Ensembles in Adversarial Classification for Spam

Deepak Chinavle
amazon.com
705 5th Avenue South
Suite 220
Seattle, WA 98104
deepchin@amazon.com

Pranam Kolari
Yahoo! Labs
2821 Mission College Blvd
Santa Clara, CA 95054
pranam@yahoo-inc.com

Tim Oates and Tim Finin
University of Maryland,
Baltimore County
1000 Hilltop Circle
Baltimore, MD 21250
{oates,finin}@umbc.edu

ABSTRACT

The standard method for combating spam, either in email or on the web, is to train a classifier on manually labeled instances. As the spammers change their tactics, the performance of such classifiers tends to decrease over time. Gathering and labeling more data to periodically retrain the classifier is expensive. We present a method based on an ensemble of classifiers that can detect when its performance might be degrading and retrain itself, all without manual intervention. Experiments with a real-world dataset from the blog domain show that our methods can significantly reduce the number of times classifiers are retrained when compared to a fixed retraining schedule, and they maintain classification accuracy even in the absence of manually labeled examples.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; I.5.2 [**Pattern Recognition**]: Design Methodology—*classifier design and evaluation*; I.5.4 [**Pattern Recognition**]: Applications—*text processing*

Keywords

Spam, Weblogs, Ensembles, Adversarial Classification, Non-stationarity, Retraining

General Terms

Algorithms, Experimentation

1. INTRODUCTION

As we build more robust classifiers to detect spam, the methods used by spammers evolve so that they can continue to get their messages through. This leads to changes in the distribution of the data seen by classifiers used to weed out spam. Such changes can also occur naturally, for example, as topics change in a blog or one company is bought by another and the internal email traffic changes in form and content. Unless this non-stationarity of the data is taken into account, classifier performance will decrease over time. For example, one common method used by spammers is to add text from legitimate sources (e.g., emails, blogs, newswires, books) to their communications in an attempt to get past the classifiers as false negatives [4]. When many of these messages get through, classifiers are often updated, which leads to an arms race of sorts.

Two natural questions arise. First, how does one determine when classifier performance has degraded? Second, how are the classifiers retrained? The most common answers to these questions involve a continual stream of labeled examples, such as current emails labeled according to whether or not they are spam. These labeled examples can be used to test the accuracy of the classifiers and to retrain them when accuracy becomes too low. The problem with this approach is that obtaining labeled examples is expensive.

In this paper we present a method based on an ensemble of classifiers that automatically, without human intervention, deals with the two questions above. First, we use mutual agreement between classifiers in the ensemble to detect possible changes in classifier accuracy. The mutual agreement between a pair of classifiers is the fraction of time they assign an instance the same class label. Note that the true label need not be known to compute mutual agreement. We show, using real data from the blog domain, that changes in mutual agreement are indicative of decreased classification accuracy. Second, we use the output of the ensemble as a proxy for the true label for new instances and retrain individual classifiers identified as possibly weak via mutual agreement.

There is significant work on using ensembles to tackle concept drift. These approaches are motivated by the fact that it is often useful to have multiple (weighted) opinions before making a decision. [6] presented an ensemble method for concept drift by assigning weights to classifiers based on their accuracies. [3] extended this work by dynamically creating and removing weighted experts in response to changes in performance. Both methods rely on a stream of labeled examples, whereas ours does not. [5] presented another ensemble-based method for concept drift, claiming it to have low computational cost and still be comparable with other ensemble methods. [1] presented a random decision tree ensemble-based engine to mine data streams. They demonstrated the ability to detect concept drift on the fly and discussed ways to combine old data with new data for computing optimal models. All the data they used was synthetic.

CIKM'09, November 2–6, 2009, Hong Kong, China.

2. BACKGROUND

Spam blogs (splogs) are typically constructed automatically by combining legitimate content taken from the web with ads or links to other sites in an attempt to boost their rank in search engines. Spam blogs are a tremendous problem for intermediate servers and blog search engines as discussed and studied in [2].

The empirical results reported in Section 4 are all based on two blog datasets, called SPLOG2005 and SPLOG2006, that we manually constructed. SPLOG2005 was obtained by sampling the update ping streams at a blog search engine (Technorati, http://technorati.com) in the year 2005. 700 positive examples (splogs) and 700 negative examples (blogs) were identified and labeled by hand. SPLOG2006 was created by processing the update ping streams at a ping server (http://weblogs.com) in the year 2006. 750 positive and 750 negative examples were also manually labeled for SPLOG2006. Training and testing on SPLOG2006 in general resulted in better classifier performance given that it was sampled at a ping server. SPLOG2005, which was sampled through a blog search engine, contained splogs that had already passed one set of filters and were thus possibly more difficult to automatically identify as such.

