI: 10.14738/tmlai.33.1313
Publication Date: 27th June, 2015
URL: http://dx.doi.org/10.14738/tmlai.33.1313

can capture an evader in any graph with a higher probability. The expected time of capture with two pursuers is polynomial in the vertices of graph. It was also shown in [5] that when the evader has 2-visibility, the pursuers number becomes unbounded. One approach of worst-case pursuit-evasion is to consider evader as adversarial target having an infinite speed, as compared to slow pursuers. The search number was defined in [1] by T.D. Parsons as the minimum number of pursuers necessary for capture. Determining the search number of a graph was found to be NP-Hard [6], and to be NP-Complete due to monotonicity of optimal edge search schedules [7][8][2]. In [9] an offline, greedy and iterative algorithm is proposed for indoor pursuit-evasion that searches a graph for an adversarial evader, who is actively trying to avoid capture. The algorithm guarantees the capture of even an adversarial evader. The problem with this greedy and iterative algorithm is that it sometimes need more pursuers to search the graph for a guaranteed capture than other already present algorithms.

Sometimes the temporal aspects of the game is lost, when the game is taken place on a graph that is an abstraction of a geometric environment. It was showed in [10] that a pursuer can catch a non-deterministic evader in any simply-connected polygonal environment using a randomized strategy. In [11] it was shown that three pursuers can capture an evader in a polygon (even with holes and obstacles). In this problem, the holes were considered to be finite, the problem was not adversarial as both pursuers and evader can see each other all the time. A visibility based pursuit-evasion was proposed in [12][13][14]. A visibility based pursuit-evasion in a polygonal environment was proposed in [15]. The evader is considered to be adversarial, with unknown initial and current position. A guaranteed search with spanning trees is proposed in [16], an anytime algorithm for multi-robot search. The proposed GSST algorithm clears the environment of any adversarial evader using fewest number of pursuers. The NP-hard problem on arbitrary graph can be solved using spanning trees in linear time. In [17] a group of mobile searchers have to find mobile evaders in a polygonal region. Upper and lower case bounds on search number of polygon are also discussed.

Unlike traditional search strategies that try to find a guaranteed solution for a graph or a polygonal environment, the probabilistic search methods consider optimization of the expected value of a search objective, such as maximal probability of detection or minimal time to capture. Many probabilistic search methods have been proposed so far. A mixed observability based robotic tasks planning algorithm under uncertainty is proposed in [18]. These type of probabilistic algorithms try to maximize the capture probability of an evader. The probability is calculated in many ways, such as using the probabilistic motion model of the evader or calculating the evader probability on the basis of previous search history. It is NP-hard to find an evader in a polygon or in a graph using probabilistic search techniques. Unlike classical pursuit-evasion and graph search which rely on evader motion model or uncertainty in search strategy, an alternative formulation of multi-robot search problems uses a probabilistic approach to model the location of the evader or the movement of the searchers[2].

Many search problems can be formulated as a Markov Decision Process (MDP) if the target's position is known [19] or a Partially Observable Markov Decision Process (POMDP) if it is unknown. In [20] it is shown that how belief compression can be used to make a POMDP search problem tractable for a single pursuer. This approach fails when the team size scales up or the state space is increased. The POMDP search problem can be searched with two pursuers using mixed observability (e.g. when the pursuer's position is fully known but the evader's position is unknown) and the Markovian target motion model is given, as shown in [18]. Most of the probabilistic search models assume the target to be non-adversarial

and the search environment to be known beforehand. Most if not all the probabilistic search techniques suffer with scalability issues as the team size scales up or the environment becomes more complex.

From the above discussion it is evident that the graph search algorithms and polygon search algorithms are greedy in nature and guarantee the capture of an adversarial evader in finite time [2], but they need a lot of resources and incur a high cost. They are not feasible for higher dimensional environments. On the other hand the probabilistic algorithms either need a motion model of the evader or they incur a high cost in finding the evader, probabilistic algorithms also do not guarantee the capture. Another problem with these algorithms is that they are complicated and become hard to implement as the team size scales up or the environment or graph becomes more complex. Such solutions sacrifice completeness over cost minimization. They may or may not find an evader, even if there exists one.

This paper presents an approach to find an adversarial evader using a modified form of the Partially Observable Markov Decision Process (POMDP) to minimize the overall cost, while guaranteeing capture.

