Privacy Preserving Collaborative Anti-Spam Filter

B.Tech Project submitted in partial fulfillment

of the requirements for the degree of

Bachelors of Technology

in CSE by Vidyadhar Rao Panga

200601100

vidyadhar@students.iiit.ac.in

International Institute of Information Technology

Hyderabad - 500 032, INDIA

December, 2009

International Institute of Information Technology

Hyderabad, India

CERTIFICATE

It is certified that the work contained in this project, titled "Privacy Preserving Collaborative Anti-Spam Filter"by Vidyadhar Rao Panga, has been carried out under my

supervision for a partial fulfillment of degree.

Date

Signature

(Adviser: Dr.Kannan Srinathan)

Abstract

Rising amounts of spam is a potential threat to all major email service providers such as Gmail, YahooMail etc. A collaborative effort by the service providers can be an effective tool in designing an effective anti-spam filter. However, in light of the recent arguments being raised against the privacy of emails getting compromised in a collaborative anti-spam filtering, a better approach to allay apprehension was in order. The ideas presented here were reached at after a comprehensive review of the problem and the existing approaches of anti-spam filtering both in practice and in research. In particular, a major emphasis was laid on the evaluation of the performance of the SVM based anti-spam filter. The previously unexplored usage of shingles as features for the SVM was looked into. Investigations were also done on the performance of the SVM classifier based on parameters like window size, training corpus size and feature vector size of the email. Privacy concerns of the service providers are addressed as privacy preserving SVMs are well studied in literature. The privacy of our system is ensured by adapting one such protocol which is known to be efficient and thus is practical for our setup.

Contents

Chapter 1

Over view

1.1 Data Interpretation

With the amount of internet data gathered doubling every three years[**?**], data interpretation has become an interestingly important in research. In order to transform this data into information various techniques have been proposed both in research and in practice. In literature theses techniques are a blend of Statistics, AI [Artificial Intelligence], Data Mining and Machine Learning research. Data interpretation is a process that involves analysis of very large data to find patterns that are

1. *Valid*: Hold on new data with some certainty.
2. *Novel*: Non-obvious to the system.
3. *Useful*: Should be possible to act on the item.
4. *Understandable*: Humans should be able to interpret the pattern.

In maintaining the above requirements, the data interpretation techniques are applied in an iterative process involving,

1. *Data Collection*: Collecitng data from large number of clients or from internet.

2. *Pre-Processing*: Data from different sources is cleaned to remove duplicates or supply missing details to obtain meaningful data.

3. *Transformation*: Mapping the cleaned data to features.

4. *Choosing a method*: To apply some machine learning or statistical techniques relevant to the problem.

5. *Result Evaluation*: Evaluating the results on new data.

Data interpretation has found numerous applications in a wide range of profiling practices like marketing(identify likely responders to promotions), fraud detection(from an online stream of event identify fraudulent events), medicine(analyze patient disease history to find relationships between diseases or symptoms), and scientific discovery(identify new galaxies by searching for sub clusters). However, there has been growing concern that use of data interpretation technology is violating individual privacy [**?**]. The goal of any of the data interpretation approaches is to develop generalized knowledge, rather than identify information about specific individuals.

1.2 Motivation

As an example[**?**], consider the example of mining of supermarket transaction data,

1. Retailer offers discount cards to consumers who are willing to have their purchases tracked.

2. Generating Association Rules from such data is common in Data Mining Technique leading to insight into buyer behavior that can be used to redesign store layouts, develop retailing promotions,etc.

3. This data can also be shared with suppliers which would inturn support their product development and marketing efforts. Unless substantial demographical information is removed, this could pose a privacy risk. Even then there will be a risk to the retailer.

4. Uilizing information from multiple retailers, the supplier can develop promotions that favour one retailer over the other retailer i.e., supplier can enhance his revenue at the "expense of the retailer".

5. This is a serious problem to the retailer who expected to increase his sales by sharing information.

Privacy Preserving Solution[?]

1. Instead, the retailers collaborate to produce globally valid association rules for the benefit of the supplier, without disclosing their own contribution to either the supplier or other retailers.

2. This allows the supplier to improve product and marketing simultaneously "benefitting all retailers" , but does not provide the information to single out one retailer. In other words, the supplier has no control to eliminate a particular retailer inorder to enchance his own benefits.

3. Also the individual data need not leave the retaier, solving the privacy problem raised by disclosing the consumer data.

Our goal to find an application which raises the privacy issues of the clients and to enable such WIN-WIN-WIN situations i.e., *"The knowledge present in the data is extracted for use, the individuals privacy is protected, and the data holder is protected against misuse of the data"*.

We have identify one such application "Privacy Preserving Collaborative Anti-Spam Filtering" and analyzed the privacy concerns to make it a practical setup.

