Journal of Intelligent Learning Systems and Applications, **2013, 5, 108-114**
http://dx.doi.org/10.4236/jilsa.2013.52012 Published Online May 2013 (http://www.scirp.org/journal/jilsa)

Automatic Classification of Unstructured Blog Text

Mita K. Dalal[1], Mukesh A. Zaveri[2]

[1]Information Technology Department, Sarvajanik College of Engineering & Technology, Surat, India; [2]Computer Engineering Department, S. V. National Institute of Technology, Surat, India.
Email: parikhmita@gmail.com, mazaveri@coed.svnit.ac.in

Received December 14th, 2012; revised February 16th, 2013; accepted February 24th, 2013

ABSTRACT

Automatic classification of blog entries is generally treated as a semi-supervised machine learning task, in which the blog entries are automatically assigned to one of a set of pre-defined classes based on the features extracted from their textual content. This paper attempts automatic classification of unstructured blog entries by following pre-processing steps like tokenization, stop-word elimination and stemming; statistical techniques for feature set extraction, and feature set enhancement using semantic resources followed by modeling using two alternative machine learning models—the naïve Bayesian model and the artificial neural network model. Empirical evaluations indicate that this multi-step classification approach has resulted in good overall classification accuracy over unstructured blog text datasets with both machine learning model alternatives. However, the naïve Bayesian classification model clearly out-performs the ANN based classification model when a smaller feature-set is available which is usually the case when a blog topic is recent and the number of training datasets available is restricted.

Keywords: Automatic Blog Text Classification; Feature Extraction; Machine Learning Models; Semi-Supervised Learning

1. Introduction

Automatic classification of blog entries is generally treated as a semi-supervised machine learning task, in which the blog entries are automatically assigned to one of a set of pre-defined classes based on the features extracted from their textual content. Usually this task involves several subtasks in natural language processing like tokenization, stop-word removal, stemming and spell-error correction followed by feature set construction, modeling using an appropriate machine learning technique and finally, classification using the trained model.

Blogging is a popular way of communicating, information sharing and opining on the Internet. There are blogs devoted to sports, politics, technology, education, movies, finance etc. Popular blogs have millions of visitors annually, so they are also important platforms for mining consumer preferences and targeted advertisement. Most of the content posted on blogs is textual and unstructured. Classifying blog text is a challenging task because blog posts and readers' comments on them are usually short, frequently contain grammatical errors and make use of domain-specific abbreviations and slang terms which do not match dictionary words. They are also punctuated inappropriately making tokenization and parsing using automated tools more difficult. The blog posts of Internet users are organized in one of three ways [1]—1) Pre-classified; 2) Semi-classified; or 3) Un-classified. These three categories are briefly explained next.

1) Pre-classified—Pre-classified blogs have separate web-pages allocated to each sub-class, so that the content posted is automatically sorted. For example, a blog that posts updates on computer technology could have previously allocated pages for categories like "hardware", "software", "outsourcing", "jobs" etc.

2) Semi-classified—Semi-classified blogs are those which have some web-pages pre-classified exclusively for popular categories, while the rest of the posts appear as mixed-bag. For example, a sports blog might contain separate web-pages for popular sports which are often commented upon, while posts on less popular sports appear as a jumble, often simply referred to as the category "Others".

3) Un-classified—Un-classified blogs contain no fine-grained classification and allow all blog postings to ap-

pear in an ad-hoc manner. For example, an amateur's movie blog could contain posts on movies, music, actors, viewers' opinions and replies to other bloggers etc. all on one page.

Most blogs fall into the semi-classified or un-classified category. So, application of automatic text classification techniques for the long term content management of these blogs has generated interest. Machine learning techniques like naïve Bayesian [1-3], Artificial Neural Networks [4], Support Vector Machines [5,6] as well techniques that combine various machine learning methods [7,8] have been used by researchers for automatic text classification. In addition to content management, classification and summarization of blog text data has several important applications such as—product review mining [9], political sentiment mining [10], mining movie reviews [11] and content-analysis for strategic display of online advertisements.

The rest of the paper is organized as follows. Section 2 describes some of the major issues in text classification and the methods adopted in literature to solve these issues. Section 3 gives the design of our unstructured blog text classifier and explains in detail, the various phases involved such as the Pre-processing Phase, Feature Extraction and Enhancement Phase, Classifier Modeling Phase using two machine learning techniques and the Evaluation Phase. Section 4 discusses our implementation of the classification strategy and the performance analysis based on experimentation. Finally, Section 5 gives conclusion and pointer to future work.

