Information Gain Based Dimensionality Selection for Classifying Text Documents

Dumidu Wijayasekara, Milos Manic
University of Idaho,
Idaho Falls, ID, USA
wija2589@vandals.uidaho.edu, misko@uidaho.edu

Miles McQueen
Idaho National Laboratory
Idaho Falls, ID, USA
miles.mcqueen@inl.gov

***Abstract*—Selecting the optimal dimensions for various knowledge extraction applications is an essential component of data mining. Dimensionality selection techniques are utilized in classification applications to increase the classification accuracy and reduce the computational complexity. In text classification, where the dimensionality of the dataset is extremely high, dimensionality selection is even more important. This paper presents a novel, genetic algorithm based methodology, for dimensionality selection in text mining applications that utilizes information gain. The presented methodology uses information gain of each dimension to change the mutation probability of chromosomes dynamically. Since the information gain is calculated a priori, the computational complexity is not affected. The presented method was tested on a specific text classification problem and compared with conventional genetic algorithm based dimensionality selection. The results show an improvement of 3% in the true positives and 1.6% in the true negatives over conventional dimensionality selection methods.**

Keywords—Genetic Algorithms, Information Gain, Dimensionality Selection, Text mining, Vulnerability Discovery

I. INTRODUCTION

Dimensionality selection is used as an important step in knowledge extraction and data mining applications for better understanding of data [1], [2]. Proper usage of dimensionality selection methodologies can result in lower computation time and achieve higher classification accuracy in classification applications [1], [3]. These methodologies are especially useful highly multi dimensional datasets where the high dimensionality increases computation time significantly.

Typical text mining applications investigate large number of documents and extract syntactical information by means of unique words occurring in the documents [4]. This type of information extraction results in highly multi dimensional datasets with a sparse matrix. Thus dimensionality selection methodologies are employed in text mining applications to identify the optimal set of dimensions that yield the best classification results possible [5].

Dimensionality selection (feature selection) is a form of transformation of representation [6], where a set of dimensions *D*, is derived from the original set of dimensions D_0 that maximizes some criterion and is at least as good as D_0 in that criterion [7]. In classification applications the criterion is the classification accuracy [7]. Dimensionality selection has been successfully performed using genetic algorithms for text mining [8], [9], [10] and other applications [3], [8], [10]. Information Gain (IG) [11] has also been used successfully in-conjunction with genetic algorithms for dimensionality selection in classification and other data mining problems [9], [10]. However, these studies use IG as either a data pre-processing step [12] or as the fitness function of the genetic algorithm [3], [5], [13], [14].

This paper presents a novel methodology of genetic algorithm based dimensionality selection that utilizes the IG of each dimension in the dataset to calculate dynamic mutation probabilities for chromosomes. Thus the probability of selecting or deselecting a given dimension at each mutation step is dependent on the IG of that dimension. This dynamic selective mutation favors dimensions with higher IG and enables the genetic algorithm to converge to a more optimal solution faster. Furthermore, since IG of each dimension is independent from any other dimension in the dataset, IG can be calculated prior to the execution of the genetic algorithm. This leads to the computation time of presented IG based method to be the same as conventional methods. The presented methodology was tested on a text mining application for identifying software vulnerabilities using textual bug descriptions. The results showed that, compared to a conventional mutation scheme, where the mutation probability is the same for each dimension, the presented methodology was able to achieve better classification results.

The rest of the paper is organized as follows. Section II gives a brief overview of related literature. Section III details the presented methodology of dimensionality selection. Section IV describes the dataset and the text mining steps used in this paper. Section V presents experimental results and finally Section IV concludes the paper.

II. RELATED WORK

Various evolutionary approaches have been explored in knowledge discovery and classification applications in the text mining domain [10]. Knowledge discovery and information extraction from text databases have been performed

This manuscript has been authored by Battelle Energy Alliance, LLC under Contract No. DE-AC07-05ID14517 with the U. S. Department of Energy. The United States Government retains a nonexclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes.

successfully by utilizing genetic programming and other evolutionary algorithms in [15] and [16]. Text clustering has also been successfully performed using Genetic algorithms [17], [18].

