Text Classification of Twenty Newsgroups Text Data via EM algorithm

Yufei Cai
Department of Statistics and Data Science
Southern University of Science and Technology
Shenzhen, China 518055
caiyf2018@mail.sustech.edu.cn

Xiaochen Tan
School of Computer Science and Engineering
Northeastern University
Shenyang, China 110167
20206472@stu.neu.edu.cn

Binghong Wu
School of Automation
Beijing Institute of Technology
Beijing, China 100081
1120200744@bit.edu.cn

Boming Miao
College of Science
Northeastern University
Shenyang, China 110004
20191202@stu.neu.edu.cn

1 Background

1.1 Problem

The project is aimed to investigate semi-supervised learning algorithms based on the expectation maximization (EM) classification algorithm and hierarchical supervised learning algorithms on the dataset - Twenty Newsgroups text data.

Since real-world text datasets often contain very limited labels because reading and classifying articles manually is a tiring job, we are going to try to figure out how many labels we need in order to make the accuracy of the semi-supervised classifier achieve a satisfactory level, which classification algorithm should be used and how it should be inserted in the EM semi-supervised framework so that it works optimally.

Also, since the labels of the dataset are hierarchical, we plan as well to investigate how different types of hierarchical classifiers work on the dataset and how to build an efficient one.

1.2 Literature Survey

[**?**] proposes the EM algorithm for the first time and it states the mathematical derivation and computational method on how to use the algorithm to evaluate the maximum likelihood estimates. In this project, the basic idea of semi-supervised learning algorithm is based on the framework proposed from the paper.

[**?**] has summarized a typical semi-supervised text classification method based on the naive bayes algorithm and EM classification framework. The idea of this paper is actually the same as part of things done in the first half of the project.

Preprint. Under review.

[**?**] puts forward the method - Hierarchy-guided BERT with Global and Local hierarchies (HBGL) to fully exploit both structural and semantic information underlying the hierarchical structure. This algorithm is used when we try to build more efficient semi-supervised and hierarchical supervised classifiers and intended as a comparison against the semi-supervised learning algorithm based on the naive bayes classifier and EM framework.

[**?**] forms a basis to interpret the TF-IDF term weights as making relevance decisions. It simulates the local relevance decision-making for every location of a document, and combines all of these "local" relevance decisions as the "document-wide" relevance decision for the document. TF-IDF is used in the project as an important natural language processing (NLP) model so this paper is fundamental to the preprocessing part.

2 Methods

2.1 Approach

2.1.1 Data Preprocessing

1. TF-IDF encoder. TF-IDF is a traditional NLP model which transforms the text into a high-dimensional vector. TF represents "Term Frequency", and that is the number of times a term occurs in a document. However, because some term like "the" is so common, term frequency will tend to incorrectly emphasize documents which happen to use the word "the" more frequently. So, an inverse document frequency factor is incorporated which diminishes the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely. When we multiply TF and IDF, we get TF-IDF, if this number is bigger, we believe the word is more important in the article.
2. BERT encoder. BERT stands for "Bidirectional Encoder Representations from Transformers". Given pre-trained parameters, BERT maps the tokens from original text into vectors that represents words' meaning in the context using multi-head attention. The sentences representation are then built through concatenating word vectors. Considering that many texts in the 20 newsgroups data exceed the maximum length that BERT can take, truncation is made in order to limit and unify the length of input text.

2.1.2 Classification

1. Naive bayes classifier. Naive bayes is a typical classification model based on the Bayes theorem and the assumption of multivariate normality and independence. Given the class variable y and dependent feature vector x_1 through x_n, $P(y \mid x_1, \ldots, x_n) = \frac{P(y)P(x_1,\ldots,x_n|y)}{P(x_1,\ldots,x_n)}$ by the Bayes theorem. Using the independence assumption which states $x_1, \ldots x_n$ are mutually independent, $P(x_i|y, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = P(x_i|y)$ for all i, so $P(y \mid x_1, \ldots, x_n) = \frac{P(y)\prod_{i=1}^{n} P(x_i|y)}{P(x_1,\ldots,x_n)}$. Since $P(x_1, \ldots, x_n)$ is a constant given the input, the conditional probability satisfies $P(y \mid x_1, \ldots, x_n) \propto P(y)\prod_{i=1}^{n} P(x_i \mid y)$ and the classification rule becomes $\hat{y} = \arg_y \max P(y)\prod_{i=1}^{n} P(x_i \mid y)$ where $P(y)$ and $P(x_i|y)$ can be easily estimated from the training set.
2. BERT classifier. With an extra output layer, BERT is capable of classifying text into categories after fine-tuning the pre-trained parameters on 20 newsgroups data, due to its extraordinary performance in capturing features based on long-range context. The final hidden state corresponding to [CLS] token is used as the aggregate sequence representation for classification tasks.