2. Related Work

This section describes some of the major issues in automatic text classification and provides a brief pointer to related work done in dealing with these issues. The major issues discussed are: pre-processing raw text for dimensionality reduction, extracting the word feature set useful for classification and handling difficulties caused by synonyms and polysemes in text classification.

2.1. Pre-Processing of Raw Text for Dimensionality Reduction

Unstructured text needs to be pre-processed before features can be extracted from it. In text processing, the features are usually the words of the text itself. Since natural languages have a large vocabulary, pre-processing steps need to be performed to reduce the amount of term matching involved. The pre-processing commonly performed on text for dimensionality reduction involves stop-word removal [1-3] and stemming [12].

Stop-words are functional words which occur frequently in the vocabulary of a language and are not indicative of any particular class of documents, hence, not useful in classification. Words like "the", "is", "in", "or", "it", "for" etc. are stop-words in English. Removing stop-words reduces the size of the text to be processed by the classification algorithm.

Stemming reduces a word to its root or base form and thus reduces the number of word features to be processed. Stemming is based on the observation that words with common stems usually have similar meanings [12], so they can be treated as a common token. For example, words like "directed", "directing", "direction", "directions", "directs", "director" etc. can all be conflated to the same root word "direct" by stripping suffixes like -ed, -ing, -ion, -ions, -s and -or respectively. Performing stemming significantly reduces the size of the data to be processed.

In addition, unstructured text, especially blog posts often contains spell-errors, incorrect punctuations, abbreviations, special characters which are non-text etc. Spell-errors can be located and corrected using document processing tools. Domain-specific semantic resources generated over time [1] can be used to identify common abbreviations and achieve fine-grained classification.

2.2. Identifying Features for Text Classification

The most significant words or the words with highest discriminatory power need to be identified as features for classification. This is usually performed using statistical and semi-semantic techniques. Well-known feature extraction techniques include TF-IDF [13-15], LSI [15-17] and Multi-words [15,18].

TF-IDF is an acronym for term frequency-inverse document frequency [13,14]. It is a probability-based statistical measure to determine the significance of a word feature in a text document corpus. It is based on the heuristic that a term is a good discriminator if it occurs frequently in a document but does not occur in many distinct documents of the corpus. LSI is an acronym for Latent Semantic Indexing [15,16]. It uses term-document matrix manipulations and singular value decomposition to derive association between feature terms and text documents. It also attempts to address the problems of synonymy and polysemy to some extent [16]. This method is computationally expensive and more suitable for query-oriented search.

Multi-words [15,18] are an ordered sequence of words that are more semantically indicative of a domain than if the words in the group are taken individually.

2.3. Handling "Synonymy" and "Polysemy"

"Synonymy" and "Polysemy" are two frequently occurring problems in text classification. "Synonymy" is the capacity for different words to have the same meaning. For example, "wealthy" and "rich" are synonyms. A rele-

vant document could be omitted during key-word based retrieval or misclassified if it uses a synonym of the feature term instead of the exact same feature term. "Polysemy" is the capacity for the same word to have different senses. For example, the word "mouse" could mean a "creature of rodent family" or an "input device to computer". A polyseme of a feature term could cause an irrelevant document to be retrieved during search or misclassification a text document into a non-relevant category. During the task of performing text classification using word features, the problem of "synonymy" can be solved by using a thesaurus or an online lexical database like WordNet [19,20]. "Polysemy" is more difficult to deal with, however techniques for word sense disambiguation [21] have been partially successful in handling this issue.

3. Automatic Classification of Blog Entries Using Machine Learning Techniques

This section discusses the classification strategy implemented by us for the classification of unstructured blog posts. This is depicted in **Figure 1**. The dataset used for experimentation consisted of a variety of blog entries of type *Sports*, *Computer Technology* and *Environment* as indicated in **Table 1**. Regardless of the machine learning technique used, the automatic classification of blog text has four major phases—1) Pre-Processing Phase; 2) Feature Extraction and Enrichment Phase; 3) Classifier Modeling and Training Phase; and 4) Evaluation Phase. These phases are briefly described next.

