Text Categorization Research Based on Cluster Idea

Mrs.S.A. Khalatkar, Mr. Abhay Khalatkar, Ms. Ketki Khante
Computer Engineer Department, RTM Nagpur University, Mechanical Engineer Department, RTM Nagpur University,
Computer Engineer Department, RTM Nagpur University,
as22@rediffmail.com, amkhalakar@iiitdmj.ac.in, ketki_khante@yahoo.co.in

Abstract — Classification and clustering are frequently-used methods in data excavation technology. Entropy model is the base structure of automated word categorizing. In this model, words appear consecutive frequently will place in different groups. Although this method is not correct always, in the most cases, obtained results simulate real situation. Text-based matching is performed to generate “soft” seeds, which are then used to guide clustering in the basic feature space. Because of NP-Complete structure of clustering problems, the entropy model cannot be solved by an optimal algorithm, so a number of heuristic algorithms were developed to solve this problem. Some examples of these heuristics are artificial neural networks, genetic algorithms, greedy algorithms and so on. In this paper, we used a clustering ensemble method for text categorization. The method uses a new feature space generated by k-means for clustering. The results show that an improvement in categorization is obtained.

Index Terms — text categorization, text clustering, KNN algorithm, K-Means algorithm.

I. INTRODUCTION

With the explosion of data volumes and Internet usage, many everyday applications rely on some form of automated categorization to avoid information overload and improve user experience. While matching of objects to predefined categories can often be carried out via supervised classification, there are many cases where classification is difficult due to inability to generate training examples for all categories. This is particularly true in situations where the pre-defined categories are highly dynamic due to constantly evolving applications and user needs. This makes it impossible to rely on manual labeling or sample selection, as they would have to be conducted whenever the taxonomy changes. In this paper we present a semi-supervised clustering methodology to address these challenges, and apply descriptions, generated independently of the category descriptions.

A set of assumptions captures well the characteristics of many real world applications. For example, in a stock photo database, images are organized according to predefined categories, such as “Portraits” or “Macro”, with corresponding descriptions. Selected image features can be computed for all photos in the collection, forming the basic feature set, while some photographers enter the description of their photos, which then represent optional data descriptions.

Another example, which we will use to develop and validate our methodology, is in the area of business analytics, and relates to categorization problems often encountered in project management tools.

Such tools are used to track projects and compute business metrics according to a set of predefined categories aligned with products/services sold by the company. However, because of the dynamic business environments and changing customer needs, the solution portfolios are constantly evolving and frequently redefined, limiting the ability of project managers to categorize projects accurately. Hence, there is a need for an automated methodology to assist with project categorization.

The text auto-categorization technology is the important basis of information retrieval and text excavation. Text categorization is the method which can effectively store and manage electronic documents. According to the categorization rules got from training document storehouses, a new document is placed into a certain categorization which is defined in advance, and thus documents are effectively managed. Clustering is the process of dividing the set of physical or abstract objects into categories composed of similar objects.

The difference between text clustering and text categorization is that text clustering has no subject category being set in advance. The purpose of it is to divide multi-document aggregation into different clusters. The similarity degree of the content of the documents in the same cluster should be to the most, while in different clusters to the least.

This method consists in counting occurrences of word pairs in text and using a greedy, hierarchical clustering technique on the frequency data to obtain a classification of words into linguistic categories. As a distance criterion in the
clustering process, it uses the loss of mutual information caused by combining two clusters into a single new cluster.

II. TEXT CATEGORIZATION AND INFORMATION TECHNOLOGY

Text categorization is the process of grouping text documents into one or more predefined categories based on their content. A number of statistical classification and machine learning techniques have been applied to text categorization, including regression models, Bayesian classifiers, decision trees, nearest neighbor classifiers, neural networks, KNN (K-Nearest Neighbor), as a simple, effective and non-parametric method, has aroused broad attention and has made very good achievement in document categorization. $k$ is the most important parameter in a text categorization system based on the $k$-nearest neighbor algorithm (kNN). To classify a new document, the $k$-nearest documents in the training set are determined first. The prediction of categories for this document can then be made according to the category distribution among the $k$ nearest neighbors. Generally speaking, the class distribution in a training set is not even; some classes may have more samples than others. The system's performance is very sensitive to the choice of the parameter $k$. And it is very likely that a fixed $k$ value will result in a bias for large categories, and will not make full use of the information in the training set. To deal with these problems, an improved kNN strategy, in which different numbers of nearest neighbors for different categories are used instead of a fixed number across all categories, is proposed in this article. More samples (nearest neighbors) will be used to decide whether a test document should be classified in a category that has more samples in the training set. The numbers of nearest neighbors selected for different categories are adaptive to their sample size in the training set. Experiments on two different datasets show that our methods are less sensitive to the parameter $k$ than the traditional ones, and can properly classify documents belonging to smaller classes with a large $k$.