2.1.3 EM Semi-supervised Learning for Classification

In statistics, an expectation–maximization (EM) algorithm is an iterative method to find (local) maximum likelihood or maximum a posterior (MAP) estimates of parameters in statistical models, where the model depends on unobserved latent variables. The EM iteration alternates between performing an expectation (E) step, which creates a function for the expectation of the log-likelihood evaluated using the current estimate for the parameters, and a maximization (M) step, which computes parameters maximizing the expected log-likelihood found on the E step. These parameter-estimates are then used to determine the distribution of the latent variables in the next E step.

To use the EM algorithm to conduct semi-supervised learning for classification, it is required to know what the E step and M step are.

Suppose a base classifier which can only be trained by supervised learning is denoted by $classifier$ and it can be trained by the method $classifier.\text{train}(\boldsymbol{X}, \boldsymbol{y})$ where $\boldsymbol{X}$ denotes the matrix of feature vectors of the training set and $\boldsymbol{y}$ denotes the vector of corresponding labels of them. Also suppose the classifier can be used to predict the labels of a given matrix of feature vectors $\boldsymbol{X}$ via the method $classifier.\text{predict}(\boldsymbol{X})$ whose return value is a vector of predicted labels $\boldsymbol{y}$.

In the training set, we denote the labeled data as $(\boldsymbol{X_l}, \boldsymbol{y_l})$ where $\boldsymbol{X_l}$ is the matrix of feature vectors and $\boldsymbol{y_l}$ is the vector of corresponding labels, and we denote the unlabeled data by $\boldsymbol{X_u}$ which is the matrix of feature vectors.

Algorithm 1 EM Semi-supervised Learning for Classification

Input: $classifier$, $(\boldsymbol{X_l}, \boldsymbol{y_l})$, $\boldsymbol{X_u}$, $iteration\_times$
Output: $classifier$

1: **function** EM_SEMI_SUPERVISED_LEARNING_FOR_CLASSIFICATION(*classifier*, $(\boldsymbol{X_l}, \boldsymbol{y_l})$, $\boldsymbol{X_u}$, *iteration_times*)
2: $classifier.\text{train}(\boldsymbol{X_l}, \boldsymbol{y_l})$
3: $\boldsymbol{X} \leftarrow \boldsymbol{X_l} \cup \boldsymbol{X_u}$
4: **for** $i = 1$ to $iteration\_times$ **do**
5: $\boldsymbol{y_u} \leftarrow classifier.\text{predict}(\boldsymbol{X_u})$
6: $\boldsymbol{y} \leftarrow \boldsymbol{y_l} \cup \boldsymbol{y_u}$
7: $classifier.\text{train}(\boldsymbol{X}, \boldsymbol{y})$
8: **end for**
9: **end function**

In the EM algorithm above, line 5 is the E step and line 7 is the M step. This framework works for all types of classifiers which are designed to do supervised learning. The iteration times can be adjusted to ensure the EM algorithm to converge and 10 is enough in most cases.

2.1.4 Hierarchical Classifier

The labels of the dataset have the hierarchical property. Figure 1 illustrates the structure of these hierarchical labels by a tree.

When a node on the tree has two or more children, we call it as a *key node*. Then we can easily find that all key nodes on the tree are as follows:

- *
- comp
- comp.sys
- misc
- rec
- rec.sport
- sci
- talk
- talk.politics

, where * represents the root of the tree.

So we come to a natural hierarchical classification method. First we build a classifier to classify all data into 7 classes under the tree root. Then we build a classifier to classify data under each key node at the first level to classify the data on the key node into nodes (classes) at the second level. If the node is not a key node but has one child, we can transmit data on it down to the second level directly. Repeat this process until all data reach the leaves of the tree and then the classification is finished.

