Knowl Inf Syst (2012) 33:117–136
DOI 10.1007/s10115-011-0461-x

REGULAR PAPER

Online active multi-field learning for efficient email spam filtering

Wuying Liu · Ting Wang

Received: 5 August 2010 / Revised: 10 October 2011 / Accepted: 15 November 2011 /
Published online: 30 November 2011

Abstract Email spam causes a serious waste of time and resources. This paper addresses the email spam filtering problem and proposes an online active multi-field learning approach, which is based on the following ideas: (1) Email spam filtering is an online application, which suggests an online learning idea; (2) Email document has a multi-field text structure, which suggests a multi-field learning idea; and (3) It is costly to obtain a label for a real-world email spam filter, which suggests an active learning idea. The online learner regards the email spam filtering as an incremental supervised binary streaming text classification. The multi-field learner combines multiple results predicted by field classifiers in a novel compound weight schema, and each field classifier calculates the arithmetical average of multiple conditional probabilities calculated from feature strings according to a data structure of string-frequency index. Comparing the current variance of field classifying results with the historical variance, the active learner evaluates the classifying confidence and takes the more uncertain email as the more informative sample for which to request a label. The experimental results show that the proposed approach can achieve the state-of-the-art performance with greatly reduced label requirements and very low space-time costs. The performance of our online active multi-field learning, the standard (1-ROCA)% measurement, even exceeds the full feedback performance of some advanced individual text classification algorithms.

Keywords Online learning · Multi-field learning · Active learning · Email spam filtering · TREC spam track

W. Liu (✉) · T. Wang
College of Computer, National University of Defense Technology, 410073 Changsha, Hunan, China
e-mail: wyliu@nudt.edu.cn

T. Wang
e-mail: tingwang@nudt.edu.cn

W. Liu
Department of Language Engineering, PLA University of Foreign Languages,
471003 Luoyang, Henan, China

1 Introduction

Email spam is the bulk, promotional, and unsolicited message. Email spam filtering has been widely investigated since the early days of Information Filtering, and many statistical text classification (TC) algorithms have been proposed [1]. However, the rapid development of computing and communicating has made spam messages glutting over the world in recent years. It is critical to develop practical, efficient filtering approaches to facilitate the email spam filtering.

Real-world email spam filtering, an online application, is normally defined as an incremental supervised binary streaming TC task [2,3]. At the beginning of the task, the filter has no labeled emails. Emails are classified by the filter in their chronological sequence. According to the user's feedback (category label), the filter will update itself incrementally. This online learning mode requires space-time-efficient statistical TC algorithms.

In many previous statistical TC algorithms, an email is often treated as a single plain-text document, and text features are extracted within this single document. Actually, a full email (often including five natural text fields: Header, From, ToCcBcc, Subject, and Body) is a multi-field text document. Feature extraction in plain-text email documents makes many text features disturb each other, and text features from one field are often noisy to other fields. The multi-field text structure brings an opportunity, applying an ensemble multi-classifier [4,5], to improve the performance of email spam filtering.

Supposing to obtain a label without any cost, we can ideally implement the email spam filter as a full feedback filter [6]. The user's feedback for each email is communicated to the filter immediately following binary classification. Many previous statistical TC algorithms have been largely successful when given large-scale, fully labeled email sets. However, in practice, it is costly to obtain a label for a real-world email spam filter. Especially, it is unreasonable to require a user labeling every email in time, which defeats the full feedback filter. Active learning has been developed to reduce labeling costs by identifying only the informative samples to request labels. It has been proved that it is sufficient to label a small portion of a large unlabeled data set, in order to train an active classifier that can achieve high classification performance [7,8]. The active filter has a preset quota parameter of emails for which feedback can be requested immediately until the quota is exhausted. After each email is classified, the active filter must decide to request or decline feedback for the email. Declining feedback for some uninformative emails can save the quota for requesting feedback for later informative emails.

In this paper, we try to integrate online learning, multi-field learning, and active learning to form an improved approach. In the next section, we firstly review related works on the email spam filtering and then describe online supervised learning, multi-field learning, and active learning separately for the binary streaming TC. Secondly, we investigate the multi-field text structure of email documents and propose an online active multi-field learning approach in Sect. 3, including a historical-variance-based active learning method and a compound-weight-based linear combining method. Thirdly, we propose a lightweight field TC algorithm in Sect. 4. Finally, from the experiments described in Sect. 5, we find strong results with the proposed approach, greatly reducing the number of labels needed to achieve strong classification performance in email spam filtering. These results even exceed the full feedback performance of some advanced individual statistical TC algorithms. We conclude with a discussion on the implication of these results for real-world email spam filtering in Sect. 6.

2 Related works

There have been many online supervised binary statistical TC algorithms proposed for the email spam filtering up to the present. For instance: (1) Based on the Vector Space Model (VSM), the online Bayesian algorithm [9] uses the joint probabilities of words and categories to estimate the probabilities of categories for a given message; (2) The Relaxed Online Support Vector Machines (ROSVM) algorithm [10] relaxes the maximum margin requirement and produces nearly equivalent results, which has gained several best results at the TREC 2007 spam track; and (3) The online fusion of Dynamic Markov Compression (DMC) and logistic regression on character 4-gram algorithm [11] is the winner in the full feedback task of trec07p data set [12]. Moreover, the fuzzy rule-based classifying algorithm [13] through a two-stage genetic search and the personalized email prioritization algorithm [14] based on social networks are also interesting researches.

In addition to the above individual TC algorithms, ensemble approaches are also effective [15]. Our research [16] has proved that the multi-field text structure of emails is very useful, and our multi-field learning framework, an ensemble learning structure, can improve the overall performance (1-ROCA)% [17] of many individual TC algorithms, such as the online Bayesian algorithm and the relaxed online SVMs algorithm.

Actually, the greatest importance of active learning is to evaluate the informativeness of unlabeled samples. Each online supervised binary statistical TC algorithm can be improved to its corresponding active learning version by adding an evaluating procedure. The uncertainty sampling is widely used to evaluate the informativeness, normally based on a fixed threshold of classifying confidence. For instance: (1) The above ROSVM algorithm can be improved to its active learning version by the fixed-margin sampling method [18], which is the winner in the active learning task of trec07p data set; and (2) The 4-gram-based logistic regression algorithm [19], an online gradient ascent implementation, can be adapted to train only on emails for which the absolute value of the confidence score was less than a threshold. The threshold-based active learning algorithm is the improved version of the logistic regression algorithm, and this improved algorithm has gained the second rank in the TREC 2007 active learning task.

