Journal of Theoretical and Applied Information Technology

www.jatit.org

TEXT CLASSIFICATION USING MACHINE LEARNING

[1]MOHAMMED.ABDUL.WAJEED (PhD), [2]Dr.T.ADILAKSHMI

[1]Associate Professor, SCSI, SNIST Ghatkesar, Hyderabad.

[2]Prof. & Head, CSE Dept., Vasavi College of Eng., Hyderabad.

Email : Wajeed.mtech@gmail.com , t_adilakshmi@rediffmail.com

ABSTRACT

Now-a-days due to high availability of computing facilities, large amount of data in electronic form is generated. The data generated is to be analyzed so as to maximize the benefits, for intelligent decision making. If the data generated is in the structured form then a large amount of work in analyzing such structured data is available. But in case of textual data, which is also termed as unstructured data, the need to analyze such data has become inevitable. This paper attempts in exploring the methods in analyzing the unstructured data.

Keywords: *Flat and hierarchical classification, Lexicon, hard and soft classification, stemming, stop-words*

1. INTODUCTION

Modern information age produces vast amount of textual data, which can be termed in other words as unstructured data. Internet and corporate spread across the globe produces textual data in exponential growth, which needs to be shared, on need basis by individuals. If the data generated is properly organized, classified then retrieving the needed data can be made easily with least efforts. Hence the need of automatic methods to organize, classify the documents becomes inevitable due to such exponential growth in documents, very especially after the increase usage of internet by individuals. Automatic classification refers to assigning the documents to a set of pre-defined classes based on the textual content of the document. The classification can be flat or hierarchical. As show in the figure 1 flat classification, categories are considered parallel, i.e., one category does not supersede another. If the class

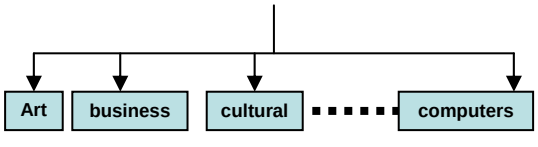

Figure 1. Flat classification

categories grow significantly large in number say, in thousands then searching with such a large number of categories becomes very difficult. This difficulty leads to have hierarchical classification in which the thematic relationship between the classifications is also used, in searching of documents. Figure 2 gives hierarchical classification.

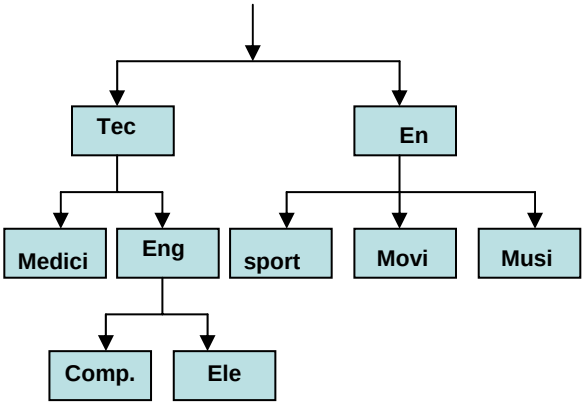

Figure 2. Hierarchical classification

Text Categorization (TC), also known as Text Classification, is the task of automatically classifying a set of text documents into different categories from a predefined set. Consider the case of sorting and organizing emails, files in folder hierarchies so that topic identification that would support topic specific operations be made. On such attempt is the yahoo web directory. If such classification is to be done manually it has several disadvantages [6].

i. It needs domain experts in the areas of pre-defined categories.
ii. It is time-consuming, leads to frustration.
iii. It is error-prone and could be employee biased (subject biased).
iv. Human decision among two experts may disagree.
v. Need to repeat the process for new documents (possibly of another domain).

So the need to employee machine learning to automate the classification is needed. In machine learning generally two types of learning algorithms are found in the literature: *supervised learning algorithms* or *unsupervised learning algorithms*. We restrict in the paper about supervised learning.

In the process of TC construction of a generalize model (classifier) is considered, which is constructed using pre-classified training documents for each category so as to classify correctly the unseen documents. The model is expected to be very effective, and would be independent of the domain of the documents to be classified.

2. BASIC CONCEPTS

The main goal of Text Categorization or Text Classification (TC) is to derive methods for the classification of natural language text [2]. The objective is to automatically derive methods that, given a set of training documents $D = \{d_1, \ldots, d_r\}$ with known categories $C = \{c_1, \ldots, c_q\}$ and a new document q, which is usually called the query, will predict the query's category, that is, will associate with q one or more of the categories in C.