1.3 Email spam

The problem of spam is ubiquitous in most of the internet applications. The service providers are willing to provide services like *Electronic mail (E-mail)*, product reviews, opinion sharing on a wide variety of topics such as sports reviews, restaurant reviews, comments on various aspects. The users want to get benefited from such services. Among all these services *e-mail* is often the mediator that promotes advertising to these commercial and social networking services. This in turn is establishing interconnections among users and service providers. Although these services have created many opportunities for the internet users, *unsolicited junk e-mail* (popularly known as *'spam'*), is an obnoxious, unwelcome and commercial byproduct for them. The estimated portion of spam email of the global emails sent is in fact 94% [**?**][**?**]. Hence, the problem of spam is creating threats to both service providers and users. As there is no guarantee to avoid all the spam, its been a challenge to avoid looking at most of it.

The users communicate using email to the service providers and other users in a social networking environment. The information exchanged constitutes the *legitimate messages* (known as *'ham'*) that are personal to the users and the spam messages. Though the definition of spam may vary from user to user, most of the time spam is considered as a distribution of similar or identical messages targeted at the large number of user addresses. Spammers usually gather huge lists of e-mail addresses from websites, news sites, blogs and worms which keep track of user addresses. Alternatively, a dictionary attack is used in enumerating all possible user addresses of some fixed length (usually very small). In practice, many of these addresses are invalid or malformed or undeliverable. Such messages

waste the Internet resources like bandwidth and time of network administrators and finally the users have to bear the cost of delivery, storage and processing. The number of valid messages sent is given by the amount of private data i.e, valid user addresses hit by the spammer. Such attacks can be limited by increasing the overhead costs for each spam sent to a valid address. That is, the email address is constrained to be of atleast a minimum length, so that in practice, the number of active addresses is far lesser than those obtained using a dictionary attack over this length.

It is also noticed that the spam messages constitutes different topics as rarely mentioned in legitimate messages and so it makes sense in using text classifier for anti-spam filtering. Earlier well studied machine learning algorithms like Bayesian Probability Models[**?**], Support Vector Machines(SVM)[**?**], Boosting and Decision Trees[**?**] were used to solve this problem of text classification. For the purpose of using these algorithms, each message is converted into a feature vector using message transformation techniques like MD5[**?**]. The main concern here is that the transformation technique should preserve the textual features of the message, that is if two emails have similar syntactic structures then their features should be similar. The supervised algorithm is trained initially on known messages to build a classifier and then the new incoming emails are filtered with the classifier.

A collaborative effort by major service providers can help in training an effective and an up-to-date anti-spam classifier. In order to do this, the service providers need to collectively train the classifier, which in turn means sharing their training data (emails) within themselves. However, this in general can be perceived as a potential privacy breach of the end-users. In this report we review the ideas presented in the prior work and their approaches to overcome the problem of spam. We then propose a better approach to the existing approaches and evaulate its performance on a publicly available TREC[**?**] email corpus.

1.4 Related Work

Email Anti-Spam classification problem is well studied using Distributed Adaptive Blacklists, Machine Learning Algorithms, Rule-Based Spam Filtering, Collaborative Spam Filtering etc.

Approaches like Distributed Checksum Clearinghouse (DCC)[**?**], Vipul's Razor [**?**] were implemented effectively. These approaches address spam as very similar messages that are distributed to a large number of users. The privacy of the emails is ensured in these systems as the clients and server share only fingerprints of emails and not the actual emails. The fingerprints are generated by using hash functions like MD5 over the email content. But the problem with these hash functions is that the hash values changes by a large number even if there is a slight change in the content. Spammers can easily get around these techniques by just replacing a few characters in the email (eg. movie and muvie).

Naive Bayesian Filtering[**?**] has been widely used in email spam filtering for the reasons like linear training time and also adaptability to changing environment over time with incremental updates. The probabilistic estimation is done as frequency ratios of the words present in the training corpus. The spammers attempt to degrade the effectiveness of the classifier by editing few characters of the words (eg. movie and moovie).

Collaborative Spam Filtering with Heterogeneous Agents[**?**] proposes a collaborative learning framework that parallely filter spam with heterogeneous agents. In this framework, emails that are difficult to classify with a single agent, can be detected and filtered through the collaborative filtering architecture based on the collaborative weighting and learning architecture. Besides the privacy of the emails, the system has to survive the communication overhead in adopting such collaborative techniques.

Spam Assassin[**?**] uses a large set of rules to determine whether an email is spam or ham. These rules are mostly based on regular expressions which are matched with the

email header or content. Spam assassin can be used as a standalone application or as a client that can communicate with a daemon. The limitations being that in the first case the classifier becomes a stand alone while in the later it introduces additional security risks.

To the best of our knowledge, only one attempt has ever been made to address these two problems (standalone and security) in Alpacas[**?**]. Alpacas is also a collaborative system where the feature sets of the emails are vertically partitioned and shared among the email clients. The classification of a new email is done based on the simple classification strategy by querying the features to the respective email clients. Each email client matches the query to their local knowledge and sends the classification result along with a partial feature set of the matching spam or ham email. Though the system performs with an overall low privacy breach by querying smaller sets of features, there is a trade off between the classification accuracy and the amount of private data of emails revealed. However, if the rendezvous agents of an email exchange the feature elements they have received as a part of the query message, then they have a better chance of guessing the contents of the email. Such attacks are termed as *collaborative inference attacks* which are of great concern to the users in the collaborative frameworks.