Genetic algorithms and other swarm based heuristic approaches have been previously used for dimensionality selection [9], [10], [19]. In [20] genetic programming was used for dimensionality selection for classification of skewed data. Similarly, genetic programming was used in [21] for feature weighing. Other evolutionary algorithms such as Memetic algorithms [22] and particle swarm optimization [23] have been investigated as possible methods of dimensionality selection with promising results.

Several recent articles focused on dimensionality selection by utilizing IG for various different applications, including text mining [9], [10]. In [3], [5], [13] and [14] the authors used IG and information entropy as the fitness function for the genetic algorithm utilized in dimensionality selection. The authors used a genetic programming approach in [3], [13] and [14]. An ant-colony based hybrid methodology was proposed in [5]. IG was used as a pre-processing step that ranks the dimensions in [12].

The main difference in the methodology presented in the present paper is that in the present work, IG is used to dynamically control mutation probabilities. However, in previous work IG has been used as either the fitness function or as a pre-processing step.

III. Information Gain Based Dimensionality Selection using Genetic Algorithms

This section first introduces Information Gain (IG) and then details the presented dimensionality selection methodology.

A. Information Gain (IG)

The information entropy of a dataset defines the distribution of the dataset in classes. Higher information entropy describes a uniform class distribution meaning more information is required to identify each class separately. Similarly lower information entropy describes a variable class distribution and less information is required to identify each class [24]. The information entropy of a dataset S can be calculated using:

$$Entropy\ (S) = -\sum_{j=1}^{c} p_j \times \log_2 p_j \tag{1}$$

Where, c is the number of distinct classes in S and p_j is the proportion of cases in S that belong to class j [11]. Similarly the information entropy of a subset of the dataset can be calculated as:

$$Entropy(S_i) = -\sum_{j=1}^{c} q_j \times \log_2 q_j \tag{2}$$

Where, S_i is a subset of the dataset S and c is the total number of distinct classes in S and q_j is the proportion of cases in S_i that belong to class j.

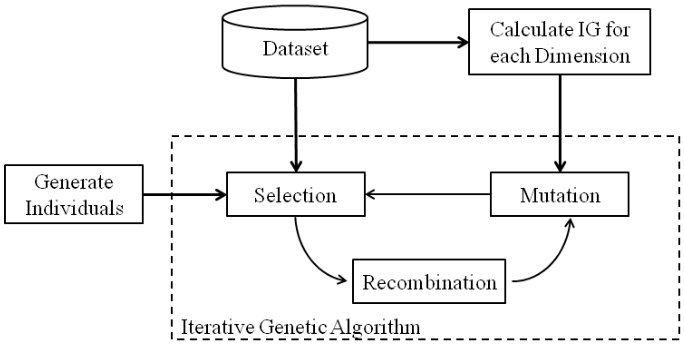

Fig. 1 IG based dimensionality selection using genetic algorithms

Furthermore, the information entropy of the dataset S can be calculated given a dimension d is known:

$$Entropy(S \mid d) = -\sum_{k=1}^{K} \frac{n_k}{N} \times Entropy(s_k) \tag{3}$$

Where, N is the total number of data points in the dataset S, and K is the number of distinct partitions caused by dimension d. n_k is the number of cases in S that belong to the partition k and s_k is the partition of data caused by k. The entropy of s_k is calculated using (2) [11].

The entropy of the dataset given that a dimension is known calculated using (3), shows the *additional* amount of information required to identify each class separately. Thus *Entropy(S)* and *Entropy(S|d)* can be used to calculate the information gained by dimension d:

$$IG(d) = Entropy(S) - Entropy(S \mid d) \tag{4}$$

Where, *IG(d)* is the information that can be gained if dimension d is known [24]. Thus, IG of any one dimension of the dataset is independent from any other dimension in the dataset.