3.1. Pre-Processing Phase

The pre-processing steps performed on the blog text were tokenization, stop-word elimination, stemming and spell-error correction as explained in the previous section. Individual words occurring in the blog text were treated as tokens. Whitespace, special characters and punctuations were eliminated during tokenization. The tokens were matched with the words in a pre-compiled stop-word list and eliminated to reduce overhead. We also performed stemming using Porter's suffix stripping algorithm [12]. It has been proved empirically that Porter's stemming algorithm reduces the size of the English vocabulary by approximately one-third [12]. Spell-errors were corrected using a standard word processor.

3.2. Feature Extraction and Feature Enrichment Phase

In text classification, features are extracted from the words of the text, which in our case are the blog posts.

Even after stop-word elimination and stemming performed in the previous phase, the vocabulary of the blog posts from hundreds of users is still very large. All these words are not useful in classification. In order to extract significant word features, we use the well-known statistical measure tf-idf [13-15] which is an acronym for "term frequency-inverse document frequency". In our case a document means a single blog post. We sorted features based on their tf-idf values and extracted the top ranking 30% words to be used as discriminators for unstructured blog text classification. We also extracted frequently occurring multi-words [15,18] to be used for classification. As explained in the previous section, multi-words are group of words which form a semantic sense. For example, a sports blog post referring to *cricket* might contain multi-words like "batting average" or "caught

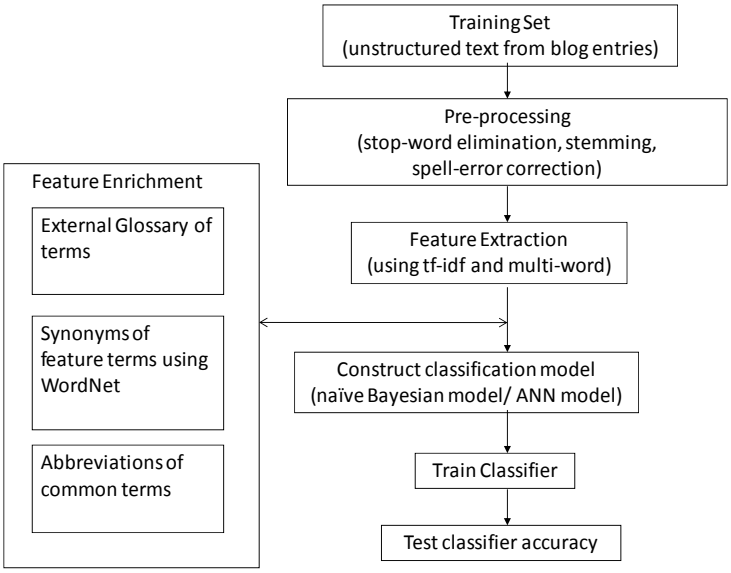

Figure 1. Strategy for automatic classification of unstructured blog text.

Table 1. Dataset description.

Sr. No.	Category	Sub-Categories
1	Sports	Cricket Field Hockey Tennis
2	Computer Technology	Social Media Gaming Outsourcing
3	Environment	Global Warming Pollution Wildlife

Table 2. Partial list of acronyms for "cricket".

Acronym	Expanded Form
BCCI	Board of Control for Cricket in India
ICC	International Cricket Council
LBW	Leg before Wicket
ODI	One Day International
T20	Twenty 20

behind" while a blog post referring to *tennis* might contain multi-words like "forced error" or "match point".

We enriched the feature set in two ways. Firstly, we added synonyms of the single word features using the online lexical resource Word Net 2.1 (available at <http://wordnet.princeton.edu>). Secondly, we added words and acronyms from online external glossaries of terms wherever available. For example, **Table 2** shows the partial list of compiled acronyms for the sport "*Cricket*". Thus, while feature-extraction is done in a completely automatic way using statistical measures, the feature-enrichment step requires human supervision, which makes our approach semi-supervised.

However, it is important to note that the feature- enrichment step is optional, and classification can still be performed using the overall multi-step strategy without adding additional features from semantic resources or external glossaries. Feature set extraction and enhancement was followed by the classifier modeling and training phase which is described in the next phase.

