ISSN:0975-9646

A Survey of Text Categorization and its Various Approaches

M. Maharasi MCA, M.phil, M.E., (1), N. Antony Sophia M.E., (2)

[1,2] *Assistant Professor, Department of Computer Applications,*
Dr. Sivanthi Aditanar College of Engineering
Tiruchendur - 628215,
Tuticorin Dist, India

***Abstract*— Text categorization is a task of automatically sorting a set of documents into categories from a predefined set. Text categorization also known as text classification. This task has several applications, including automated indexing of scientific articles according to predefined thesauri of technical terms, filing patents into patent directories, spam filtering, identification of document genre etc. In this paper we discuss several techniques of text categorization.**

***Keywords*— Text mining, text classification, feature selection**

I. INTRODUCTION

Text categorization (TC – also known as text classification, or topic spotting) is the task of automatically sorting a set of documents into categories (or classes, or topics) from a predefined set. This task, includes information retrieval (IR) and machine learning (ML), has witnessed an interest in the last ten years from researchers and developers. The capacity of storing data becomes enormous as the technology of computer hardware develops. So amount of the information required by the users become varies actually user's deal with textual data more than the numerical data. It is very difficult to apply techniques of data mining to textual data instead of numerical data. Therefore it becomes necessary to develop techniques applied to textual data that are different from the numerical data. Instead of numerical data the mining of the textual data is called text mining. Text mining is procedure of synthesizing the information by analyzing relations, the patterns and rules from the textual data. A key element is the linking together of the extracted information together to form new facts or new hypotheses to be explored further by more conventional means of experimentation. Text mining is different from what are familiar with in web search. In search, the user is typically looking for something that is already known and has been written by someone else. In text mining, the goal is to discover unknown information, something that no one yet knows and so could not have yet written down. Text categorization (or text classification) is the assignment of natural language documents to predefined categories according to their content Text classification is the act of dividing a set of input documents into two or more classes where each document can be said to belong to one or multiple classes. Huge growth of information flows and especially the explosive growth of Internet promoted growth of automated text classification. a general inductive process automatically builds a classifier by learning, from a set of pre classified documents, the characteristics of the categories. Text classification is commonly used to handle spam emails, classify large text collections into topical categories, used to manage knowledge and also to help Internet search engines. The accuracy of modern text classification systems rivals that of trained human professionals, a combination of information retrieval (IR) technology and machine learning (ML) technology. This chapter will outline the fundamental traits of the technologies involved, of the applications that can feasibly be tackled through text classification, and of the tools and resources that are available to the researcher and developer wishing to take up these technologies for deploying real-world applications.

II. THE FUNDAMENTAL PICTURE

TC may be formalized as the task of approximating the unknown *target function* $\Phi : D \times C \rightarrow \{T,F\}$ (that describes how documents to be classified, according to a authoritative expert) by means of a function $\hat{\Phi} : D \times C \rightarrow \{T,F\}$ called the *classifier*, where $C = \{c1, \ldots, c|C|\}$ is a predefined set of categories and D is a (possibly infinite) set of documents. If $\Phi(dj, ci) = T$, then dj is called a *positive example* (or a *member*) of ci, while if $\Phi(dj, ci) = F$ it is called a *negative example* of ci. The categories are just symbolic labels: no additional knowledge of their meaning is usually available, and it is often the case that no metadata (such as e.g. publication date, document type, and publication source) is available. In these cases, classification must be accomplished only on the basis of knowledge extracted from the documents themselves. Depending on the application, TC may be either a *single-label* task (i.e. exactly one $ci \in C$ must be assigned to each $dj \in D$), or a *multi-label* task.

