Improved Online Support Vector Machines Spam Filtering Using String Kernels

Ola Amayri and Nizar Bouguila

Concordia University, Montreal,
Quebec, Canada H3G 2W1
{o_amayri,bouguila}@encs.concordia.ca

Abstract. A major bottleneck in electronic communications is the enormous dissemination of spam emails. Developing of suitable filters that can adequately capture those emails and achieve high performance rate become a main concern. Support vector machines (SVMs) have made a large contribution to the development of spam email filtering. Based on SVMs, the crucial problems in email classification are feature mapping of input emails and the choice of the kernels. In this paper, we present thorough investigation of several distance-based kernels and propose the use of string kernels and prove its efficiency in blocking spam emails. We detail a feature mapping variants in text classification (TC) that yield improved performance for the standard SVMs in filtering task. Furthermore, to cope for realtime scenarios we propose an online active framework for spam filtering.

Keywords: Support Vector Machines, Feature Mapping, Spam, Online Active, String Kernels.

1 Introduction

Electronic mail has gained immense usage in everyday communication for different purposes, due to its convenient, economical, fast and easy to use nature over traditional methods. Beyond the rapid proliferation of legitimate emails lies adaptive proliferation of unwanted emails that take the advantage of the internet, known as spam emails. Variety of techniques have been developed to mitigate sufferings of spam emails. In particular, many machine learning (ML) techniques have been employed in the sake of spam filtering such as Boosting Trees, k-nearest neighbor classifier, Rocchio algorithm, Naive Bayesian classifier, Ripper and SVMs [4]. SVMs have made a large contribution to the development of spam email filtering. Based on SVMs, different schemes have been proposed through TC approaches. Recent studies on spam filtering, using SVMs, have focused on deploying classical kernels which neglects the structure and the nature of the text. Along with common Bag-of-Word (BoW) feature mapping approach in a batch mode [6]. In this paper, we propose an automated spam filtering in realtime, that improves the blocking of spam emails and reduce the misclassification of legitimate emails. To reach this goal, we focus on three key aspects:

E. Bayro-Corrochano and J.-O. Eklundh (Eds.): CIARP 2009, LNCS 5856, pp. 621–628, 2009.

first we explore several feature mapping strategies in context of text categorization. We intensively investigate the effect of various combinations of term frequency, importance weight and normalization on spam filtering performance. Second, we compare and analyze the use of various string kernels and different distance-based kernels for spam filtering. In addition, we provide detailed results for a fair comparison between different feature mapping and kernel classes using typical spam filtering criteria. Finally, we propose a framework of various online modes for spam filtering. We discuss the use of online SVMs, Transductive Support Vector Machines (TSVMs) and Active Online SVMs for spam filtering. We study proposed modes using different feature mapping and kernel classes, also.

This paper is organized as follows: in next section number of feature mapping choices that have been employed to transform email data into feature vectors usable by machine learning methods are outlined. In section 3 we briefly introduce Support Vector Machine, along with investigation of different kernels classes used in spam filtering tasks. Section 4 describes different online SVMs modes. In section 5 we report empirical results of proposed approaches. Finally conclusions are presented in Section 6.

2 Input Data Format

Many researchers have pointed out the importance of text representation in the performance of TC using SVMs. In this section, briefly, we discuss different approaches that have been applied in text representation. Generally, supervised TC is engaged into three main phases: term selection, term weighting, and classifier learning. Among existing approaches, the text representation dissimilarity can be shown either on what one regards the meaningful units of text or what approach one seeks to compute term weight. Terms are usually identified with words syntactically or statistically. In BoW, for instance, the extraction of features is based on defining a substring of contiguous characters *word* w, where word boundary is specified using a set of symbolic delimiters such as whitespace, etc. Using k-mer (i.e. k-gram)approach, however, the document can be represented by predefined sequences of contiguous characters (i.e. sub-strings) of length k. Moreover, term weighting phase is a vital step in TC, involves converting each document d to vector space which can be efficiently processed by SVMs. Term weights can be considered by occurrence of term in the corpus (term frequency) or by its presence or absence (binary). In our experiments we adopted seven term weighting schemes similar to [8]. In particular, the first three term weighting schemes are different variants of term frequency which are: TF, $\log TF$ and ITF. Next four schemes are different combinations of term frequency and importance weight which are: $TF\text{-}IDF$, $\log TF\text{-}IDF$ and $ITF\text{-}IDF$. For large corpus, if we consider each distinct feature for spam filtering then a very dense feature space F is constructed. To solve this issue researchers suggest *Stop words* and *Stemming*. More sophisticated feature selection is found by computing the probability of dependency between term w and category c such as Information Gain (IG), CHI statistic (χ^2) and Term strength (TS). In addition, spammers

