An Adaptive Fusion Algorithm for Spam Detection

Congfu Xu, Baojun Su, and Yunbiao Cheng, *Zhejiang University*

Weike Pan, *Shenzhen University, Hong Kong Baptist University*

Li Chen, *Hong Kong Baptist University*

Using email services as an example, an adaptive fusion algorithm for spam detection offers a general content-based approach. The method can be applied to non-email spam detection tasks with little additional effort.

Spam detection has become a critical component in various online systems to filter harmful information, for example, false information in email or SNS services, malicious clicks in advertising engines, fake user-generated content in social networks, and so on. Most commercial systems adopt a machine learning classifier, such as Naive Bayes, logistic regression, or support vector machines, to detect the spams. However, one single classifier might not be able to capture diverse aspects of spams, which can change dynamically. As a response, we designed a fusion algorithm based on a set of online learners, instead of relying on a single base model. We use email spam detection as an example, although our algorithm isn't limited to the email domain.

An email spam is defined as an unsolicited email sent indiscriminately, directly or indirectly, by a sender having no current relationship with the recipient.[1] Email spams will affect employees' working efficiency and cause bandwidth wastage. Besides email spams, there are an increasing number of similar abuses in social medias[2] and mobile services.[3] We're surrounded by spams in our daily life, motivating us to detect and filter them accurately.

There exist a variety of popular methods for fighting spams, such as DNS-based Blackhole Lists,[4] greylisting,[5] spamtraps,[6] extrusion,[7] online machine learning models,[8] feature engineering,[3] matrix factorization,[2] and so on. As spammers become more sophisticated and manage to outsmart static antispam methods, content-based approaches have shown promising accuracy in combating them. In this article, we also focus on content-based approaches.

To overcome the limitations of a single machine learning classifier, here we borrow ideas from the information fusion community and devise an adaptive fusion algorithm for spam detection (AFSD). AFSD aims to build an integrated spam detector from a collection of lightweight online learners in an adaptive manner. As far as we know, AFSD holds the best area-under-curve (AUC) score on the Text Retrieval Conference (TREC) spam competition datasets. (For others' work on spam detection, see the "Related Work in Spam Detection" sidebar.)

Adaptive Fusion for Spam Detection

We take some real-time arriving text, such as emails, $\{(\boldsymbol{x}, y)\}$, where $\boldsymbol{x} \in \mathbb{R}^{d\times 1}$ is the feature representation of a certain email and $y \in \{1, 0\}$

1541-1672/14/$31.00 © 2014 IEEE
Published by the IEEE Computer Society

is the label denoting whether the email is spam "1" or ham (that is, nonspam or good email) "0". We also have k online learners, $f_j(\boldsymbol{x}; \boldsymbol{\theta})$, j = 1, …, k, with the prediction of the jth learner on email $\boldsymbol{x}$ as follows: $y_j = f_j(\boldsymbol{x}; \boldsymbol{\theta})$.

Our goal is to learn an adaptively integrated prediction model,

$$f(\boldsymbol{x}) = \{f_j(\boldsymbol{x};\theta)\}_{j=1}^{k},$$

which minimizes the accumulated error during the entire online learning and prediction procedure. As far as we know, we're the first in designing an adaptive fusion algorithm for real-time email spam detection.

Feature Representation

Various methods have been proposed to extract features from text, among which tokenization is probably the most popular one. However, tokenization might not obtain good results when facing spammers' intentional obfuscation or good word attack, especially for the task of email spam detection. We thus drop tokenization and adopt n-grams of non-tokenized text strings, which is a simple yet effective method.[9] The feature space includes all n-character substrings of the training data. We construct a binary feature vector for each email,

$$\boldsymbol{x} = [x_i]_{i=1}^{d} \in \{1,0\}^{d\times 1},$$

where x_i indicates the existence of the corresponding ith feature,

$$x_i = \begin{cases} 1, & \text{if the } i\text{th feature exists} \\ 0, & \text{otherwise.} \end{cases}$$

Note that such representation is efficient for online learning and prediction environments.