Some of the applications of TC are in finding answers to similar questions that have been answered before say in some legal court case, classifying news by subject or newsgroup; sorting spam from legitimate e-mail messages; finding Internet pages on a given subject, among others. In each case, the goal is to assign the appropriate category or label to each document that needs to be classified.

Initially machine learning was applied for a binary classification, where the classifier determines whether the document belongs to some pre-defined category or not. A special case of single-label classification is binary classification, in which, given a category c_k, each d_j € D must be assigned either to c_k or to its complement c_k. An example of binary classification is distinguishing an e-mail message between spams and legitimate. Figure 3 gives binary classification. In hard classifier a document at any instance of time belongs to a single category, it cannot be a member of two classes.

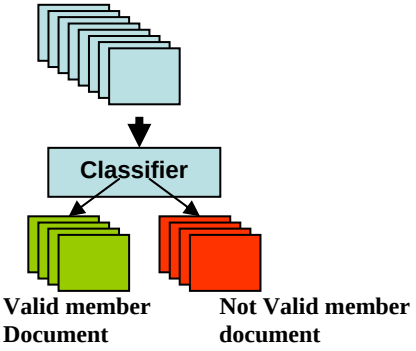

Figure 3. Binary hard classification

In some situations, documents can have only one classification, but the pre-defined categories need not be binary, they could be large number of categories[1]. As the need demanded for a document to be classified into several categories, the binary classifier was modified for the classification of documents to multiple categories. This kind of classification is called single-label classification, where exactly one class c_k € C must be assigned to each document d_j € D. If the classification is made in such a way that one document belongs to utmost only one class at a time, then we refer to such a classification as single-label multi-class classification task. Figure 4 shows abstract multi-class single label classifier.

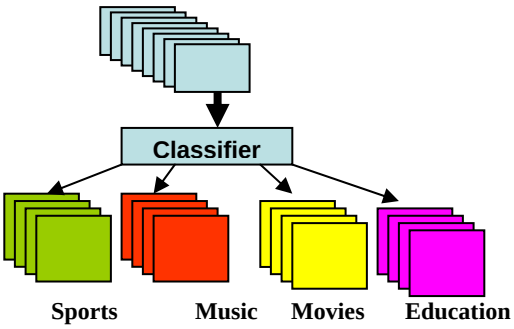

Figure 4. Multi-class single label classification

A problem of multi-label classification under $C = \{c_1, \ldots, c_q\}$ can be tackled as |C| independent binary classification problems under $\{c_k, c_k'\}$, for i = 1, . . . , |C|. In this case, a classifier for C is thus actually composed of |C| binary classifiers. The case a document belong to more than a single category at the same instance of time then such a classifier is termed as multi-class,

multi-label classification. Depending on the application, documents can be classified into one or more classes. For instance, a piece of news regarding how the funds were spent in developing irrigation projects by the government can be classified both as development in irrigation, funds usage. This kind of classification is called multi-label classification, where any number $0 < n_j <= |C|$ of classes may be assigned to each document d_j € D. Figure 5 shows multi-class, multi-label classifier.

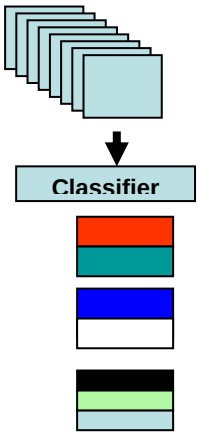

Figure 5. Multi-class multi-label hard classification.

Another line of classifier types is the hard classification and soft classification. In hard classification the result of the classifier is a Boolean type stating whether a document belongs to the category or not. But in case of soft classification, using fuzzy technique the result gives the degree stating to what

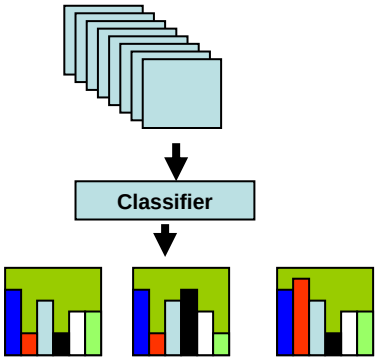

Figure 6. Multi-class soft classification.

extent the document belongs to the category. Combining all the above classifier we have binary classification, multi-class single label with hard classification, multi-class with multi-label with hard classification, multi-class with soft classification as shown in the figure as the important types of the classification.