Our work contributes the advantage of using Shingles[**?**] as features for the emails to resist against character replacement attacks. To the best of our knowledge this is for the first time SVM classifier is evaluated by using shingles as features for the emails. We completely eliminate the problem of collaborative inference attacks, thus preserving the privacy of the emails using cryptographic protocols[**?**]. To reduce the communication overhead involved in the process of execution of these protocols we train the SVM using an incremental algorithm[**?**].

1.5 Our Approach

The complete solution can be thought of as a two step process, i.e. the training and the classification part. For the training, an SVM classifier is trained on the joint data of the k service providers. Privacy preserving training ensures that joint classifier is obtained without disclosing the data of each service provider to others. The trained classifier is then postprocessed to preserve the privacy of the support vectors. The resulting, privacy preserving classifier is then released to every service provider. We now describe in detail each of the sub-steps involved.

1.5.1 Feature Generation

Extraction of features from text can be done in a variety of ways, such as, a vector of keyword weights based on the concept of term frequency and inverse document frequency [TF-IDF] as applied in document classification problems or as a bag of word vectors as applied in many natural language processing tasks. We use *shingles* as a fingerprint of each email which characterizes the message content as used in Alpacas. The concept of shingles is a result of work on the resemblance and containment of documents[**?**] which were also used to detect duplicate or redundant content of webpages over the Internet[**?**].

Each email is first preprocessed to remove the hyper-content and meta-content like header and HTML information. Then a sliding window algorithm is applied in which a window of length (w) slides through the message content. The algorithm computes w-*shingles* of an email as w consecutive tokens (of characters or words) that fall within the window. Algorithm 2 describes the pseudo-code of the proposed feature generation. For the sake of simplicity, we make an assumption that each email is a string of characters without any blank spaces in between. For an L length message, we obtain a set of $L-w+1$ shingles. For example, if the message in the email is: `"we ask you to advise us by an email"`.

Then the $3-shingles$ of the above message, is the set: {`wea, eas, ask, ···, mai, ail`}.

Algorithm 1 Feature Generation using Shingles

1: Pre-process the email to remove the header and HTML tags and blank spaces.
2: **for** $i = 0$ to $i \leq L - w$ **do**
3: Shingle = Email$(i : i + w)$
4: Features[i] = RabinFingerprint (Shingle)
5: **end for**
6: Sort_Ascending (Features)
7: Select top s features.

The number of shingles is larger than original number of words in the email itself by a factor of w. This not only requires more storage space, but there is also a computational overhead involved in calculating its distance/similarity with other emails. However, it is not uncommon that the message length varies for every email. So, we make use of Rabin's Fingerprint technique[**?**] to overcome these problems. The Rabin Fingerprint Algorithm is a one-way hash function known for its efficient implementation of random sampling that can be done independently for each message.

The obtained hash values of the shingles are sorted in ascending order of which the smallest s are choosen as the feature vector of the email. The advantage of using Rabin Fingerprint's rather than some random hash function like MD5 is that their probability of collision is well understood and also have the advantage of their algebraic properties when we compute the fingerprints of sliding windows. It was also proven that using such a sampling method still preserves the syntactic properties of the message. Detailed mathematical foundations of the algorithm are given in [**?**].

1.5.2 Incremental SVM Learning

The features thus extracted are used in training a 2-class classifier. *Support Vector Machines (SVM)* [**?**] are a powerful classifier, which has been used in a variety of problems in pattern recognition and function regression. The basic principle of SVM is to find an optimal seperating hyperplane (support vectors) so as to seperate the two classes of data with maximum margin. The location of the seperating hyperplane is specified via a real valued weights of the training examples. Without loss of generality, consider the i^{th} training example to be (x_i, y_i) where $x_i \in R^n$ is the vector of n features and $y_i \in \{+1, -1\}$ is class label for the example. SVM constructs the decision boundary in the kernal induced feature space by sloving the quadratic programming optimization problem[**?**]:

$$arg\ min_\alpha \tfrac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j K(x_i, x_j) - \sum_{i=1}^{m} \alpha_i \text{ subject to}$$

$$\sum_{i=1}^{m} \alpha_i y_i = 0, 0 \leq \alpha_i \leq C, \text{ for } i = 1, ..., m$$

m is the total number of training examples, $K(x, y)$ is a kernal function, and $C \geq 0$ is the cost parameter. For a linear SVM $K(x, y) = x.y$ and for non-linear SVMs, we have $K(x, y) = \exp(-g \parallel x - y \parallel^2)$ for RBF kernal, where $g > 0$ is a parameter. The kernal function implicity maps data into higher dimensional Reproducing Kernal Hilbert Space(RKHS)[**?**] and computes their dot product there without actually mapping the data. The constant γ is calculated from KKT complementarity conditions[**?**]. Thus the classification function is $f(x) = \sum_{i=1}^{m} \alpha_i y_i K(x_i, x) + \gamma$, where x is a testing example. The curve corresponding to $f(x) = 0$ is the decision boundary.