A. $k$-nearest neighbor method

It is a kind of example-based text categorization method, and is also a sort of vector space similarity degree between testing text and every text in the training sample set, find out $k$ training texts which are similar to the most, and then it counts out among the $k$s training samples the number of the text which belongs to the same category. The testing sample belongs to the category of the biggest number. The merit of this algorithm is that it’s a simple one. However, the problem of it is that it needs to store all the samples into computer and compare the distance between the being-identified sample and all the other training samples. However, it is all known that a categorization system always has to provide enough amounts of training documents to achieve the categorization accuracy rating, which results in the deceleration of categorization. Text categorization is the process of grouping text documents into one or more predefined categories based on their content. A number of statistical classification and machine learning techniques have been applied to text categorization, including regression models, Bayesian classifiers, decision trees, nearest neighbor classifiers, neural networks, and support vector machines. The idea behind k-Nearest Neighbor algorithm is quite straightforward. To classify a new document, the system finds the $k$ nearest neighbors among the training documents, and uses the categories of the $k$ nearest neighbors to weight the category candidates.

B. K-Means algorithm

The text clustering K-Means algorithm is also called k means value clustering method, which is a partition-based and simple clustering method. Because it’s theoretically reliable, simple and speedy, it has been popularized in fundamental steps of K-Means algorithm are as follows:

Step 1: Randomly select $k$ texts to be the original cluster centroid. $k$ is the number of given clusters.

Step 2: According to the similarity degree between every text and every category centroid, category it to the most similar category, and then recalculate the centroid of every category.

Step 3: Iterate the two steps above again and again until the criterion function becomes convergent.

The purpose of K-Means clustering method is to cut down the square difference of every point and the cluster center in every cluster. This algorithm has advantage in time complexity degree, but it requires to giving $k$, the number of cluster. And it is sensitive to the isolated point.

C. Hiberarchy clustering algorithm

The thought of hierarchy clustering algorithm is: Firstly treat every data object as a cluster, the center of which is the data object. Combine two clusters of maximum similarity degree every time until the number of clusters is $k$, and then calculate the center of every cluster. During the process of clustering, the similar data objects gradually become one cluster. Hiberarchy clustering can automatically generate clustering models of different arrangements. The time complexity degree of hierarchy clustering is high ($O(n^2)$ in general), but the algorithm cost is acceptable when it works on small samples through random sampling. When two clusters, clu1 and clu2, combine into a new cluster, the central vector of it can be obtained from the central vector of clu1 and clu2. Suppose the numbers of data point in clu1 and clu2 are n1 and n2, the centre vectors are cen1 and cen2, and the characteristic vector of the new cluster is cen, then the calculation formula will be:

\[ cen = \frac{n_1 * cen_1 + n_2 * cen_2}{n_1 + n_2} \]

III. ALGORITHM DESIGN AND DESCRIPTION

All paragraphs must be indented. All paragraphs must be justified, i.e. both left-justified and right-justified. In order to improve time efficiency of KNN categorization, we attempt to introduce clustering thought into KNN categorization system. We reduce the dimension of training text set in clustering method, thus to cut down KNN’s calculation amounts. The fundamental thought is: re-cluster training text on the basis of identified category, and generate more concentrated distribution clusters. Therefore every element belongs to the
same category, and then we compare the testing document vector with the entire cluster central vector in the training document set. Both KNN and K-Means use the method of text similarity degree operation. KNN calculates the similarity degree between the being-categorized text and every training text, with its categorization accuracy higher, while time efficiency relatively lower; K-Means evaluates the similarity degree between the being-clustered text and every clustering center, with its time efficiency fairly higher, while clustering accuracy relatively lower, and it also has the problems of requiring artificially ascertain parameter k and isolated points sensitivity. We will try to introduce clustering thought into categorization system, so as to reduce the dimension of training text set and to balance the categorization accuracy and time efficiency in categorization algorithm. One of the drawbacks of kNN algorithm is its efficiency, as it needs to compare a test document with all samples in the training set. In addition, the performance of this algorithm greatly depends on two factors, that is, a suitable similarity function and an appropriate value for the parameter k. If k is too large, big classes will overwhelm small ones. On the other hand, if k is too small, the advantage of kNN algorithm, which could make use of many experts, will not be exhibited. In practice, the value of k is usually optimized by many trials on the training and validation sets. But this method is not feasible in some cases where we have no chance to do cross-validation, such as online classification. To deal with this problem, we propose a revised k-Nearest Neighbor algorithm, which uses different k values for different classes, rather than a fixed k value for all classes.