While in hierarchical classification, the categories are organized in a hierarchical tree-like structure, in which each category may have a number of subcategories.

In case of hierarchical classification we have different possible structures of the hierarchy. In most of the hierarchical classification methods, the categories are organized in tree like structures. On the whole, we can identify four distinct category structures for text classification. They are:

1. *Virtual category tree:* In this category structure, categories are organized as a tree. Each category can belong to at most one parent category and documents can *only* be assigned to the leaf categories [2].
2. *Category tree:* This is an extension of the virtual category tree that allows documents to be assigned into both internal and leaf categories [5].
3. *Virtual directed acyclic category graph:* In this category structure, categories are organized as a Directed Acyclic Graph (DAG). Similar to the virtual category tree, documents can only be assigned to leaf categories.
4. *Directed acyclic category graph:* This is perhaps the most commonly-used structure in the popular web directory services such as Yahoo! [3] and Open Directory Project [4]. Documents can be assigned to both internal and leaf categories.

TC can employs tools developed by Information Retrieval (IR), Machine Learning (ML) researchers because it is a content-based document management task, sharing many characteristics with other IR tasks, such as text search, where the goal is to find the set of documents (or document passages) most relevant to a particular query.

3. PROPOSED PRE-PROCESSING

Documents are given in terms of raw data which usually needs a transformation into machine process-able representation. Cleaning text from insignificant terms and stemming are prevalent part of preprocessing phase. So a process of conversion of the raw data in to a machine readable format is needed i.e. the process of transforming the texts into a representation suitable for the learning algorithms. This section deals with two strategies of encoding documents for tasks of text mining, such as text categorization and text clustering.

The documents can be encoded into numerical vectors for text categorization or text clustering. The process of encoding is done in two parts.

a. The first part of the process deals with extracting words as feature candidates from a corpus and the process of selecting some of them as features. Lexicon is the set which contains all the selected features.
b. The second part of the process assigns values corresponding to the features, which makes the final step of generating numerical vectors.

A collection of documents is given as a corpus in advance. The corpus is given as input of this stage. A list of words and their frequencies is generated as its output. The algorithm for *Lexicon generation* is given below.

a. Concatenation of all documents into a single document.
b. Conversion of sentences into individual words.
c. Sort all the words so that repeated words are clustered (collected) together.
d. Perform stemming on the words and remove stop words (optional step).
e. Find the frequency of each word.
f. Select cut-off for the frequency; include all the words above the cut-off into Lexicon.

Pre-processing stage consists of three steps, as illustrated in figure 7. As illustrated in figure 7, a document or documents may be given as input of this stage which is concatenated into a full text. The integrated full text becomes the target for the tokenization. The full text is tokenized into tokens by a white space or a punctuation mark. Therefore, the output of this step is a list of tokens.

The next step to the concatenation & tokenization is the stemming & exception handling, as illustrated in figure 7. In this step, each token is converted into its root form. Before doing that, rules of stemming and exception handling are saved into a file. When the program encoding documents is activated, the rules are loaded into memory and the corresponding rules are applied to each token. The output of this step is a list of tokens converted into their root forms. The last step of extracting feature candidates from a corpus is to remove stop words as illustrated in figure 7. This step can also be the second step i.e. either we do stop words removal first then stemming or stemming followed by stop word removal. Here, stop words are defined as words which function only grammatically without their relevance to content of their document; articles (a an, or the), prepositions (in, on, into, or at), pronoun (he, she, I, or me), and conjunctions (and, or, but, and so on) belong to this kind of words. It is necessary to remove this kind of words for more efficient processing. After removing stop words, frequencies of remaining words are counted. Therefore, a list of the remaining words and their frequencies is generated as the final output from the process illustrated in figure 7.