A large training dataset, uniformly chosen from several mail-servers can help in reducing the generalization error. However, in practice, training with large datasets is very costly in

time and memory consumption and more over in our scenario the data is private to each mail server, who might be unwilling to share it with others. The (x_i, y_i) pairs with non-zero α_i are *support vectors*. The specification of the decision boundary is contributed by the support vectors alone. Thus the decision function reduces to $f(x) = \sum_{i=1}^{\#SV} \alpha_i y_i K(SV_i, x) + \gamma$, where SV_i is the support vector and y_i its corresponding label, and $\#SV$ denotes the number of support vectors.

Hence, we use an *incremental learning algorithm* to handle the vast data which is computationally efficient for training over a distributed data. It was shown in [**?**], that the SVM works well with an incremental learning model with impressive performance for its outstanding power to generalize the data space and also its behaviour is well understood.

In our current scenario, the training data is horizontally partitioned among the mail-servers and the objective is to learn the weight vector while respecting the privacy of everyones data and keeping the communication costs low. Our experiments shows that adopting incremental SVM learning in our setup significantly reduces the communication costs. In general if the data is partitioned among k servers, the learning algorithm requires $k - 1$ updates to arrive at an optimal weight vector. At each update, the training data to the classifier SVM_i at the server_i is obtained by combining the support vectors SV_{i-1}'s from the previous update and the training data present at the server_i. The final classifier thus learned can be made known to each service-provider. Figure 3.2 shows the data flow during the incremental learning method.

We evaluated the performance of SVM classifier for spam filtering and found to get the state of art accuracies. Furthermore, the ability to privately and efficiently execute the SVMs over a collaborative setup, makes it a natural model to work with.

1.5.3 Private Training

We now present the details of the privacy preserving SVM classification which constructs the joint SVM classifier from the data distributed at multiple service providers. The data of each serivce provider is kept private while the final classifier is relvealed to all the service providers. It is assumed that each service provider are semi-honest i.e they follow the protocol strictly, but may try to learn additional details from what they see during the execution. While the assumption is quite strict, it enables an efficient algorithm.

To securely build the decision boundary over the distributed data without revealing the data of each server to the others, we securely compute the *kernel matrix* $K_{ij} = K(x_i, x_j)$. As the kernel matrix contains only the dot products of the data vectors, securely computing the *gram matrix G where* $G_{ij} = x_i.x_j$ is enough. Secure computation of dot-product of two vectors can be done by applying Oblivious Transfer Protocol[**?**] as applied in Blind Vision[**?**].

Algorithm 2 Secure Dot Product

1: Service Provider A has input vector $\mathbf{x}\ \epsilon R^n$.

2: Service Provider B has input vector $\mathbf{y}\ \epsilon R^n$.

3: B generates a random vector $\mathbf{b}\ \epsilon R^n$.

4: **for** $i = 1$ to $i \leq n$ **do**

5: B enumerates all possible x_i values and constructs a h-dimensional vector $\mathbf{a}$, s.t $a_i = y_i * x_i - b_i$, h is the maximum hash value of the rabin function used and $x_i \epsilon [0...h]$

6: A uses OT_1^h with x_i as index, to choose the appropriate element from the vector $\mathbf{a}$ and stores it as a_i.

7: **end for**

8: A and B sum their private vectors $\mathbf{a}$ and $\mathbf{b}$, respectively, to obtain the shares $a = \sum_{j=1}^{n} a_j$ and $b = \sum_{j=1}^{n} b_j$ of the dot product $x^T.y$.

The output of the above protocol is secure in the sense that A and B obtain their private shares $\mathbf{a}$ and $\mathbf{b}$ such that $a + b = x^T.y$. This process is applied in an iterative manner as shown in Figure 3.2 to obtain the final classifer. At the end of the iteration the classifier is revealed to all the service providers.

1.5.4 Security Analysis

The protocol is clearly correct and secure for both service providers. In the above protocol,

1. *From A to B*: In step 6, A uses OT with x_i as an index to choose an element from the vector a. Because OT is secure, B can not learn which element A has chosen and hence can learn nothing about the vector $\mathbf{x}$.

2. *From B to A*: For each element, B lets A pick one element from vector $\mathbf{a}$ and since $\mathbf{a}$ is the sum of the vector $\mathbf{y}$ with some random vector $\mathbf{b}$, A can learn nothing about $\mathbf{y}$

from **a**.