A. Document Indexing

Document indexing denotes the activity of mapping a document dj into a compact representation of its content that can be directly interpreted (i) by a classifier building algorithm and (ii) by a classifier, once it has been built. The document indexing methods usually employed in TC are borrowed from IR, where a text dj is typically represented as a vector of term weights _dj = _w1j, . . . , w|T |j _. Here, T is the dictionary, i.e. the set of terms (also known as features) that occur at least once in at least k documents (in TC: in at least k training documents), and $0 \leq wkj \leq 1$

quantifies the importance of tk in characterizing the semantics of dj . Typical values of k are between 1 and 5. An indexing method is characterized by (i) a definition of what a term is, and (ii) a method to compute term weights. Concerning (i), the most frequent choice is to identify terms either with the words occurring in the document (with the exception of stop words, i.e. topic-neutral words such as articles and prepositions, which are eliminated in a pre-processing phase), or with their stems (i.e. their morphological roots, obtained by applying a stemming algorithm . Concerning (ii), term weights may be binary-valued (i.e. wkj ∈ {0, 1}) or real-valued (i.e. $0 \leq wkj \leq 1$), depending on whether the classifier-building algorithm and the classifiers, once they have been built, require binary input or not. When weights are binary, these simply indicate presence/absence of the term in the document. When weights are non-binary, they are computed by either statistical or probabilistic techniques. One popular class of statistical term weighting functions is tf * idf, where two intuitions are at play: (a) the more frequently tk occurs in dj, the more important for dj it is (the term frequency intuition); (b) the more documents tk occurs in, the less discriminating it is, i.e. the smaller its contribution is in characterizing the semantics of a document in which it occurs (the inverse document frequency intuition). Weights computed by tf * idf techniques are often normalized so as to contrast the tendency of tf * idf to emphasize long documents. In TC, unlike in IR, a dimensionality reduction phase is often applied so as to reduce the size of the document representations from T to a much smaller, predefined number. This has both the effect of reducing over fitting (i.e. the tendency of the classifier to better classify the data it has been trained on than new unseen data), and to make the problem more manageable for the learning method, since many such methods are known not to scale well to high problem sizes. Dimensionality reduction often takes the form of feature selection: each term is scored by means of a scoring function that captures its degree of (positive, and sometimes also negative) correlation with ci, and only the highest scoring terms are used for document representation. Alternatively, dimensionality reduction may take the form of feature extraction: a set of "artificial" terms is generated from the original term set in such a way that the newly generated terms are both fewer and stochastically more independent from each other than the original ones used.

B. *Classifier learning*

A text classifier for ci is automatically generated by a general inductive process (the learner) which, by observing the characteristics of a set of documents pre classified under ci or ci, gleans the characteristics that a new unseen document should have in order to belong to ci. In order to build classifiers for C, one thus needs a set Ω of documents such that the value of Φ(dj, ci) is known for every _dj, ci_ ∈ Ω × C. In experimental TC it is customary to partition Ω into three disjoint sets Tr (the training set), V a (the validation set), and Te (the test set). The training set is the set of documents observing which the learner builds the classifier. The validation set is the set of documents on which the engineer fine-tunes the classifier, e.g. choosing for a parameter p on which the classifier depends, the value that has yielded the best effectiveness when evaluated on V a. The test set is the set on which the effectiveness of the classifier is finally evaluated. In both the validation and test phase, "evaluating the effectiveness" means running the classifier on a set of pre classified documents (V a or Te) and checking the degree of correspondence between the output of the classifier and the pre assigned classes. Different learners have been applied in the TC literature. Some of these methods generate binary-valued classifiers of the required form $\hat{\Phi} : D \times C \rightarrow \{T,F\}$, but some others generate real-valued functions of the form $CSV : D \times C \rightarrow [0, 1]$ (CSV standing for categorization status value). For these latter, a set of thresholds τi needs to be determined (typically, by experimentation on a validation set) allowing to turn real-valued CSVs into the final binary decisions. It is worthwhile to notice that in several applications, the fact that a method implements a real-valued function can be profitably used, in which case determining thresholds is not needed. For instance, in applications in which the quality of the classification is of critical importance (e.g. in filing patents into patent directories), post-editing of the classifier output by a human professional is often necessary. In this case, having the documents ranked in terms of their estimated relevance to the category may be useful, since the human editor can scan the ranked list starting from the documents deemed most appropriate for the category, and stop when desired.