B. IG based dimensionality selection

Typical genetic algorithm based dimensionality selection encodes the dimensions of the dataset as bits in a chromosome:

$$C_x = \{b_1, b_2, ..., b_D\} \tag{5}$$

Where, C_x is the chromosome of individual x, and D is the number of dimensions in the dataset. b_i is a bit that represents whether dimension i is selected or not. At each iteration of the genetic algorithm, the chromosome of an individual may change during recombination or mutation phases. This enables the population to evolve, and eventually reach an optimum, where the most optimum set of dimensions are selected by the individual with highest fitness.

The presented IG based dimensionality selection methodology utilizes Information Gain (IG) of each dimension to dynamically vary the mutation probability of chromosomes. The mutation probabilities are dynamically varied such that it favors the dimension with a higher IG. Since a dimension with higher IG means that more information about the class separation is gained by using the said dimension, such a selective mutation enables the genetic algorithm to reach the

TABLE I. NUMBER OF REPORTED BUGS PER YEAR IN THE REDHAT BUGZILLA [27] DATABASE

Year	Number of bug reports	Number of bugs per day
Prior to 2006	59,885	21.1
2006	15,283	41.9
2007	17,263	47.3
2008	20,916	57.3
2009	27,052	74.1
2010	43,301	118.6
2011 (to April)	19,185	139
Unknown	11	-
Total	**202.896**	**44.5**

optimal value faster. A simple block diagram of the presented methodology is shown in Fig. 1.

As shown in Section IIIA, the IG of a dimension is independent from any other dimension in the dataset. Thus IG can be calculated for each dimension prior to the execution of the genetic algorithm. Once the IG is calculated, it is normalized between 0 and 1 using:

$$IG(d_i) = \frac{IG(d_i) - IG(min)}{IG(max) - IG(min)} \qquad (6)$$

Where, *IG(min)* and *IG(max)* are minimum and maximum information gain for all the dimensions in the dataset *S*, respectively.

This calculated IG is then used to dynamically vary the mutation probability of each dimension using:

$$p(C_x, i) = (IG(d_i) \times (p_{max} - p_{min})) + p_{min} \qquad (7)$$

Or,

$$p(C_x, i) = ((1 - IG(d_i)) \times (p_{max} - p_{min})) + p_{min} \qquad (8)$$

Where, $p(C_x, i)$ is the probability that the i^{th} bit of individual C_x is mutated, and $IG(d_i)$ is the information gain of dimension *i*. p_{min} and p_{max} are preset probabilities that define the maximum mutation probability and minimum mutation probability respectively, and are set such that $p_{max} > p_{min} > 0$. If bit *i* of individual C_x is 0, meaning the dimension is currently deselected, (7) is used to calculate the mutation probability and (8) is used otherwise. Therefore, a dimension with higher information gain has a higher probability of being selected and lower probability of being deselected.

IV. TEXT MINING DATASET

This section first describes the text mining problem that the dimensionality selection methodology was applied to and then describes the text mining process that was used to extract syntactical information from the dataset.

A. *Software vulenrability identification via text mining*

It has been shown that a significant percentage of software vulnerabilities are identified as vulnerabilities, only after they have been reported as bugs [4], [25]. In other words, the true security impact of certain bugs was identified some time after they have been reported to bug databases. These bugs are known as Hidden Impact Bugs (HIBs) [4]. If these HIBs could be identified correctly as vulnerabilities, as they are being reported to bug databases, time that critical systems are vulnerable to attacks could be reduced.

It has been shown in previous work that bug reports in publically available bug databases may contain textual information that could be used to identify HIBs [4]. Thus, this paper uses a publically available bug database and a vulnerability database to classify bugs as HIBs and regular bugs using the textual description of the bug reports.

