IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727
PP 14-17
www.iosrjournals.org

Clustering Technique To Boost Text Classification

M. Praveena[1], Dr.V.Jaiganesh[2]
Assistant Professor, Dept. of Computer Science, Dr.SNS College of Arts and Science, Coimbatore, Tamil Nadu, India[1]
Professor, Dept. of PG & Research, Dr. NGP Arts and Science College, Coimbatore, Tamil Nadu, India[2]

Abstract: *Clustering is a technique of data mining. It aims at decision natural separation of data. Objects are grouped depending on the similar nature they share. Similarity is measured depending on different parameters like number of parameters which are in common or lowest allowed difference between any parameters of an object. Data mining techniques include classification, clustering, mining frequent pattern, and correlation. Out of all these, in few decades clustering has gain wide attention of researchers. Clustering involves grouping of data objects which are similar in nature. This helps in the abstraction process of huge amount of data. Once the abstraction process is complete group of data can be represented in more compact manner. This is nothing but data compression. To describe the objects more precisely it needs to be defined by all the possible and meaningful dimensions. As the number of attributes increases finding similarity between objects become difficult. A general boosting algorithm for clustering tasks is proposed, and solutions for directly optimizing two loss functions according to this frame- work are obtained. Experimental results show how the performance of relatively simple and computationally efficient base clustering algorithms could be boosted using the pro- posed algorithm.*

I. Introduction

In this paper we evaluate the performance of two such methods for the automatic categorization of articles published by an international journal in the geological sciences. The first method we consider is the Naïve Bayes (NB) method, a popular categorization method. Not surprisingly, we observe that when the domain expert, in addition to labeling duties also provides a list of keywords and associated weighting the performance of this algorithm is greatly enhanced. The second method we consider is Principle Direction Divisive Partitioning (PDDP), an unsupervised text clustering approach based on principal components analysis. Basically in this approach we cluster both the labeled and unlabeled data together and let the labeled documents within each cluster determine the labels of the unlabeled documents. For the dataset we considered the PDDP-based approaches outperformed the NB-based approaches in terms of the overall quality of performance (as measured by classification or categorization error).However, when taking into account execution time the NB-based approaches were much faster. We then considered a method that combines both the NB-based approach and the PDDP approach. The new approach uses PDDP to cluster the training data to help the method identify pure nodes (representing pure clusters). The text documents that are clustered in the pure nodes are then used to seed the Naïve Bayes approach. We find that the overall quality of this hybrid approach improves the quality of the results as compared with the baseline NB-based approaches. This is attributed to the minimization of noise in the training data by using only the documents from the pure nodes to train the classifier. Also since the PDDP clustering is applied only once and that to on a small portion of the entire dataset (the training data) the hybrid approach also does not suffer from poor performance and is therefore much more scalable than the baseline PDDP-based approaches. We also evaluate the performance of these methods and their variants on papers that are considered to be a mixture of categories. The rest of this article is organized as follows. We evaluated along with the experimental results. we discuss the results obtained and some additional features of our work (visualization etc.). We also outline directions for future work in this section.

II. Methods Of Clustering

Two popular methods of clustering are hierarchical clustering and partitioning clustering.

Hierarchical Clustering

In this method data set is partitioned into distinct partitions. Again procedure of partitioning can be done in two ways-agglomerative and divisive. In agglomerative approach initially each object is treated as single cluster, later objects are merged depending on the criteria followed. In divisive approach initially

entire data set is consider as a cluster. Entire cluster is then separated into smaller sets depending on their similarity. This method of hierarchical clustering generates a data structure called as dendogram. Dendogram describes the order in which data objects are merged. Dendogram gives the hierarchy of clusters and hence the name. When we cut the dendogram at a point then we get the cluster at that level. Similarity measure can be any distance calculating measure such as Euclidian distance or Manhattan distance.

Partitioning Method

In this method each object is placed in exactly one set. Membership of an object in two different sets is not allowed. Here user has to give an input the required number of clusters. This method gives another class of clustering algorithms called as non-overlapping clustering algorithms. Output of partitioning method depends on input given. Same algorithm with same data set can give different output with large difference between each of them depending on required number of clusters.