3.3. Classifier Modeling and Training Phase

We treat text classification as an application of supervised machine learning. The blog posts in the testing set were represented as binary feature vectors using the Multivariate Bernoulli model [2,3]. This model simply indicates the presence/absence of each feature term. For example, if there are n features extracted in the feature extraction phase, then, a blog post entry "e" would be internally represented as an ordered sequence, $e = (i_1, i_2, i_3, \cdots, i_n)$ where each "i_k" is a binary variable indicating "1" for presence of feature term and "0" for absence. The feature terms are the words and multi-words with high tf-idf values extracted in the Feature Extraction Phase.

We used two well-known machine learning models to classify unstructured blog text data—1) Naïve Bayesian Model; and 2) Artificial Neural Network Model. These models are briefly described next.

3.3.1. Naïve Bayesian Model

The naïve Bayesian model is a probabilistic approach to classification. It is based on the simplifying assumption of conditional independence among attributes [22]. Given a training set containing *attribute values* and corresponding *target values* (*classes*), the naïve Bayesian classifier predicts the class of an unseen (new) instance, based on previously observed probabilities of the feature terms occurring in that instance. Let C indicate the set of pre-defined classes to which the blog post may belong. Let B indicate the training set of pre-processed blog posts, while B_c is a pre-labeled subset of B that contains blog posts of some class $c \in C$. Let F be the final feature set generated during the Feature Extraction and Enrichment Phase. The probabilities of occurrence each of the features in the feature set F for each class, was computed by making one pass over the blog training set. First, the naïve Bayesian classifier [1,22] computes the prior probabilities of each class $c \in C$ as indicated by Equation (1).

for each class $c \in C$ *do*

$$P(c) = \frac{|B_c|}{|B|} \tag{1}$$

In order to classify a new blog entry e, the probability of it belonging to each class is predicted as shown in Equation (2). In Equation (2) the $P(f_i/c)$ terms indicate the statistical probability of occurrence of the i^{th} feature term in a blog entry of category c.

$$p_e(c) = P(c)\prod_{i=1}^{|F|} P(f_i/c) \tag{2}$$

The blog post is then assigned to the class with the highest probability, as indicated by Equation (3).

$$maxprob = \arg\max_{c \in C}\left(p_e(c)\right) \tag{3}$$

We empirically obtained good average classification accuracy of over 87% using the naïve Bayesian classification model.

3.3.2. Artificial Neural Network Model

We developed our ANN based blog text classification model using Matlab version 7.8 (R2009a). We generated a 3-layer feed-forward neural network trained using the "Backpropagation" algorithm. The input layer had the

same number of neurons as the number of features. The input was a binary vector indicating the absence/presence of feature terms in the training document. The hidden layer neurons used the *sigmoid* activation function. The number of neurons in the output layer depends on the number of output classes. For example, if the output layer has n neurons, it can represent upto 2^n output classes (00, 01, 10 and 11). So, if a blog is to be classified into 4 sub-categories (e.g. Sports blog posts are classified as: *Cricket*, *Hockey*, *Tennis* or *Other*), then 2 output neurons are sufficient. We empirically obtained good average classification accuracy of over 85% using the basic ANN classification model with a sufficiently large feature set.

Since both the naïve Bayesian and the ANN based models used the same feature set, it is interesting to compare their classification accuracies. This comparison has been performed in the next phase.

3.4. Evaluation Phase

We performed pre-processing and feature set generation over 3 categories of blogs with 3 sub-categories under each type. These categories have been listed in **Table 1**. We evaluated the classification accuracy of both the naïve Bayesian and ANN models in terms of precision, recall and f-measure over our blog dataset. We used 70% of the dataset for training a model, and 30% for testing, as this has been known to give good classification accuracy [1]. We repeated our experiments with varying sizes of feature-set. The empirical evaluation is explained in Section 4, and the results obtained have been summarized in **Table 3**.

4. Empirical Evaluation and Results

We implemented our classifier for blog datasets as per the strategy outlined in Section 3. The training corpus consisted of 3000 blog posts and comments (about 1000 posts from each of the three categories Sports, Computer Technology and Environment) collected from several popular blogs on the Internet using a web-crawler. The categories and sub-categories are as indicated in **Table 1**. First, we applied pre-processing as explained in Section 3.1. Then, we extracted top-ranking features using our implementation of the tf-idf statistical measure and also added multi-words to the feature set. We also enhanced the feature-set as explained in Section 3.2. After this we performed final classification using both alternatives—the naïve Bayesian and ANN based machine learning models. Our aim is to find out which model works comparatively better after identical pre-processing and given same feature-set.