This paper focuses on Linux Kernel vulnerabilities and the MITRE CVE vulnerability database [26] was used to identify HIBs for the Linux kernel. The Redhat Bugzilla bug database [27] was used as the bug database. All the bugs and vulnerabilities explored in this paper are within the time period from January 2006 to April 2011.

The set of Linux Kernel vulnerabilities extracted from the MITRE CVE database was pruned by selecting vulnerabilities that affected 1) multiple processors, 2) multiple distributions and 3) Linux kernel 2.6 and above. This was done in order to identify vulnerabilities that are most applicable and most relevant. HIBs were identified as vulnerabilities that had at least 2 weeks of impact delay, where the impact delay was defined as the time from the public disclosure of the bug via a patch to the time a CVE was assigned to the vulnerability in the MITRE database. This means that each of the HIBs was known to the public and the Linux development team as a bug at least two weeks before the true security impact was identified.

Out of the Linux kernel vulnerabilities reported from January 2006 to April 2011 in the MITRE CVE database, 185 vulnerabilities were selected by using the rules mentioned above. Out of these 73 (39%) were selected as HIBs since they showed an impact delay of at least 2 weeks [4].

Redhat Bugzilla bug database was selected because 1) it is one of the most extensive publically available bug databases, 2) all other Bugzilla bug databases generally follow the same format, 3) most of the Linux vulnerabilities examined in this paper were associated with bugs in the Redhat Bugzilla database [4]. Although the Redhat Bugzilla database "*is not an avenue for technical assistance or support, but simply a bug tracking system*" [27], it has been shown that certain details in the bug reports can be used for various forms of classification [28], [29], [30].

As of 2011-4-30 the Redhat Bugzilla database contained 202,896 entries. Table I shows the distribution of bugs per year and the average number of bugs reported per day for each year. The “Unknown” bugs in Table I refer to bugs that were not considered due to no report date, no textual descriptions or due to denied access.

For the classification, a set of bugs that contained two classes: HIB and Regular bugs, was compiled. The HIB class contained the 73 identified HIBs mentioned above, while the regular bug class contained 6000 randomly selected bugs reported from January 2006 to April 2011. Since the number of bugs reported per year is different for each year (see Table I), the random set was constructed to reflect the proportion of

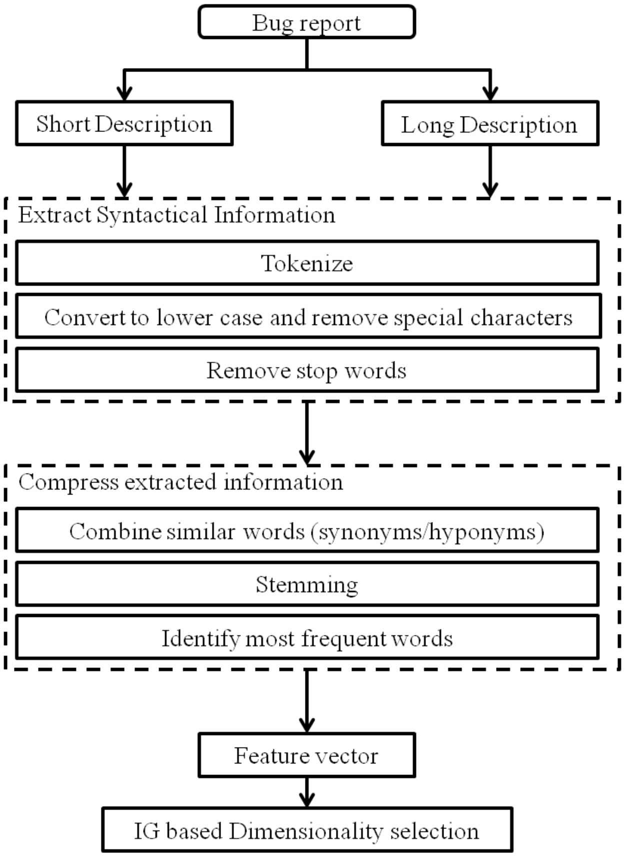
Bug report
Short Description
Long Description
Extract Syntactical Information
Tokenize
Convert to lower case and remove special characters
Remove stop words
Compress extracted information
Combine similar words (synonyms/hyponyms)
Stemming
Identify most frequent words
Feature vector
IG based Dimensionality selection

Fig. 2.Text mining process

bugs reported for each year, in order to avoid misrepresenting any year.