The Algorithm

In this section, we describe our algorithm in detail, including the link function, mistake-driven training, and adaptive fusion.

Link function. Considering that the prediction scores of different online learners are usually in different ranges, we thus adopt a commonly used sigmoid function,

$$\sigma(z) = 1/(1 + e^{-z})$$

to map raw prediction scores returned by online learners to a common range between 0 and 1.

To make the scores from different online learners more comparable, we follow the approaches used in data normalization and introduce a bias parameter y_0 and an offset parameter y_Δ in the link function to achieve effect of centering and scaling,[10]

$$P_j(\boldsymbol{x}) = \sigma\left(\frac{y_j - y_0}{y_\Delta}\right) = \sigma\left(\frac{f_j(\boldsymbol{x};\theta) - y_0}{y_\Delta}\right),$$

where different bias parameter and offset parameter values shall be used for different online learners, which can be determined empirically via cross validation. In our experiments, we set the value of bias and offset empirically in order to make the scores of different base classifiers to be in a similar range, which will then make the scores more comparable.

Mistake-driven training of online learners. We consider a qualified online learner from four perspectives. First, it shall be a vector space model or can be transformed into a vector space model, because then the email text only needs to be processed once, and can be used for all online learners. Second, it shall be a lightweight classifier with acceptable accuracy, which will most likely help achieve high prediction accuracy in the final algorithm. Third, the model parameters can be learned incrementally, because it will be trained in a mistake-driven manner to make the classifier more competitive. Fourth, the output of a model should be a score

IEEE Intelligent Systems

IEEE Computer Society Publications Office
10662 Los Vaqueros Circle, PO Box 3014
Los Alamitos, CA 90720-1314

Lead Editor
Brian Kirk
bkirk@computer.org

Editorial Management
Tammi Titsworth

Manager, Editorial Services
Jenny Stout

Publications Coordinator
isystems@computer.org

Director, Products & Services
Evan Butterfield

Senior Manager, Editorial Services
Robin Baldwin

Digital Library Marketing Manager
Georgann Carter

Senior Business Development Manager
Sandra Brown

Senior Advertising Coordinator
Marian Anderson
manderson@computer.org

Submissions: For detailed instructions and formatting, see the author guidelines at www.computer.org/intelligent/author.htm or log onto *IEEE Intelligent Systems*' author center at Manuscript Central (www.computer.org/mc/intelligent/author.htm). Visit www.computer.org/intelligent for editorial guidelines.

Editorial: Unless otherwise stated, bylined articles, as well as product and service descriptions, reflect the author's or firm's opinion. Inclusion in *IEEE Intelligent Systems* does not necessarily constitute endorsement by the IEEE or the IEEE Computer Society. All submissions are subject to editing for style, clarity, and length.

Related Work in Spam Detection

Spam detection—as a critical component in various online systems—has attracted much attention from both researchers and practitioners. Parallel to the categorization of recommender systems,[1] we can divide spam-detection algorithms into content-based[2] and collaborative-based approaches.[3] Most previous works are content-based approaches, while some recent works exploit social networks for spam detection[4–6] or spammer detection.[3] Our proposed algorithm of adaptive fusion for spam detection (AFSD) belongs to the content-based class.

Fusion algorithms or ensemble methods[7–9] have achieved great success in classification tasks. Lior Rokach[7] categorized the ensemble methods from different dimensions. Sašo Džeroski and Bernard Ženko[8] showed that a well-designed stacking-based ensemble method can beat a best single method. Eitan Menahem and his colleagues[9] proposed a three-layer stacking-based ensemble learning method, which works well for multiclass classification problems. Thomas Lynam and his colleagues[10] built a fused model by combining all the Text Retrieval Conference participants' spam filters, and also studied the fusion approaches thoroughly. They showed that a set of independently developed spam filters can be combined in simple ways to provide substantially better performance than any individual filter. Those so-called simple fusion approaches[10] are similar to the bagging approaches in our experiments.

