Exploring the Techniques to Improve the Performance of Naïve Bayes for Spam Email Filtering and Email Categorization

Chao Wang and Chao Li

1 INTRODUCTION

In the domain of personal email messages, text categorization methods have been widely applied to the problem of spam filtering and email categorizing into folders. The spam filtering task is a two-value classification: spam and non-spam, while the email categorizing is considered as a multi-class categorization problem like what Gmail does.

Naïve Bayes classifier is a simple generative classification algorithm that has reasonably good performance. But it makes strong assumption that all words in a document are independent. This assumption is clearly lack of context information which violates natural language rules [5]. Therefore, Naïve Bayes classifier needs some modification to improve its performance.

In our project, we seek to implement and compare some existing approaches which can improve the performance of Naïve Bayes in spam email filtering and email categorization. Through such implementation and comparison, we hope we can find our own ways to improve the classification accuracy by doing some modification to Naïve Bayes.

2 THE ALGORITHMS

2.1 NAÏVE BAYES CLASSIFIER

Given a document *d*, the Naïve Bayes classiier selects the class with maximum posterior probability as the output class:

$$c^*(d) = \arg\max_j \; p(c_j)\, p(d \mid c_j)$$

And according to conditional independence assumption, features are independent of each other. Therefore, the distribution of document in each class can be evaluated as

$$p(d \mid c_j) = p(w_1 \mid c_j)\, p(w_2 \mid c_j) \cdots p(w_{|d|} \mid c_j)$$

And the final class of the input document can be expressed as

$$c^*(d) = \arg\max_j \; p(c_j) \prod_{i=1}^{|d|} p(w_i \mid c_j)$$

During the testing process, in order to get rid of the zero probability caused by the new word, the add-one smoothing is often used to compute the conditional probability

$$p(w_k \mid c_j) = \frac{n_k + 1}{n + |Vocabulary|}$$

where n_k is the number of occurrence of w_k in *d*.

2.2 IMPROVEMENT OF NAÏVE BAYES CLASSIFIER

The conditional independence assumption used in Naive Bayes model treats each occurrence of a word in a document independently of any other occurrence of the same word. In reality, however, multiple occurrences of the same word in a document are not independent. When a word occurs once, it is likely to occur again, i.e. the probability of the second occurrence is much higher than that of the first occurrence. Therefore, this circumstance may results in a large underestimation of the probability of documents with multiple occurrences of the same word [5].

In order to better fit the document data to the probabilistic model, the Ref. [5] takes the word frequencies in all documents into account to improve the traditional Naïve Bayes model. Considering that the method proposed in [5] does not have a specific name, we call this method *Improved Bayes* in this report. In this Improved Bayes method, the conditional probability is computed as

$$p(w_k \mid c_j) = \frac{d(c_j, w_t) + 1}{\sum_{i=1}^{|V|} d(c_j, w_i) + |Vocabulary|}$$

where $d(c_j, w_t)$ is the number of documents with class c_j that contains the word w_t. This modification treats the probability of a word in a class as the "document frequency" of this word in all documents in this class. Based on this modification, the final category of input document *d* is also obtained as the class with maximum posterior probability of all classes.

2.3 LOCALLY WEIGHTED BAYES

The basic idea of locally weighted Naïve Bayes is to obtain a classification from the naive Bayes model taking the attribute values of the test instance as input [4].

The subsets of data used to train each locally weighted naive Bayes model are determined by a nearest neighbors algorithm. A user-specified parameter k controls how many instances are used. Let d_i be the Euclidean distance to the i*th* nearest neighbor x_i. They assume that all attributes have been normalized to lie between zero and one before the distance is computed, and that nominal attributes have been binarized. Let f be a weighting function with f(y) = 0 for all $y \geq 1$. They then set the weight w_i of each instance xi to $w_i = f\left(\frac{d_i}{d_k}\right)$ This means that instance x_k receives weight zero, all instances that are further away from the test instance also receive weight zero, and an instance identical to the test instance receives weight one. Any monotonically decreasing function with the above property is a

candidate weighting function. In their experiments they used a linear weighting function flinear defined as $f_{linear}(y) = 1 - y \; for \; y \in [0,1]$

In other words, they let the weight decrease linearly with the distance. And then they can use these computed weights to modify the conditional probability of attribute ai and the prior probability of class cl.

