SPAM FILTERING USING K - NN

S.Ananthi
Senior Lecturer, Dept. of Applied Sciences,
Sethu Institute of Technology,
Viruthunagar (Dt)

Dr. S. Sathyabama
Professor, Dept. of MCA,
K.S. Rangasamy College of Technology
Tiruchengode-637 215

Abstract - This project performs a survey on the current spam filtering techniques, explores the use of k-Nearest-Neighbor algorithm as the basis for personalized spam filters. Several other classifiers such as Naive Bayesian classifier, Random Forest Tree, and Heuristic rules are combined to construct hybrid spam filtering systems to, if possible, improve the performance of classification. At the same time, heavy experiments are performed in preprocessing steps to compare their impacts on the same algorithm and the results are reported. Finally, this project participates spam filtering using k-NN algorithm.

INTRODUCTION

According to www.dictionary.com, spam is "unsolicited e-mail, often of a commercial nature, sent indiscriminately to multiple mailing lists, individuals, or newsgroups". Legitimate email, also called ham, on the opposite, is the email we want. It is reported that the first spam appeared in 1970s.Nowadays, spam is a problem for everyone with e-mail. Spam Cop, a web site to help users to report spam, estimated that between 60 and 80% of all emails is spam. There are hundreds of millions of spam per week and the number is still growing constantly. It is inconvenient and time-consuming to read and delete spam manually. Spammers bear little or no cost for distribution, yet normal receivers are forced to spend substantial time and effort deleting unwanted emails from their mailboxes. Fortunately, help has arrived. In the recent past, a large number of freely available software helps users to filter out annoying emails in a considerably good performance. Such software is called spam filter. The following section introduces the typical use of a spam filter, a categorization of spam filters, and a unified model of spam filtration.

SPAM FILTER

"A spam filter is a piece of software which takes an input of an email message. For its output, it might pass the message through unchanged for delivery to the user's mailbox, it might redirect the message for delivery elsewhere, or it might even throw the message away. Some spam filters are able to edit message during processing".

ALGORITHM APPLICATION

K - NN Algorithm

The "k"-nearest neighbors algorithm'' is amongst the simplest of all [machine learning] algorithms. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its "k" nearest neighbors. ''k" is a positive [integer], typically small. If "k" = 1, then the object is simply assigned to the class of its nearest neighbor. In binary (two class) classification problems, it is helpful to choose "k" to be an odd number as this avoids tied votes. The same method can be used for regression, by simply assigning the property value for the object to be the average of the values of its "k" nearest neighbors. It can be useful to weight the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones.

The neighbors are taken from a set of objects for which the correct classification (or, in the case of regression, the value of the property) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required. In order to identify neighbors, the objects are represented by position vectors in a multidimensional feature space. It is usual to use the Euclidean distance, though other distance measures, such as the [[Manhattan distance]] could in principle be used instead. The "k"-nearest neighbor algorithm is sensitive to the local structure of the data. The test sample (green circle) should be classified either to the first class of blue squares or to the second class of red triangles. If "k = 3" it is classified to the second class because there are 2 triangles and only 1 square inside the inner circle. If "k = 5" it is classified to first class (3 squares vs. 2 triangles inside the outer circle).]

The training examples are vectors in a multidimensional feature space. The space is partitioned into regions by locations and labels of the training samples. A point in the space is assigned to the class "c" if it is the most frequent class label among the "k" nearest training samples. Usually Euclidean distance is used as the distance metric, however this will only work with numerical values. In cases such as text classification another metric, such as the "'overlap metric"' or Hamming distance can be used. The training phase of the algorithm consists only of storing the feature vectors and class labels of the training samples. In the actual classification phase, the test sample (whose class is not known) is represented as a vector in the feature space. Distances from the new vector to all stored vectors are computed and "k" closest samples are selected. There are a number of ways to classify the new vector to a particular class, one of the most used techniques is to predict the new vector to the most common class amongst the K nearest neighbors. A major drawback to using this technique to classify a new vector to a class is that the classes with the more frequent examples tend to dominate the prediction of the new vector, as they tend to come up in the K nearest neighbors when the neighbors are computed due to their large number. One of the ways to overcome this problem is to take into account the distance of each K nearest neighbors with the new vector that is to be classified and predict the class of the new vector based on these distances.