The protocol is linear in n- the dimentionality of the input vectors **x** and **y**. However, applying the protocol iteratively has a serious limitation i.e., *Utilizing the classifier from previous service providers the service provider(adversary or spammer) at the tail can develop a classifer with poor performance.* This is a serious problem to the remaining providers who have shared the information inorder to enhance the performance of the Anti-spam classifer. In other words this iterative protocol has lost control over the tailenders in enhancing the performance of the classifer.

One possible approach to overcome this limitation would be to pair up the clients and then run this iterative protocol. Each pair executes the secure dot protodct protocol. This would not yield any improvements if both the clients paired are collaborating each other. So, the pairing should be in such a way that the clients are not collaborating or friends. The potential competitors are to be identified, pair them up and moreover, they have to be placed at the end of the iteration to reduce the effect of potential spammers.

A more rigorous solution to this problem could be obtained by using random graphs. Initially the graph is a forest with all nodes(clients) and zero edges. At each iteration an edge randomly chosen from some uniform distribution is added between any two trees. Adding an edge to the graph is equivalent to executing the secure dot product algorithm to obtain an intermediate classifer. After k iterations we obtain the final classifier which is revealed to all clients. However, we haven‘t analyzed formally the effect of this approach in our work and this will be apart of our future work.

Chapter 2

Conclusions

2.1 Experiments and Discussion

We have argued that using shingles as features for the email would enhance the performance of the classifier as the syntactic and semantic properties of similar emails have been restored by generating the similar set of features. Further to reduce the amount of communication overhead involved in training the SVM we adopted an alternate approach, Incremental SVM training.

In this section we test these methods on the the publicly available TREC07 [**?**] email corpus which is a standard benchmark set for anti-spam filter evaluations. Emails having a minimum of 500 characters have been selected from the corpus. The selected data set of 16590 emails contains an equal number of ham and spam messages. The training data is randomly partitioned in batches of fixed size of 1000 emails, each batch corresponding to a distint service provider. Incremental SVM is updated at each service provider, where it uses the previously learned support vectors and the new data set to upgrade the SVM classifier.

Figure 3.3 indicates the observed accuracies of the SVM classifier against the percentage

of emails trained. As expected, the use of shingles in our approach has a strong feature preserving capabilities and shows a better accuracy with large number of shingles used as features for the emails. The results also demonstrate that the notion of collaborative efforts is knowledge sharing. With 50% of messages trained, the SVM classifier is best found to performance with an accuracy of 94.46% when the parameters settings are w=3 and features=480. The results obtained are comparable to the current approaches like Collaborative spam filtering with heterogeneous agents, Alpacas. Even with comparable accuracies, using shingles as a features for the emails is preferred because of its resistance towards character replacement attacks.

Figure 3.4 shows the variations of accuracies with the window size. Intuitively, for a very small window size the features of emails (spam or ham) are likely to be highly similar because of commonly occurring sequences of characters. And for large window size, the features are likely to be very different even for similar emails because each character appears in several windows. So, it is expected that the classification results are greatly affected by the variations in the window size. But, the results on using SVM as a classifier in our approach, confirm that the window size does not affect the accuracies of the classifier. The expected variations in the classification results are not observed in our approach in contrary to the observations in Alpacas. The classifier used in Alpacas is based on a similarity measure between the feature sets of the emails and so the variations are observed. SVM is very robust, adaptability to large number of features by optimally finding the decision boundary. This nature of SVM made adjustments to the classifer with the variations in the window size.

Figure 3.5 shows the comparison of the communication overheads involved by adapting incremental SVM learning with the normal SVM. Communication overhead is used in the sense that the number of emails circulated between the client and the server at each update

of the SVM. The results indicate that the communication cost has been reduced by almost 67%. We further plan to do evaluations on various sizes on data sets having different spam and ham percentages.

2.2 Conclusion

Our work presents a better approach for privacy preserving anti-spam filtering. Experiments on using shingles as features for emails has been proven very effective with SVM classifier. In particular we made use of Incremental SVM learning to reduce the communication overhead invoved in the data flow among the clients. In the earlier approach, the Alpacas system, the performance with an overall low privacy breach is achieved by querying smaller set of features. Our approach complements it for collaborative inference attacks i.e, the privacy of emails is ensured by following secure dot product. However, we have noticed a limitation in our system that the tailenders have a major control on the final classifer which might be targeted by spammers. In our future we are exploring ways to overcome on this problem using random graphs.