You can find more details in the reference paper [4]. We did some modification of this kind of method to fit our email categorization problem. The feature set is the set of tokens we get from training files. The distance between two instances is the Euclidean distance of frequency of tokens.

2.4 SELECTIVE NAÏVE BAYES

The goal of *selective Naïve Bayes* is to modify the naive Bayesian classifier to achieve improved accuracy in domains with redundant attributes. The selective Bayesian classifier is a variant of the naive method that uses only a subset of the given attributes in making predictions. In other words, the performance component of the algorithm computes $p(d|C_j)$ as the product of conditional probabilities, $p(w_j|C_j)$, for selected attributes w_j from the original feature set. The learning component of the selective classifier augments the original algorithm with the ability to exclude attributes that introduce dependencies. This process consists of a search through the space of attribute subsets [6].

The author makes a number of choices in designing the search process. First, the direction of search is a forward selection method which starts with the empty set and successively adds attributes until the accuracy doesn't degrade. Second, they use a greedy method to traverse the space. That is, at each point in the search, the algorithm considers all local changes to the current set of attributes, makes its best selection, and never reconsiders this choice. This gives a worst-case time complexity of $O(a^2)$ for a attributes. Third, they consider the leave-one-out technique for estimating accuracy from the training set, since this is the most accurate method of cross validation. Finally, they adopt a more conservative strategy of continuing to select attributes as long as they do not degrade accuracy for halting the search process.

3 EXPERIMENTS

3.1 SPAM EMAIL FILTERING

In the spam email filtering work, we present results on the Apache Spam email dataset [3]. The Apache Spam email data contains the following datasets overall:

- **spam**: 500 spam messages, all received from non-spam-trap sources.
- **easy_ham**: 2500 non-spam messages. These are typically quite easy to differentiate from spam, since they frequently do not contain any spammish signatures (like HTML etc). This data is easy to be detected as non-spam.
- **hard_ham**: 250 non-spam messages which are closer in many respects to typical spam: use of HTML, unusual HTML markup, coloured text, "spammish-sounding" phrases etc. This database is very hard to be detected as non-spam with respect to **easy_ham**.
- **easy_ham_2**: 1400 non-spam messages. A more recent addition to the set.
- **spam_2**: 1397 spam messages. Again, more recent.

And in our experiment, we use two recent datasets **easy_ham_2** and **spam_2** as training data, since they have almost the same number of emails and more recent with respect to **spam** and **easy_ham.** Based on the training data we choose, we try to filter the spam emails from the testing datasets **easy_ham** and **spam**. The reason we abandon **hard_ham** is that, the emails in this dataset are very hard to be detected and the accuracy is below 50%, no matter which method we choose in our experiment. By the way, we only take the first 500 emails of **easy_ham** as testing data in order to have the same number of non-spam emails as well as the spam emails in the dataset **spam**. In summary, we train a dataset with about 1400 spam and non-spam emails by each classifier, and test the classifier on the testing data with 500 spam and non-spam emails.

Table 1 Spam email filtering results with different training datasets

No.	Training Data	Testing Data	Accuracy			
			Naïve Bayes	Improved Bayes	Weighted Bayes	Selective Bayes
1	**spam_2**, **easy_ham_2**	**spam**	0.774	0.826	0.844	0.714
		easy_ham	0.996	0.996	0.996	0.802
		together	0.885	0.911	0.920	0.758
2	half of **spam_2**, **easy_ham_2,**	**spam**	0.702	0.762	0.790	0.496
		easy_ham	0.996	0.994	0.996	0.992
		together	0.849	0.878	0.893	0.744
3	**spam_2**, half of **easy_ham_2**	**spam**	0.792	0.854	0.867	0.734
		easy_ham	0.99	0.988	0.991	0.767
		together	0.891	0.921	0.929	0.751
4	half of **spam_2**, half of **easy_ham_2**	**spam**	0.718	0.824	0.835	0.362
		easy_ham	0.996	0.984	0.996	0.998
		together	0.857	0.904	0.916	0.680