The differences between our proposed fusion algorithm, AFSD, and other fusion approaches can be identified from three aspects: we introduce a link function with the effect of result scaling, which makes the prediction scores of online learners more comparable; we train our online learners in a mistake-driven manner, which allows us to obtain a series of highly competitive online learners; and we design a fusion algorithm to adaptively integrate the predictions of online learners.

References

1. F. Ricci et al., eds., *Recommender Systems Handbook*, Springer, 2011.
2. B. Su and C. Xu, "Not So Naive Online Bayesian Spam Filter," *Proc. 21st Conf. Innovative Applications of Artificial Intelligence*, 2009, pp. 147–152.
3. Y. Zhu et al., "Discovering Spammers in Social Networks," *Proc. 26th AAAI Conf. Artificial Intelligence*, 2012.
4. H.-Y. Lam and D.-Y. Yeung, "A Learning Approach to Spam Detection Based on Social Networks," *Proc. 4th Conf. Email and AntiSpam*, 2007, pp. 832–840.
5. Q. Xu et al., "SMS Spam Detection Using Contentless Features," *IEEE Intelligent Systems*, vol. 27, no. 6, 2012, pp. 44–51.
6. C.-Y. Tseng and M.-S. Chen, "Incremental SVM Model for Spam Detection on Dynamic Email Social Networks," *Proc. IEEE 15th Int'l Conf. Computational Science and Engineering*, 2009, pp. 128–135.
7. L. Rokach, "Taxonomy for Characterizing Ensemble Methods in Classification Tasks: A Review and Annotated Bibliography," *Computational Statistics and Data Analysis*, vol. 53, no. 12, 2009, pp. 4046–4072.
8. S. Džeroski and B. Ženko, "Is Combining Classifiers with Stacking Better than Selecting the Best One?" *J. Machine Learning*, vol. 54, no. 3, 2004, pp. 255–273.
9. E. Menahem, L. Rokach, and Y. Elovici, "Troika—An Improved Stacking Schema for Classification Tasks," *J. Information Sciences*, vol. 179, no. 24, 2009, pp. 4097–4122.
10. T.R. Lynam, G.V. Cormack, and D.R. Cheriton, "On-Line Spam Filter Fusion," *Proc. 29th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval*, 2006, pp. 123–130.

or probability. Any classifier meeting these four requirements is acceptable in our adaptive fusion algorithm.

In our AFSD algorithm, we implement eight online learners: Naive Bayes, not so Naive Bayes (NSNB),[11] Winnow,[12] Balance Winnow,[13] online logistic regression,[9] an algorithm by the Harbin Institute of Technology (HIT),[14] passive aggressive,[15] and online Perceptron Algorithm with Margins.[16]

We use thick thresholding in our mistake-driven training procedure of the online learners. For a labeled email instance, the model parameters of an online learner won't be updated if the email has been well classified as a spam or ham. Otherwise, this email will be used to train the online learner until it has been well classified. An email is considered as not well classified by the online learner j in the following cases:

$$\begin{cases} P_j(\boldsymbol{x}) \le 0.75, & \text{if } \boldsymbol{x} \text{ is spam} \\ P_j(\boldsymbol{x}) \ge 0.25, & \text{if } \boldsymbol{x} \text{ is ham,} \end{cases} \quad (1)$$

which means that an email is well classified only if the prediction score is larger than 0.75 or smaller than 0.25. We consider an email with a prediction score of larger than 0.75 as a spam with high confidence; and with a prediction score of smaller than 0.25 as a ham with little uncertainty. The mechanism of thick thresholding will thus emphasize the difficult-to-classify email instances, and finally produce a well-trained online learner with prediction scores well away from the uncertain decision range—for example, scores located in the range between 0.25 and 0.75. Models trained in this way, usually have high generalization ability, which is also observed in our experiments when we compare our online learners with the champion solutions of the corresponding datasets.