The hierarchical classification method above requires us to build a classifier on every key node on the tree. However, it does not tell us which type of classifier should be used on each key node. For convenience, we can use the same type of classifier on each key node.

2.2 Rationale

The idea of using EM algorithm to do semi-supervised learning for classification is very common in the field of text classification and we have found many papers doing the same thing. Real-world datasets often contain few labels since manually labelling the categories of so many articles is a time-consuming affair and the cost of it is also high. As a result, usually we have only a few labels and the number of unlabeled data is much larger than that of the labeled ones. Under such circumstances, semi-supervised learning is important for the classifiers.

When we first saw the labels of the dataset, it can be found that the labels of the data form a perfect tree structure. But few researchers have constructed hierarchical classifiers on the tree to classify the data while we were doing literature survey. So the idea of hierarchical classification appears.

TF-IDF and naive bayes are the methods of NLP and classification suggested by the provider of the dataset so it is suitable for doing supervised learning on the dataset. The use of BERT, a deep learning NLP model, is due to the existing work that has been done on the dataset which shows that the accuracy of non-hierarchical classification can achieve 95%. So when we use it to do semi-supervised learning for classification and hierarchical classification, it is expected to be quite accurate and powerful.

3 Plan & Experiment

3.1 Datasets

Our algorithms are run on 20 newsgroups, which is a data set containing 1000 text articles posted to each of 20 online newsgroups, for a total of 20,000 articles. The label of each article is which of the 20 newsgroups it belongs to. The newsgroups (labels) are hierarchically organized (e.g., "sports", "hockey").

Figure 1 presents all hierarchical labels through a tree. It can be clearly seen that the labels are logically hierarchical and organized in such a structure.

3.2 Hypotheses

We are going to test two main hypotheses below:

1. When limited amount of labels of each class are kept and other labels are ignored, the semi-supervised learning for classification via EM algorithm can achieve high accuracy although the unlabeled data are much more than the labeled ones.
2. When choosing a specific classification method and appropriately constructing a hierarchical classifier, the accuracy of the hierarchical classifier (composed of a bunch of classifiers) can be higher than that of the non-hierarchical classifier with the same type of basic method for classification.

3.3 Experimental Design

1. Construct a typical naive bayes classifier to do non-hierarchical classification. Test the classifier via both train-test split and 5-fold cross validation.
2. Construct a hierarchical classifier whose inner classifiers are based on naive bayes. Test the classifier via 5-fold cross validation. Compare the results to the previous non-hierarchical classifier. (This is to test the second hypothesis.)
3. Use the EM algorithm combined with a naive bayes classifier to do semi-supervised training. Train another naive bayes classifier by the data with labels kept only. Compare them to verify that the unlabeled data are useful. Use 5-fold cross validation to evaluate the accuracies. Try to change the number of labels kept of each class. (This is to test the first hypothesis.)
4. Construct a BERT classifier to do non-hierarchical classification. Test the classifier via both train-test split and 5-fold cross validation.
5. Construct a hierarchical classifier whose inner classifiers are based on BERT. Test the classifier via 5-fold cross validation. Compare the results to the previous non-hierarchical classifier. (This is to test the second hypothesis.)