attempt to defeat the spam filtering by writing short emails. To handle such problem, in our experiments, we normalize emails by using L_2-normalization which yields generally to best error bounds.

3 Support Vector Machines: Kernels

SVMs are known to give accurate discrimination in high feature space [3]. Furthermore, they received a great attention in many applications such as text classification. The state of the art of SVMs evolved mapping the learning data from input space into higher dimensional feature space where the classification performance is increased. This has been developed by applying several kernels each with individual characteristics. Lately, the choice of the kernel became a widely discussed issue, since it reveals different performance result for various applications. SVMs in classification problems, such as spam filtering, explore the similarity between input emails implicitly using inner product $K(X,Y) = \langle \phi(X), \phi(Y) \rangle$ i.e. kernel functions. In distance based learning [12] the data samples $\overrightarrow{x}$ are not given explicitly but only by a distance function $d(\overrightarrow{x}, \overrightarrow{x}')$. In our experiments we compare the effectiveness of different kernels in this class which are: *Gaussian*, *Laplacian*, χ^2, *Inv multi*, *Polynomial*, and *Sigmoid* [12]. In contrast of distance-based kernels, string kernels define the similarity between pair of documents by measuring the total occurrence of shared substrings of length k in feature space F. In this case, the kernel is defined via an explicit feature map. In our experiments we adopted two classes of string kernels: the position-aware string kernel which takes advantage of positional information of characters/substrings in their parent strings and the position-unaware string kernel which does not. We applied Weighted Degree kernel (WD) and Weighted Degree kernel with Shift (WDs) [11] for position-aware kernels. Additionally, for position-unaware kernels, Subsequence String kernel (SSK) [10], Spectrum kernel and Inexact String Kernels such as Mismatch kernel, Wildcard kernel and Gappy kernel [9].

4 Support Vector Machines: Learning and Classification

In reality, spam filtering is typically tested and deployed in an online setting, by proceeding incrementally. To this end, Online SVM model presents to the filter a sequence of emails, where sequence order is determined by the design (i.e. it might be in chronological order or even randomized). We adopted a simple algorithm introduced in [7] to adapt batch model to online model. Initially, suppose a spam filter is trained on training set. In SVM model, examples closer to the hyperplane are most uncertain and informative. Those examples are presented by support vectors (SVs). Furthermore, SVs are able to summarize the data space and preserve the essential class boundary. Consequently, in our model, we use SVs as seeds (starting point) for the future retraining and discard all non-SVs samples. To this end, labeled data sets are not often affordable prior classification and label data set is time consuming and tedious process. To overcome this

problem, TSVM constructs a maximum margin by employing large collection of unlabeled data jointly with a few labeled examples for improving generalization performance [5]. To cope with realtime scenario, Online Active SVM presents messages to the filter in a stream, where the filter must classify them one by one. Each time a new example is presented to the filter, the filter has the option of requesting a label for the given message using *Angle diversity* approach [1].