Adaptive fusion of online learners. Once we've trained the online learners, we have to find a way to integrate them for final prediction. We use $w_1, w_2, \ldots, w_k$ to denote the weight of those k online learners. For any incoming email $\boldsymbol{x}$, we calculate the final prediction score via a weighted combination,

$$P(\boldsymbol{x}) = \sum_{j=1}^{k} w_j P_j(\boldsymbol{x}) / \sum_{j=1}^{k} w_j,$$

where the weight of each online learner is initialized as 1, and will be updated

adaptively according to the corresponding online learner's performance. Once we've learned the integrated prediction model, we can decide whether the email $\boldsymbol{x}$ is spam or ham via $\delta(P(\boldsymbol{x}))$ with

$$\delta(z) = \begin{cases} 1, & z > 0.5 \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

During the training procedure of adaptive fusion, for an online learner $f_j(\boldsymbol{x}; \boldsymbol{\theta})$, if its prediction is the same as that of the final prediction, $\delta(P_j(\boldsymbol{x})) = \delta(P(\boldsymbol{x}))$, the corresponding weight, w_j, won't be updated; otherwise, w_j will be updated adaptively. More specifically, if an online learner makes a correct prediction while the integrated model makes an incorrect prediction, $\delta(P_j(\boldsymbol{x})) = y$ and $\delta(P(\boldsymbol{x})) \neq y$, the weight w_j will be increased, otherwise w_j will be decreased,

$$w_j = \begin{cases} w_j + \gamma \Delta w_+, & \text{if } \delta(P_j(\boldsymbol{x})) = y, \delta(P_j(\boldsymbol{x})) \neq y \\ w_j + \gamma \Delta w_-, & \text{if } \delta(P_j(\boldsymbol{x})) \neq y, \delta(P_j(\boldsymbol{x})) = y \end{cases} \quad (3)$$

where y is the true label of email $\boldsymbol{x}$, $\Delta w_+ > 0$ and $\Delta w_- < 0$ are the award and punishment on the weight, respectively, and γ is the learning rate. In our experiments, we fix the award $\Delta w_+ = 20$, the punishment $\Delta w_- = -1$, and the learning rate $\gamma = 0.02$. From Equation 3, we can see that a classifier with correct prediction will be awarded with more weight, while a classifier with incorrect prediction will be punished with reduced weight.

Finally, we have the complete AFSD algorithm:

Input: The real-time arriving text $\{(\boldsymbol{x}, y)\}$, k online learners $f_j(\boldsymbol{x}; \boldsymbol{\theta})$, $j = 1, \ldots, k$;

Output: The learned k online learners $f_j(\boldsymbol{x}; \boldsymbol{\theta})$ and the corresponding weight w_j, $j = 1, \ldots, k$.

1. Feature extraction of the text;
2. Mistake-driven training of each online learner as shown in Equation 1;
3. Adaptive fusion of online learners as shown in Equation 3.

The AFSD algorithm can be implemented efficiently, because the extracted features can be used for all online learners (vector space models); the online learners are trained independently and thus can be implemented via multithread programming or in a distributed platform; and the adaptive fusion procedure won't update the model parameters of the trained online learners, but only the weight. Furthermore, both the model parameters learned in the mistake-driven training step and the weight learned in the adaptive fusion step can be updated online, which means that our fusion algorithm AFSD is actually an online learning algorithm with the ability to receive the training data on the fly. Our experiments are also conducted in an online setting.