The best choice of "k" depends upon the data; generally, larger values of "k" reduce the effect of noise on the classification, but make boundaries between classes less distinct. A good "k" can be selected by various [[heuristic (computer science)|heuristic]] techniques, for example, cross-validation. The special case where the class is predicted to be the class of the closest training sample (i.e. when "k" = 1) is called the nearest neighbor algorithm. The accuracy of the "k"-NN algorithm can be severely degraded by the presence of noisy or irrelevant features, or if the feature scales are not consistent with their importance. Much research effort has been put into feature selection selecting or scaling features to improve classification. A particularly popular approach is the use of evolutionary algorithms to optimize feature scaling. Another popular approach is to scale features by the mutual information of the training data with the training classes.

The naive version of the algorithm is easy to implement by computing the distances from the test sample to all stored vectors, but it is computationally intensive, especially when the size of the training set grows. Many nearest neighbor search algorithms have been proposed over the years; these generally seek to reduce the number of distance evaluations actually performed. Some optimizations involve partitioning the feature space, and only computing distances within specific nearby volumes. Several different types of nearest neighbor. The nearest neighbor algorithm has some strong consistency (statistics) consistency results. As the amount of data approaches infinity, the algorithm is guaranteed to yield an error rate no worse than twice the [Bayes error rate] (the minimum achievable error rate given the distribution of the data). "k"-nearest neighbor is guaranteed to approach the Bayes error rate, for some value of "k" (where "k" increases as a function of the number of data points).

The "k"-NN algorithm can also be adapted for use in estimating continuous variables. One such implementation uses an inverse distance weighted average of the "k"-nearest multivariate neighbors. This algorithm functions as follows:

# Compute Euclidean from target plot to those that were sampled.

# Order samples taking for account calculated distances.

# Choose heuristically optimal "k" nearest neighbor based on RMSE done by cross validation technique.

# Calculate an inverse distance weighted average with the "k"-nearest multivariate neighbors.

The k-nearest neighbors algorithm is amongst the simplest of all machine learning algorithms. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbors. k is a positive integer, typically small. If k = 1, then the object is simply assigned to the class of its nearest neighbor. In binary (two class) classification problems, it is helpful to choose k to be an odd number as this avoids tied votes. The same method can be used for regression, by simply assigning the property value for the object to be the average of the values of its k nearest neighbors. It can be useful to weight the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. the neighbors are taken from a set of objects for which the correct classification (or, in the case of regression, the value of the property) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required. In order to identify neighbors, the objects are represented by position vectors in a multidimensional feature space. It is usual to use the Euclidean distance, though other distance measures, such as the Manhattan distance could in principle be used instead. The k-nearest neighbor algorithm is sensitive to the local structure of the data.

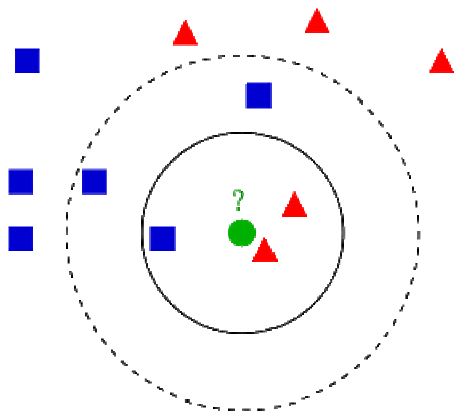

Fig.1. K-NN Classification

Example of *k*-NN classification. The test sample (green circle) should be classified either to the first class of blue squares or to the second class of red triangles. If $k = 3$ it is classified to the second class because there are 2 triangles and only 1 square inside the inner circle. If $k = 5$ it is classified to first class (3 squares vs. 2 triangles inside the outer circle).

The training examples are vectors in a multidimensional feature space. The space is partitioned into regions by locations and labels of the training samples. A point in the space is assigned to the class *c* if it is the most frequent class label among the *k* nearest training samples. Usually Euclidean distance is used as the distance metric, however this will only work with numerical values. In cases such as text classification another metric, such as the overlap metric (or Hamming distance) can be used.