Email spam filtering is an online application. Unfortunately, the previous online supervised binary statistical TC algorithms, normally using the VSM, have to align vector dimensions, select features, and often lead to high-dimensional sparse and time-consuming problems. Moreover, the online application also faces an open incremental problem of the text feature space and cannot foreknow the vector space dimension. Some advanced individual TC algorithms may achieve high classification accuracy at the cost of complex training or updating, which undoubtedly causes space-time inefficiency, and are unsuitable for the online application. In this paper, the divide-and-conquer strategy breaks the complex TC problem into multiple simple sub-problems according to the multi-field text structure of emails, which can reduce the complexity of training or updating, and multiple field classifying results has been combined to improve the final classification accuracy.

In order to reduce the labeling cost, we make use of active learning to improve the previous algorithms. But the previous researches of active learning paid more attention to various sampling methods, without adopting the application-specific information. In this paper, the historical information of spam filtering and the multiple field classifying results are both helpful to improve sampling methods for the active learner to make a decision.

This paper proposes an online active multi-field learning approach, which has following features: (1) The lightweight field TC algorithm calculates the arithmetical average of multiple conditional probabilities predicted from feature strings according to the

Table 1 Dimensions of k-gram feature vector space under the two representations: SPTM and MFM

	1-gram	2-gram	3-gram	4-gram
SPTM	1,037,395	4,189,054	9,447,962	13,869,560
MFM	1,258,491	4,906,594	10,390,571	14,880,647

string-frequency index; (2) The compound-weight-based linear combining method considers the historical field classifying performance and the current field classifying contribution together; and (3) The historical-variance-based active learning method compares the current variance of field classifying results with the historical variance to choose the uncertain email as the informative sample. The goal of all these efforts is to seek solving the email spam filtering problem practically and efficiently.

3 Online active multi-field learning

Online active multi-field learning, a sort of ensemble learning, breaks a complex TC problem into multiple simple sub-problems according to the email text structure, and each sub-problem may have its own text features. The combined multiple field classifying results will be expected to improve the final classification accuracy, and the multiple field classifying results may also help the active learner to make a decision. The multi-field text structure of emails motivates our divide-and-conquer approach.

3.1 Multi-field text structure

In many statistical TC algorithms, a text document is normally represented as a text feature vector. The dimension of feature vector space, the total number of text features, reflects the representational granularity of the VSM. Previous research has shown that the overlapping word-level k-gram model can achieve promising results [20]. For email documents, single plain-text model (SPTM) and multi-field model (MFM) are two different representations. The SPTM ignores the field information of the text feature, regarding the same string occurrence in different fields as a single text feature, while the MFM treats it as distinct text features. The dimension of feature vector space for trec07p email set is shown in Table 1. For the two email representations, four overlapping word-level models are applied, respectively. For the MFM, the field structure is considered according to the five natural text fields of email documents.

Table 1 shows that the dimension of the MFM is larger than that of the SPTM for each k-gram model. For instance, this obvious difference between the two representations becomes 1,011,087 for the 4-gram model. The results of Table 1 indicate that text feature noises indeed exist in the SPTM. Because the more finely granular text feature can reduce the noises and increase the TC accuracy, this paper proposes an online active multi-field learning (OAMFL) framework, which is an alignment technique of text feature sources. In the OAMFL framework, text features are enhanced by their field information, and the undesired influences among text features from different fields are expected to be reduced.

3.2 OAMFL framework

Figure 1 shows the OAMFL framework for the binary TC of emails, including a splitter, several field classifiers, a combiner, and an active learner. The online learning process of

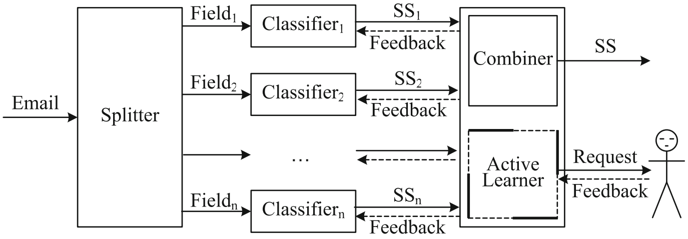


Fig. 1 Online active multi-field learning framework

the email spam filtering includes: (1) The splitter analyzes an email document and splits it into multiple text fields. (2) Each field classifier is obligated to process its corresponding text field, and outputs a result. The extracting of text features, the training or updating of field TC model, and the predicting of field classifying result are only localized within the corresponding text fields for each field classifier. (3) The combiner combines multiple results from the field classifiers to form the final result. (4) The active learner evaluates the classifying confidence according to the results from the field classifiers and makes a decision to request a label or not. (5) If the active learner decides to request a label, then it will send the feedback to the field classifiers, and each field classifier will incrementally update itself immediately.

Within the OAMFL framework, each field classifying result is not the traditional binary output but a spamminess score (SS), which is a real number reflecting the likelihood that the classified document is spam. The classical Bayesian conditional probability $P(spam|doc)$, shown in Eq. (1), reflects that likelihood.

$$P(spam|doc) = \frac{P(doc|spam)P(spam)}{P(doc|spam)P(spam) + P(doc|ham)P(ham)} \tag{1}$$

If the $P(spam|doc)$ is chosen to estimate the SS, the SS threshold T, shown in Eq. (2), can be used to make a binary judgment.

$$T = \frac{P(spam)}{P(spam) + P(ham)} \tag{2}$$

However, the value of SS and threshold is affected by the number distribution of labeled spams and hams, and the number of two categories labeled data is not fixed during the time of online filtering. In order to eliminate that influence of the number distribution and make the same SS value has the equivalent likelihood during the whole online filtering, this paper scales up the number of two categories labeled data to make $P(spam) = P(ham)$ and uses the scaled Bayesian conditional probability $P(spam|doc)$, shown in Eq. (3), to represent the SS, and then the SS threshold $T = 0.5$ will be a fixed point.

$$P(spam|doc) = \frac{P(doc|spam)}{P(doc|spam) + P(doc|ham)} \tag{3}$$

The effectiveness of the OAMFL framework depends on the following factors: the splitting strategy, the combining strategy, and the active learning strategy. And the space-time complexity of the OAMFL framework is mostly determined by the field TC algorithm.

3.3 Splitting strategy

The explicit splitting strategy is based on the native multi-field text structure. For instance, the splitter can easily extract five native fields (Header, From, ToCcBcc, Subject, and Body) for email documents.

Except the explicit splitting strategy, this paper also proposes a novel artificial splitting strategy, which is motivated by that there are normally some special texts with many strong distinguishing features in the header or body of an email. These special texts may include IP address texts, email address texts, URL texts, and phone number texts. We can extract these special texts by some regular expression rules to form artificial fields that do not really exist in actual email documents. This artificial splitting strategy is equivalent to increasing the statistical weight of these texts.

On the one hand, spammers may insert a forged email header to hide the true origin of the email [21] and try to confuse the spam body text. On the other hand, spammers dare not misspell the critical IP, email box, URL, and phone number with the expectation to be called back by spam recipients. Fortunately, the effectiveness of our artificial splitting strategy is based on the repeatability of texts and is independent of the reality and forgery. So, our artificial splitting strategy can be used in a real and forged email header.