The features extracted from data are crucial for the success of machine learning methods. Features that work in the email and web spam domains may not work in the blog domain. Therefore, we used a variety of features, some of which are novel, as described below.

- Bag of words: These features are simply counts of the number of times each word occurs in a blog.
- Bag of 2 or 3 words, or word n-grams: These features are simply counts of the number of times each pair and triple of words occur consecutively in a blog.
- Character n-grams: This feature is similar to the above two features but we instead consider 2 or 3 *characters* at a time.
- Anchor text: This is the text appearing between "a href" tags.
- Tokenized URLs or outlinks: In this feature both local and outgoing links are tokenized using "/" and "." as delimiters.
- HTML tags: This feature looks at html tags, like "h1" and "bold".
- URL: This is one feature which is not dependent on the content of the blog and hence is very fast to extract.

3. MUTUAL AGREEMENT TO TRACK CONCEPT DRIFT

The utility of using an ensemble of classifiers is that the ensemble can perform well even if some of its member classifiers perform poorly. A good ensemble has classifiers that are diverse, perhaps having disjoint feature sets. An adversary, such as a spammer, is unlikely to change all the features at the same time, so only the classifiers based on the features that change will be affected, not the others. The other classifiers can still keep the overall ensemble performance high. Eventually, though, the underperforming classifiers will have to be retrained to prevent further changes to the features from reducing the accuracy of the ensemble.

Whenever two different classifiers give the same label to an instance, we say they agree. By doing this test on a sufficiently large number of instances, we can estimate their mutual agreement. Let $f_i(x_k)$ be the output of the i^{th} classifer on the k^{th} instance. Let $\delta(p)$ evaluate to 1 if p, a predicate, is true. Otherwise, it evaluates to 0. Then the mutual agreement between classifiers i and j can be estimated as follows:

$$\mathrm{MA}_{i,j} = \sum_{k=1}^{n} \frac{\delta(f_i(x_k) = f_j(x_k))}{n}$$

Over a period of time, if the mutual agreement between a pair of classifiers decreases, it may indicate that one or both the classifiers are performing poorly. Using mutual agreement as a potential indicator of performance, we were able to do the following.

- Pairs of classifiers performing poorly: In an ensemble setting, we can monitor the agreement between all possible pairs of the classifiers. Periodically, all the agreement values are checked. If the agreement values have dropped below a fraction of the initial agreement, both the classifiers of the pair are retrained.
- Weakest classifier: This approach is similar to the first except for the way retraining is done. While checking agreement values periodically, we check which classifiers occur in most of the weak pairs and only retrain those.

4. EXPERIMENTAL RESULTS

This section describes two experiments with our method. The first shows the usefulness of mutual agreement in reducing retraining time. The second experiment shows how mutual agreement can be used to improve the accuracy of an ensemble as compared to performing no retraining and that it is as good as frequent retraining with significantly less computation.

4.1 Usefulness of Mutual Agreement

The first experiment demonstrates how mutual agreement can be used to reduce retraining time and track performance of individual classifiers in an ensemble. The following cases were considered in this experiment.

- Case 1: 10 fold cross validation on the SPLOG2005 dataset was performed, i.e, the classifiers were trained and tested for agreement on the SPLOG2005 dataset. The initial agreement between the classifiers based on the text and tags features was found to be 76%. Let this be the initial agreement between the two classifiers.
- Case 2: The classifiers were trained on SPLOG2005 and tested on the SPLOG2006 dataset with no retraining. The agreement was 69.5%. This sets the baseline.

For the cases that follow, the classifiers were trained on SPLOG2005, and then selectively retrained based on different criteria.

- Case 3: In this experiment with retraining, agreement between the classifiers was checked after every 100 instances of the SPLOG2006 dataset. Only when the agreement fell below 95% of the initial agreement, i.e, 95% of 76% which is around 72.2%, the classifiers were retrained. There were 14 iterations, after which the agreement value was checked and, if necessary, retraining was performed.

Case	Mutual Agreement	Retrainings
95% threshold	79.48	1.2
100% threshold	80.44	2
Always retrain	82.74	14

Table 1: Experiment 1a: Retraining to maintain mutual agreement using ensemble labels

Case	Mutual Agreement	Retrainings
95% threshold	80.48	1
100% threshold	81.5	1.6
Always retrain 5	84.26	14

Table 2: Experiment 1b: Retraining to maintain mutual agreement using true labels

- Case 4: Case 3 was repeated with a 100% threshold instead of 95%. That is, classifiers were retrained if their agreement dropped at all from that found on the SPLOG2005 dataset.
- Case 5: The classifiers were retrained after every 100 instances, regardless of agreement values.