We tested our classification strategy empirically on the three datasets shown in **Table 1**. We recorded classification accuracy using the naïve Bayesian and ANN models with varying sizes of feature set and tabulated them as shown in **Table 3**. Testing using different feature-set sizes is important because blog datasets are generated incrementally as users add their observations and comments over time. So, the feature set also becomes available gradually. Thus, blog data tends to collect incrementally, and blog posts classification requires a machine learning technique that could efficiently perform batch-incremental classification.

The results displayed in **Table 3** clearly indicate that for performing the task of automatic classification of unstructured blog text the naïve Bayesian classifier gives better overall classification accuracy than basic neural-network based classification. This is especially evident when the feature-set size is restricted, as is usually the case during the initial stage of launching of a new blog or when a blog topic is relatively new.

Table 3. Classification accuracy over blog datasets.

FS Size	Category of Blog Post	Naïve Bayesian Model			ANN Model		
		Avg. Precision	*Avg. Recall*	*Avg. F-Measure*	*Avg. Precision*	*Avg. Recall*	*Avg. F-Measure*
10%	Sports	0.8483	0.8134	0.8304	0.7520	0.7484	0.7501
	Computer Technology	0.8444	0.8396	0.8419	0.7210	0.7112	0.7160
	Environment	0.8224	0.8210	0.8216	0.7682	0.7488	0.7583
20%	Sports	0.8630	0.8598	0.8613	0.8498	0.8254	0.8374
	Computer Technology	0.8626	0.8630	0.8628	0.8624	0.8622	0.8623
	Environment	0.8520	0.8444	0.8481	0.8462	0.8450	0.8456
30%	Sports	0.8820	0.8792	0.8805	0.8676	0.8636	0.8655
	Computer Technology	0.8810	0.8781	0.8795	0.8669	0.8666	0.8667
	Environment	0.8740	0.8672	0.8705	0.8555	0.8542	0.8548

Empirical results shown in **Table 3** indicate that when the feature-set size is restricted to 10%, the naive Bayesian model records an F-measure value of over 82% for all three categories of blog posts, while the ANN model shows a comparatively poor F-measure value of 71% to 75%. When a sufficiently large feature set size of top ranking 30% words is used, the naïve Bayesian model shows only marginal improvement over the ANN model.

5. Conclusion and Future Work

We attempted automatic classification of unstructured blog posts using a semi-supervised machine learning approach. Empirical studies indicate that the multi-step classification strategy outlined can classify blog text with good accuracy. We confirmed that the combination of tf-idf and multi-word heuristics is an effective statistical feature-set extractor for blog entries. Moreover, our empirical results indicate that the naïve Bayesian classification model clearly out-performs the basic ANN based classification model for highly domain-dependent unstructured blog text classification especially when a restricted feature-set is available. However, we would like to repeat our experiments with larger and more varied datasets. We would also like to investigate the effect of changing neural network configuration on blog text classification accuracy.

REFERENCES

[1] M. K. Dalal and M. A. Zaveri, "Automatic Text Classification of Sports Blog Data," *Proceedings of the IEEE International Conference on Computing, Communications and Applications* (*ComComAp* 2012), Hong Kong, 11-13 January 2012, pp. 219-222.

[2] S. Kim, K. Han, H. Rim and S. H. Myaeng, "Some Effective Techniques for Naïve Bayes Text Classification," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 18, No. 11, 2006, pp. 1457-1466. doi:10.1109/TKDE.2006.180

[3] M. J. Meena and K. R. Chandran, "Naïve Bayes Text Classification with Positive Features Selected by Statistical Method," *Proceedings of the IEEE International Conference on Advanced Computing*, Chennai, 13-15 December 2009, pp. 28-33. doi:10.1109/ICADVC.2009.5378273

[4] Z. Wang, Y. He and M. Jiang, "A Comparison among Three Neural Networks for Text Classification," *Proceedings of the IEEE* 8*th International Conference on Signal Processing*, Beijing, 16-20 November 2006, pp. 1883-1886. doi:10.1109/ICOSP.2006.345923