Table 1 shows the accuracy results of all methods introduced in Chapter 2 on the testing datasets with different size of training datasets. We selected four different groups to train each classifier separately and test each classifier on the **spam** dataset**, easy_ham** dataset and the two sets together**.** From the result we can see that the Improved Bayes and Weighted Bayes classifiers are both more effectively than traditional Naïve Bayes classifier. This demonstrates that these two classifiers indeed improve the traditional Naïve Bayes Classifier by their techniques shown in Chapter 2, respectively. However, the results of Selective Bayes method is not very well. We think the reason is that we only use part of the words with most frequencies from all the vocabulary during the training process. Then, we use this model to test the testing dataset with many new words. Therefore, the training model cannot deal with some testing datasets with many new words, which makes Selective Bayes method unsuitable for this case. Figure 1 to 3 shows the accuracy-training size curves of all four methods on different testing datasets. These figures could clearly illustrate the effectiveness of Improved Bayes and Locally Weighted Bayes.

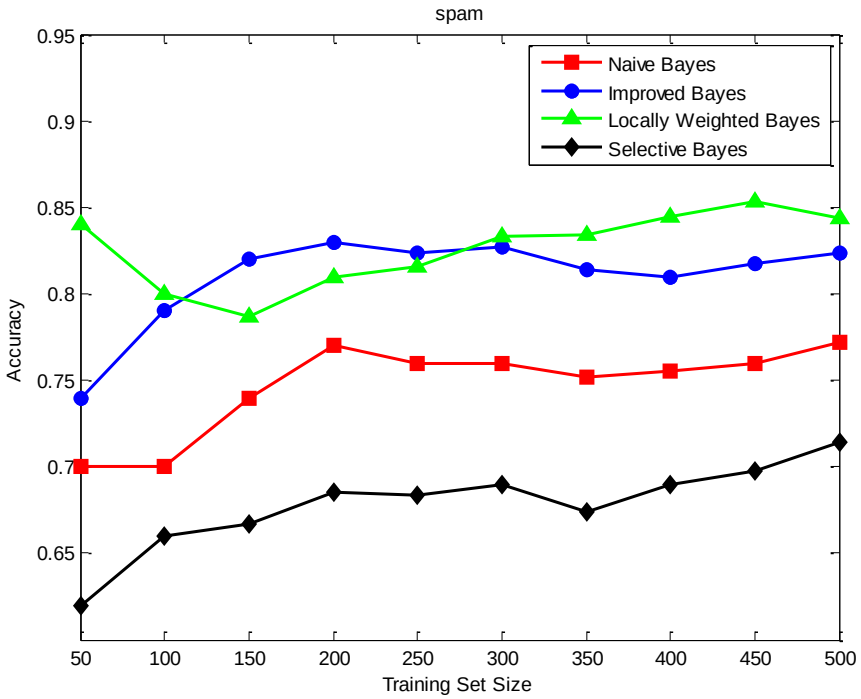

Figure 1 Accuracy-Training Size curves on **spam** dataset by all methods

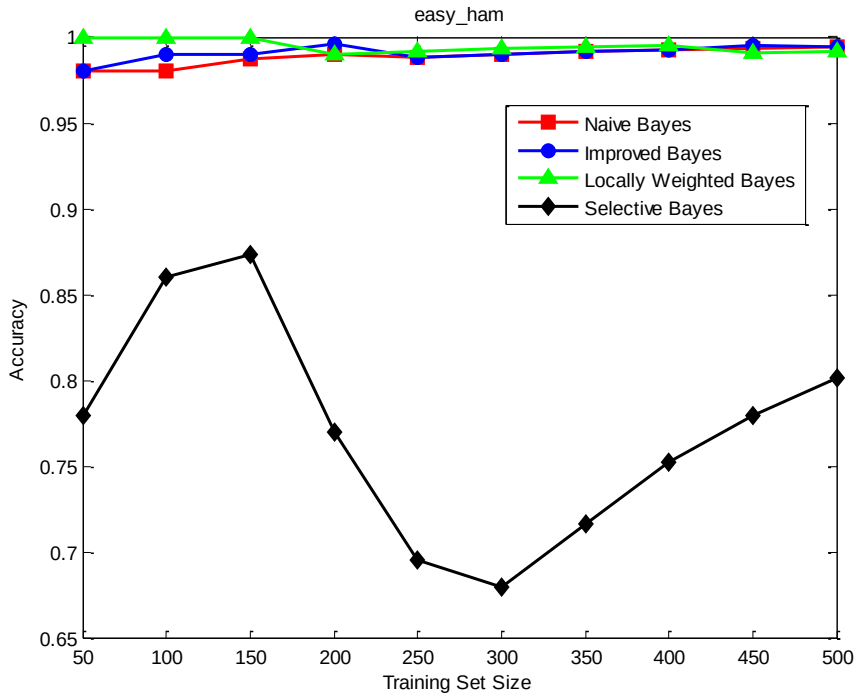

Figure 2 Accuracy-Training Size curves on **easy_ham** dataset by all methods

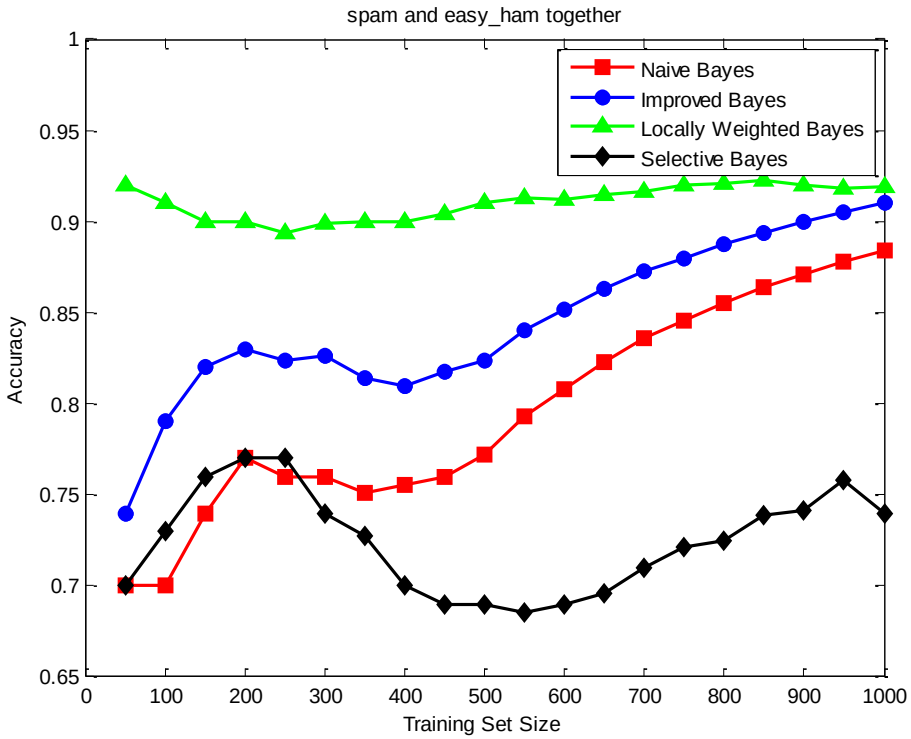

Figure 3 Accuracy-Training Size curves on **spam** and **easy_ham** datasets together by all methods