This paper extracts two artificial fields (H.IP, H.EmailBox) for email documents. The H.IP field contains IP address texts and the H.EmailBox field contains email address texts in the Header field of email documents. Considering both natural fields and artificial fields within the OAMFL framework of this paper, the splitter implements a 7-field framework for email documents.

3.4 Combining strategy

The combining strategy used in the combiner is the key point to guarantee the effectiveness of the OAMFL framework. The linear combination of spamminess scores from field classifiers is a simple and efficient method, which is defined in Eq. (4); where the SS denotes the final spamminess score, the n denotes the number of field classifiers, and the SS_i denotes the spamminess score predicted by the ith field classifier. The coefficient α_i (a real number) can be set by different weighted strategies. The straightforward weighted strategy is arithmetical average calculating method: $\alpha_i = 1/n$, abbreviated as *cs1* combining strategy.

$$\mathrm{SS} = \sum_{i=1}^{n} \alpha_i \mathrm{SS}_i \tag{4}$$

The normalized historical classification accuracy rates of field classifiers can also be used to estimate the linear combination coefficients. Within the OAMFL framework, each field classifier's historical SS values can be plotted to a receiver operating characteristic (ROC) curve. The percentage of the area below the ROC curve, abbreviated as ROCA, indicates the historical classifying ability. So each ROCA value is reasonable to estimate the classification accuracy rate of each field classifier. This historical performance weighted strategy was proposed in our previous research [16], abbreviated as *cs2* combining strategy, where the normalized current n values of ROCA were used to set the coefficient α_i before a document was classified. Our research has also proved that the overall performance of *cs2* is better than that of *cs1*.

Furthermore, the information of the current classified document will also affect the classification accuracy at the time of online predicting. The number of characters in each text field can be used as the measure of the information for each field, which formed the current classifying contribution weighted strategy, abbreviated as *cs3* combining strategy. In this strategy, the normalized number of characters in text fields is used to set the coefficient α_i.

In fact, the *cs2* and *cs3* strategies are two sides of the same coin. The two strategies, the historical performance weighted strategy and the current classifying contribution weighted strategy, will affect the classification accuracy together. This paper presents a compound weight considering the *cs2* and *cs3* strategies on the assumption that the two strategies contribute equally to a correct classification. Let α_i^{cs2} and α_i^{cs3} denote separately the coefficient of the *cs2* and *cs3*, then a compound weight, shown in Eq. (5), is used as the coefficient α_i. This compound weight strategy is abbreviated as *cs4*.

$$\alpha_i = \frac{\alpha_i^{cs2} + \alpha_i^{cs3}}{2} \tag{5}$$

The four linear combination strategies are refined step by step from *cs1* to *cs4*, especially the *cs4* strategy considers the two influences thoroughly, and can ensure the high classification accuracy. The combiner can integrate the scores of multiple field classifiers to form the final SS by one of the above four strategies. If the final SS $\in [0, T]$, the document will be predicted as a ham; otherwise, if the final SS $\in (T, 1]$, it will be predicted as a spam. Here T denotes the SS threshold and $T = 0.5$ in this paper.

After the binary classification decision, the active learning process is triggered to evaluate the current classifying confidence and make a decision whether a user feedback for the current email is requested. By contrast, the full feedback filter lacks this active learning process and it unconditionally requests a user feedback immediately after the binary judgment. The user feedback will be sent to each field classifier for its TC model updating.

3.5 Active learning strategy

How to choose more informative training samples is a key point of active learning. The widely used uncertainty sampling [22,23] is an efficient method, which selects those easily incorrectly classified samples for training. The reason is that the more uncertain sample can highly improve the training.

In this paper, the active learner has a preset quota parameter of emails for which feedback can be requested immediately until the quota is exhausted. This quota is much less than the total number of emails. There are several strategies to spend this quota. The simplest strategy is the first coming priority strategy, abbreviated as *as1*, which requests a label for each coming message until the quota is exhausted. The *as1* strategy is not a real active learning strategy, but a baseline compared with other active learning strategies.

We can also set a heuristic uncertain range (0.4, 0.6) of spamminess scores and estimate whether the current SS, output from the combiner, belongs to the uncertain range. This explicit uncertainty sampling method is abbreviated as *as2*.

For each email, we can get several spamminess scores from field classifiers within the OAMFL framework. These spamminess scores, the same inputs of the combiner, can also be used by the active learner. Just because the email documents have multi-field text structures and our OAMFL framework can make use of that structural feature, the several results from different field decision-makers can be combined to measure the difference, which also indicates the uncertainty of the current classification. We can use the variance of several spamminess scores to measure the uncertainty. So, we propose a historical-variance-based active learning strategy, abbreviated as *as3*. The detailed algorithm is shown in Fig. 2.

```
// HVAL: Historical Variance based Active Learning algorithm.
// SS: Field TC Results; D: Historical Average Variance; C: Training Sample Count; Q: Quota.
// b: Sampling Rate; This paper sets (b = 1).
HVAL (ArrayList<Float> SS; Float D; Integer C; Integer Q)
(1) Float V := ComputeVariance (SS);
(2) If (V > b*D) And (Q > 0) Then:
    (2.1) Q := Q-1;
    (2.2) Request a Label L;
    (2.3) D := D*C+V;
    (2.4) C := C+1;
    (2.5) D := D/C;
(3) Output: L and D; C; Q.

ComputeVariance (SS) //For n real numbers SS_i∈SS, compute their variance.
```

Fig. 2 HVAL algorithm

From Fig. 2, we find that the space-time complexity of the HVAL algorithm is very low. The main space cost is space for several real numbers, and the main time cost is several multiplications. The space-time-efficient active learning algorithm will improve our OAMFL framework's performance.

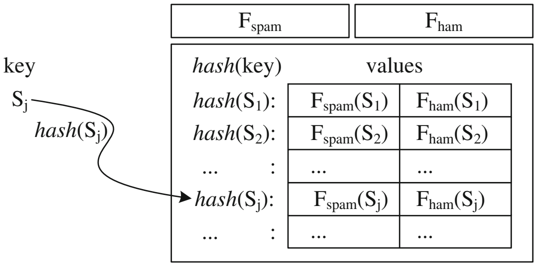
From the perspective of machine learning, the OAMFL framework adds a document-level category label to each field document. Each field classifier can develop more sophisticated features and train a TC model in its own feature space, which reduces the feature disturbance among the fields and makes the TC model more precise. The *as3* active learning strategy makes use of multiple decision-makers to estimate the classifying confidence and tends to regard that a high divergence of opinions indicates more uncertainty. The OAMFL framework is a general structure, easily to be applied to previous TC algorithms, because previous TC algorithms can be used to implement the field classifier by changing a binary result output to a continuous SS output.