In addition, cases 3, 4, and 5 were performed in the setting with true labels instead of ensemble labels. Also, these cases were repeated 5 times with different random orders of the SPLOG2006 dataset instances. The average results are provided in Table 1. All of the agreement values are given in percentages and the "%" sign has been dropped. "Retrainings" means the number of times the classifiers were retrained. "Ensemble labels" means retraining was done using samples labeled by the ensemble. "True labels" stand for the case of using true labels for retraining.

Clearly, with no retraining, the agreement value goes down. Experimental results show that by using mutual agreement for retraining we reduce retraining time drastically while maintaining the initial agreement between classifiers. We next show how mutual agreement can be used in an ensemble setting for dynamic retraining of base classifiers.

4.2 Graph Agreement

To demonstrate how members of an ensemble can be trained dynamically based on mutual agreement as a trigger, we considered 5 classifiers in all possible pairs. Performance is measured in terms of the accuracy of the ensemble. The 5 classifiers were based on the text, character, outlink, anchor, and tag feature sets. Classifiers based on character-gram and word-gram feature sets were left out as they are similar to the classifier based on text features, and therefore contribute little to ensemble diversity. All of the classifiers were trained on SPLOG2005 and initial agreements were determined by performing 10 fold cross-validation as in the earlier experiment. This gave rise to a fully connected graph of 5 classifiers where the nodes are the classifiers and the arc labels are the agreement values between the classifiers. The classifiers were then exposed to the SPLOG2006 dataset and, as in the earlier experiment, the agreement values for the classifiers were checked after every 100 instances. So with every iteration we update the edges of the graphs with agreement values for that iteration. Unlike the previous experiment, we make definite claims about accuracy here. This experiment was carried out in the following settings, mainly differing in the retraining algorithm.

Case	Accuracy	Retraining
Case 2	95.04	17.6
Case 3	94.88	6.8
Case 4	95.14	65

Table 3: Experiment 2a: Impact on ensemble accuracy, retraining using ensemble labels

Case	Accuracy	Retraining
Case 2	95.86	14.75
Case 3	95.08	7
Case 4	96.4	65

Table 4: Experiment 2b: Impact on ensemble accuracy, retraining using true labels

- Case 1 - No retraining: There was no retraining at all. This forms the baseline. The end accuracy of the ensemble was 93%.
- Case 2 - Retraining all of the classifiers in weak pairs: In this setting, all of the classifiers whose agreement had fallen below a threshold of 95% of the initial agreement were retrained.
- Case 3 - Retraining only the weakest: The classifier that participated in the most weak pairs was the only one retrained. Again, the threshold for retraining was 95% of the initial agreement.
- Case 4 - Retraining all of the classifiers: In this method, all of the classifiers were retrained in every iteration.

Cases 2, 3, and 4 were also repeated with true labels instead of ensemble labels. Each case was repeated 5 times with different random orderings of the SPLOG2006 dataset instances. Some of the experiments were also performed with a 100% threshold but no significant improvement in accuracy was observed. Therefore, we only report results with a 95% threshold. All of the results are shown in Table 4 where the columns are the same as Table 1. The only difference is that, instead of agreement values, we report the accuracies of the ensemble. The accuracy values are given in percentages and the "%" sign is dropped.

Figure 1 shows a plot of accuracies for no retraining, and for full retraining using both true and ensemble labels. This plot shows lower and upper bounds on performance. Two additional accuracy plots are shown - one for experiments using true labels, Figure 3, and the other for experiments

(a) Accuracies

Retraining	Ensemble Labels	True Labels
None	90	-
Weak pairs	93	91
Weakest	97	97
All	96	97

(b) Classifiers to retrain

Retraining	Ensemble Labels	True Labels
Weak pairs	None	Tag,Anchor
Weakest	Text	Text

Table 5: Summary of agreement graph after two iterations

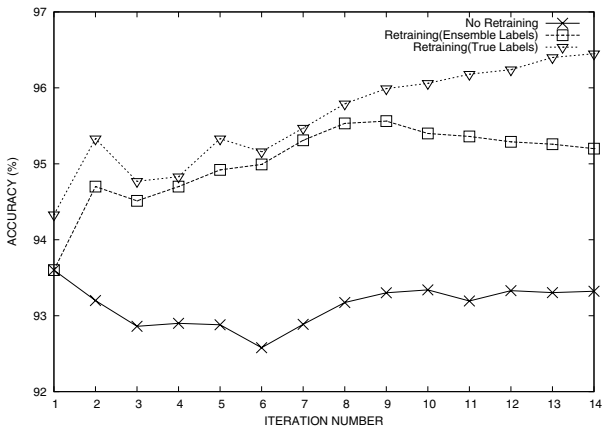

Figure 1: Full retraining accuracy using ensemble vs. true labels and no retraining