B. *Text mining process*

Bug reports in the Redhat Bugzilla bug database contain large amount of information such as a heading, when the bug was reported, who reported the bug, severity of the bug, version of the release affected by the bug, detailed long description of the bug, etc. However, for this application the short description which is the heading of the bug and the long description were used to extract information. The complete text mining procedure is shown in Fig. 2.

The text mining process consists of two main steps: extracting textual information and compression of extracted information (see Fig. 2). In the first step all the unique words are extracted from each document. This process is called tokenizing. Second, these words are converted to lower case and all numbers and special characters are removed. This is because numbers and special characters carry very little to no information when taken out of context. Then, most common words in the English language, known as *stop-words* are removed, since these words also carry little to no information.

In the second step of the text mining process (see Fig. 2), the extracted information is compressed. This is done by first identifying synonyms and combining them. In order to identify synonyms, the lexical database Wordnet was used [31]. Second words are deconstructed in to their basic for by using a method call Porter Stemming [32]. This also allows identification of similar words, and thus, reduction of the dimensionality. Finally, the most frequently used words in bug descriptions are identified. Since, a large number of words exist in the English language, some words may occur in a small percentage of bugs. In order to alleviate these words, only the top 500 words in the short description and the top 500 words in the long description that were used in bug reports was selected in this paper. Thus the final dimensionality of the dataset was reduced to 1000. Table II shows the dimensionality of the dataset after each step of the data mining process.

TABLE II. DIMENSIONALITY OF THE DATASET AFTER EACH STEP

Type	**Tokenization**	**Combining Synonyms**	**Combining Stemmed**	**Selecting most frequent**
Short Description	7279	4451	4005	500
Long Description	27685	20800	19113	500

TABLE III. CONFUSION MATRIX

		Classified as	
		HIB	***Normal***
Actual Class	***HIB***	True Positives (TP)	False Negatives (FN)
	Normal	False Positives (FP)	True Negatives (TN)

TABLE IV. CLASSIFICATION RESULTS FOR EACH METHOD

Parameter	**No Dimensionality Selection**	**Conventional Dimensionality Selection**	**IG Based Dimensionality Selection**
TP Rate	0.671	0.881	**0.912**
TN Rate	0.908	0.915	**0.931**
BD Rate	0.085	0.116	**0.143**

V. EXPERIMENTAL RESULTS

In order to identify the effectiveness of the presented IG based dimensionality selection methodology, it was applied to the text classification problem detailed in Section IV. Furthermore, the presented dimensionality selection methodology was compared to a conventional dimensionality selection methodology.

The presented methodology was compared to conventional genetic dimensionality selection by using a generational genetic algorithm. This type of genetic algorithm uses recombination as part of the evolutionary process, along with mutation. Both the genetic algorithms were tested using 50 individuals, with a tournament size of 10. The minimum and maximum mutation probabilities (p_{min} , p_{max}) for the presented IG based method were set at 5% and 10% respectively while the mutation probability of the conventional genetic algorithm was set at 10%. The classification accuracy using a Naïve Bayes Multinomial classifier with 10 fold cross validation was used as the fitness function of both genetic algorithms. The genetic algorithms were run for 200 iterations. Each method