The total space-time cost within the OAMFL framework depends on the space-time complexity of each field classifier. But previous TC algorithms often face the high-dimensional sparseness problem and the online open problem of the text feature space, and they depend on some complex time-consuming operations to achieve high classification accuracy for some advanced individual TC algorithms. The problems make previous TC algorithms unsuitable to be implemented as the field classifier for efficient email spam filtering. So this paper next explores a lightweight space-time-efficient TC algorithm to implement the field classifiers.

4 Field text classification

In this section, we address the efficient online binary field TC problem and propose a general lightweight field TC algorithm (named as SFITC), which is independent of any concrete field. Based on our proposed data structure of string-frequency index (SFI), the SFITC algorithm converts the online training and classifying processes into index incremental updating and retrieving processes, and smoothly solves the online open problem of the text feature space. The SFITC algorithm is suitable to be implemented as the field classifiers owing to the space-time-efficient SFI data structure.

Fig. 3 String-frequency index



4.1 String-frequency index

The feature string frequency of historical labeled field texts gives rich classification information [24] and must be stored efficiently for online training. This paper applies the overlapping word-level 4-gram model to define feature strings and lets a field text *FT* be represented as a sequence of feature strings in the form $FT = S_j, (j = 1, 2, \ldots, N)$. The string-frequency index is a data structure to store the feature string information of labeled field texts, from which we can conveniently calculate spamminess score of each feature string according to the scaled Bayesian conditional probability $P(spam|S_j)$ and straightforwardly combine the scores to form the field's final score.

Figure 3 shows the SFI structure for a field classifier including two integers and a hash table. The integers F_{spam} and F_{ham} denote separately the total number of historical labeled spam and ham field texts, which are then scaled up in order to make $P(spam) = P(ham)$. Each table entry is a key-value pair <Key, Value>, where each key is a feature string and each value consists of two integers. The integers $F_{spam}(S_j)$ and $F_{ham}(S_j)$ denote separately the number of occurrences of feature string S_j in historical labeled spam and ham field texts, and the S_j denotes the jth feature string in the field text. The hash function maps the feature string S_j to the address of two integers $F_{spam}(S_j)$ and $F_{ham}(S_j)$. In this paper, we call the default hash function of the java class (*java.util.HashMap*).

4.2 SFITC Algorithm

Supported by the SFI, the SFITC algorithm takes the online classifying process of a field text as an index retrieving process and also takes the online training process as an incremental updating process of the index. Figure 4 gives the pseudo-code for the SFITC algorithm consisting of two main procedures: PREDICT and UPDATE.

When a new (Label = null) field text arrives, the PREDICT procedure is triggered: (1) It extracts the feature string sequence from the field text based on the overlapping word-level 4-gram model; (2) It retrieves the current SFI and calculates each feature string's SS according to the scaled Bayesian conditional probability described in Eq. (6); and (3) It assumes that each feature string's contribution to the final SS is equivalent and uses the arithmetical average to calculate the final SS described in Eq. (7).

$$\mathrm{SS}_j = P(spam|S_j) = \frac{F_{spam}(S_j)/F_{spam}}{F_{spam}(S_j)/F_{spam} + F_{ham}(S_j)/F_{ham}} \tag{6}$$

$$\mathrm{SS}_i = \frac{1}{N}\sum_{j=1}^{N} \mathrm{SS}_j. \tag{7}$$

```
// SFITC: String-Frequency Index Text Classification algorithm.
// FT: Field Text; L: Binary Category Label; SFI: String-Frequency Index.
SFITC (FT; L; SFI)
(1) If (L = null) Then: PREDICT (FT; SFI);
(2) Else: UPDATE (FT; L; SFI).

// PREDICT: Online classifying procedure.
PREDICT (FT; SFI)
(1) String[] $S$ := FEATURE(FT);
(2) Integer $I_s$ := SFI.$F_{spam}$;
(3) Integer $I_h$ := SFI.$F_{ham}$;
(4) New ArrayList<Float> $F$;
(5) If ($I_s = 0$) Or ($I_h = 0$) Then: Float $SS_i$ := 0.5;
(6) Else:
   (6.1) Loop: For Each $S_j \in S$ Do:
      (6.1.1) If (SFI.containKey($S_j$)) Then:
         (6.1.1.1) Integer $I_{sj}$ := SFI.$F_{spam}(S_j)$;
         (6.1.1.2) Integer $I_{hj}$ := SFI.$F_{ham}(S_j)$;
         (6.1.1.3) Float $SS_j := (I_{sj}/I_s)/(I_{sj}/I_s+I_{hj}/I_h)$;
         (6.1.1.4) $F$.add($SS_j$);
   (6.2) Integer $N$ := $F$.length;
   (6.3) If ($N = 0$) Then: Float $SS_i$ := 0.5;
   (6.4) Else: Float $SS_i := (1/N)\sum SS_j$; // $SS_j \in F$
(7) If ($SS_i > 0.5$) Then: Label L := spam;
(8) Else: Label L := ham;
(9) Output: $SS_i$ and L.

// UPDATE: Online training procedure.
UPDATE (FT; L; SFI)
(1) String[] $S$ := FEATURE(FT);
(2) If (L = spam) Then:
   (2.1) SFI.$F_{spam}$ := SFI.$F_{spam}$ + 1;
   (2.2) Loop: For Each $S_j \in S$ Do:
      (2.2.1) If SFI.containKey($S_j$) Then: SFI.$F_{spam}(S_j)$ := SFI.$F_{spam}(S_j)$ + 1;
      (2.2.2) Else: SFI.putKey($S_j$), And SFI.$F_{spam}(S_j)$ := 1, SFI.$F_{ham}(S_j)$ := 0;
(3) Else If (L = ham) Then:
   (3.1) SFI.$F_{ham}$ := SFI.$F_{ham}$ + 1;
   (3.2) Loop: For Each $S_j \in S$ Do:
      (3.2.1) If (SFI.containKey($S_j$)) Then: SFI.$F_{ham}(S_j)$ := SFI.$F_{ham}(S_j)$ + 1;
      (3.2.2) Else: SFI.putKey($S_j$), And SFI.$F_{spam}(S_j)$ := 0, SFI.$F_{ham}(S_j)$ := 1.

FEATURE (FT) //Extract the feature string sequence from FT based on word-level 4-gram model.
```

Fig. 4 Pseudo-code for the SFITC algorithm

When a new labeled field text arrives, it is only required that the field text's feature strings are put into the SFI. The UPDATE procedure extracts the feature string sequence and updates the frequency or adds a new index entry to the SFI according to the feature strings within the sequence.

4.3 Space-time complexity

Based on the SFI, the SFITC algorithm, an online Bayesian algorithm, overcomes some disadvantages caused by traditional VSM and smoothly solves the online open problem of the text feature space. Some time-consuming operations, such as vector alignment and feature selection, are avoided in the SFITC algorithm. The SFITC algorithm, independent of any concrete field, is a general robust field TC algorithm, whose space-time complexity depends on the SFI storage space and the loops in the PREDICT and the UPDATE procedures.