3.2 EMAIL CATEGORIZATION

In the email categorization work, we present results on the Enron email dataset [2]. The Enron dataset was collected and prepared by the CALO Project (A Cognitive Assistant that Learns and Organizes). The raw Enron corpus contains 619,446 messages belonging to 158 users organized into folders. The corpus contains a total of about 0.5M messages. Although the size of the dataset is large, many users' folders are sparsely populated. In our experiment, we select four popular categorization: **calendar**, **discussion_threads**, **personal** and **university** from several different persons in the Enron data. Each category is described as follows:

- **calendar**: emails related to calendar, such as time and work schedule, meeting, date, etc;
- **discussion_threads**: emails related to meeting, talk and discussion content;
- **personal**: private emails;
- **university**: emails related to school, university and college, i.e. emails from or to university staff or students.

The reason we choose these four categories is that the data of these four categories is abundant and easy to find in Enron datasets, and also very common in normal life. Another reason is that different persons in Enron dataset share only a few common categories, of which these four are the most frequent ones and also contains the most number of datasets.

Table 2 Traning and Testing data in email categorization from Enron datasets

No.	Category	Training data	Number of emails in training data	Testing data	Number of emails in testing data
1	**calendar**	beck-s, corman-s haedicke-m, kaminski-v, lavorato-j, mccarty-d	560	whalley-g, white-s	292
2	**discussion_threads**	beck-s, lokay-m, mann-k	8508	mcconnell-m, love-p	1275
3	**personal**	kaminski-v, farmer-d, griffith-j, williams-w3	643	beck-s, davis-d	89
4	**university**	kaminski-v	367	rogers-b, kean-s	36

Table 2 shows the source and size of our training and testing data from Enron in our experiments. Here each data name, such as “beck-s”, is the folder name, also the person name from the Enron dataset. And in this email categorization experiment, we use all the emails of our chosen training datasets with different number of emails in each category. And according to the results of spam email filtering experiment in Chapter 3.1, the result of Selective Bayes is not good, so we abandon this method in email categorization and just use the other three methods for testing.

Table 3 Email categorization

No.	Testing data	Accuracy		
		Naïve Bayes	Improved Bayes	Weighted Bayes
1	**calendar** (whalley-g)	1	1	1
2	**calendar** (white-s)	1	0.993007	1
3	**discussion_threads** (mcconnell-m)	0.946915	0.978479	0.910254
4	**discussion_threads** (love-p)	0.693772	0.795848	0.722773
5	**personal** (beck-s)	0.166667	0.0833333	0.612245
6	**personal** (davis-d)	0.219512	0.0731707	0.47619
7	**university** (rogers-b and kean-s)	0.888889	0.416667	0.864865

Table 3 shows the accuracies of these three methods on different testing data. From the table we can see that the categorization results of **calendar, discussion_threads** and **university** by Improved Bayes and Locally Weighted Bayes are satisfying, but the **personal** case is very bad for Naïve Bayes and Improved Bayes. The problem is very likely be caused by the small size of data of **personal** category and large size of **discussion_threads** category. And it is possibly that many personal emails contains key words or frequent words from other categories, which makes these emails very hard to be categorized.

4 REFERENCE

[1] Ron Bekkerman. Automatic categorization of email into folders: Benchmark experiments on enron and sri corpora. 2004.
[2] Enron Email Dataset. http://www.cs.cmu.edu/~enron/.
[3] Apache Spam email dataset. http://spamassassin.apache.org/publiccorpus/.
[4] Eibe Frank, Mark Hall, and Bernhard Pfahringer. Locally weighted naive bayes. In Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence, pages 249–256. Morgan Kaufmann Publishers Inc., 2002.
[5] Karl-Michael Schneider. Techniques for improving the performance of naive bayes for text classification. In Computational Linguistics and Intelligent Text Processing, pages 682–693. Springer, 2005.
[6] Pat Langley and Stephanie Sage, Induction of Selective Bayesian Classifiers, CONFERENCE ON UNCERTAINTY IN ARTIFICIAL INTELLIGENCE, pages 399—406. Morgan Kaufmann, 1994.