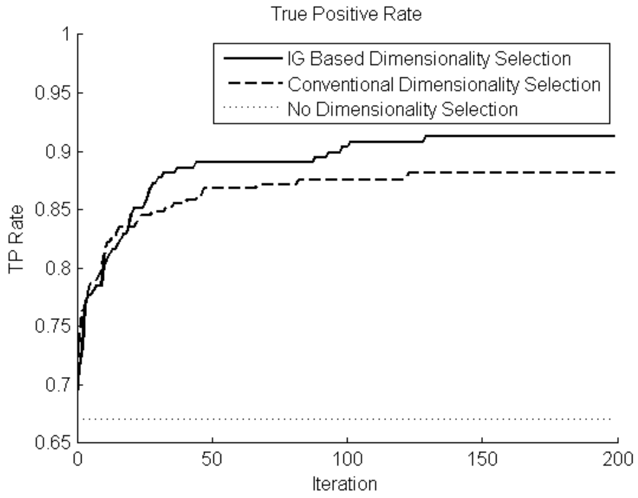

Fig. 3.True positive rate for each iteration

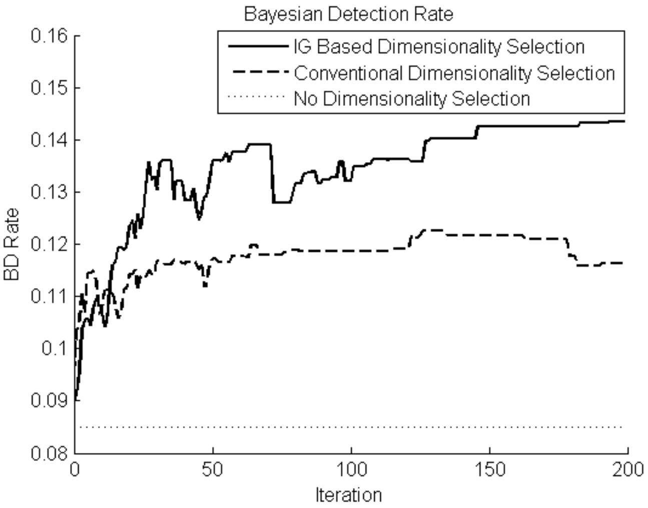

Fig. 5.Bayesian detection rate for each iteration

was executed 10 times with a different starting population and the final results were averaged.

The classification results are shown in True Positive (TP), True Negative (TN) and Bayesian Detection (BD) rates:

$$TP\ Rate = \frac{TP}{(TP + FN)} \tag{9}$$

$$TN\ Rate = \frac{TN}{(TN + FP)} \tag{10}$$

$$BD\,Rate = \frac{TP}{(TP + FP)} \tag{11}$$

Where TP, FN, TN, and FP are true positives, false negatives, true negatives and false positives, respectively. Table III shows the confusion matrix from which each can be identified. The Bayesian Detection rate defines what percentage of bugs that were classified by the classifier as HIBs are actually vulnerabilities.

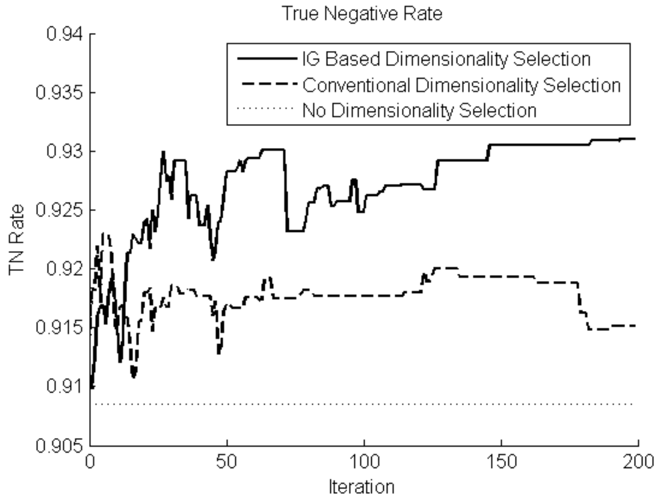

Fig. 4.True negative rate for each iteration

Table IV shows the final classification results with no dimensionality selection, conventional dimensionality selection and the presented IG based dimensionality selection. Figures 3, 4 and 5 plot the TP rate, TN rate and the BD rate for each method respectively. Both dimensionality selection methods performed better than when the full 1000 dimensions are used. The presented IG based method shows more than 3% improvement over the conventional dimensionality selection method for TP rate and the BD rate and a improvement of 1.6% for TN rate.