The SFI storage space is efficient owing to the inherent compressible property of index files. The SFI is an improved version of traditional inverted files [25], which simplifies the position and document ID information to two integers, only reflecting the occurrence frequency of feature strings. The hash list structure, prevailingly employed in Information Retrieval, has a lower compression ratio of raw texts. Though the training field texts will mount in the wake of the increasing of online feedbacks, the SFI storage space will increase slowly. Theoretically, the inherent compressible property of index files ensures that the SFI storage space is proportional to the total number of feature strings and is independent of the number of training documents.

The incremental updating or retrieving of SFI has constant time complexity according to a hash function. The major time cost of the online classifying procedure is the time for $3N+1$ divisions in the loop (see 6.1 of Fig. 4). The online training procedure is lazy, requiring no retraining when a new labeled field text is added. From Fig. 4, it is found that the time cost of per updating is only proportional to the total number of feature strings in the field text. Except the loop (see 2.2 and 3.2 of Fig. 4) according to the number of feature strings, there is no time-consuming operations. Above time complexity is acceptable in the practical email spam filtering application.

We implement each field classifier according to the SFITC algorithm and integrate these field classifiers into the OAMFL framework to form an OAMFL binary streaming TC approach. The maximal space complexity O(nm) and the maximal time complexity O(nM) are both low. Here, n means the number of fields, usually a low number; m means the number of feature strings in a field; and M means the total number of messages.

5 Experiments

In this section, we validate the effectiveness of the online active multi-field learning approach described in Sect. 3 for online email spam filtering. In our experiments, we implement each field classifier according to the field TC algorithm described in Sect. 4. The experimental results show strong support for the use of the online active multi-field learning in the fast email spam filtering.

5.1 Data and evaluation

We use a large-scale, publicly available benchmark data set from the TREC spam filtering competition trec07p [12], which contains 75,419 total email messages (25,220 hams and 50,199 spams).

The TREC spam filter evaluation toolkit and the associated evaluation methodology are applied. The evaluation toolkit is run under the constraint of the TREC default setups [6], which prohibit filters from using network resources, and constrained temporary disk storage (1 GB), RAM (1 GB), and runtime (2 s/message, amortized).

Table 2 Combining strategies and learning strategies (feedbacks) of spam filters

	Combining strategies	Learning strategies (feedbacks)
ndtF1	cs1	Full
ndtF2	cs2	Full
ndtF3	cs3	Full
ndtF4	cs4	Full
ndtF5	cs4	as1 (Quota = 10,000)
ndtF6	cs4	as2 (Quota = 10,000)
ndtF7	cs4	as3 (Quota = 10,000)
ndtF8	cs4	as1 (Quota = 1,000)
ndtF9	cs4	as2 (Quota = 1,000)
ndtF10	cs4	as3 (Quota = 1,000)

Here the identifier prefix (ndt) of our filters is shortened from the name of our university

We report the overall performance measurement (1-ROCA)%, the area above the ROC curve percentage where 0 is optimal, and the total running time to evaluate the filter's performance. We also report two measurements: the ham misclassification percentage (hm%) is the fraction of all ham classified as spam; and the spam misclassification percentage (sm%) is the fraction of all spam classified as ham.

All above measurements are automatically computed by the TREC spam filter evaluation toolkit. The toolkit can also plot a ROC curve and a ROC learning curve for the ROC analysis. The ROC curve is the graphical representation of spam misclassification percentage and ham misclassification percentage. The ROC learning curve shows the cumulative (1-ROCA)% as a function of the number of messages processed.

5.2 Implementations

We implement 10 spam filters in total. Each filter has a 7-field splitter (five natural fields and two artificial fields) as described in Sect. 3.3. All field classifiers apply the SFITC algorithm described in Sect. 4, and the feature strings are based on overlapping word-level 4-gram model. There are a total of four combining strategies described in Sect. 3.4 and three active learning strategies described in Sect. 3.5 in the spam filters. For each active learning strategy, we separately run with quota values (10,000 and 1,000). The detailed configuration of the spam filters is shown in Table 2.

Three full feedback filters are chosen as baselines: (1) the bogo [26,27] filter (bogo-0.93.4) is a classical implementation of the VSM-based online Bayesian algorithm; (2) the tftS3F [28] filter is based on the relaxed online SVMs algorithm and has gained several best results at the TREC2007 spam track; and (3) the wat3 [29] filter, the winner in the trec07p full feedback task, is based on the online fusion of DMC and logistic regression, and whose overall performance (1-ROCA)% is the best one (0.0055). The two filters can be run in the same environment with our filters, and we can compare their running time to evaluate time complexity.

Three active learning filters (tftS2F [28], wat4 [29], crm1 [30]) are chosen as baselines compared with our active learning filters. The tftS2F filter uses the fixed-margin sampling method, which is the winner in the active learning task of trec07p data set. The wat4 filter is the implementation of the logistic regression algorithm with a threshold-based active learning, which has gained the second rank in the TREC 2007 active learning task. The crm1 filter is based on the SVM algorithm with the single-sided thick threshold training, which got the third rank in the TREC 2007 active learning task.

Table 3 Results of experiment I

	Time (s)	(1-ROCA)%
bogo	25,100	0.1558
bogo.cs2	111,709	0.0103
tftS3F	62,554	0.0093
tftS3F.cs2	446,765	0.0083

The hardware environment for running experiments is a PC with RAM (1 GB) and CPU (2.80 GHz Pentium D). The operating system is Red Hat Linux (linux-2.4.20-8smp).

5.3 Results and discussion

We verify the effectiveness of the online active multi-field learning for email spam filtering through four experiments. The experiment I is to prove the OAMFL framework's effect on improving the performance of previous TC algorithms in the full feedback task. The experiment II is also the full feedback experiment, in which we try to compare the performance of the four combining strategies. We also evaluate the performance between our filter with the best combining strategy and other three full feedback filters (bogo, tftS3F, wat3). The experiment III is the active learning with the 10,000 quota, in which we try to compare the performance of the three active learning strategies. The experiment IV is similar to the experiment III except with the only 1,000 quota. Under the few quota, we also evaluate the performance between our active learning filter and the top three filters (tftS2F, wat4, crm1) in the TREC 2007 active learning task.

In the experiment I, we choose two typical online TC algorithms separately to be integrated within the OAMFL framework. We use bogo which applies VSM-based online Bayesian algorithm to implement each field classifier and combine the 7 field classifying results with *cs2* combining strategy form a bogo.cs2 filter. In the same way, we use tftS3F, which applies the relaxed online SVMs algorithm to implement each field classifier, and also combine the results with *cs2* to form a tftS3F.cs2 filter. The four filters bogo, bogo.cs2, tftS3F, and tftS3F.cs2 separately run in the full feedback task. The detailed experimental results are shown in Table 3. The experimental results show that the bogo filter's (1-ROCA)% is optimized from original 0.1558 to bogo.cs2's 0.0103, and the tftS3F filter's (1-ROCA)% is also optimized from original 0.0093 to tftS3F.cs2's 0.0083. The (1-ROCA)% performance of both the online Bayesian and relaxed online SVMs algorithms all can be improved within the OAMFL framework, which demonstrates the advantage of the OAMFL framework. The improvement of OAMFL framework can be explained in two main reasons: (1) it can reduce the disturbances among text features from different fields and (2) also has statistical, computational, and representational advantages [31].