C. *Classifier Evaluation*

Training efficiency (i.e. average time required to build a classifier $\hat{\Phi}i$ from a given corpus Ω), as well as classification efficiency (i.e. average time required to classify a document by means of $\hat{\Phi}i$), and effectiveness (i.e. average correctness of $\hat{\Phi}i$'s classification behavior) are all important measures of success for a learner. In TC research, effectiveness is usually considered the most important criterion, since it is the most reliable one when it comes to experimentally comparing different learners or different TC methodologies, given that efficiency depends on too volatile parameters (e.g. different sw/hw platforms). In TC applications, however, all three parameters are important. In applications involving interaction with the user, a classifier with low classification efficiency is unsuitable. On the contrary, in multi-label TC applications involving thousands of categories, effectiveness tends to be the primary criterion in operational contexts too, since in most applications an ineffective although efficient classifier will be hardly useful, or will involve too much post-editing work on the part of human professionals, which might defy the purpose of using an automated system. In single-label TC, effectiveness is usually measured by accuracy, i.e. the percentage of correct classification decisions However, in binary (in multi-label) TC, accuracy is not an adequate measure. In this case, building a classifier that has high accuracy is trivial, since the trivial rejector, i.e. the classifier that trivially assigns all documents to the most heavily populated category (i.e. ci), has indeed very high accuracy.

III. Different Classifiers

A. Decision Trees

A Decision Tree text classifier is a tree in which internal nodes are labeled by terms, branches departing from them are labeled by the weight that the term has in the text document and leafs are labeled by categories. Decision Tree constructs using 'divide and conquer' strategy. Each node in a tree is associated with set of cases. This strategy checks whether all the training examples have the same label and if not then select a term partitioning from the pooled classes of documents that have same values for term and place each such class in a separate subtree.

B. Decision Rule

Decision rules classification method uses the rule-based inference to classify documents to their annotated categories. A popular format for interpretable solutions is the disjunctive normal form (DNF) model. A classifier for category ci built by an inductive rule learning method consists of a disjunctive normal form (DNF) rule. In the case of handling a dataset with large number of features for each category, heuristics implementation is recommended to reduce the size of rules set without affecting the performance of the classification.

C. Naïve Bayes Algorithm

Naive Bayes classifier is a simple probabilistic classifier based on applying Baye's Theorem with strong independence assumptions. This algorithm computes the posterior probability of the document belongs to different classes and it assigns document to the class with the highest posterior probability. This probability model would be independent feature model so that the present of one feature does not affect other features in classification tasks.

D. K-Nearest Neighbors

K-NN classifier is a case-based learning algorithm that is based on a distance or similarity function for pairs of observations, such as the Euclidean distance or Cosine similarity measure's. This method is try for many application Because of its effectiveness, non-parametric and easy to implementation properties, however the classification time is long and difficult to find optimal value of k .The best choice of k depends upon the data; generally, larger values of k reduce the effect of noise on the classification.

E. Support Vector Machines (SVM)

The support vector machine method has been introduced in text classification by Joachim. Support vector Machines were used for separate the different classes. Linear support vector machines have the advantages of simplicity and interpretability. Normally, Text data which are correlated with one another and organized into the linearly separable categories. Support vector machines can be applied to the Email data classification. The performance of support vector machine is compared to the other classification techniques like decision trees, the rule based classifier and rocchio method it should provides the more robust and flexible performance. The support vector machine classifier has been well suited for large amount of unlabeled data and small amount of labeled data. To solve the quadratic programming problem and two-class pattern recognition problem, support vector machine can be applied. Hyper planes are chosen for the separator for high dimensional surfaces. It should classify the positive and negative margins in the high dimensional surface. This method should not need any human and machines help for tuning on a validation set of parameters, default choices are available in the support vector machines. There are error-estimating formulas are helpful for predicting the classification and eliminating the need of cross validation on the test and training set of data. It is very easy to select the features from the high dimensional space.

SVM classification algorithms, proposed by Vapnik to solve two-class problems, are based on finding a separation between hyperplanes defined by classes of data shown in Figure

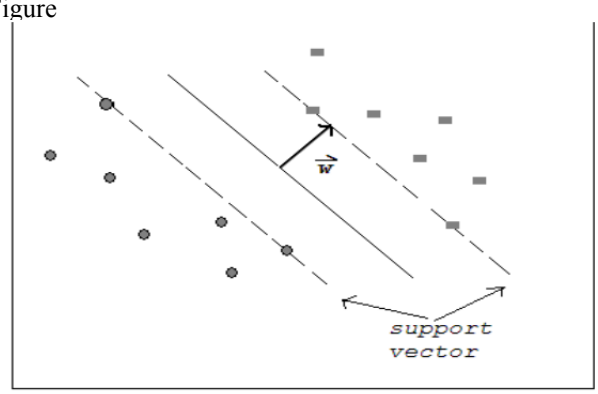

Fig. Example of SVM hyper plane pattern

IV. Comparative Observations

The performance of a classification algorithm is greatly affected by the quality of data source. Irrelevant and redundant features of data not only increase the cost of mining process, but also reduce the Quality of the result in some cases]. Each algorithm has its own advantages and with their time complexity. The the most common method in most cases support machine have better effect than other classifiers.