3 Problem Formulation

The objective is to locate an adversarial evader in a graph with minimal computational effort. In this study, PE is set to take place in an indoor environment. Moreover it is assumed that, both pursuer and evader cannot leave the environment. The evader is adversarial having infinite speed, but can only move along edges and can hide in nodes. The pursuer has a unit speed and can move only one step at a time. Both the pursuer and evader know the map, evader also knows the pursuer position but pursuer does not know the evader position. The capture happens when either the evader and pursuer reside in the same node or evader comes in line of sight of pursuer. The pursuer has a 360 degree field of view camera mounted on it and can see an evader in its line of sight [9].

Consider the map of Fig.1.a. This map can be converted to a graph using discretization, as shown in Fig.1.b.

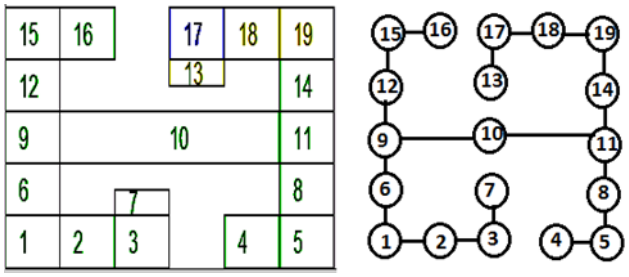
Some steps to convert the polygonal indoor environment to a graph are as following:

1. The indoor environment is discretized into cells, each of these cells correspond to a node of an (*undirected*) *navigation graph*. The discretization takes place on *critical visibility events* [9].
2. A (*directed*) *information space* is obtained from the *navigation graph*. The information graph is a *state transition diagram* of the search process [9].
3. Search takes place in the information graph to find a path from a starting state to a terminal or clear state [9].

As we know that a graph G= (V, E) is described by a set of nodes or vertices V and edges E, such that $E \subseteq V \times V$. The nodes are labelled by integers i.e. for a graph of N vertices we take $V = \{1, 2, 3, \ldots, N\}$. We also have $|V| = N$ (using the *set cardinality notation*). The map to graph conversion is performed by:

1. Discretize the environment into cells.
2. Associate a node to each cell
3. Insert edges to nodes which correspond to adjacent nodes and also an edge to the node itself.

Given a graph G = (V; E) with N nodes, the adjacency of node $n \in V$ is denoted by N (n) and defined by



(a) (b)

Figure 1. Conversion of a map to graph (a) a map of an indoor environment (b) a graph of 1.a.[9]

$$N(n) = \{m \in V: (n, m) \in E\}.$$

The $N \times N$ adjacency matrix **A** of the graph is defined by

$$A_{mn} = \begin{cases} 1 \; iff \;\; n \in N(m); \\ 0 \; else. \end{cases}$$

The adjacency matrix of graph in Fig.1.b is

$$A=$$

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

(1)

For a given graph the *visibility region* of node $n \in V$ is denoted as $V(n)$ and is defined as

$$V(n) = \{m \in V: m \text{ is visible from } n\}.$$

The condition "*m* is visible from *n*" is evaluated under "straight line visibility" and *iff* every point of *m* is visible from every point of *n*. The $N \times N$ visibility matrix **C** of the graph is defined by

$$C_{mn} = \begin{cases} 1 \; iff \;\; n \in V(m); \\ 0 \; else. \end{cases}$$

The visibility matrix of graph in Fig.1.b is

$$\mathbf{C}=\begin{bmatrix}
1&1&1&0&0&1&0&0&1&0&0&1&0&0&1&0&0&0&0\\
1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&1&0&0&1&0&0&1&0&0&1&0&0&0&0&1\\
1&0&0&0&0&1&0&0&1&0&0&1&0&0&1&0&0&0&0\\
0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&1&0&0&1&0&0&1&0&0&0&0&1\\
1&0&0&0&0&1&0&0&1&1&1&1&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&1&1&1&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&1&1&1&1&0&0&1&0&0&0&0&1\\
1&0&0&0&0&1&0&0&1&0&0&1&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0\\
0&0&0&0&1&0&0&1&0&0&1&0&0&1&0&0&0&0&1\\
1&0&0&0&0&1&0&0&1&0&0&1&0&0&1&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&1&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1\\
0&0&0&0&1&0&0&1&0&0&1&0&0&1&0&0&1&1&1
\end{bmatrix}$$