III. Naïve Bayes Classifier

The basic idea in Naïve Bayes methods is to use the joint probabilities of words and categories to estimate the probabilities of categories given a document. The naïve aspect of the method has to do with the fact that the dependencies between words are ignored,i.e. the conditional probability of a word given a category is assumed to be independent from the conditional probabilities of other words given that category. The assumption is necessary for efficiency reasons. We next describe this approach more formally using the notation from. Assume that each text document is described by a conjunction of the English words (ai) it contains. Let vj be the set of user-defined categories. A set of labeled (training) documents in every category is provided to the classifier. After the training phase (described below) when a new document is presented, described by the tuple of its words a1, a2…an>, the learner is asked to predict the category for this new document.The Bayesian approach to classifying the new document is to assign the most probable category it fits in, vMAP -- the maximum a posteriori (MAP) hypothesis, given the words <(<a1, a2…an>) contained in the document.vMAP = argmax P(vj| a1, a2…an)We can use Bayes theorem to rewrite this expression as vMAP = argmax P(a1, a2…an|vj)P(vj) (1) Estimating the different P(a1, a2…an|vj) terms in this fashion is not feasible unless there is an exceptionally large set of training data. The problem is that the number of these terms is equal to the number of possible documents times the number of possible categories.Therefore, we need to see every document in the training set many times in order to obtain reliable estimates. The Naïve Bayes classifier is based on the simplifying assumption that the contained words are conditionally independent in the category. This assumption reduces equation 1 to equation 2.
vNB = argmax P(vj) Pi P(ai|vj) Here vNB denotes the probability output by naïve Bayes classifier. Notice that with the naïve Bayes classifier the number of distinct P(ai|vj) terms that must be estimated from the training data is just the number of distinct words times the number of distinct categories – a much smaller number than if we were to estimate the P(a1, a2…an|vj) terms first contemplated. Whenever the naïve Bayes assumption of conditional independence is satisfied, this naïve Bayes classification vNB is identical to the MAP classification vMAP obtained from equation. Even when this assumption is not met, as in the case of learning to classify text, the naïve Bayes classifier is often quite effective. Completing the design of the learning algorithm requires choosing a method for estimating the probability P(ai|vj) terms. We adopt the m-estimate with uniform priors and with m equal to the size of the word vocabulary. Thus, the estimate for P(ai|vj) will be (nk+1)/(n+ |Vocabulary|),where n is the total number of word positions in all training examples whose category is vj, nk is the number of times word ai is found among these n word positions, and |Vocabulary| is the total number of distinct words (and other tokens) found within training data. We adopt the implementation of the algorithm proposed in. In addition to the base implementation we also implement a variant that allows us to embed domain specific knowledge within the classifier. This domain knowledge is embedded in terms of keywords associated with a particular category and weights associated with each keyword. Essentially, instead of using the pre-classified papers as training data, we build our own collections of words with different weights based on the domain expert's experience with the data. Note, that the Naïve Bayes classifier needs several training papers for each topic to classify an unknown paper. The classifier is dependent on the quality of papers in the training set. Instead of using the entire paper if one can instead train the classifier based on a bag of words deemed important by an expert, the classifier is less likely to be affected by noise effects.

PDDP (Principal Direction Divisive Partitioning)

PDDP is an unsupervised technique that clusters together related documents. It constructs a binary tree, in which each node is a data structure holding the documents associated with that node, the various

quantities computed from that set of documents, and the pointers to the two children nodes. The information stored in each node consists of the documents in the cluster associated with that node, the centroid (mean vector), the leading singular value and associated singular vector, and the pointers to the children nodes. The "scatter" value is also stored in the node, which will be discussed later.

The PDDP tree starts with the root cluster representing all documents. The algorithm then recursively splits each leaf cluster into two children until some criterion is satisfied. This partitioning constructs a binary tree, the PDDP tree. Each document is represented by an n-vector d of word counts. Each n-vector is scaled so that ||d|| = 1 to ensure that the values are independent of document length. The vectors of all the m documents to be clustered are assembled into term frequency matrix M(n x m) = (d1 … dm). This term frequency matrix along with the mean vector (centroid) is used to obtain the principle direction and the hyper-plane partition that is used to split the documents within a given node into two partitions. To decide at each stage which node should be split next, one choice is to try to keep the binary tree balanced by splitting all the nodes at a given level before proceeding to the next level. However, by doing so the resulting clusters are often imbalanced with a few large clusters and many small clusters, even singletons. PDDP uses the notion of "scatter" to determine the best node to split. The scatter value is simply a measure of how cohesive the documents within a cluster are.