Chapter 3

Figures

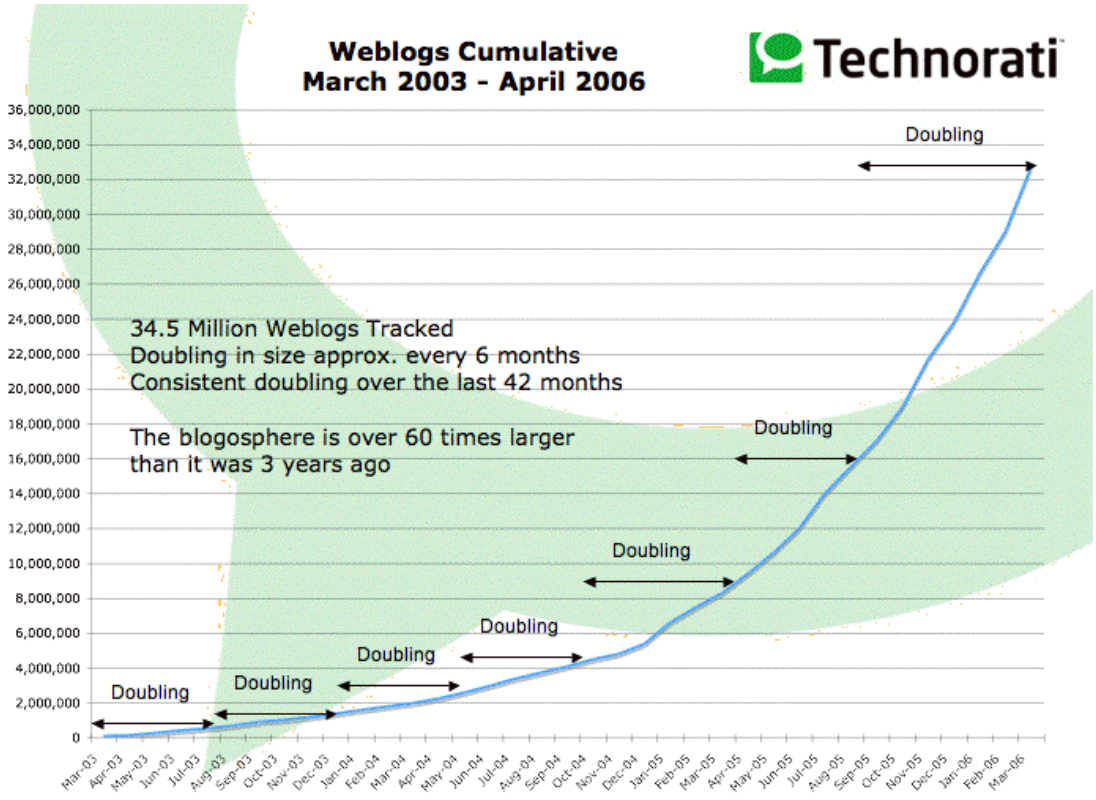

Figure 3.1: Statistics of Spam

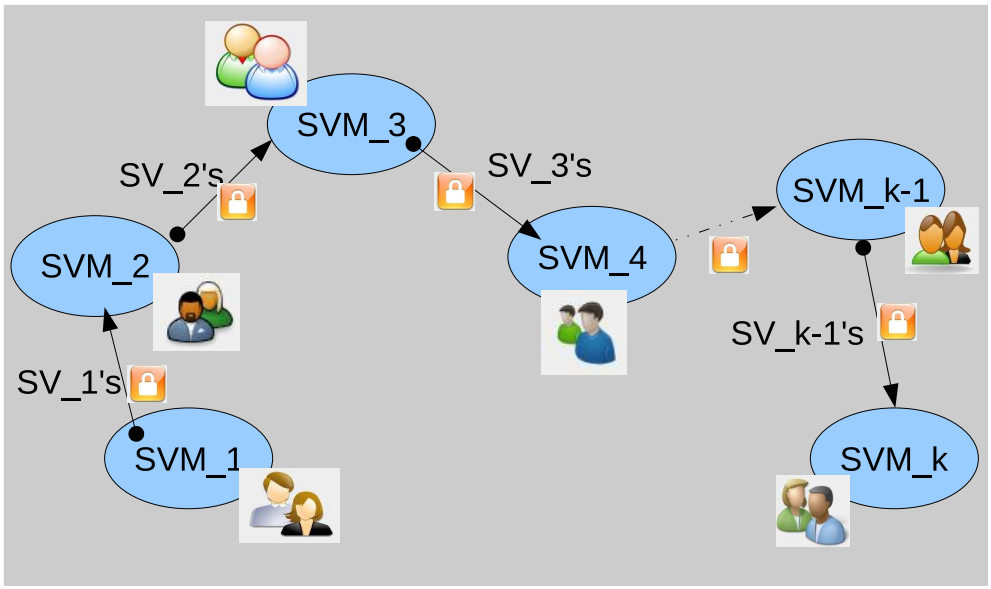

Figure 3.2: Incremental SVM Learning. At each service provider, the classifier is built upon the support vectors learned earlier. The final classifier, thus learned, is made known to all.