5 Experimental Results

Recently, spam filtering using SVM classifier has been tested and deployed using linear kernel weighted using binary weighting schemes [13,2,4]. We extend previous research on spam filtering, as we consider three main tasks. Firstly, we compare the use of various feature mapping techniques described in section 2 for spam email filtering. Secondly, we investigate the use of string kernels with a number of classical kernels and exploring that in terms of accuracy, precision, recall, F1 and running classification time. Thirdly, we report results from experiments testing the effectiveness of the online, TSVM and online active learning methods, presented in previous sections. In seek of comparison, the performance of each task is examined using the same version of spam data set which is trec05-p1[1] (92,189 labeled spam and legitimate emails) and the same pre-processing is applied for different kernels. In the purpose of comparison evaluation, SVM^{light}[2] package was used as an implementation of SVMs. We set the value of ρ in

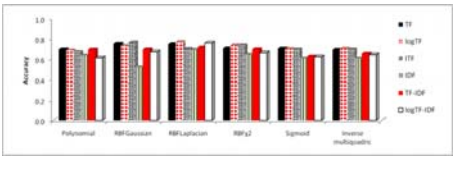

Fig. 1. The performance of SVM spam filtering on trec05-1, where IG has been applied

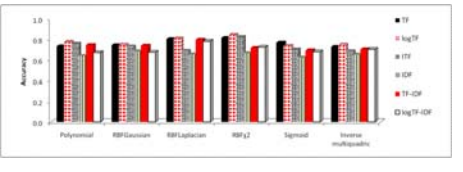

Fig. 2. The performance of SVM spam filtering on trec05-1, where χ^2 has been applied

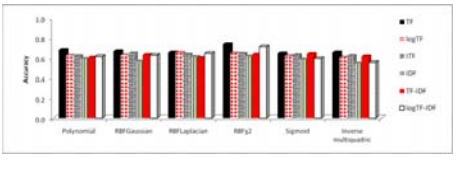

Fig. 3. The performance of SVM spam filtering on trec05-1, where TS has been applied

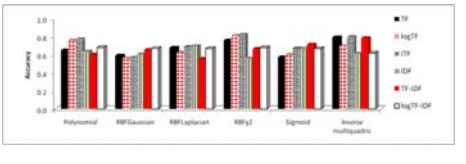

Fig. 4. The performance of SVM spam filtering on trec05-1, where stop words list and stemming have been applied

[1] http://plg1.cs.uwaterloo.ca/cgi-bin/cgiwrap/gvcormac/foo
[2] http://svmlight.joachims.org/

Table 1. The performance of batch SVM (SVM), TSVM, Online SVM (ON) and Online Active SVM (ONA) spam filtering on trec05-1 using distance-based kernels normalized using L_2-norm, and without removing stop words

