Online Discriminative Spam Filter Training

Joshua Goodman
Microsoft Research
One Microsoft Way
Redmond, WA, USA
joshuago@microsoft.com

Wen-tau Yih
Microsoft Research
One Microsoft Way
Redmond, WA, USA
scottyih@microsoft.com

ABSTRACT

We describe a very simple technique for discriminatively training a spam filter. Our results on the TREC Enron spam corpus would have been the best for the Ham at .1% measure, and second best by the 1-ROCA measure. For the Mr. X corpus, our 1-ROCA measure was a close second best, and third best by the Ham at .1% measure. We use a very simple feature extractor (all words in the subject and headers). Our learning algorithm is also very simple: gradient descent of a logistic regression model.

1. INTRODUCTION

Machine learning techniques can roughly be divided into two types: generative models (like Naive Bayes), and discriminative models (like Support Vector Machines and Logistic Regression.) In most text classification tasks, especially when there is sufficient training data, discriminative models have outperformed generative models. Hulten and Goodman [4, Slide 103] surveyed results on the PU-1 spam corpus, and found that discriminative methods typically beat generative methods. It was thus surprising in the 2006 TREC Spam competition that all methods except for one variation of a single entry were best described as generative.

Part of the reason for this was that the rules of the competition made it difficult to enter discriminative techniques. Because feedback was provided to the system after every single message, a system that wanted optimal performance needed to update its parameters after each message. Updating parameters is expensive for most discriminative techniques, which strongly discouraged their use; indeed, the only method that was used was Winnow, which can easily be run with an incremental update scheme.

In this paper, we describe a very simple discriminative technique that performs extremely well on standard spam corpora. Our results would have been first, second or third best on two datasets from the TREC 2005 spam competition, depending on the corpus and measure.

2. METHOD

Our overall method is very simple. For those familiar with machine learning, it can be described as a simple linear model, whose features are the words in the body and headers of each message; weights for the model are trained using online gradient descent of a logistic regression model.

For those less familiar with the preceding terminology, the description is not much more complex. We will learn a set of weights for each word in the body or header of the message. When a new message arrives, we find this list of words, and sum the weights associated with those words. In mathematical terms, we will write $\overline{w} \cdot \overline{x}$. In this notation, $\overline{w}$ is a vector of weights, and $\overline{x}$ is a vector of 1's and 0's, with a 1 in the position corresponding to each word that was found in the message. The notation $\overline{w} \cdot \overline{x}$ simply means to take the sum of weights associated with each of these words. We then convert this sum of weights to a probability, using the "logistic" function,

$$P(Y = spam|\overline{x}) = \frac{\exp(\overline{w} \cdot \overline{x})}{1 + \exp(\overline{w} \cdot \overline{x})} \quad (1)$$

This simple equation converts a number between $-\infty$ and $+\infty$ to a probability between 0 and 1. If this probability is over some threshold, say .5 or .9, we predict that the message is spam; otherwise we predict that the message is ham.

Next, we update the weights. Technically, this is done with a method known as gradient descent, which means computing the derivative of the function in such a way as to make the correct prediction more likely. In practice, the function we try to maximize is the log of the probability of the training data. As it happens, for logistic regression, the gradient is particularly simple, just $(1 - p) \times \overline{x}$ or $p \times \overline{x}$ depending on the desired prediction, spam or ham, and where p is the current predicted probability. We use a learning rate, which makes sure that the step taken in the direction of the gradient is not too large. Typically, we use $rate = 0.02$.

The actual algorithm is very simple. Denoting a sequence of messages by $\overline{x}_i, y_i$, it is:

$\overline{w} = 0$; // initialize weights to 0
for each $\overline{x}_i$, y_i
  $p = \exp(\overline{x}_i \cdot \overline{w})/(1 + \exp(\overline{x}_i \cdot \overline{w})$
  if $(p > .5)$
    predict spam;
  else
    predict ham;
  if $(y_i == 1)$
    $\overline{w} = \overline{w} + (1 - p) \times x_i \times rate$
  else
    $\overline{w} = \overline{w} - p \times x_i \times rate$

The only detail remaining is the exact definition of a word. A word is defined as either a contiguous sequence of al-

CEAS 2006 - Third Conference on Email and Anti-Spam, July 27-28, 2006, Mountain View, California USA

phanumeric characters, or as a contiguous sequence of non-alphanumeric, non-whitespace characters. We distinguish between body features and header features. For header features, we also distinguish based on the first word of the header. So, "John" occurring in the Subject: header line is a different feature than "John" occurring in the To: lines. No feature selection is performed: all features are kept, even those that occur exactly once. If a word occurs multiple times in the body of a message, or multiple times in the same header type, it is counted only once. Note that we did not do any decoding of MIME encoded body parts, or internationally encoded subject lines: these were all simply processed in their encoded form. We suspect that adding decoding would lead to improvements, and might be particularly important for international users.