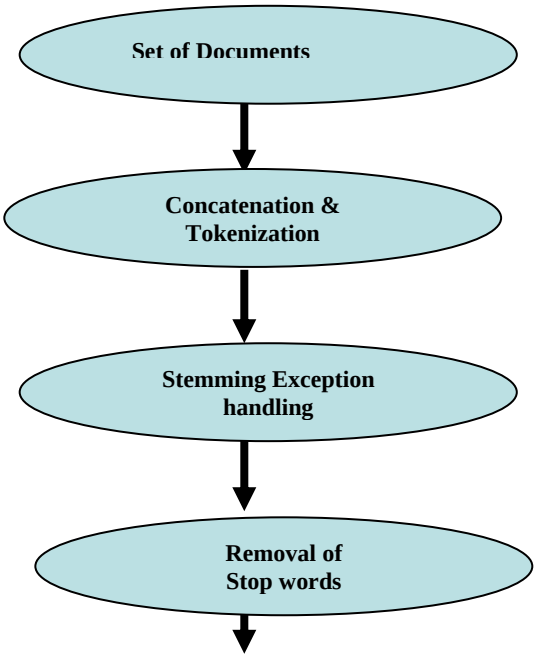

Figure 7. steps in pre-processing

As too many feature candidates are usually extracted from a corpus, some of them should be selected as features. We have many ways to choose the feature among; frequencies of words are set as the criteria for selecting features for simplicity. Words with their highest frequencies are selected as features.

Once features are selected as attributes of numerical vectors, values should be assigned to the features. There are the three ways for assigning values to features for encoding documents into numerical vectors.

i. To each feature, a binary value is set, indicating its absence or presence; a document encoded into a binary vector in this way.

ii. To each feature, its frequency in the document is set as its value; a numerical vector representing a document has integers as its elements, in this way.

iii. We set values of features as weights of words computed by equation given below

$$\text{weight}_i(w_k) = tf_i(w_k)(\log_2 df(w_k) + 1$$

where

- $\text{weight}_i(w_k)$ gives the weight of the word w_k, which indicates its content based importance in the document, i.
- $tf_i(w_k)$ gives the frequency of the word wk in the document i.
- $df(w_k)$ is the number of document including the word w_k and D is the total number of documents in the corpus

4. APPROACH FOR CLASSIFICATION

Basically two approaches can be adopted in the process of hierarchical classification. They are Big-bang approach and top-down level based approach.

In case of the big-bang approach with multi-label classification a single classifier is given the task to classify the documents to its respective category trees, while in case of multi-class classification the single classifier is given the task to classify the document to its respective category tree. Again depending upon the structure of the tree the document can be assigned to the internal node or to the leaf nodes of the hierarchy. The performance of the big-bang approach is very simple where in, one needs to observe the number of correctly classified documents or to observe the incorrectly classified documents.

The disadvantage of big-bang approach is that it can use the information carried by the category structure during the training phase not the classification phase. Moreover the discriminative features at a parent category may not be discriminative at the child categories. It is very difficult for a classification method using big-bang approach to exploit the different sets of features at different category levels. It is also not flexible enough to cater for changes to the category structure. There is a need to retrain once the category structure is changed.

In case of the top-down level based approach at each level one or more classifier are built similar to the flat classification. At the root level the classifier classifies the document to one or more lower level category.

The document would be further classified into lower level categories until it reaches a final category where from it cannot be lowered further.

The most disadvantage of the top-down level based approach is that if the document is misclassified at the parent (ancestor level) category, this forces a document to be excluded from the child categories before it could be examined by the classifiers of the child categories. More-ever top-down level based classification approach needs more training examples due to multiple classifiers construction at each level. Each classifier needs different training instances.

5. CONCLUSIONS

In this paper, the need to classify the unstructured data was introduced. The need of such a classification and types of classification structure were stated. In case of large number of documents the ease in using the hierarchical classification over the non-hierarchical classification was emphasized. Major part of the paper contributes towards how the Lexicon generation can be made. Lexicon is the set of words the document contains. These words are used as the features of the learning model (the classifier). Once the Lexicon is generated then, based on it feature vectors of the documents need to be generated. Using the feature vector of the document the classifier would classify the documents into different labels.

REFERENCES:

[1]. Fabrizio Sebastiani. Text Classification for Web Filtering Final POESIA Workshop “Present and Future of Open-Source Content-Based Web filtering” Pisa, IT — 21-22 January, 2004.

[2]. Jason D. M. Rennie. Improving multi-class text classification with naive bayes. Master's thesis, Massachusetts Institute of Technology, 2001.

[3]. Michelangelo Ceci and Donato Malerba: Classifying web documents in a hierarchy of categories: A comprehensive study.

[4]. Xiaojin Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, Madison, WI, 2005.

[5]. Stopwords list for the english language. From the Weka software. http://www.cs.waikato.ac.nz/ml/weka.

[6].Fabrizio Sebastiani. Text classification, automatic. In Keith Brown (ed.), *The Encyclopedia of Language and Linguistics*, 2nd Edition, Vol. 14, Elsevier Science, Amsterdam, NL, 2004.