VI. CONCLUSION

This paper presented a novel Information Gain (IG) based dimensionality selection methodology for text mining applications using genetic algorithms. The presented methodology dynamically varies mutation probability of bits in the chromosome according to the IG of each dimension.

The presented methodology was tested using a software vulnerability detection method that utilizes textual information of bugs in publically available bug databases. The presented methodology was applied to this text mining problem and compared with a conventional genetic algorithm with static mutation probabilities. The results show an increase of 3% for the true positives and the Bayesian detection rate and an increase of 1.6% for the true negatives in 200 iterations.

As future work, the usability of the presented methodology will be investigated on different applications. Furthermore, the effects of different sized populations with different minimum and maximum mutation probabilities will be explored. Methodologies where the recombination can be affected by data driven features such as IG will also be explored in the future.

REFERENCES

[1] M. L. Raymer, W. L. Punch, E. D. Goodman, L. A. Kuhn, A. K. Jain, "Dimensionality reduction using genetic algorithms," in IEEE Transactions on Evolutionary Computation, vol. 4, no. 2, pp. 164-171, Jul 2000.

[2] R. Battiti, "Using mutual information for selecting features in supervised neural net learning," in IEEE Transactions on Neural Networks, vol. 5, no. 4, pp. 537-550, Jul 1994.

[3] F. E. B. Otero, M. M. S. Silva, A. A. Freitas, J. C. Nievola, "Genetic Programming for Attribute Construction in Data Mining," in Proc. of 6th European Conference on Genetic Programing, pp. 384-393, Apr 2003.

[4] D. Wijayasekara, M. Manic, J. L. Wright, M. McQueen "Mining Bug Databases for Unidentified Software Vulnerabilities," in Proc. of IEEE Intl. Conference on Human System Interaction, Jun 2012.

[5] M. E. Basiri, S. Nemati, "A novel hybrid ACO-GA algorithm for text feature selection," in Proc. of IEEE Congress on Evolutionary Computation, pp. 2561-2568, May 2009.

[6] H. Liu and H. Motoda, Feature Extraction, Construction and Selection: A Data Mining Perspectiv (The Springer International Series in Engineering and Computer Science Series, 453). Berlin, Germany: Springer-Verlag, 1998.

[7] A. Jain, D. Zongker, "Feature selection: evaluation, application, and small sample performance," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 2, pp. 153-158, Feb 1997.

[8] P. G. Espejo, S. Ventura, F. Herrera, "A Survey on the Application of Genetic Programming to Classification," in IEEE Transactions on Systems, Man, and Cybernetics, vol. 40, no. 2, pp. 121-144, Mar 2010.

[9] F. Sebastiani, C. N. D. Ricerche, "Machine learning in automated text categorization," in ACM Computing Surveys, vol. 34, pp. 1-47, 2002.

[10] Y. Yang, J. O. Pedersen, "A comparative study on feature selection in text categorization," in Proc. of the 14th International Conference on Machine Learning, pp. 412–420, 1997.

[11] C. E. Shannon, "A Mathematical Theory of Communication," in The Bell System Technical Journal, Vol. 27, pp. 379–423, July, 1948.

[12] H. Uguz, "A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm," in Knowledge-Based Systems, vol. 24, no. 7, pp. 1024-1032, Oct. 2011.

[13] K. Neshatian, M. Zhang, "Genetic programming and class-wise orthogonal transformation for dimension reduction in classification problems," in Proc. of the 11th European Conference on Genetic Programing, Mar. 2008.

[14] M. Muharram and G. D. Smith, "Evolutionary constructive induction," in IEEE Transactions on Knowledge and Data Engineering, vol. 17, no. 11, pp. 1518–1528, Nov. 2005.