A word-by-document matrix A is used for a collection of documents, where each entry represents the occurrence of a word in a document, i.e., $a_{ij}$, where $a_{ij}$ is the weight of word $i$ in document $j$. There are several ways of determining the weight $a_{ij}$. Let $f_{ij}$ be the frequency of word $i$ in document $j$, $N$ the number of documents in the collection, $M$ the number of distinct words in the collection, and $f_{ij}$ the total number of times word $i$ occurs in the whole collection. The simplest approach is Boolean weighting, which sets the weight $a_{ij}$ to 1 if the word occurs in the document and 0 otherwise. Another simple approach uses the frequency of the word in the document, i.e., $tf_{ij}$. A more common weighting approach is the so-called $tf \cdot idf$ (term frequency - inverse document frequency) weighting:

$$A_{ij} = \frac{f_{ij}}{log(N/n_i)} \quad (1)$$

A slight variation of the $tf \cdot idf$ weighting, which takes into account that documents may be of different lengths, is the following:

$$w_{ij} = \frac{f_{ij}}{\sqrt{\sum f_{ij}^2} \times log\left( \frac{N}{n_i} \right)}.$$  

For matrix A, the number of rows corresponds to the number of words $M$ in the document collection. There could be hundreds of thousands of different words. In order to reduce the high dimensionality, stop-word (frequent word that carries no information) removal, word stemming (suffix removal) and additional dimensionality reduction techniques, feature selection or re-parameterization are usually employed.

To classify a class-unknown document $X$, the $k$-Nearest Neighbor classifier algorithm ranks the document's neighbors among the training document vectors, and uses the class labels of the $k$ most similar neighbors to predict the class of the new document. The classes of these neighbors are weighted using the similarity of each neighbor to $X$, where similarity is measured by Euclidean distance or the cosine value between two document vectors. The cosine similarity is defined as follows:

$$\text{sim}(X, D_j) = \frac{\sum_{i \in X} x_{ij} \cdot d_{ij}}{\|X\|_2 \cdot \|D_j\|_2}$$

where $X$ is the test document, represented as a vector; $D_j$ is the $j$th training document; $n_{ij}$ is a word shared by $X$ and $D_j$; $u_{ij}$ is the weight of word $i$ in $X$; $v_{ij}$ is the weight of word $i$ in document $D_j$; $\|X\|_2$ is the norm of $X$, and

$$\|X\|_2 = \sqrt{u_{11}^2 + u_{22}^2 + u_{33}^2 + \ldots}$$

is the norm of $X$. A cutoff threshold is needed to assign the new document to a known class. The kNN classifier is based on the assumption that the classification of an instance is most similar to the classification of other instances that are nearby in the vector space. Compared to other text categorization methods such as Bayesian classifier, kNN does not rely on prior probabilities, and it is computationally efficient. The main computation is the sorting of training documents in order to find the $k$ nearest neighbors for the test document.

C. Categorization judgment

As to a specific text needing to be categorized, the description of the algorithm of judging its category based on clustering thought is as follows:

Input: One text waiting for categorization
Output: The category of the waiting text
1) Calculate the similarity degree $sim_{ij}$ between each text vector and the central vector of every cluster $T_i$.
2) Find out $k'$ clusters whose similarity degrees are maximal to the text, namely the nearest $k'$ clusters.
3) Calculate the sum of the similarity degree of the cluster central vectors belonging to the same category in $k'$ clusters, and then rank this category on grades, at last specify the category of the text waiting for categorization according to their grades.
4) Classify this text into the nearest cluster of the specified category; meanwhile adjust the central vector of the cluster.

While using kNN algorithm, after k nearest neighbours are found, several strategies could be taken to predict the category of a test document based on them. But a fixed k value is usually used for all classes in these methods, regardless of their different distributions.

IV. EXPERIMENT AND ANALYSIS

The training texts are extracted by the manpower to ensure the correctness of the training data. We just did a pretreatment on the text treatment without using any characteristic choosing algorithm. But due to the short of priori knowledge, and the hardness of giving cluster granularity, we artificially set the cluster number k. We did the categorization respectively using clustering-based text categorization algorithm, KNN categorization algorithm, and K-Means categorization algorithm.

The result of the experiments indicates that the clustering-based text categorization algorithm is inferior to K-Means in time efficiency, but superior to KNN; superior to K-Means in accuracy of classification, but inferior to KNN. It can balance time efficiency and categorization accuracy during the process of categorization. Clustering-based text categorization algorithm uses the K-Means method for reference to try to remove the dependence of categorization accuracy rate to the scale of the training set in the purpose of improving time efficiency of the categorization algorithm. We also noticed that with the increase of the cluster granularity, the required calculating amount decreased, time complicated degree reduced, while accuracy rate reduced as well.

V. CONCLUSION

We have presented an improved kNN classifier, combining it with the idea of the Centroid-based method. The improvement consists in removing outliers from the categories of the training dataset. Our method shows almost 10% better the original Centroid-based classifier, which was reported in as the most accurate text categorization method.

In the future, automatic choice of the threshold value ε is to be considered.

REFERENCES