The algorithm chooses the cluster with the largest scatter value to split. The total scatter value represents the distance between each document in the cluster and the overall mean of the cluster. Using the total scatter value to choose the next cluster to be split usually results in clusters all having more or less similar numbers of documents. This scatter value is the only component of this algorithm based on "distance" measure. It would be possible to use other measures not based on a "distance" measure and appropriate for particular data sets. At each pass through the main loop, a node is selected based on total scatter value, and the mean vector and principal direction are obtained for the documents associated with that node. Then the collected mean vector and principal vector are used to split the node into two child nodes. In a geometric sense, the process is splitting the documents using hyper plane normal to the principal direction passing through the mean vector. The execution time of the algorithm depends on the size of the term frequency matrix. Much like we did for the Naïve Bayes approach we have implemented a variant of the base PDDP algorithm where instead of using all the distinct words in the documents to build the term frequency matrix we limit the words to the set of words chosen by domain experts for the problem on hand. This variant reduces the execution time of the algorithm significantly (since the size of the term frequency matrix is reduced) without affecting the quality of results obtained by much as we shall see in the next section.

IV. Experiments And Results

The data we used is from a key journal in the water sciences called "Water Resources Research". We evaluated articles over 7 years, i.e., from 1990-1996. There were basically five major categories of water sciences research ("precipitation", "unsaturated", "groundwater", "river-lake", and "estuary-ocean") amongst these articles along with several articles that involve a mixture of research topics. Unless otherwise stated we used a training set of 116 papers (roughly an equal number from each category) and a testing set of 233 papers to evaluate classifier performance. Of the 233 papers in our testing set the domain experts identified 51 papers as hard to classify (this is also borne out in our classification results). We report classification accuracies over the entire test dataset as well as over this hard subset of test papers. We evaluated the following classifiers:

1. NB0 (Naïve Bayes): This classifier served as a baseline against which we evaluate the other NB variants.

This is the classical approach that uses a training set of documents and evaluates the classifier on an independent testing set. While the overall performance of this classifier was not bad (81% accuracy), only 13 out of 51 papers (25% accuracy) from "hard" set were correctly classified.

2. NB1 (Naïve Bayes with experts' chosen keywords and weighting)

The second classifier we evaluated is the variant. Instead of sending in the pre-labeled chosen papers as training set, we used experts' chosen keywords and associated weights to seed the classifier. In this experiment, the memory needed for storing vocabularies (128 words) and running time is much smaller and more efficient than using training papers (116 pre-labeled training papers created more than 13,000 words). The overall accuracy of this classifier was 87% and the accuracy on the hard subset was 39% (20/51).

References

[1]. Y. Yang. An evaluation of statistical approaches to text categorization. Journal of information Retrieval, 1(1/2):67—88, 1999.

[2]. K. Tzeras and S. Hartman. Automatic indexing based on bayesian inference networks. In Proc 16th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'93).

[3]. C. Apte, F. Damerau, and S. Weiss. Text mining with decision rules and decision trees. In Proceedings of the Conference on Automated Learning and Discovery, Workshop 6: Learning from Text and the Web, 1998.

[4]. D. D. Lewis and M. Ringuette. Comparison of two learning algorithms for text categorization. In Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval (SDAIR'94), 1994.

[5]. Thorsten Joachims. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In European Conference on Machine Learning (ECML).

[6]. Tom Mitchell. Machine Learning. McGraw Hill, 1996.

[7]. Domingos, P., and Pazzani, M. 1996. Beyond independence: Conditions for Optimality of the simple Bayesian classifier.

[8]. Cestnik, B. 1990. Estimating probabilities: A crucial task in machine learning. Proceedings of the Ninth European Conference on Artificial Intelligence (pp. 147-149). London: Pitman.

[9]. McCallum, A., 1996. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/~mccallum/bow. 1996.

[10]. I. Moulinier. Is learning bias an issue on the text categorization problem? In Technical report, LAFORIA-LIP6, Universite Paris VI, 1997.

[11]. D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In The Fourteenth International Conference on Machine Learning (ICML'97).