Figure 5 shows the ROC curve and the ROC learning curve of the experiment I, which also indicates above results. We also evaluated the *cs1*, *cs3*, and *cs4* combining strategies, and the results are similar to the *cs2* combining strategy. From Table 3, we also find that the time spending increases both greatly, which indicates that the time-consuming operations make some advanced individual TC algorithms unsuitable to be implemented as the field classifier within the OAMFL framework.

In the experiment II, the bogo, tftS3F, and our four filters run in the full feedback all have been evaluated on the trec07p corpus separately, whose detailed experimental results are shown in Table 4. The results show that the ndtF4 filter can complete filtering task in high speed (2,834 s), whose overall performance (1-ROCA)% is comparable to the best wat3

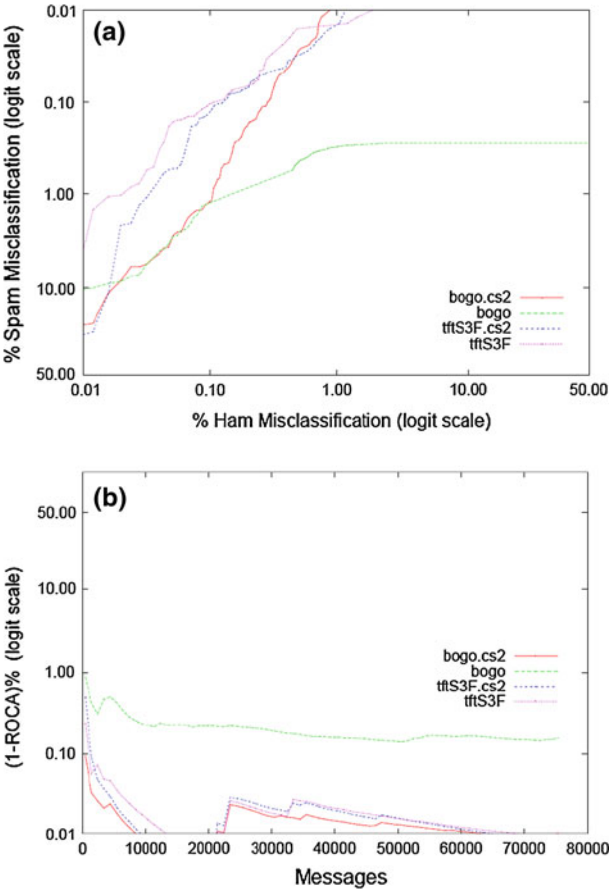


Fig. 5 Experiment I: **a** ROC curves and **b** ROC learning curves in the immediate full feedback task

Table 4 Results of experiment II

	Time (s)	(1-ROCA)%	sm%	hm%	TREC 2007 full feedback rank
ndtF4	2,834	0.0055	0.21	0.11	
wat3		0.0055			1
ndtF2	2,776	0.0067	0.16	0.15	
ndtF3	1,910	0.0070	0.40	0.08	
ndtF1	1,863	0.0074			
tftS3F	62,554	0.0093			2
bogo	25,100	0.1558			

filter's (0.0055) among the participators at the trec07p evaluation. The time and (1-ROCA)% performance of our four filters overcome the bogo's and the tftS3F's obviously. Comparing the ndtF2 and the ndtF3 in the percent of misclassified spams and hams, we find that the *cs2* strategy optimizes spam's decision ($0.16 < 0.40$) and the *cs3* strategy optimizes ham's decision ($0.08 < 0.15$). The sm% and hm% of ndtF4 indicate that the compound weight

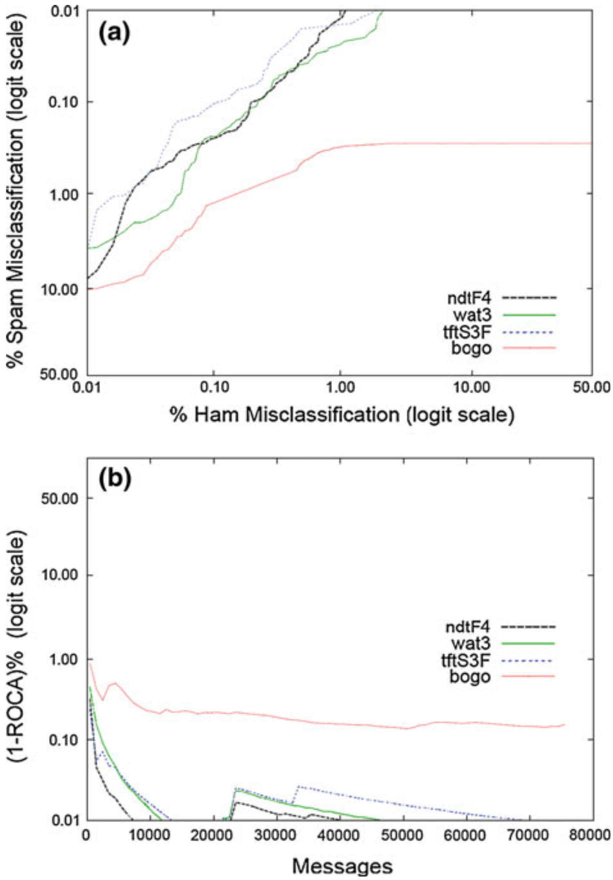


Fig. 6 Experiment II: **a** ROC curves and **b** ROC learning curves in the immediate full feedback task

indeed covers the both two aspects: the historical classifying ability of each field classifier and the classifying contribution of each text field in the current classified email.

Figure 6 shows the ROC curve and the ROC learning curve of the bogo, tftS3F, wat3, and our best ndtF4 filter, respectively. In the ROC curve, the area surrounded by the left border, the top border, and the ndtF4 curve is relatively small, which means that the overall filtering performance of the ndtF4 filter is promising. The ROC curve also shows that the overall performance is comparable among the tftS3F, wat3, and ndtF4 filters. In the ROC learning curve, around 7,000 training samples, the ndtF4 curve achieves the ideal (1-ROCA)% performance (0.01). Comparing the learning curves of ndtF4, tftS3F, and wat3, we find that the performances are all dropping near 20,000 training samples. However, when close to 40,000 training samples, the ndtF4 can quickly return the ideal steady-state and the average overall performance (1-ROCA)% can reach 0.0055. This indicates that the SFITC algorithm applying *cs4* strategy of the OAMFL framework has a strong online learning ability.