(2a)

The direction matrix **F** of graph in Fig.1.b is given as

$$\mathbf{F}=\begin{bmatrix}
\mathbf{0}&\boldsymbol{N}&\boldsymbol{S}&\boldsymbol{E}&\boldsymbol{W}\\
\mathbf{1}&\mathbf{6}&\mathbf{0}&\mathbf{2}&\mathbf{0}\\
\mathbf{2}&\mathbf{0}&\mathbf{0}&\mathbf{3}&\mathbf{1}\\
\mathbf{3}&\mathbf{7}&\mathbf{0}&\mathbf{0}&\mathbf{2}\\
\mathbf{4}&\mathbf{0}&\mathbf{0}&\mathbf{5}&\mathbf{0}\\
\mathbf{5}&\mathbf{8}&\mathbf{0}&\mathbf{0}&\mathbf{4}\\
\mathbf{6}&\mathbf{9}&\mathbf{1}&\mathbf{0}&\mathbf{0}\\
\mathbf{7}&\mathbf{0}&\mathbf{3}&\mathbf{0}&\mathbf{0}\\
\mathbf{8}&\mathbf{11}&\mathbf{5}&\mathbf{0}&\mathbf{0}\\
\mathbf{9}&\mathbf{12}&\mathbf{6}&\mathbf{10}&\mathbf{0}\\
\mathbf{10}&\mathbf{0}&\mathbf{0}&\mathbf{11}&\mathbf{9}\\
\mathbf{11}&\mathbf{14}&\mathbf{8}&\mathbf{0}&\mathbf{10}\\
\mathbf{12}&\mathbf{15}&\mathbf{9}&\mathbf{0}&\mathbf{0}\\
\mathbf{13}&\mathbf{17}&\mathbf{0}&\mathbf{0}&\mathbf{0}\\
\mathbf{14}&\mathbf{19}&\mathbf{11}&\mathbf{0}&\mathbf{0}\\
\mathbf{15}&\mathbf{0}&\mathbf{12}&\mathbf{16}&\mathbf{0}\\
\mathbf{16}&\mathbf{0}&\mathbf{0}&\mathbf{0}&\mathbf{15}\\
\mathbf{17}&\mathbf{0}&\mathbf{13}&\mathbf{18}&\mathbf{0}\\
\mathbf{18}&\mathbf{0}&\mathbf{0}&\mathbf{19}&\mathbf{17}\\
\mathbf{19}&\mathbf{0}&\mathbf{14}&\mathbf{0}&\mathbf{18}
\end{bmatrix}$$

(2b)

Note that the graph can be cleared by only one pursuer, so it is assumed that the PE takes place in G= (*V, E*) with *V*= {1, 2, …. , N}. The position of pursuer at time *t* is *x* (*t*). If a node *might* contain evader, the node is called *dirty* otherwise *clear.* A node is clear if pursuer is present in it or falls inside the pursuer's *visibility region* and remains *clear* until there is no free (pursuer's *visibility free*) path from it to a dirty node. The node is re-contaminated if there is a pursuer's *visibility free* path from it to a dirty node after it is cleared. A set of all the dirty nodes is denoted by *D*.

The indicator vector for *D* is **d**= {d_1, d_2, …., d_N }, where d_N = 1 *iff n* ∈ *D,* 0 else. The task of the pursuer is to capture the evader by clearing the nodes and in return converting the *dirty* set to *clear* set. At any time *t,* the *state vector* is the *position* of pursuer at time *t* and the *dirty* node set. The *state vector* is denoted as:

$$\mathbf{z}\,(t) = [x\,(t), \mathbf{d}\,(t)] \tag{3}$$

The task of the pursuer is to make the state vector at some time *t* as **z** = (*x*, 0), that is no more dirty nodes should be present in the graph.