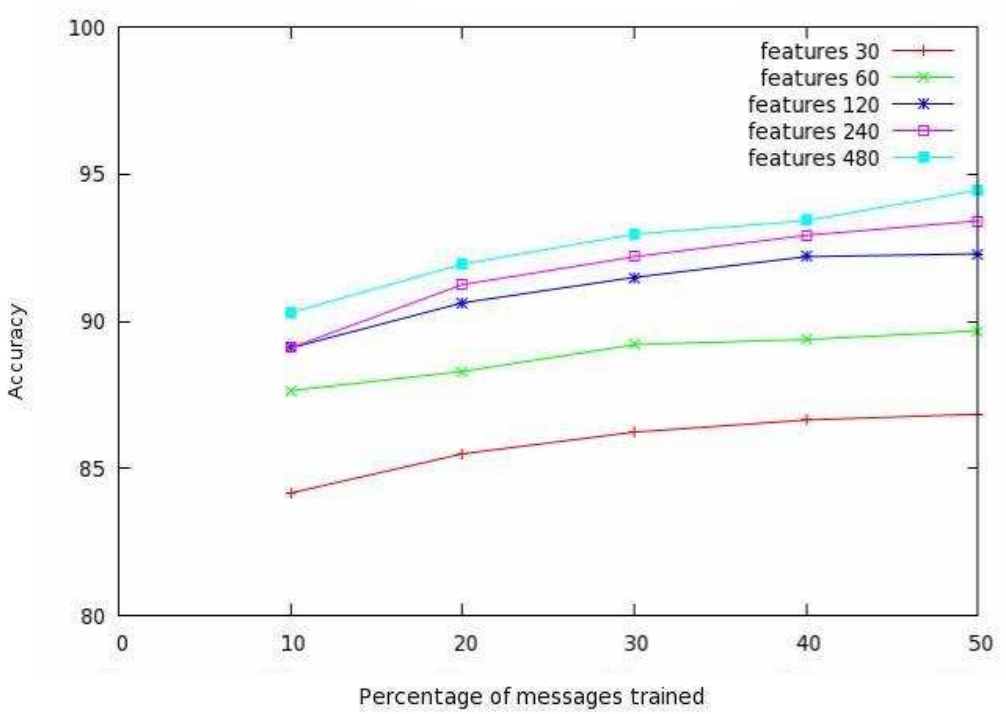

Figure 3.3: Performance of Incremental SVM against messages trained or updates

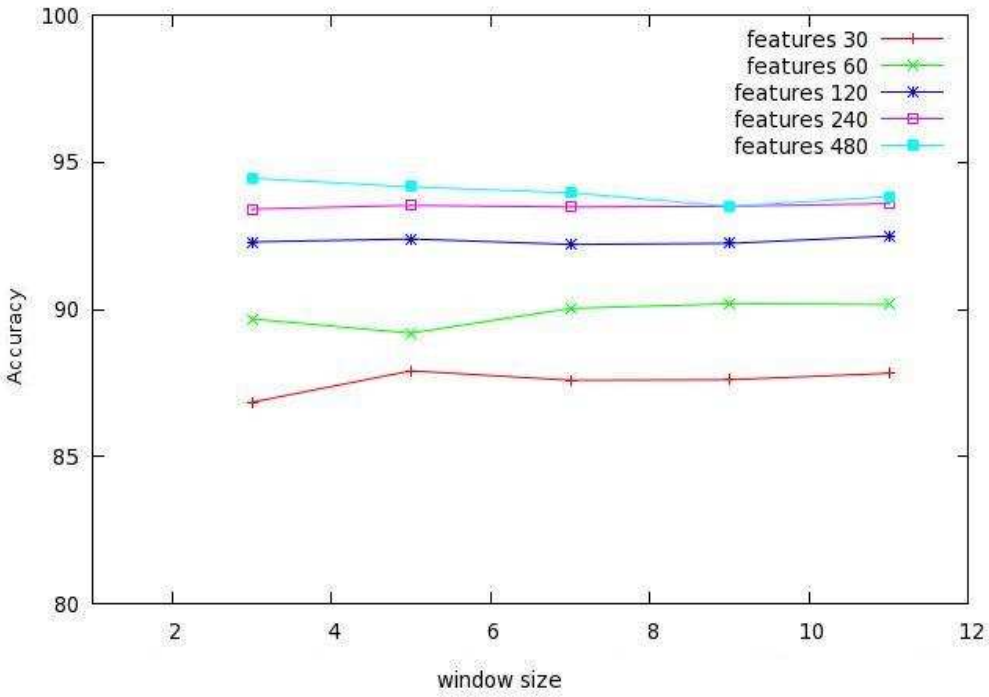

Figure 3.4: Performance of Incremental SVM against Window size

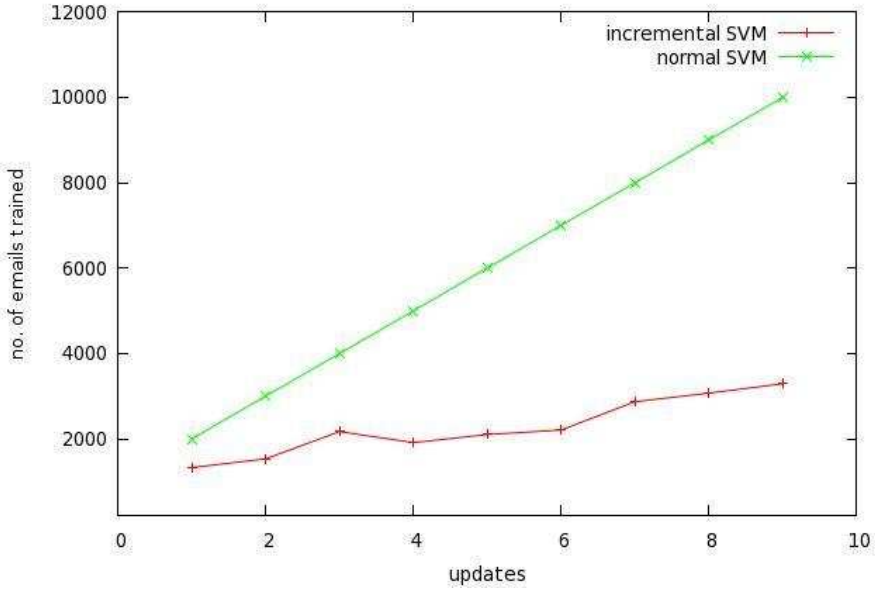

Figure 3.5: Communication Overhead of incremental and normal SVM

Bibliography

[1] Androutsopoulos, I., Koutsias, J., Chandrinos, K., Paliouras, G., and Spyropoulos, C. An evaluation of naive bayesian anti-spam filtering. *Arxiv preprint cs/0006013* (2000).

[2] Avidan, S., and Butman, M. Blind vision. *LECTURE NOTES IN COMPUTER SCIENCE 3953* (2006), 1.

[3] Broder, A. Some applications of Rabins fingerprinting method. In *Sequences II: Methods in Communications, Security, and Computer Science* (1993), Citeseer, pp. 143–152.

[4] Broder, A. On the resemblance and containment of documents. In *Compression and Complexity of Sequences 1997. Proceedings* (1997), pp. 21–29.

[5] Clifton. Chris, Zhu. Michael, V. J. Privacy Preserving Data Mining. *Springer* (2006).

[6] Cormack, G., and Lynam, T. Spam track preliminary guidelines-TREC 2005, 2005.

[7] De Guerre, J. The mechanics of Vipul's Razor technology. *Network Security 2007*, 9 (2007), 15–17.

[8] Domeniconi, C., and Gunopulos, D. Incremental support vector machine construction. In *icdm* (2001), Citeseer, pp. 589–592.

[9] Feingold, M., Corzine, M., Wyden, M., and Nelson, M. Data-mining moratorium act of 2003. *US Senate Bill (proposed)* (2003).

[10] Goldreich, O. Secure multi-party computation. *Working Draft* (2000).

[11] Lyman, P., Varian, H., Dunn, J., Strygin, A., and Swearingen, K. How much information, 2003.

[12] Nicholas, T. Using adaboost and decision stumps to identify spam e-mail.