[15] J. Atkinson-Abutridy, C. Mellish, S. Aitken, "A semantically guided and domain-independent evolutionary model for knowledge discovery from texts," in IEEE Transactions on Evolutionary Computation, vol. 7, no. 6, pp. 546-560, Dec. 2003.

[16] J. Atkinson-Abutridy, C. Mellish, S. Aitken, "Combining information extraction with genetic algorithms for text mining," in IEEE Intelligent Systems, vol. 19, no. 3, pp. 22-30, May 2004.

[17] M. Gao, B. Wang, "Text clustering ensemble based on genetic algorithms," in Proc of 2012 International Conference on Systems and Informatics, pp. 2329-2332, May 2012.

[18] W. Song, S. C. Park, "Genetic algorithm for text clustering based on latent semantic indexing," in Computers & Mathematics with Applications, vol. 57, no. 12, pp. 1901-1907, Jun 2009.

[19] C. Tsai, W. Eberle, C. Chu, "Genetic algorithms in feature and instance selection," in Knowledge-Based Systems, vol. 39, pp. 240-247, Feb. 2013.

[20] I. Sandin, G. Andrade, F. Viegas, D. Madeira, L. Rocha, T. Salles, M. Goncalves, "Aggressive and effective feature selection using genetic programming," in Proc of IEEE Congress on Evolutionary Computation, pp. 1-8, Jun 2012.

[21] A. Friedlander, K. Neshatian, M. Zhang, "Meta-learning and feature ranking using genetic programming for classification: Variable terminal weighting," in Proc of IEEE Congress on Evolutionary Computation, pp. 941-948, Jun 2011.

[22] Y. Wen, H. Xu, "A cooperative coevolution-based pittsburgh learning classifier system embedded with memetic feature selection," in Proc of IEEE Congress on Evolutionary Computation, pp. 2415-2422, Jun 2011.

[23] A. S. J. Tjiong, S. T. Monteiro, "Feature selection with PSO and kernel methods for hyperspectral classification," in Proc of IEEE Congress on Evolutionary Computation, pp. 1762-1769, Jun 2011.

[24] L. Wenke X. Dong, "Information-theoretic measures for anomaly detection," in Proc of IEEE Symposium on Security and Privacy, pp.130-143, 2001.

[25] J. Arnold, T. Abbott, W. Daher, G. Price, N. Elhage, G. Thomas, A. Kaseorg, "Security Impact Ratings Considered Harmful," in Proc. of the 12th Conf. on Hot Topics in Operating Systems, USENIX, May 2009

[26] The MITRE Corporation (10 Feb. 2012), *Common Vulnerabilities and Exposures (CVE)* [Online]. Available: http://cve.mitre.org/

[27] Redhat, Inc. (10 Feb. 2012), *Redhat Bugzilla Main Page* [Online]. Available: https://bugzilla.redhat.com/.

[28] A. J. Ko, B. A. Myers, D. H. Chau, "A Linguistic Analysis of How People Describe Software Problems," in Proc. of IEEE Symp. on Visual Languages and Human-Centric Computing, pp. 127–134, Sep. 2006.

[29] A. Lamkanfi, S. Demeyer, E. Giger, B. Goethals, "Predicting the severity of a reported bug," in Proc. of the 7th IEEE Working Conf. on Mining Software Repositories, pp.1–10, May 2010.

[30] A. Lamkanfi, S. Demeyer, Q. D. Soetens, T. Verdonck, "Comparing Mining Algorithms for Predicting the Severity of a Reported Bug," in Proc. of the 15th European Conf. on Software Maintenance and Reengineering, pp.249–258, Mar. 2011.

[31] C. Fellbaum, *WordNet: An Electronic Lexical Database*, Cambridge, MA: MIT Press, 1998.

[32] M. F. Porter, "An algorithm for suffix stripping," in *Program*, vol. 14, no. 3, pp. 130−137, 1980.