Suppose for a moment that the pursuer is removed from the graph and the evader moves with unit speed. Then, the possible locations of the evader at time t +1 are the ones adjacent to its possible locations at time t. Mathematically this is expressed by [9].

$$\mathbf{d}\,(t+1) = \mathbf{d}\,(t) * A; \tag{4}$$

If the evader has speed M (it crosses M edges in a single time step) then instead of (4) we have

$$\mathbf{d}\,(t+1) = \mathbf{d}\,(t) * A^{[M]} \tag{5}$$

4 The Search Algorithm

Since it is assumed that the evader is adversarial so the pursuer does not know the evader position. At any time t the pursuer knows its own position with certainty but the evader position is unknown, only the probability of evader can be calculated from the dirty node set, that is why the search algorithm used is a probabilistic algorithm and is called Partially Observable Markov Decision Process (POMDP)[21][22]. The algorithm is slightly modified according to the problem, as the terminal state is unknown, the standard policy and value iteration solutions cannot be applied.

In the PE process, suppose that the pursuer starts at an initial node zin= (xin; din). The pursuer must move through nodes in a manner such that the 1's in the d part progressively decrease (the dirty set shrinks) until eventually the pursuer reaches one of the clear states, say zfin= (xfin; dfin), where d fin= 0.

As the given graph in Fig.1.b contains 19 nodes, a dirty node vector of 14 elements is defined as $\mathbf{d}$ = [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]

The pursuer's task is to make all the elements of this dirty node vector zero. A (pursuer's *visibility region*) free path between a dirty node and a cleared node makes the cleared node re-contaminated.

In the given problem, the set of actions is denoted by **E** and is defined as

E = [North, South, East, West]

It is assumed that the sensors of pursuer are accurate, that is the pursuer knows its position with certainty, and the observations are accurate. The actuators however are not accurate enough, and the action probability of pursuer is 90%. However the state transition probabilities are considered to be 100% for simplicity and to minimize the resources used.

The Search Algorithm

Input Graph G= (*V, E*); Adjacency matrix **A**; Visibility matrix **C**; Direction matrix **F**; Dirty vector **d**
Find hard nodes; **H** = (nodes visible from minimum nodes, from visibility matrix **C**)
Input Starting node *i* (must be a member of **H**)
Initialization
Assign value to *i* = 0;
Assign values to remaining **H** = -100;
While (~assigned values to all the nodes)
Assign value to nodes adjacent to previously assigned nodes;
If (previous node is adjacent to initial node)
node value = *abs (previous adjacent node value)* + 5
else
node value = *abs (previous adjacent node value)* – 5
end if
end while
Main
While (**d** is not empty)
Move to the adjacent node with maximum value;
Assign current node value = 0;
Previous node value = previous value + 3;
Add previous node to **visited set**;
H nodes value = –100;
If (a clear node is re-contaminated)
Assign value to contaminated node;
Contaminated node value = +90;
end if
end While
Total cost = [{(no of steps taken before **d** becomes empty) × –10} + {(no of times contaminations occurred) × –30}]
Output the policy generated (from the movement history of robot)

The algorithm can be defined as.

1. In the first step of the algorithm, the Hard Nodes vector **H** is formed by finding the nodes that are visible from only one node from the visibility matrix **C**.
2. Then a node is selected from the **H** and is assigned a value equal to zero. All the other nodes of **H** are assigned values equal to -100. The hard nodes should be assigned a negative value because, if the pursuer enters a hard node it will lose visibility of the graph and in return the whole graph will be re-contaminated.
3. After assigning a value to the initial node and hard nodes, values are assigned to the adjacent nodes of **H**. The nodes adjacent to the initial node are assigned a value equal to the value of initial node + 5. The nodes adjacent to the remaining hard nodes are assigned a value equal to the absolute value of hard node – 5;
4. The process of assigning values to nodes continues until all the nodes are assigned with some value. The values of nodes adjacent to the initial node are assigned in increasing order, while the nodes adjacent to the remaining hard nodes get a value in decreasing order as compared to their parent node.
5. After all the nodes are assigned with a value, the pursuer starts to move from the initial node by first finding possible actions from direction matrix **F**, and then finding an action that maximizes the overall value. At the same time avoiding recontamination by entering the hard nodes.