The training phase of the algorithm consists only of storing the feature vectors and class labels of the training samples. In the actual classification phase, the test sample (whose class is not known) is represented as a vector in the feature space. Distances from the new vector to all stored vectors are computed and *k* closest samples are selected. There are a number of ways to classify the new vector to a particular class, one of the most used techniques is to predict the new vector to the most common class amongst the K nearest neighbors. A major drawback to using this technique to classify a new vector to a class is that the classes with the more frequent examples tend to dominate the prediction of the new vector, as they tend to come up in the K nearest neighbors when the neighbors are computed due to their large number. One of the ways to overcome this problem is to take into account the distance of each K nearest neighbors with the new vector that is to be classified and predict the class of the new vector based on these distances.

PARAMETER SELECTION

The best choice of *k* depends upon the data; generally, larger values of *k* reduce the effect of noise on the classification, but make boundaries between classes less distinct. A good *k* can be selected by various heuristic techniques, for example, cross-validation. The special case where the class is predicted to be the class of the closest training sample (i.e. when $k = 1$) is called the nearest neighbor algorithm. The accuracy of the *k*-NN algorithm can be severely degraded by the presence of noisy or irrelevant features, or if the feature scales are not consistent with their importance. Much research effort has been put into selecting or scaling features to improve classification. A particularly popular approach is the use of evolutionary algorithms to optimize feature scaling. Another popular approach is to scale features by the mutual information of the training data with the training classes.

In Sorting Spam with K-Nearest-Neighbor and Hyperspace Classifiers Spam classification[1] continues to provide an interesting, if not vexing, field of research. This particular classifier problem is unique in the machine-learning field as most other problems in machine learning are not continuously made more difficult by intelligent and motivated minds. A number of different approaches have been taken for spam filtering beyond the single-ended machine learning filter and a reasonable survey really requires a full book; in this paper we will restrict ourselves to post-SMTP acceptance filtering and the sorting of email into two classes: good and spam.

Even within post-acceptance, single-ended filtering, there are a number of techniques available. One of the most common is a Naive Bayesian filter (usually using a limiting window of the most significant N words. Other common variations use chi-squared analysis, a Markov random field even compressibility of the unknown text given basis vectors representing the good and spam classes [Willets 2003]. Although K-NN filters [Fix and Hodges, 1951] [Cover and Hart, 1967] [have been considered for spam classification in the past (John Graham-Cumming's early POP file used a K-NN) they have fallen into disfavor among most filter authors. We reconsider the use of K-NNs for classification and attempt to quantize their qualities.

PURE VERSUS INCREMENTALLY TRAINED K-NN

One disadvantage of standard Cover and Hart style K-NNs is every known input is added to the stored data; this can cause very long compute times. To mitigate this, we have used selectively trained K-NNs; rather than adding every known text immediately to the stored data, we incrementally test each known text and only add the known text to the stored data if the known text was judged incorrectly. This speeds up filter classification tremendously. Those familiar with Cover and Hart's limit theorem should realize that this modification results in a K- NN that the Cover and Hart limit theorem does not necessarily apply to. We will consider extension of the Cover and Hart theorem to cover incremental trained K-NNs in future work.

In Non-Parametric Spam Filtering Based On K-NN and LSA[2]. The paper proposes a non-parametric approach to filtering of unsolicited commercial e-mail messages, also known as spam. The email messages text is represented as an LSA vector, which is then fed into a k - NN classifier. The method shows a high accuracy on a collection of recent personal email messages. Tests on the standard LINGSPAM collection achieve an accuracy of over 99.65%, which is an improvement on the best-published results to date.