Kernel	Precision				Recall				F1			
	SVM	TSVM	ON	ONA	SVM	TSVM	ON	ONA	SVM	TSVM	ON	ONA
Polynomial.TF	0.779	0.800	0.798	0.787	0.805	0.807	0.810	0.817	0.792	0.804	0.804	0.802
Gaussian.TF	0.876	0.824	0.870	0.869	0.850	0.804	0.858	0.842	0.863	0.814	0.864	0.855
Laplacian.TF	0.919	0.903	0.904	0.919	0.879	0.843	0.868	0.863	0.899	0.872	0.886	0.890
χ^2.TF	0.893	0.892	0.891	0.898	0.879	0.876	0.880	0.844	0.886	0.884	0.886	0.870
Sigmoid.TF	0.897	0.897	0.876	0.899	0.817	0.788	0.820	0.794	0.855	0.839	0.847	0.843
Inv multi.TF	0.779	0.791	0.798	0.787	0.824	0.807	0.868	0.813	0.801	0.799	0.831	0.799
Polynomial.logTF	0.854	0.858	0.850	0.861	0.838	0.836	0.842	0.823	0.846	0.847	0.846	0.842
Gaussian.logTF	0.819	0.815	0.833	0.837	0.849	0.824	0.853	0.835	0.834	0.820	0.843	0.836
Laplacian.logTF	0.901	0.871	**0.910**	0.885	0.893	0.887	0.890	0.888	0.897	0.879	0.900	0.886
χ^2.logTF	**0.920**	**0.918**	0.903	**0.921**	**0.915**	**0.908**	**0.916**	**0.906**	**0.917**	**0.913**	**0.910**	**0.913**
Sigmoid.logTF	0.803	0.803	0.815	0.795	0.836	0.812	0.840	0.825	0.819	0.808	0.827	0.809
Inv multi.logTF	0.797	0.799	0.789	0.765	0.873	0.841	0.882	0.859	0.833	0.820	0.833	0.809
Polynomial.ITF	0.835	0.857	0.831	0.827	0.836	0.803	0.845	0.818	0.835	0.829	0.838	0.822
Gaussian.ITF	0.872	0.812	0.889	0.863	0.796	0.792	0.801	0.790	0.832	0.802	0.843	0.825
Laplacian.ITF	0.793	0.749	0.836	0.765	0.790	0.772	0.790	0.772	0.792	0.760	0.812	0.768
χ^2.ITF	0.899	0.892	0.905	0.897	0.893	0.892	0.883	0.887	0.896	0.892	0.894	0.892
Sigmoid.ITF	0.769	0.783	0.797	0.774	0.795	0.765	0.783	0.793	0.782	0.774	0.790	0.783
Inv multi.ITF	0.795	0.742	0.800	0.778	0.788	0.769	0.795	0.760	0.791	0.756	0.797	0.769
Polynomial.IDF	0.683	0.708	0.698	0.679	0.768	0.716	0.778	0.757	0.723	0.712	0.736	0.716
Gaussian.IDF	0.778	0.743	0.759	0.758	0.765	0.769	0.780	0.785	0.771	0.756	0.769	0.771
Laplacian.IDF	0.747	0.746	0.760	0.729	0.801	0.709	0.812	0.786	0.773	0.727	0.785	0.756
χ^2.IDF	0.719	0.713	0.729	0.732	0.737	0.767	0.759	0.778	0.728	0.739	0.744	0.755
Sigmoid.IDF	0.748	0.703	0.799	0.735	0.709	0.689	0.714	0.696	0.728	0.696	0.754	0.715
Inv multi.IDF	0.728	0.749	0.718	0.737	0.698	0.701	0.733	0.674	0.713	0.725	0.725	0.704
Polynomial.TF-IDF	0.854	0.807	0.850	0.836	0.823	0.825	0.835	0.809	0.838	0.816	0.842	0.822
Gaussian.TF -IDF	0.836	0.802	0.842	0.831	0.835	0.817	0.846	0.826	0.835	0.809	0.844	0.829
Laplacian.TF-IDF	0.832	0.850	0.820	0.861	0.889	0.885	0.898	0.893	0.860	0.867	0.857	0.877
χ^2.TF-IDF	0.816	0.816	0.818	0.808	0.793	0.761	0.796	0.785	0.804	0.788	0.807	0.796
Sigmoid.TF-IDF	0.797	0.749	0.790	0.794	0.831	0.782	0.842	0.809	0.814	0.765	0.815	0.801
Inv multi.TF-IDF	0.796	0.749	0.790	0.788	0.811	0.804	0.820	0.816	0.803	0.776	0.805	0.801
Polynomial.logTF-IDF	0.742	0.747	0.768	0.732	0.752	0.740	0.769	0.745	0.747	0.743	0.768	0.739
Gaussian.logTF-IDF	0.726	0.793	0.758	0.739	0.769	0.705	0.773	0.747	0.747	0.747	0.766	0.743
Laplacian.logTF-IDF	0.827	0.840	0.831	0.818	0.866	0.863	0.866	0.888	0.846	0.851	0.848	0.851
χ^2.logTF-IDF	0.817	0.803	0.820	0.800	0.891	0.786	0.892	0.793	0.852	0.795	0.855	0.797
Sigmoid.logTF-IDF	0.813	0.790	0.832	0.810	0.795	0.712	0.787	0.767	0.804	0.749	0.809	0.788
Inv multi.logTF-IDF	0.789	0.824	0.779	0.799	0.775	0.732	0.786	0.766	0.782	0.776	0.782	0.782