Experimental Results

The benchmarks that we used in our experiments consist of the commonly used 2005 to 2007 TREC datasets (see http://trec.nist.gov/data/spam.html), the 2008 Collaboration, Electronic Messaging, Anti-Abuse, and Spam Conference (CEAS) dataset (see http://plg.uwaterloo.ca/~gvcormac/ceascorpus), and the NetEase dataset (authorized from the largest email service provider in China, NetEase, see http://www.163.com). Specifically, TREC05p, TREC06p, TREC06c, TREC07p, CEAS08, and NetEase have 92,189, 37,822, 64,620, 75,419, 137,705, and 208,350 instances, respectively. There are four types of emails in the NetEase dataset: spam, advertisement, subscription, and regular emails. For the NetEase dataset, we convert the spam detection task into a binary classification task, spam versus ham (advertisement, subscription, and regular).

For each dataset, we use 4 grams to extract features from character strings of an email and use binary coding to represent the corresponding feature's existence. To reduce the impact of long messages, we only keep the first 3,000 characters of each message.

To take the false positive rate into consideration, we use (1-AUC) percent[17] in our evaluations, which is commonly used in email spam detection.

For evaluation, we use the standard TREC spam detection evaluation toolkit (see http://plg.uwaterloo.ca/~gvcormac/jig), which ensures that all the results obtained by different approaches on the same datasets are comparable.

Note that the baseline of Winner in our results refers to the champion solutions of the corresponding competitions: TREC05p,[18] TREC06p,[19] TREC07p,[20] and CEAS08 (see www.ceas.cc/2008/challenge/results.pdf). The baseline 53-ensemble refers to the fusion algorithm with 53 base classifiers.[21]

Study of Online Learners

To ensure reliable performance of AFSD, we must guarantee the performance of each online learner. Furthermore, the predictability of each online learner is also useful in the analysis of fusion approaches and the selection of a subset of online learners.

Table 1 shows the results of (1-AUC) percent, from which we can see that NSNB[11] has the best performance on TREC05p, TREC06p, and TREC06c; HIT has the best performance on TREC06c and CEAS08; and passive aggressive has the best performance on NetEase. Winner is the best only on TREC07p, and Balance Winnow is close (0.0061 compared to 0.0055). When we consider the total (1-AUC) percent of all six datasets, the result of the champion solutions of each year is only slightly better than the worst online learner (Winnow) in our AFSD

Table 1. The (1-AUC) percent scores of online learners.

Dataset	Winner	Naive Bayes	Not so Naive Bayes (NSNB)	Winnow	Balance Winnow	Logistic regression	HIT	Passive aggressive	Perceptron Algorithm with Margins
TREC05p*	0.0190	0.0197	**0.0073**	0.0356	0.0152	0.0150	0.0108	0.0137	0.0137
TREC06p	0.0540	0.0495	**0.0278**	0.0599	0.0624	0.0359	0.0305	0.0347	0.0348
TREC06c	0.0023	0.0137	**0.0003**	0.0061	0.0023	0.0006	**0.0003**	0.0006	0.0006
TREC07p	**0.0055**	0.0163	0.0079	0.0189	0.0061	0.0070	0.0086	0.0066	0.0066
CEAS08	0.0233	0.0039	0.0006	0.0037	0.0009	0.0013	**0.0005**	0.0012	0.0013
NetEase	-	0.0218	0.0149	0.0220	0.0133	0.0135	0.0128	**0.0127**	0.0128
Total	-	0.1249	**0.0588**	0.1462	0.1002	0.0733	0.0635	0.0695	0.0698

*TREC = Text Retrieval Conference; CEAS = Collaboration, Electronic Messaging, Anti-Abuse, and Spam Conference. The bold numbers are the best results on the corresponding datasets.

Table 2. The (1-AUC) percent scores of our adaptive fusion algorithm AFSD and other fusion approaches.