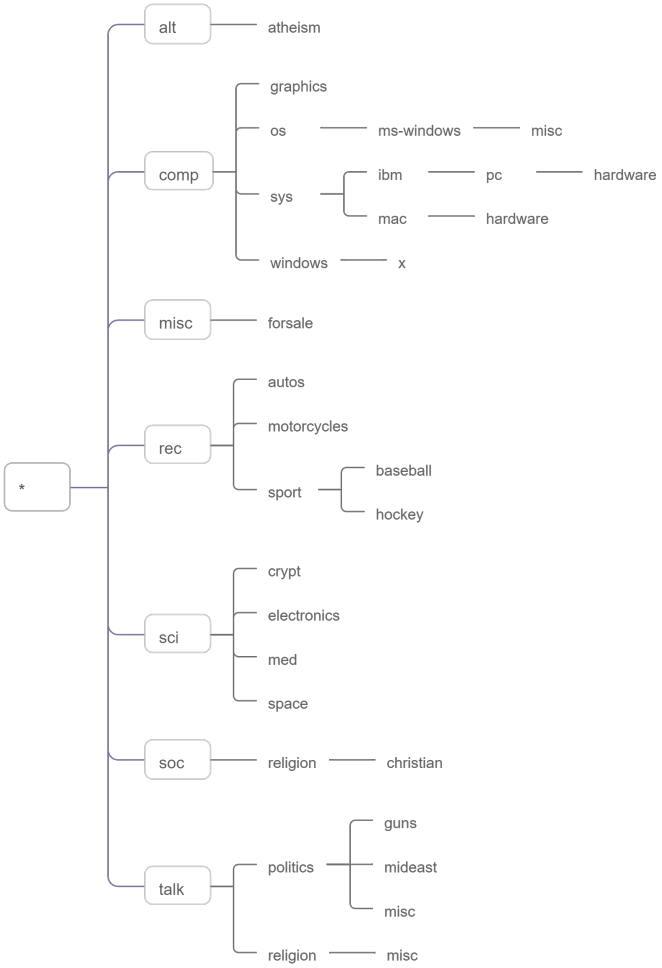

Figure 1: Tree of Hierarchical Labels

6. Use the EM algorithm combined with a BERT classifier to do semi-supervised training. Train another BERT classifier by the data with labels kept only. Compare them to verify that the unlabeled data are useful. Use 5-fold cross validation to evaluate the accuracies. Try to change the number of labels kept of each class. (This is to test the first hypothesis.)
7. Other experiments if needed.

4 Results

4.1 Results

The goals we have completed for our midterm milestone are as follows:

- A typical non-hierarchical naive bayes classifier has been built. The accuracy on train-test split is 0.7002124269782263 and the average accuracy of 5-fold cross validation is 0.7719402665750824.
- A hierarchical classifier whose inner classifiers are based on naive bayes have been built. The average accuracy of 5-fold cross validation is 0.7392549579038266.
- A non-hierarchical BERT classifier has been built. The accuracy is 0.95.

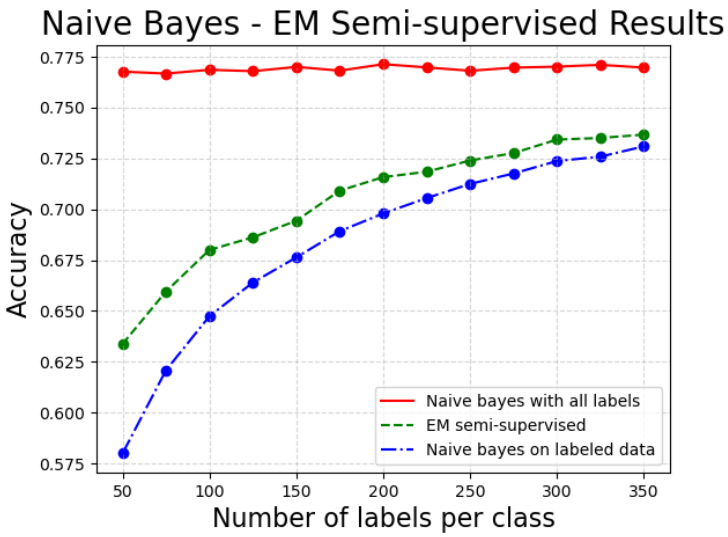

Figure 2: Naive Bayes - EM Semi-supervised Results

- Naive bayes classifiers trained by semi-supervised EM algorithm and the corresponding naive bayes classifiers trained by labeled data kept only are built. The number of labels kept of each class varies from 50 to 350 with the step length 25. 5-fold cross validation is done and average accuracies are computed. Figure 2 illustrates the results where the red line indicates the accuracies of naive bayes classifiers trained by all labels, the green line indicates the accuracies of naive bayes classifiers trained by the semi-supervised EM framework and the blue line indicates the accuracies of naive bayes classifiers trained by data with labels kept only.

4.2 Critical Evaluation

The accuracy of the hierarchical naive bayes classifier is lower than that of the non-hierarchical one, which is the opposite of our second hypothesis. The phenomenon could be owing to the fact that each classifier on the tree is train by less data so that it cannot distinguish different classes with less information given.

The results of the semi-supervised EM algorithm combined with naive bayes classifiers is as expected and in good agreement with our first hypothesis.

5 Conclusions

At this stage, the first hypothesis is more likely to be true but the second seems to be false. We will try to use BERT to increase the accuracy of the semi-supervised learning and try to use BERT to test the second hypothesis in the following weeks.