RBF kernels, and C for the soft margin via 10-fold cross-validation. For TSVM, the value of C^* is set similar to C. We ran experiments for similar length of substrings used in string kernels (value of k parameter). We varied the value of the decay vector λ for SSK to see its influence in the performance, where the higher value of λ gives more weight to non-contiguous substrings ($\lambda = 0.4$ has provided a better performance). For mismatch kernel (k, m), wildcard kernel (k,w) and gappy kernel (g,k), the experiments have taken place with fixed values of allowed mismatch, wild card and gaps which are $m = 1$, $w = 2$, $k = 2$,

Table 2. The performance of batch SVM, TSVM, Online SVM (ON) and Online Active SVM (ONA)spam filtering on trec05-1 using string kernels

Kernel	Precision				Recall				F1			
	TSVM	SVM	ONA	ON	TSVM	SVM	ONA	ON	TSVM	SVM	ONA	ON
SSk	0.9278	0.9509	0.9505	0.9590	0.9281	0.9531	0.9468	0.9729	0.9279	0.9404	0.9341	0.9372
Spectrum	0.9249	0.9657	0.9466	0.9800	0.9362	0.9345	0.9478	0.9479	0.9305	0.9325	0.9315	0.9320
Mismatch	0.8745	0.8967	0.9012	0.9033	0.9100	0.9277	0.9145	0.9265	0.8919	0.9094	0.9006	0.9050
Wildcard	0.9356	0.9689	0.9432	0.9900	0.8890	0.9056	0.8952	0.9112	0.9117	0.9086	0.9102	0.9094
Gappy	0.9190	0.9678	0.9256	0.9943	0.9167	0.9012	0.9189	0.9043	0.9178	0.9094	0.9136	0.9115
WD	0.8978	0.9571	0.9189	0.9700	0.9021	0.9100	0.9067	0.9190	0.8999	0.9049	0.9024	0.9037
WDs	0.8987	0.9124	0.9109	0.9167	0.9189	0.9080	0.9001	0.9088	0.9087	0.9083	0.9085	0.9084

Table 3. Execution time for different combinations of frequency and distance kernels in different modes: batch SVM (SVM) and online SVM (ON). These times do not include time spent in feature mapping.

	Polynomail		Gaussian		Laplacian		χ^2		Sigmoid		Inv multi	
	SVM	ON	SVM	ON	SVM	ON	SVM	ON	SVM	ON	SVM	ON
TF	0.03	0.03	0.4	0.3	3.48	2.4	0.49	0.38	0.1	0.1	3.43	3.1
logTF	0.45	0.4	1.25	1.1	5.54	4.44	5.47	5	6.36	6	3.21	2.45
ITF	1.55	1.35	2.39	1.55	**6.48**	**6.30**	5.49	4.55	3.03	3.03	4.02	4
IDF	0.47	0.46	2.02	2	2.05	2	4.57	4.5	3.07	3.01	4.11	4.05
TF-IDF	0.02	0.02	0.39	0.37	3	3	0.35	0.34	0.55	0.53	3.43	3.1
logTF-IDF	0.45	0.4	1.2	1.1	4.3	3.25	5	5	6.4	5.59	3.2	2.55

Table 4. Execution time for string kernels in different modes: batch SVM (SVM) and online SVM (ON). These times do not include time spent in feature mapping.

	SSk	Spectrum	Mismatch	Wildcard	Gappy	WD	WDs
SVM	20.08	19.45	**21.43**	20.47	20.02	19.56	20.32
ON	7.37	18.55	19.30	20.20	**19.32**	19.10	20.00

respectively, as allowing higher mismatch will increase the computation cost. To examine the use of different feature mapping, we evaluate trec05p-1 data set using generic combinations of feature mapping approaches along with different distance-based kernels (i.e. $Polynomial.TF$ all normalized using L_2). Clearly, classification performance is better when no feature selection techniques were applied. Figures 1, 2, 3 and 4 show slight degradation in performance comparing with the performance of spam filtering using all distinct words. Tables 1 through 2 show the comparison results obtained for distance-based and string kernels along with different feature mapping combinations deployed in different SVMs modes. We achieved best performance with emails weighted using different variant of TF and normalized using $L2$-norm. Besides, the kernel choice is crucial in classification problem. The good kernel is the kernel that gives a