2.1 Comparison to Other Methods

It is interesting to compare our method to the other methods at TREC 2005. We start by a quick discussion of generative versus discriminative techniques. Roughly, a generative model attempts to describe the distribution of all the data; that is, a generative model for distinguishing birds from mammals would contain a long description of each one, sufficient to draw many different bird-like creatures, or many different mammalian creatures. A discriminative model would focus on the differences between them, e.g. mammals almost always have hair and always feed their young with milk, while birds almost always have feathers and almost always have wings. Mathematically, denoting the spam versus ham distinction as y and the actual message as $\overline{x}$, a generative model G will typically find the best G for a model of the form $P(y, \overline{x}|G)$, while a discriminative model D will typically find the best model of the form $P(y|\overline{x}, G)$. In other words, the discriminative model focuses on learning the distinction of interest (spam versus ham in our case), rather than a complete description of the data.

We surveyed all of the major entries in the TREC 2005 spam track – all those which did well and had readily available descriptions. All but one – one of the 4 variations of CRM114 – would be better described as a generative model than as a discriminative model, although most were not strictly speaking generative.

Bratko *et al.* [2] entered perhaps the best performing method, which was also a truly generative model. Essentially, the method is an n-gram language model (or, equivalently, a PPM compression model), with clever online adaptation.

Segal [7] entered a method that used "Less Naive Bayes." Few details are given, but it appears that this technique is a generative technique that attempts to model some of the dependencies that Naive Bayes normally ignores.

DBACL [3] is yet another generative model. Note that one option for DBACL is to use a discriminative model – a maxent model – as a component of the overall generative model. Because of the way the probabilities of this model are trained – to maximize the probability of the observed strings, not to maximize the probabilities of the predicted classes – overall, this is still a generative model.

CRM114 [1] was tested in 4 different configurations. Three of them used variations on Naive Bayes, while one configuration used Winnow training. Winnow is a discriminatively trained linear model. In the CRM114 experiments, Naive-Bayes-like methods beat Winnow on most, but not all measures.

SpamBayes [5] is yet another filter based roughly on Naive Bayes techniques, though with many tweaks.

3. RESULTS

Our results are extremely competitive. On the Enron Full corpus, our results are by some measures better than any entered into the competition; by all measures they are very competitive. One key measure used in TREC is the Ham at .1% measure (percentage of spam caught when the threshold is adjusted for a .1% false positive rate.) At that setting, we missed 1.72% of spam, compared to 1.78% of spam for the next best competitor (ijsSPAM2). For 1-ROCA (area under the curve – a measure of performance across different thresholds, lower is better), our number was .022, compared to .019 for ijsSPAM2, and .052 for the best CRM114 based system.

On the Mr. X corpus, the very best system in terms of ROCA was bogofilter, with a score of .045; our system achieved .047, which would have placed it second by this measure. For ham at .1%, our system missed 6.6% of spam, compared to 9.72% for ijsSPAM2, 3.56% for ijsSPAM3, 9.65% for the best CRM114 system, and 3.9% for bogofilter. Space precludes listing the full results of our runs, so we have made them available at `http://www.research.microsoft.com/~joshuago/goodman.full.html` and `goodman.mrx.html`

Note that these results were achieved after the TREC Spam competition. Our results on Enron data were tuned (we have a single parameter, learning rate, and we also tried several other featurization techniques). For the Mr. X corpus, we thank Gordon Cormack who ran the evaluation blindly for us: we have never seen the corpus, and we tested only a single piece of code with a single parameter setting, so this comparison is a fair one.

Our method is also quite fast: it can process about 30 messages per second on a 3.2 GHz dual processor pentium, even though it is unoptimized perl code.

4. CONCLUSION

Discriminative techniques have received relatively little attention in the spam fighting community. Some of the most influential anti-spam research was the seminal work of Sahami *et al.* [6], and the Graham's highly cited "A Plan for Spam", both of which are based on the prototypical example of a generative model, Naive Bayes. In this paper, we have shown that an extremely simple discriminative model can produce results that are competitive with, and in some cases better than, the best reported generative methods.

5. REFERENCES

[1] F. Assis, W. Yerazunis, C. Siefkes, and S. Chhabra. CRM114 versus Mr. X: CRM114 notes for the TREC 2005 spam track. In *TREC 2005 Workbook*, 2005. See `http://plg.uwaterloo.ca/ gvcormac/trecspamtrack05/`.

[2] A. Bratko and B. Filipič. Spam filtering using compression models. Technical Report IJS-DP-9227, Department of Intelligent Systems, Jožef Stefan Institute, Ljubljana, Slovenia, 2005.

[3] L. A. Breyer. The DBACL text classifier. See `http://www.lbreyer.com/preprints/dbacl.ps.gz`, 2005.

[4] G. Hulten and J. Goodman. Tutorial on junk e-mail filtering. In *ICML 2004*, 2004.

[5] T. Meyer. SpamBayes: TREC 2005 spam track notebook. In *TREC 2005*, 2005. See `http://plg.uwaterloo.ca/ gvcormac/trecspamtrack05/`.

[6] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A bayesian approach to filtering junk e-mail. In *AAAI'98 Workshop on Learning for Text Categorization*, July 1998.

[7] R. Segal. IBM SpamGuru on the TREC 2005 spam track. In *TREC 2005 Workbook*, 2005. See `http://plg.uwaterloo.ca/ gvcormac/trecspamtrack05/`.