Email Classification Using Co-EM
Project Report

Aravind Sankar (CS11B033)
Sriram V (CS11B058)
Group 11

Table of Contents

1 Introduction 3
2 Literature Survey 4
3 Methodology 5
4 Experiments and Results 7
5 Conclusions and Future Work 7
6 References 8

1 Introduction

The broad area of Text Classification has been a topic of research for a very long time now, and has been well researched especially in the case of supervised learning of classifiers. The traditional problems solved in the past by text classification include automatically cataloging news articles, web pages and spam detection in emails, etc. Text classification as defined by [1], is the activity of labeling natural language texts with thematic categories from a predefined set. Most of the earlier approaches assume the existence of sufficient training data, which has labeled instances of documents with their corresponding labels. As the process of labeling here requires human effort, our focus is therefore directed towards tackling the problem of classification of limited labeled data, otherwise referred to as semi-supervised learning, assuming we have a large number of unlabeled documents.

Although text classification is a problem that has been looked into right from the 1960s, email classification is highly relevant in today's fast-paced world, where one does not have the time to sift through vast amounts of mail but would rather like information clustered together, so that one may disregard the irrelevant ones such as promotional offers, and focus attention on the important ones at hand. Triaging has gained a lot of focus in recent times, with GMail launching personalized mail classification, and with other services such as Mailbox and Boxer focusing on improving user efficiency. Thus, this problem is of immense interest in recent times, and improving performance and accuracy, without existence of enough labeled email, is of prime concern.

One of the most popular problems addressed by email classification is that of spam filtering, i.e. separating emails into spam and ham. This problem has been widely addressed, with various classifiers and feature selection and extraction methods experimented with. Prior work has shown great results on the Enron corpus, which is the most popular dataset used for this problem. We wish to focus on the problem of categorizing emails into categories, which would help a user identify the emails important to him. If we have a idea about the areas of interest of a user, then the emails could be easily filtered according to the category and the relevant mails can be displayed first, hence making it personalized to the user.

Before we describe the exact problem we solve and our approach, a brief survey of existing techniques to solve email classification is shown in the following section.

2 Literature Survey

As mentioned earlier, spam filtering was one of the popular problems addressed. Initially, the mails were classified as spam using the presence of certain predefined tokens in the email body. As these methods require manual specification of these tokens, the focus shifted to learning, which required the mails to defined as a set of features. Among the various feature extraction techniques, the Bag-of-Words feature representation has been widely used. This representation gives a set of binary features, for each mail, where each feature shows the presence or absence of a particular term. Some of the other methods such as [2] include other spam features, that are domain specific. Other techniques such as [3] have used temporal features which identify temporal relations in an email sequence in the form of temporal sequential patterns using the timestamps. Behavioral features have also been identified by [4], by looking at outgoing messages from a user.

Once the features have been extracted from the emails, the standard supervised classification techniques have been put to use. The most predominant ones, being the Naive Bayes and the SVM classifiers. The Naive Bayes classifier assumes that each of the features $f_1, f_2, ...f_n$ of the email are conditionally independent given the class label C. Surprisingly, this has proved to quite effective for email classification. Studies have been performed, comparing different classification approaches, eg: the Naive Bayes has been compared with Rule learning approaches by [5] and has been shown that Naive Bayes performs much better. SVM's have also been widely used, which try to separate the two classes by an optimum hyperplane. Another comparative study by [6] showed the effectiveness of a decision tree classifier in spam filtering in comparison to SVM's and Neural Networks.