valuable information about the nature of data, and report good performance. RBF kernels have the higher performance among distance based kernels in most of experiments. For instance, string kernels, in particular SSK, yields improved performance compared to batch supervised learning, with reduced number of labels and reasonable computational time. On the basis of kernels comparison string kernels performed better than distance-based kernels. Besides F1, precision, and recall, we evaluate involved kernels in terms of their computational efficiency, in order to provide insight into the kernels impact on filtering time. We measured the duration of computation for all kernels (see results in Tables 3 and 4). As expected, string kernels were defeated by their computational cost [9]. In addition, results show a clear dominance of online active learning methods, compared to both Online SVM and TSVM.

6 Conclusion

The ultimate goal of our extensive study of automated spam filtering using SVMs is to develop a devoted filter for spam problem in order to improve the blocking rate of spam emails (high precision) and reduce the misclassification rate of legitimate emails (high recall). The path towards such powerful filter is a thorough study of powerful classifier to accurately distinguish spam emails from legitimate emails and to consider the dynamic nature of spam problem. In this paper, particularly, we intensively study SVM email classification performance given by deployed kernels in realtime environment. Indeed, we described the use of string kernels in order to improve spam filter performance. We implemented, tested, integrated various preprocessing algorithms based on term frequency, importance weight with normalization to investigate their impact on classifier performance. Moreover, we applied algorithms to adapt batch theoretical models to online real world models using string kernels and well-performed preprocessing combinations, and hence maximize the overall performance. Further enhancement can be made by taking into account user feedback and the structure of emails which is richer than only text.

References

1. Brinker, K.: Incorporating diversity in active learning with support vector machines. In: Proceedings of the Twentieth International Conference on Machine Learning, pp. 59–66 (2003)
2. Cormack, G.V., Bratko, A.: Batch and on-line spam filter comparison. In: Proceedings of the Third Conference on Email and Anti-Spam, California, USA (2006)
3. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning 20(1), 229–273 (1995)
4. Drucker, H., Vapnik, V., Wu, D.: Support vector machines for spam categorization. IEEE Transactions on Neural Networks 10(5), 1048–1054 (1999)
5. Joachims, T.: Transductive inference for text classification using support vector machines. In: Proceedings of the sixteenth International Conference on Machine Learning (ICML 1999), San Francisco, US, pp. 200–209 (1999)

6. Kolcz, A., Alspector, J.: Svm-based filtering of e-mail spam with content-specific misclassification costs. In: Proceedings of the Workshop on Text Mining, California, USA, pp. 123–130 (2001)
7. Lau, K.W., Wu, Q.H.: Online training of support vector machine. Pattern Recognition 36(8), 1913–1920 (2003)
8. Leopold, E., Kindermann, J.: Text categorization with support vector machines. how to represent texts in input space? Machine Learning 46(13), 423–444 (2002)
9. Leslie, C., Kuang, R.: Fast string kernels using inexact matching for protein sequences. Journal of Machine Learning Research 5, 1435–1455 (2004)
10. Lodhi, H., Saunders, C., Shawe-Taylor, J., Cristianini, N., Watkins, C.: Text classification using string kernels. The Journal of Machine Learning Research 2(1), 419–444 (2002)
11. Rtsch, G., Sonnenburg, S., Schlkopf, B.: Rase: Recognition of alternatively spliced exons in c. elegans. Bioinformatics 21(1), i369–i377 (2005)
12. Scholkopf, B., Smola, A.: Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, Cambridge (2001)
13. Sculley, D., Wachman, G.: Relaxed online svms for spam filtering. In: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, Amsterdam, Netherlands, pp. 415–422 (2007)