Dataset	NSNB*	Bagging	Voting	AFSD
TREC05p	0.0073	0.0065 (+11.0%)	0.0070 (+ 4.1%)	**0.0055 (+24.7%)**
TREC06p	0.0278	0.0176 (+36.7%)	0.0193 (+30.6%)	**0.0155 (+44.2%)**
TREC06c	0.0003	0.0001 (+66.7%)	0.0002 (+33.3%)	**0.0001 (+66.7%)**
TREC07p	0.0079	0.0058 (+26.6%)	0.0060 (+24.1%)	**0.0058 (+26.6%)**
CEAS08	0.0006	0.0006	0.0006	**0.0004 (+33.3%)**
NetEase	0.0149	0.0095 (+36.2%)	0.0096 (+35.6%)	**0.0092 (+38.3%)**
Total	0.0588	0.0401 (+31.8%)	0.0427 (+27.4%)	**0.0365 (+37.9%)**

* We use NSNB as the best single online learner.

Table 3. The (1-AUC) percent scores of our proposed fusion algorithm (AFSD) and other approaches.

Dataset	Winner	53-ensemble	AFSD
TREC05p	0.0190	0.0070	**0.0055 (+71.1%)**
TREC06p	0.0540	0.0200	**0.0155 (+71.3%)**
TREC06c	0.0023	-	**0.0001 (+95.7%)**
TREC07p	**0.0055**	-	0.0058
CEAS08	0.0233	-	**0.0004 (+98.3%)**
Total	0.1041	-	**0.0273 (+73.8%)**

algorithm, which shows that the online learners are competitive. We think that the high prediction accuracy of our online learners is from the generalization ability of online learners trained in the mistake-driven manner, described previously.

From Table 1, we can see that NSNB outperforms other online learners. Hence, we use NSNB as the best single online learner to compare with fusion approaches in the next subsection.

Study of Fusion Algorithms

We demonstrate the effectiveness of our AFSD algorithm by comparing it with the following approaches:

- *Best online learner.* As a baseline algorithm for comparison, we use NSNB as the best single filter.
- *Bagging.* We use the average prediction scores of online learners, $\delta\left(\sum_{j=1}^{k} P_j(\boldsymbol{x};\theta)/k\right)$, where $\delta(z)$ is the same as that in Equation 2. Bagging can be considered as a special case of AFSD when the weights of online learners are fixed as $w_j = 1$, $j = 1, \ldots, k$.
- *Voting.* We use the majority votes of online learners, $\delta\left(\sum_{j=1}^{k} \delta(P_j(\boldsymbol{x};\theta))/k\right)$, where $\delta(z)$ is the same as that in Equation 2.

The results of different fusion approaches are shown in Tables 2 and 3.

From Table 2, we can see that bagging improves the baseline algorithm (that is, NSNB) on average by 31.8 percent on (1-AUC) percent, voting by 27.4 percent and AFSD by 37.9 percent. Our proposed algorithm AFSD is significantly better than both bagging and voting, which clearly shows the effect of our adaptive fusion algorithm.

From Table 3, we can see that AFSD outperforms the TREC champion solutions (that is, Winner) significantly on most datasets, and is only slightly worse than Winner on TREC07p. The total receiver-operating-characteristic (ROC) score of TREC champion solutions (Winner) is 0.1041, while AFSD gives the total score 0.0273, which improves Winner's ROC score by 73.8 percent. AFSD also achieves better results (using only eight classifiers) than a recent ensemble classifier using 53 online learners.[21]

Study of Weight on Online Learners

Generally, the more classifiers integrated, the slower the entire system

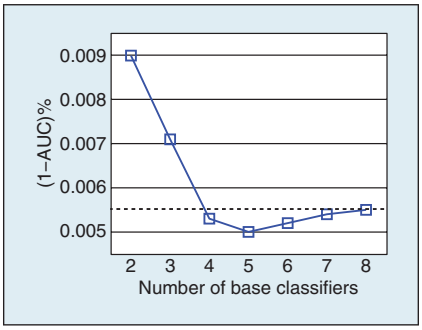

Figure 1. The (1-AUC) percent score of our proposed fusion algorithm AFSD with different numbers of online learners on the TREC05p dataset. A selected subset of classifiers is able to achieve comparable or even slightly better performance than using the whole set of classifiers.

would be when deployed. Moreover, increasing the number of online learners doesn't guarantee better prediction performance.[22] Therefore, it's important to select a relatively small subset of the online learners to be both efficient and effective.