From the experiment II, we find that the *cs4* strategy is efficient to the email spam filtering. Furthermore, we will evaluate the active learning strategies through the experiment III. In the experiment III, the ndtF5, ndtF6, and ndtF7 filters run in the active learning task (Quota = 10,000) on the trec07p corpus separately. The detailed experimental results are

Table 5 Results of experiment III

	Time (s)	(1-ROCA)%
ndtF5	1,422	0.0465
ndtF6	1,560	0.0200
ndtF7	1,976	0.0071

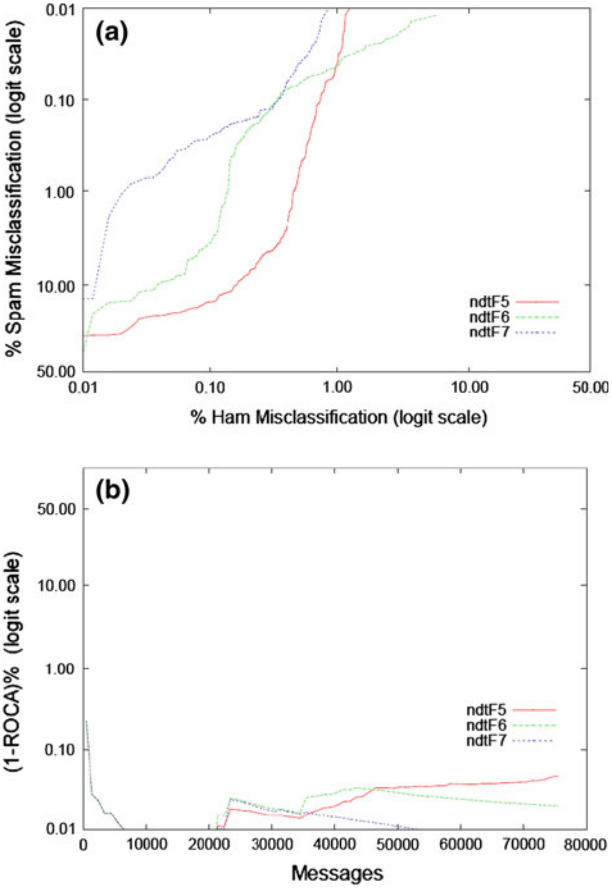


Fig. 7 Experiment III: **a** ROC curves and **b** ROC learning curves in the active learning task with the 10,000 quota

shown in Table 5. The overall performance (1-ROCA)% of the *as3* active learning strategy (0.0071) exceeds that of the *as1* strategy (0.0465) and the *as2* strategy (0.0200), which indicates that the *as3* active learning strategy can indeed choose more informative samples.

Moreover, even the time (1,976 s) and the overall performance (0.0071) of the ndtF7 filter both outgo the full feedback filtering time (62,554 s) and the overall performance (0.0093) of the tftS3F filter, which is the second rank in the TREC 2007 full feedback task. This indicates that our proposed approach can achieve the state-of-the-art performance at greatly reduced label requirements in email spam filtering. Figure 7 shows the ROC curve and the ROC learning curve of the ndtF5, ndtF6, and ndtF7 filters, respectively, which also indicates above results.

Table 6 Results of experiment IV

	Time (s)	(1-ROCA)%	TREC 2007 active learning rank
tftS2F		0.0144	1
wat4		0.0145	2
ndtF10	1,518	0.0380	
crm1		0.0401	3
ndtF8	1,392	0.0530	
ndtF9	1,486	0.0997	

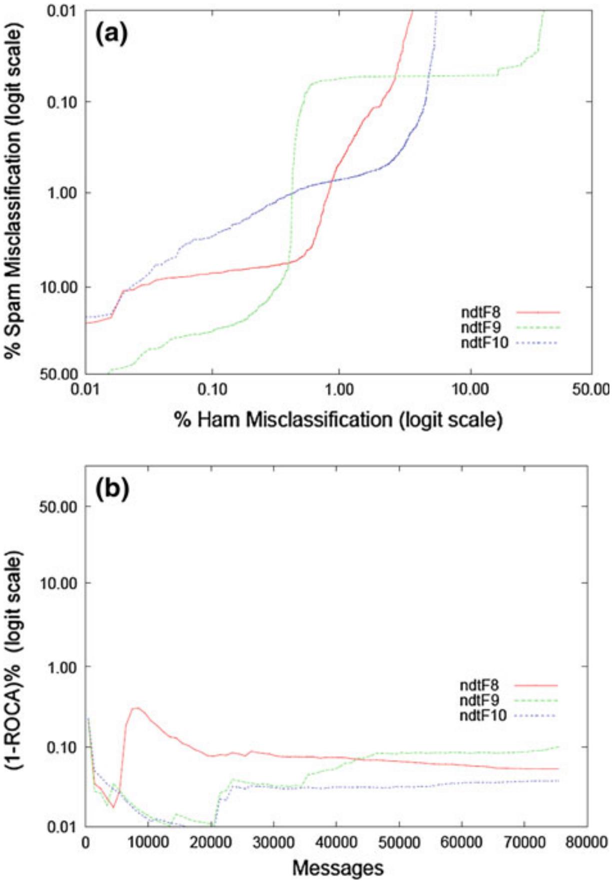


Fig. 8 Experiment IV: **a** ROC curves and **b** ROC learning curves in the active learning task with the 1,000 quota

In the experiment IV, we run the ndtF8, ndtF9, and ndtF10 filters in the active learning task (Quota = 1,000) on the trec07p corpus separately, whose results are listed in Table 6. The overall performance (1-ROCA)% of the ndtF10 filter (0.0380) overcomes that of the crm1 filter (0.0401) who got the third rank in the TREC 2007 active learning task. This indicates that our proposed approach is robust even in very small feedbacks.

Figure 8 shows the ROC curve and the ROC learning curve of the ndtF8, ndtF9, ndtF10 filter respectively. Figure 8 indicates that the ndtF10 even precedes the ndtF8, ndtF9 in very small feedbacks.

Above four experiments indicate: (1) The OAMFL framework can improve the (1-ROCA)% performance of previous TC algorithms. (2) The explicit splitting strategy and the artificial splitting strategy are both effective. (3) The compound-weight-based combining strategy is the best one among the four combining strategies. (4) The historical-variance-based active learning strategy is the best one among the three active learning strategies. (5) The SFITC algorithm is a space-time-efficient field TC algorithm. (6) Our filters all run fast, which are suitable for the email spam filtering. (7) The OAMFL approach prefers to achieve the state-of-the-art performance of email spam filtering at only about 10,000 label requirements.

6 Conclusion

Through the investigation of the multi-field text structure, this paper proposed a novel approach for the efficient online spam filtering. The experiments have proved that the proposed online active multi-field learning approach can solve the email spam problem well.