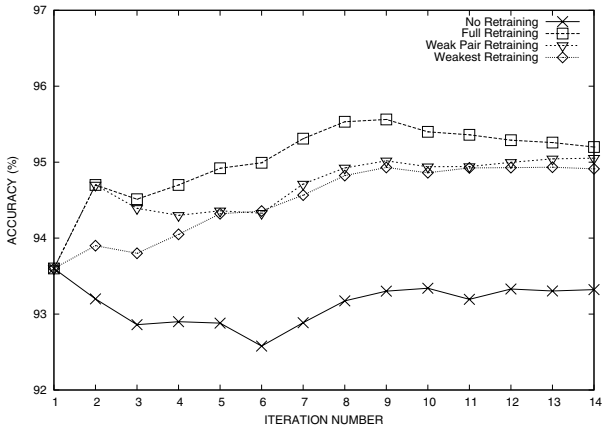

Figure 2: Accuracy with ensemble labels for retraining

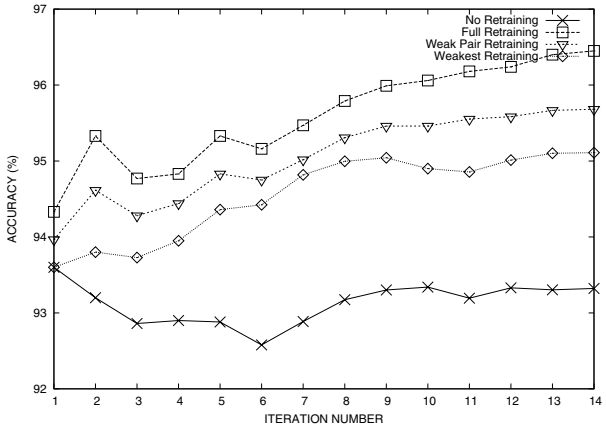

Figure 3: Accuracy with true labels for retraining

using ensemble labels, Figure 2. Note that the data series line for the "No Retraining" case has been added just for comparison, it is the same for both cases. All these experiments show how mutual agreement can be used to reduce the retraining time, while maintaining the accuracy close to that of frequent retraining.

5. DISCUSSION AND CONCLUSION

The experimental results in the previous section showed how mutual agreement can be used for dynamic retraining of base classifiers to handle adversarial classification. The main idea is that decreasing mutual agreement is an indicator that one or more classifiers are performing poorly. Said differently, when mutual agreement is above threshold, we assume that the classifiers are performing well. However, it could very well happen that, though the classifiers agree, they are both wrong. In short, the relation between mutual agreement and accuracy of the classifiers is not perfect. This can be more severe if we consider just two classifiers at a time rather than a larger set of them. The probability of both the classifiers succumbing to the spammers changing tactics in the same way cannot be neglected. But in an ensemble setting where we have more then two classifiers, the probability that most of the classifiers will be wrong at a given time decreases. The more classifiers there are, the lower the probability of them all going wrong at the same time. But, for this statement to hold true, we need to keep the classifiers diverse. So if the adversary changes some of the features, only classifiers based on those features may be affected. The rest should still perform well.

6. REFERENCES

[1] W. Fan. Streamminer: a classifier ensemble-based engine to mine concept-drifting data streams. In *VLDB '04: Proceedings of the Thirtieth international conference on Very large data bases*, pages 1257–1260. VLDB Endowment, 2004.

[2] P. Kolari, A. Java, and T. Finin. Characterizing the splogosphere. In *WWW 2006, 3rd Annual Workshop on the Weblogging Ecosystem: Aggregation, Analysis and Dynamics*, 2006.

[3] J. Z. Kolter and M. A. Maloof. Dynamic weighted majority: A new ensemble method for tracking concept drift. In *ICDM '03: Proceedings of the Third IEEE International Conference on Data Mining*, page 123, Washington, DC, USA, 2003. IEEE Computer Society.

[4] D. Lowd and C. Meek. Good word attacks on statistical spam filters. In *CEAS*, 2005.

[5] M. Scholz and R. Klinkenberg. An ensemble classifier for drifting concepts. In *In Proceedings of the Second International Workshop on Knowledge Discovery in Data Streams*, pages 53–64, 2005.

[6] H. Wang, W. Fan, P. S. Yu, and J. Han. Mining concept-drifting data streams using ensemble classifiers. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 226–235, New York, NY, USA, 2003. ACM.