Our AFSD is able to achieve this goal. After training, each online learner has a weight indicating its importance on the final results, for example, on the TREC06p dataset: Naive Bayes (13.48), NSNB (6.26), Winnow (12.14), Balance Winnow (9.88), logistic regression (5.38), HIT (12.48), passive aggressive (6.14), and Perceptron Algorithm with Margins (5.72). During subset selection, we propose to select a highly weighted online learner with high priority. For example, we will select Naive Bayes and HIT if two online learners are needed, and Naive Bayes, HIT, and Winnow if three online learners are needed.

The results of using different numbers of online learners are shown in Figure 1. We can see that if the number of online learners is smaller than four, the result is worse than that of using all eight online learners. And when four to seven online learners are integrated, we can obtain better results than that of using all online learners. Our main observation from Figure 1 is that a selected subset of classifiers is able to achieve comparable or even slightly better performance than using the whole set of classifiers.

Experimental results on five public competition and one industry dataset show that AFSD produces significantly better results than several state-of-the-art approaches, including the champion solutions of the corresponding competitions.

For future work, we're interested in continuing our work in designing strategies for automatic selection of a base classifier subset, applying our fusion algorithm to spam detection tasks in social media and mobile computing domains, and studying the generalization ability of our proposed algorithm. ■

Acknowledgments

We thank the Natural Science Foundation of China (grants 60970081 and 61272303), and the National Basic Research Program of China (973 Plan, grant 2010CB327903) for their support.

THE AUTHORS

Congfu Xu is an associate professor with the College of Computer Science, Zhejiang University. His research interests include information fusion, data mining, artificial intelligence, recommender systems, and sensor networks. Xu has a PhD in computer science from Zhejiang University. Contact him at xucongfu@zju.edu.cn.

Baojun Su works in search advertising as a software engineer at Microsoft STC-Beijing. His research interests include information fusion, data mining, and spam detection. Su has an MS in computer application technology from Zhejiang University. Contact him at freizsu@gmail.com.

Yunbiao Cheng works as a software engineer at Yahoo! Software Research and Development in Beijing. His research interests include information fusion, data mining, and recommender systems. Cheng has an MS in computer application technology from Zhejiang University. Contact him at yunbiaoch@gmail.com.

Weike Pan is a lecturer with the College of Computer Science and Software Engineering, Shenzhen University. He's also the information officer of *ACM Transactions on Intelligent Systems and Technology*. His research interests include transfer learning, recommender systems, and statistical machine learning. Pan has a PhD in computer science and engineering from the Hong Kong University of Science and Technology. He is the corresponding author. Contact him at panweike@szu.edu.cn.

Li Chen is an assistant professor in the Department of Computer Science at Hong Kong Baptist University. Her research interests include intelligent Web technologies, recommender systems, and e-commerce decision support. Chen has a PhD in computer science from the Swiss Federal Institute of Technology in Lausanne. Contact her at lichen@comp.hkbu.edu.hk.

References

1. G.V. Cormack, "Email Spam Filtering: A Systematic Review," *Foundations and Trends in Information Retrieval*, vol. 1, no. 4, 2007, pp. 335–455.
2. Y. Zhu et al., "Discovering Spammers in Social Networks," *Proc. 26th AAAI Conf. Artificial Intelligence*, 2012, pp. 171–177.
3. Q. Xu et al., "SMS Spam Detection Using Contentless Features," *IEEE Intelligent Systems*, vol. 27, no. 6, 2012, pp. 44–51.
4. J. Jung and E. Sit, "An Empirical Study of Spam Traffic and the Use of DNS Black Lists," *Proc. 4th ACM Sigcomm Conf. Internet Measurement*, 2004, pp. 370–375.
5. J.R. Levine, "Experiences with Greylisting," *Proc. 2nd Conf. Email and Anti-Spam*, 2005; http://ceas.cc/2005/papers/120.pdf.
6. M. Prince et al., "Understanding How Spammers Steal Your E-Mail Address: An Analysis of the First Six Months of Data from Project Honey Pot," *Proc. 2nd Conf. Email and Anti-Spam*, 2005; http://ceas.cc/2005/papers/163.pdf.
7. R. Clayton, "Stopping Spam by Extrusion Detection," *Proc. 1st Conf. Email and Anti-Spam* (CEAS), 2004; http://ceas.cc/2004/172.pdf.