V. Conclusion

Text categorization play a very important role in information retrieval, machine learning , text mining and it have been successful in tackling wide variety of real world applications. Key to this success have been the ever-increasing involvement of the machine learning community in text categorization, which has lately resulted in the use of the very latest machine learning technology within text categorization applications. Many approaches for text categorization are discussed here. Process of text classification is well researched, but still many improvements can be made both to the feature preparation and to the classification engine itself to optimize the classification performance for a specific application. Different algorithms perform differently depending on data collection. However, to the certain extent SVM performs well in many text classification tasks.

REFERENCES

[1]. Russell Greiner and Jonathan Schaffer, "Exploratorium – Decision Trees", Canada. 2001.

[2] KO, Y. J., Park, J., and Seo, J., "Improving text categorization using the importance of sentences", International Journal Information Processing and Management, vol. 40, no. 1, January 2004, pp. 65-79.

[3] Songbo, T., Cheng, X., Ghanem, M. M., Wnag, B. and Xu, H., "A novel refinement approach for text categorization", Proc. of 14th ACM International Conference on Information and Knowledge Management, 2005, pp.469-476.

[4] Vapnik (1995), The Nature of Statistical Learning Theory. Springer, Berlin.

[5] Salton, G. and M. McGill. (1983). Introduction toinformation retrieval. McGraw-Hill, Inc., New York, NY.

[6] Kim S. B., Rim H. C., Yook D. S. and Lim H. S., "Effective Methods for Improving Naïve Bayes Text Classifiers", LNAI 2417, 2002, pp.414-423.

[7] Klopotek M. and Woch M., "Very Large Bayesian Networks in Text Classification", ICCS 2003, LNCS 2657, 2003, pp. 397-406.

[8] Sang- Bum Kim, et al, "Some Effective Techniques for Naive Bayes Text Classification "IEEE Transactions on Knowledge and Data Engineering, Vol. 18, November 2006.

[9] Yirong Shen and Jing Jiang" Improving the Performance of Naive Bayes for Text Classification"CS224N Spring 2003.

[10] Michael J. Pazzani "Searching for dependencies in Bayesian classifiers" Proceedings of the Fifth Int. workshop on AI and, Statistics. Pearl, 1988.

[11] T.Joachims, "Text categorization with SVM: Learning with many relevant features," proc.European conf.Machine learning,pp.137-142,1988

[12] Thorsten Joachims , "A statistical learning model of text classification for support vector machines"

AUTHORS

M.Maharasi She is presently working as a Assistant Professor in Dr.Sivanthi Aditanar College of Engineering, Tiruchendur. She has done her M.E (CSE) in Dr.Sivanthi Aditanar College of Engineering, Anna University @ Tiruchendur in 2010. She received her M.Phil degree from Manonmaniam Sundaranar University at Tirunelveli in 2004. She received her MCA degree in Sri Saratha College for women @ Bharathidasan University, Karur in 1996. She received her B.sc (computer Science) degree in Cauvery College for women Bharathidasan University @ Trichy in 1993. She has 14 years of teaching experience in this field. She had presented papers in national and International Conferences.

Ms. N. Antony Sophia has obtained B. E. Degree in Computer Science and Engineering from Dr. G. U. Pope College of Engineering, in 2008 and M. E. from Anna University of Technology Tirunelveli, in 2010. She worked as a Lecturer in the department of Computer Science in Holy Cross Engineering College around a year. She is currently working as Assistant Professor in the department of Computer Applications in Dr. Sivanthi Aditanar College of Engineering. Her areas of interest are CBIR in Image Processing, Data Mining and Mobile Ad Hoc Network.