6. In order to maximize the overall value, the negative cost at each step forces the pursuer to find the terminal state in minimum steps possible.
7. The node containing the pursuer is assigned a value equal to zero. The previous node is assigned a value equal to the node's previous value + 3. In this way the value of each visited node again starts increasing that helps the pursuer to return from a corridor like structure.
8. Every time a node is cleared, it is added to the visited node set. If at some time a node that was previously a member of visited node set again becomes a member of dirty node vector, it is considered re-contaminated and is removed from visited set, as well as assigned a value equal to +90. So that to force the pursuer to clear the node again.
9. The process continues until the dirty node set becomes empty.
10. After the terminal state is obtained, a policy is generated that is a mapping of pursuer movements in graph.

The algorithm proposed in this paper resembles POMDP in a way that it uses *value iteration* to assign values to all the nodes. After values are assigned, the algorithm uses *policy iteration* to find the actions that maximize the overall value. The algorithm applies value iteration after each step to prevent the pursuer from going into a loop. It also tries to minimize the cost by finding the terminal state in minimum steps possible, while at the same time guaranteeing capture.

5 Implementation

Potential advantages of the proposed algorithm are highlighted through its implementation. Consider the graph illustrated in Fig.1.a. A single pursuer is used to clear the graph. The graph is composed of 19 nodes. First of all hard nodes are determined, after that values are assigned to these and remaining nodes of the graph. After initial values have been assigned the pursuer starts its search for the evader. At each step, the values of nodes are updated in order to facilitate pursuer in its search for the evader. If a node is re-contaminated it is assigned a higher value, so that pursuer has to clear it again.

From the graph of Fig.1.b it can be seen that the hard nodes are 4, 7, 13 and 16.

$$\mathbf{H} = [4\ 7\ 13\ 16]$$

If initial node selected is 7, then the policy generated is given as:

$$\mathbf{\Pi} = [7\ 3\ 2\ 1\ 6\ 9\ 12\ 15\ 12\ 9\ 10\ 11\ 8\ 5\ 8\ 11\ 14\ 19\ 18\ 17]$$

This policy is exactly the same as was generated in [9] for the same graph using same initial node.

If initial node is 4, the generated policy is given as:

$$\mathbf{\Pi} = [4\ 5\ 8\ 11\ 14\ 19\ 18\ 17\ 18\ 19\ 14\ 11\ 8\ 5\ 8\ 11\ 10\ 9\ 12\ 15\ 12\ 9\ 6\ 1\ 2\ 3]$$

If initial node is 13, the generated policy is given as

$$\mathbf{\Pi} = [13\ 17\ 18\ 19\ 14\ 11\ 8\ 5\ 8\ 11\ 10\ 9\ 12\ 15\ 12\ 9\ 6\ 1\ 2\ 3]$$

If initial node is 16, the generated policy is given as

$$\mathbf{\Pi} = [16\ 15\ 12\ 9\ 6\ 1\ 2\ 3\ 2\ 1\ 6\ 9\ 12\ 15\ 12\ 9\ 10\ 11\ 8\ 5\ 8\ 11\ 14\ 19\ 18\ 17]$$

The above generated policies show the effectiveness of the proposed algorithm in finding a solution to the problem with certainty and in minimum steps possible. The algorithm successfully generated the capture policies for multiple initial nodes, as compared to [9] which was considered as rooted and was used only for one initial node. The algorithm returns a policy only when the graph is cleared and any evader in the graph is successfully captured, or returns a clear graph if there is no evader found in the graph. The proposed Algorithm assigns values to the nodes on the basis of probability of presence of evader in them. The policy generated by the algorithm is in just 20 iterations for node 7 as initial node, which is in fact the minimum no of steps for any algorithm to clear the graph, 26 iterations for node 4, 20 for node 13 and 26 for node 16. All of these generated policies are generated in minimum possible iterations while considering re-contamination for any algorithm.

6 CONCLUSION

A probabilistic algorithm is proposed that uses value iteration and policy iteration concept of POMDP in finding an adversarial evader in a graph in minimum steps possible while guaranteeing capture by maximizing the overall value.

ACKNOWLEDGMENT

We would like to thank Allah Almighty. Our parents for their prayers. Mr. Mansoor Ghazi for his guidance in writing this paper, Mr. Saad Farooq, Mr. Ahsan Gull and Mr. Adnan for their motivation. NUST College of E&ME Pakistan for providing a research oriented environment.

REFRENCES

[1] T. D. Parsons, “Pursuit-Evasion in a Graph,” vol. 642, 1978.