[13] Rabin, M. Fingerprinting by random polynomials. Tech. rep., Technical Report TR-15-81, Center for Research in Computing Technology, Harvard University, 1981.

[14] Ramaswamy, L., Iyengar, A., Liu, L., and Douglis, F. Automatic detection of fragments in dynamically generated web pages. In *Proceedings of the 13th international conference on World Wide Web* (2004), ACM New York, NY, USA, pp. 443–454.

[15] Research, F. Industry Statistics. *http://www.ferris.com/research-library/industry-statistics/*.

[16] Scholkopf, B., and Smola, A. *Learning with kernels*. Citeseer, 2002.

[17] Schryver, V. Distributed checksum clearinghouse, 2005.

[18] Sculley, D., and Wachman, G. Relaxed online SVMs in the TREC Spam filtering track. In *Sixteenth Text REtrieval Conference (TREC-2007)* (2007).

[19] Shih, D., Chiang, H., and Lin, B. Collaborative spam filtering with heterogeneous agents. *Expert Systems with Applications 35*, 4 (2008), 1555–1566.

[20] Stone, B. Spam back to 94% of All E-mail. *The New York Times, http://bits.blogs.nytimes.com/2009/03/31/spam-back-to-94-of-all-e-mail/* (Aug 2009).

[21] Team, S. Spamassassin. *URL http://www. spamassassin. org/. Accessed 27*, 02–04.

[22] Vaidya, J., Yu, H., and Jiang, X. Privacy-preserving SVM classification. *Knowledge and Information Systems 14*, 2 (2008), 161–178.

[23] Vapnik, V. Structure of statistical learning theory. *Computational Learning and Probabilistic Reasoning* (1996), 3.

[24] Vapnik, V. *The nature of statistical learning theory*. Springer Verlag, 2000.

[25] William, S. Cryptography and network security. *New Jersey* (1999).

[26] Zhong, Z., Ramaswamy, L., and Li, K. ALPACAS: A Large-scale Privacy-Aware Collaborative Anti-spam System.