The amount of unsolicited commercial e-mails (also known as spam) has grown tremendously during the last few years and today it already represents the majority of the Internet traffic. Although the spam is normally easy to recognize and delete, doing so on an everyday basis is inconvenient. Once an e-mail address has entered in the widespread spam distribution lists it could become almost unusable unless some automated measures are taken. Nowadays, it is largely recognized that the constantly changing spam form and contents requires filtering using machine learning (ML) techniques, allowing an automated training on up-to-date representative collections. The potential of learning has been first demonstrated by Sahami et al who used a Naïve Bayesian classifier. Several other researchers tried this thereafter and some specialized collections have been created to train ML algorithms. The most famous one LINGSPAM (a set of e-mail messages from the *Linguist List*) is accepted as a standard set for evaluating the potential of different approaches.

TEXT CATEGORIZATION WITH K-NN

We used the k-nearest-neighbor classifier, which has been proved to be among the best performing text categorization algorithms in many cases [2]. K-NN calculates a similarity score between the document to be classified and each of the labeled documents in the training set. When k = 1 the class of the most similar document is selected. Otherwise, the classes of the k closest documents are used, taking into account their scores. We combined k - NN with latent semantic analysis (LSA). This is a popular technique for indexing, retrieval and analysis of textual data, and assumes a set of mutual latent dependencies between the terms and the contexts they are used in. This permits LSA to deal successfully with synonymy and partially with polysemy, which are the major problems with the word-based text processing techniques (due to the freedom and variability of expression). LSA is a two-stage process including learning and analysis. During the learning phase it is given a text collection and it produces a real-valued vector for each term and for each document. The second phase is the analysis when the proximity between a pair of documents or terms is calculated as the dot product between their normalized LSA vectors [2]. In our experiments, we built an LSA matrix (TF.IDF weighted) from the messages in the training set. The e-mail message to be classified is projected in the LSA space and then compared to each one from the training set. Then a K-NN classifier for a particular value of k predicts its class.

In Spam Mail Filtering Based On Network Processor[3], As the rapid development of the Internet, the occurrence of more and more spam mails becomes harmful to users. Content-based spam filtering technologies become the mainstream anti-spam mail methods so far. Support vector machine (SVM), Bayes, windows and K-N are excellent ones of these technologies and they have advantages and disadvantages respectively. The common shortage of content-based methods is that they can't filter spam mails as far as white-list-based, black-list-based or rule-based methods. This paper proposes a spam mail filtering system based on SVM[4] and Bayes which is implemented on Network Processor (NP). To content-based methods, this system preserves the filtering accuracy and takes advantage of the parallel processing abilities of NP to improve the filtering speed.

As the popularization of Internet, the efficient, convenient and cheep email has become the substitute of the conventional papery mail. But at the same time, spam mails cause lots of problems. At present, the anti-spam technologies can be divided as black-list-based, white-list-based, ruled-based and content-based. The black-list-based method has to maintain a real-time black list, which increases the management cost. The white-list-based method is able to filter 100% spam mails, but it may also filter some legitimate mails during the first communication. The rule-based method could find out 80% spam mails through simple rules, but it is difficult to improve the percentage and requires the user to be professional to build the rule-set. Compare with black-list-based, white-list-based and ruled-based methods, the content-based method has advantage of filtering accuracy, but it is slower. SVM and Bayes are two excellent methods to filter spam mails based on content. SVM is a statistical learning method which was developed during the 90s of twenty century. It classifies documents by constructing the best linear classifying plane and is recognized one of the best technologies for document classifying. .

CONCLUSION

Considering the disadvantages of discriminating the spam by mail body classification, is paper brings up a spam discriminating model based on SVM and D-S Identity Theory, the analysis of mail body and mail header features. Theory, there are a lot of ways to analyze Mail headers and mail bodies such as SVM with DSA.

REFERENCE

[1] William Yerazunis, Fidelis Assis, Christian Siefkes, Shalendra Chhabra Sorting Spam With K-Nearest-Neighbor And Hyperspace Classifiers

[2] Preslav Ivanov Nakov Panayot Markov Dobrikov Non-Parametric Spam Filtering based on k-NN and LSA

[3] Lian Lin, Zhongwen Li, Liang Shi 2008 IFIP International Conference on Network Spam Mail Filtering Based on Network Processor and Parallel Computing

[4] Chih-Chung Chang and Chih-Jen Lin_Last updated: February 27, 2009 LIBSVM: a Library for Support Vector Machines