The unsupervised approaches for email classification are broadly based upon 2 techniques, namely EM (Expectation Maximization) and Co-Training. These approaches typically start with a very small set of labeled instances and slowly label each of the unlabeled instances in subsequent iterations. The EM algorithm has been traditionally used to solve incomplete data problems, in many different areas. Here, the unlabeled emails constitute the incomplete data of the problem. The EM algorithm for text classification first introduced by [7] first trains a classifier with only the available labeled documents, and then uses the classifier to assign probabilistically-weighted class labels to each unlabeled document by calculating the expectation of the missing class labels. The idea of co-training, on the other hand is different. Co-training was used in email classification first by [8]. It starts with just a few labeled emails, and builds an initial weak classifier. It assumes the existence of 2 sets of redundant features but sufficient on their own for correct classification. We can see that in case of email, we can think of the email body and email header (including subject) as a set of 2 independent feature sets. [8] have shown that SVM's have performed better than the Naive Bayes classifier when Co-training is used.

Our approach draws inspiration from Co-Training and EM, and uses the Co-EM approach, which was proposed by [9]. To the best of our knowledge, Co-EM has not been experimented with, for the purpose of email classification. We use Co-EM for email classification, and compare with Co-Training and evaluate it's effectiveness.

3 Methodology

Before going into the details of our approach, we provide a brief description of the Co-Training algorithm, as applied to email classification by [8].

Co-Training :

Given 2 redundantly sufficient sets of features F_1 and F_2, L - the set of labeled mails (typically small) and U - the set of unlabeled mails, the algorithm switches between 2 classifiers and add mails to L, in each iteration.

```
Data: $L$,$U$,$k$ (parameter)
Result: Fully labeled set $L$ (containing mails of $U$)
while Any documents are present in $U$ do
    Learn classifier $C_1$ from $F_1$ on $L$;
    Learn classifier $C_1$ from $F_2$ on $L$;
    for Classifier $C$ in $\{C_1, C_2\}$ do
        $C$ labels mails in $U$ based on it's feature set.
        Choose $k$ most confidently predicted mails from each class, in $U$.
        Remove this set of mails from $U$ and add them to $L$, with the
        labels predicted by $C$.
    end
end
```

Algorithm 1: Co-Training

Here, the set F_1 of features used are extracted from the body of the mail, and the set F_2 from the mail header, which includes information about the sender, receiver and the subject of the mail. As these two pieces of information are mostly disjoint, and as subject and sender may prove to be a good discriminator in classifying emails, we choose this split.

The Co-EM algorithm proposed by [9] is a hybrid of EM and Co-Training, and is an iterative algorithm that uses the feature-split.

```
Data: $L$,$U$
Result: Fully labeled set $U$
Learn classifier $C_1$ from $F_1$ on $L$;
$C_1$ labels mails in $U$.
while $C_1$ and $C_2$ haven't converged do
    Learn classifier $C_2$ from $F_2$ on $L$ and $U$ (based on labels given by $C_1$);
    $C_2$ labels mails in $U$;
    Learn classifier $C_1$ from $F_1$ on $L$ and $U$ (based on labels given by $C_2$);
    $C_1$ labels mails in $U$;
end
```

Algorithm 2: Co-EM

Dataset :

We used the mails which we get on our csemail id's, and took only those mails that involved any talks, seminars or lectures that happen in the department. Our goal was to classify these mails into categories such as Machine Learning, Architecture, Networks, Theory, etc. This was motivated by the fact that we get tons of mails about PhD seminar lectures, about which we'd actually be interested only in a few of them, based on our research interests. So, if we're able to classify these mails into categories, corresponding to broad areas in computer science, the user's mail box can personalized according to his interests.
Our dataset consisted of around 250 such mails, distributed into 8 major categories. We labeled these mails manually and used a small portion of them as L and the remaining as U. So, our problem was a multi-class classification problem, in contrast to prior work on email classification, which focused mainly on 2 class spam classification.