8. J. Ma et al., "Learning to Detect Malicious URLs," *ACM Trans. Intelligent Systems Technology*, vol. 2, no. 3, 2011, article no. 30.
9. J. Goodman, "Online Discriminative Spam Filter Training," *Proc. 3rd Conf. Email and Anti-Spam*, 2006; http://ceas.cc/2006/22.pdf.
10. T.W. Anderson, *An Introduction to Multivariate Statistical Analysis*, 3rd ed., Wiley-Interscience, 2003.
11. B. Su and C. Xu, "Not So Naive Online Bayesian Spam Filter," *Proc. 21st Conf. Innovative Applications of Artificial Intelligence*, 2009, pp. 147–152.
12. J. Kivinen, M.K. Warmuth, and P. Auer, "The Perceptron Algorithm versus Winnow: Linear versus Logarithmic Mistake Bounds When Few Input Variables Are Relevant," *Artificial Intelligence*, vol. 97, nos. 1–2, 1997, pp. 325–343.
13. V.R. Carvalho and W.W. Cohen, "Single-Pass Online Learning: Performance, Voting Schemes and Online Feature Selection, *Proc. Int'l Conf. Knowledge Discovery and Data Mining*, 2006, pp. 8–13.
14. H. Qi et al., "Joint NLP Lab between HIT 2 at CEAS Spam-Filter Challenge 2008," *Proc. 5th Conf. Email and Anti-Spam*, 2008; www.ceas.cc/2008/papers/china.pdf.
15. K. Crammer et al., "Online Passive Aggressive Algorithms," *J. Machine Learning Research*, vol. 7, Mar. 2006, pp. 551–585.
16. D Sculley, G.M. Wachman, and C.E Brodley, "Spam Filtering Using Inexact String Matching in Explicit Feature Space with On-Line Linear Classifiers," *Proc. 15th Text Retrieval Conf.*, 2006; http://trec.nist.gov/pubs/trec15/papers/tuftsu.spam.final.pdf.
17. J.A. Hanley and B.J. McNeil, "The Meaning and Use of the Area Under a Receiver Operating Characteristic (ROC) Curve," *Radiology*, vol. 143, no. 1, 1982, p. 29.
18. G. Cormack and T. Lynam, *TREC 2005 Spam Track Overview*, tech. report, Univ. Waterloo, 2005; http://plg.uwaterloo.ca/~gvcormac/trecspamtrack05/trecspam05paper.pdf.
19. G. Cormack, *TREC 2006 Spam Track Overview*, tech. report, Univ. Waterloo, 2006; http://trec.nist.gov/pubs/trec15/papers/SPAM06.OVERVIEW.pdf.
20. G. Cormack, *TREC 2007 Spam Track Overview*, tech. report, Univ. Waterloo, 2007; http://trec.nist.gov/pubs/trec16/papers/SPAM.OVERVIEW16.pdf.
21. T.R. Lynam and G.V. Cormack, "On-Line Spam Filter Fusion," *Proc. 29th Ann. Int'l ACM Sigir Conf. Research and Development in Information Retrieval*, 2006, pp. 123–130.
22. Z.H. Zhou, J. Wu, and W. Tang, "Ensembling Neural Networks: Many Could Be Better Than All," *Artificial Intelligence*, vol. 137, nos. 1–2, 2002, pp. 239–263.