[5] Z. Wang, X. Sun, D. Zhang and X. Li, "An Optimal SVM-Based Text Classification Algorithm," *Proceedings of the IEEE* 5*th International Conference on Machine Learning and Cybernetics*, Dalian, 13-16 August 2006, pp. 1378-1381. doi:10.1109/ICMLC.2006.258708

[6] M. Zhang and D. Zhang, "Trained SVMs Based Rules Extraction Method for Text Classification," *Proceedings of the IEEE International Symposium on IT in Medicine and Education*, Xiamen, 12-14 December 2008, pp. 16-19. doi:10.1109/ITME.2008.4743814

[7] R. D. Goyal, "Knowledge Based Neural Network for Text Classification," *Proceedings of the IEEE International Conference on Granular Computing*, Fremont, 2-4 December 2007, pp. 542-547. doi:10.1109/GrC.2007.108

[8] D. Isa, L. H. Lee, V. P. Kallimani and R. RajKumar, "Text Document Preprocessing with the Bayes Formula for Classification Using Support Vector Machine," *Proceedings of the IEEE Transactions on Knowledge and Data Engineering*, Vol. 20, No. 9, 2008, pp. 1264-1272.

[9] J. Polpinij and A. K. Ghose, "An Ontology-Based Sentiment Classification Methodology for Online Consumer Reviews," *Proceedings of the IEEE International Conference on Web Intelligence and Intelligent Agent Technology*, Sydney, 9-12 December 2008, pp. 518-524. doi:10.1109/WIIAT.2008.68

[10] K. T. Durant and M. D. Smith, "Predicting the Political Sentiment of Web Log Posts Using Supervised Machine Learning Techniques Coupled with Feature Selection," *Lecture Notes in Computer Science*, Vol. 4811, 2007, pp. 187-206. doi:10.1007/978-3-540-77485-3_11

[11] L. Zhao and C. Li, "Ontology Based Opinion Mining for Movie Reviews," *Lecture Notes in Computer Science*, Vol. 5914, 2009, pp. 204-214. doi:10.1007/978-3-642-10488-6_22

[12] M. F. Porter, "An Algorithm for Suffix Stripping," *Program*, Vol. 14, No. 3, 1980, pp. 130-137.

[13] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation*, Vol. 60, No. 5, 2004, pp. 493-502. doi:10.1108/00220410410560573

[14] K. S. Jones, "IDF Term Weighting and IR Research Lessons," *Journal of Documentation*, Vol. 60, No. 5, 2004, pp. 521-523. doi:10.1108/00220410410560591

[15] W. Zhang, T. Yoshida and X. Tang, "TF-IDF, LSI and Multi-Word in Information Retrieval and Text Categorization," *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, Singapore City, 12-15 October 2008, pp. 108-113. doi:10.1109/ICSMC.2008.4811259

[16] S. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas and R. Harshman, "Indexing by Latent Semantic Analysis," *Journal of American Society of Information Science*, Vol. 41, No. 6, 1990, pp. 391-407. doi:10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9

[17] T. Liu, Z. Chen, B. Zhang, W. Ma and G. Wu, "Improving Text Classification Using Local Latent Semantic Indexing," *Proceedings of the* 4*th IEEE International Conference on Data Mining*, Brighton, 1-4 November 2004, pp. 162-169. doi:10.1109/ICDM.2004.10096

[18] K. W. Church and P. Hanks, "Word Association Norms, Mutual Information and Lexicography," *Computational Linguistics*, Vol. 16, No. 1, 1990, pp. 22-29.

[19] G. A. Miller, "WordNet: A Lexical Database for Eng-

lish," *Communications of the ACM*, Vol. 38, No. 11, 1995, pp. 39-41. doi:10.1145/219717.219748

[20] E. H. Hovy and C.-Y. Lin, "Automated Text Summarization in SUMMARIST," In: I. Mani and M. Maybury, Eds., *Advances in Automatic Text Summarization*, MIT Press, Cambridge, 1999, pp. 81-94.

[21] S. Vàzquez, A. Montoyo and G. Rigau, "Using Relevant Domains Resource for Word Sense Disambiguation," *International Conference on Artificial Intelligence* (*IC-AI*'04), Las Vegas, 21-24 June 2004, pp. 784-789.

[22] T. M. Mitchell, "Chapter 6: Bayesian Learning," Machine Learning, McGraw-Hill International Editions, New York, 1997, pp. 154-199.