Features used :

a) Initially, we started out with a Bag-of-Words representation of the features of the mails (for both feature sets). We remove all the stopwords such as *and*,*the*,etc and then find the features. In this representation, we indicate the presence or absence of a word in the mail, with 0/1.

b) Another feature extraction technique that we used, was a Term-Document matrix where the emails are the documents and the words that occur in the email are the terms. The entries of the matrix are the counts of each word in that email. Here, we use the *TfIdf* measure to obtain a feature representation of each mail in terms of it's constituent words, by assigning a weight to each word. These weights show the discriminative power of each of the terms, as we take a product of *Tf* the number of times the term occurs in that document, and the *Idf*, which is the inverse of the number of documents in which the word occurs. This can prove to particularly useful in email classification because, we generally have specific words that identify a category. An example might be the occurrence of words such as *wireless*, or *communication*, which would be present a lot in the mails of networks category.

c) LSI (Latent Semantic Indexing) was also another technique used to extract features. Again we have a term document matrix, which we transform using SVD to a new concept space, where both the terms and the documents are represented uniformly. This technique can be useful to capture the occurrences of terms which are similar in meaning and occur together.

Classification :

The 2 most popular classifiers, which were used earlier, were used in the Co-EM algorithm. These are the Naive Bayes and SVM classifiers.

a) **Naive Bayes classifier :**
b) **SVM (Support Vector Machine) classifier :**

4 Experiments and Results

The experiments were performed by considering 40 mails in L, and the rest around 200 in U. The

5 Conclusions and Future Work

Conclusions :
Future Work :

[10] introduces Co-EMT, a multi-view algorithm which combines semi-supervised and active-learning, to handle classification problems with incompatible, correlated views. This technique could be applied to the problem of email classification, and its performance could be compared with the proposed as well as the existing semi-supervised learning methods. This paper by [10] makes use of Naive Bayes as the underlying algorithm, but SVMs too could also be explored as an alternative classifier. This is very likely to yield much better results, as inputs from user are used at various stages, when the system needs more information, as a part of active learning. As active learning is used, the number of mails that need to be labeled by hand will be greatly reduced.

The problem that we solved was just a multi-class classification problem. We also observed that some of the mails could pertain to more than research interest area. An example of this may be, say a mail about parallel programming on multi-core systems. This could possibly be related to both the Architecture category, as well an Algorithms category. So, this becomes a multi-label classification problem, which becomes much harder to solve. This line could be explored in the future.

6 References

References

[1] Sebastiani, Fabrizio. "Machine learning in automated text categorization." ACM computing surveys (CSUR) 34.1 (2002): 1-47.

[2] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz, "A Bayesian Approach to Filtering Junk E-Mail", In Proc. of the AAAI Workshop on Learning for Text Categorization, 1998.

[3] Kiritchenko, Svetlana, Stan Matwin, and Suhayya Abu-Hakima. "Email classification with temporal features." Intelligent Information Processing and Web Mining. Springer Berlin Heidelberg, 2004. 523-533.

[4] Martin, Steve, et al. "Analyzing Behavioral Features for Email Classification." CEAS. 2005.

[5] Provost, Jefferson. "Naive-Bayes vs. Rule-Learning in Classification of Email." University of Texas at Austin (1999).

[6] Youn, Seongwook, and Dennis McLeod. "A comparative study for email classification." Advances and Innovations in Systems, Computing Sciences and Software Engineering. Springer Netherlands, 2007. 387-391.

[7] Nigam, Kamal, et al. "Text classification from labeled and unlabeled documents using EM." Machine learning 39.2-3 (2000): 103-134.

[8] Email classification with co-training. Svetlana Kiritchenko, Stan Matwin. In Proceedings of the 2001 conference of the Centre for Advanced Studies on Collaborative Research, CASCON 2001.

[9] Analyzing the Effectiveness and Applicability of Co-training. Kamal Nigam, Rayid Ghani. In Proceedings of the Ninth International Conference on Information and Knowledge Management, CIKM 2000.

[10] Active + Semi-Supervised Learning = Robust Multi-View Learning. Muslea, Ion, Steven Minton, Craig A. Knoblock. In Proceedings of the International Conference on Machine Learning, 2002.