[2] T. H. Chung, G. a. Hollinger, and V. Isler, “Search and pursuit-evasion in mobile robotics A survey,” *Auton. Robots*, vol. 31, no. 4, pp. 299–316, 2011.

[3] R. Nowakowski and P. Winkler, “Vertex-to-vertex pursuit in a graph,” *Discrete Math.*, vol. 43, no. 2–3, pp. 235–239, 1983.

[4] M. Aigner and M. Fromme, “A game of cops and robbers,” *Discret. Appl. Math.*, vol. 8, no. 1, pp. 1–12, 1984.

[5] V. Isler, S. Kannan, and S. Khanna, “Randomized Pursuit-Evasion with Local Visibility,” *SIAM J. Discret. Math.*, vol. 20, no. 1, pp. 26–41, 2006.

[6] N. Megiddo, S. L. Hakimi, M. R. Garey, D. S. Johnson, and C. H. Papadimitriou, “The complexity of searching a graph,” *22nd Annu. Symp. Found. Comput. Sci. (sfcs 1981)*, vol. 35, no. 1, pp. 18–44, 1981.

[7] D. Bienstock and P. Seymour, “Monotonicity in graph searching,” *J. Algorithms*, vol. 12, no. 2, pp. 239–245, 1991.

[8] A. S. LaPaugh, "Recontamination does not help to search a graph," *J. ACM*, vol. 40, no. 2, pp. 224–245, Apr. 1993.

[9] A. Kehagias and S. Singh, "A Graph Search Algorithm for Indoor Pursuit / Evasion," no. July, pp. 1305–1317, 2008.

[10] V. Isler, S. Kannan, and S. Khanna, "Randomized Pursuit evasion in a polygonal environment," *IEEE Trans. Robot.*, no. Wafr, 2005.

[11] D. (department of C. S. and E. of M. Bhadauria and V. (Department of C. S. and E. of M. Isler, "Capturing an Evader in a Polygonal Environment with Obstacles," no. June, pp. 16–26, 2011.

[12] B. P. Gerkey, "Visibility-based Pursuit-evasion with Limited Field of View," *The International Journal of Robotics Research*, vol. 25, no. 4. pp. 299–315, 2006.

[13] L. J. Guibas, J. L. Steven, M. L. David, and L. Rajeev, "A Visibility-Based Pursuit-Evasion Problem," pp. 1–29.

[14] L. J. GUIBAS, J.-C. LATOMBE, S. M. LAVALLE, D. LIN, and R. MOTWANI, "A VISIBILITY-BASED PURSUIT-EVASION PROBLEM," *International Journal of Computational Geometry & Applications*, vol. 09, no. 04n05. pp. 471–493, 1999.

[15] L. Guibas, J.-C. Latombe, S. Lavalle, D. Lin, and R. Motwani, "Visibility-based pursuit-evasion in a polygonal environment," *Algorithms Data Struct.*, pp. 17–30, 1997.

[16] G. Hollinger, A. Kehagias, and S. Singh, "GSST: Anytime guaranteed search," *Auton. Robots*, vol. 29, no. 1, pp. 99–118, 2010.

[17] M. (Department of E. E. U. Yamashita, E. D. D. C. A. C. Umemoto, Hideki (System Engineering Section, I. (Department of E. E. and C. S. of W. Suzuki, and T. (School of C. S. F. U. Kameda, "Searching for Mobile Intruders in a Polygonal Region by a Group of Mobile Searchers."

[18] S. C. W. Ong, Shao Wei Png, D. Hsu, and Wee Sun Lee, "Planning under Uncertainty for Robotic Tasks with Mixed Observability," *Int. J. Rob. Res.*, vol. 29, no. 8, pp. 1053–1068, 2010.

[19] J. H. Eaton and L. A. Zadeh, "Optimal Pursuit Strategies in Discrete-State Probabilistic Systems," *J. Basic Eng.*, vol. 84, no. 1, p. 23, Mar. 1962.

[20] N. Roy, G. Gordon, and S. Thrun, "Finding approximate POMDP solutions through belief compression," *J. Artif. Intell. Res.*, vol. 23, pp. 1–40, 2005.

[21] K. P. Murphy, "A survey of POMDP solution techniques," *Environment*, vol. 2, no. September, p. X3, 2000.

[22] A. Barto, "Reinforcement learning: An introduction," 1998.