The contributions mainly include three parts: (1) Based on our proposed SFI data structure, we design and implement a novel online Bayesian TC algorithm (SFITC), which is space-time-efficient and can satisfy the requirements of large-scale online filtering applications. (2) We use the multi-field text structure to break a complex problem into multiple simple sub-problems. According to a novel compound weight, the ensemble result of sub-problems can achieve promising performance. (3) The difference among the results of field classifiers motivates a novel uncertainty sampling method, and our historical-variance-based active learning algorithm can choose informative samples and greatly reduce user feedbacks.

Moreover, the OAMFL framework is suitable to a parallel running environment. If the OAMFL framework is applied on the reduplicate hardware for multiple field classifiers, then its theoretical computational time to classify a message will nearly be equal to the slowest field classifier's running time.

Based on the above researches, we can draw following conclusions:

- The index data structure has the inherent compressible property of raw texts, by which the Information Retrieval approach can be used to solve the Information Classification problem. Each incremental updating or retrieving of an index has constant time complexity, which can satisfy the space-limited and real-time requirements of online applications.
- The multi-field text structure can support the divide-and-conquer strategy. Using an optimal linear combination strategy of the compound weight, the straightforward occurrence counting of string features can obtain promising classification performance, even better than that of some advanced individual algorithms. The straightforward counting will also bring time reducing.
- The uncertainty sampling is an effective active learning method. Within the OAMFL framework, the multiple field classifiers bring an opportunity to estimate the uncertainty by the variance of multi-result. The historical-variance-based active learning algorithm is space-time-efficient, and according to which, the active learner regards the more uncertain message as the more informative sample.

In recent years, the amount of spam messages has been dramatically increased on the network. The spam concept has already generalized from email spam to various messages spam, such as short message service (SMS) spam, instant messaging (IM) spam, microblogging spam, etc. Our proposed general, space-time-efficient approach can be easily transferred to other spam filtering in a ubiquitous environment.

Further research will concern personal learning for online spam filtering. We will improve the SFI structure for both global and personal labeled text storage.

Acknowledgments The authors thank the editor-in-chief, the handling editor, and anonymous reviewers for helping to greatly improve the paper. This research is supported by the National Natural Science Foundation of China (No.60873097, No.60933005), the Program for New Century Excellent Talents in University (No.NCET-06-0926), and the Fund of Innovation of NUDT (No.B080605). We thank Dr. D. Sculley for his tftS3F filter code.

References

1. Sebastiani F (2002) Machine learning in automated text categorization. ACM Comput Surv 34(1):1–47
2. Kuncheva LI, Sánchez JS (2008) Nearest neighbour classifiers for streaming data with delayed labeling. In: ICDM 2008 Proceedings of the 8th IEEE international conference on data mining, pp 869–874
3. Wozniak M (2010) A hybrid decision tree training method using data streams. Knowl Inf Syst, Online First^{TM}, 05 Oct 2010
4. Chang M, Yih W, Meek C (2008) Partitioned logistic regression for spam filtering. In: SIGKDD 2008 Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining, pp 97–105
5. Lee C-H (2010) Learning to combine discriminative classifiers: confidence based. In: SIGKDD 2010 Proceedings of the 16th ACM SIGKDD international conference on knowledge discovery and data mining, pp 743–752
6. Cormack GV, Lynam T (2005) TREC 2005 spam track overview. In: TREC2005 Proceedings of the 14th text retrieval conference, National Institute of Standards and Technology, Special Publication 500–266
7. Tong S, Koller D (2002) Support vector machine active learning with applications to text classification. J Mach Learn Res 2:45–66
8. Cesa-Bianchi N, Gentile C, Zaniboni L (2006) Worst-case analysis of selective sampling for linear classification. J Mach Learn Res 7:1205–1230
9. Chai KMA, Chieu HL, Tou H (2002) Bayesian online classifiers for text classification and filtering. In: SIGIR'02 Proceedings of the 25th annual international ACM SIGIR conference on research and development in information retrieval, pp 97–104
10. Sculley D, Wachman GM (2007) Relaxed online SVMs for spam filtering. In: SIGIR'07 Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval, pp 415–422
11. Cormack GV (2007) University of waterloo participation in the TREC 2007 spam track. In TREC2007: Notebook of the 16th text retrieval conference, National Institute of Standards and Technology
12. Cormack GV (2007) TREC 2007 spam track overview. In: TREC2007 Proceedings of the 16th text retrieval conference, National Institute of Standards and Technology, Special Publication 500–274
13. Verikas A, Guzaitis J, Gelzinis A, Bacauskiene M (2010) A general framework for designing a fuzzy rule-based classifier. Knowl Inf Syst, Online First^{TM}, 16 Sept 2010
14. Yoo S, Yang Y, Lin F, Moon I-C (2009) Mining social networks for personalized email prioritization. In: SIGKDD 2009 Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining, pp 967–976
15. Katakis I, Tsoumakas G, Vlahavas I (2010) Tracking recurring contexts using ensemble classifiers: an application to email filtering. Knowl Inf Syst 22(3):371–391
16. Liu W, Wang T (2010) Multi-field learning for email spam filtering. In: SIGIR'10 Proceedings of the 33rd annual international ACM SIGIR conference on research and development in information retrieval, pp 745–746
17. Cormack GV (2006) TREC 2006 spam track overview. In: TREC2006 Proceedings of the 15th text retrieval conference, National Institute of Standards and Technology, Special Publication 500–272
18. Sculley D (2007) Online active learning methods for fast label-efficient spam filtering. In: CEAS2007 Proceedings of the 4th conference on email and anti-spam
19. Goodman J, Yih W (2006) Online discriminative spam filter training. In: CEAS2006 Proceedings of the 3rd conference on email and anti-spam
20. Drucker H, Wu D, Vapnik VN (1999) Support vector machines for spam categorization. IEEE Trans Neural Netw 10(5):1048–1054

21. Sanchez F, Duan Z, Dong Y (2010) Understanding forgery properties of spam delivery paths. In: CEAS2010 Proceedings of the 7th annual collaboration, electronic messaging, anti-abuse and spam conference. http://ceas.cc/2010/papers/Paper%2012.pdf
22. Lewis DD, Gale WA (1994) A sequential algorithm for training text classifiers. In: SIGIR'94 Proceedings of the 17th annual international ACM SIGIR conference on research and development in information retrieval, pp 3–12
23. Lewis DD, Catlett J (1994) Heterogeneous uncertainty sampling for supervised learning. In: Proceedings of the 11th International Conference on Machine Learning, pp 48–156
24. Malik HH, Fradkin D, Moerchen F (2010) Single pass text classification by direct feature weighting. Knowl Inf Syst, Online First$^{\text{TM}}$, 25 June 2010
25. Zobel J, Moffat A (2006) Inverted files for text search engines. ACM Comput Surv 38(2):Article 6
26. Graham P (2002) A plan for spam. http://www.paulgraham.com/spam.html
27. Graham P (2003) Better bayesian filtering. http://www.paulgraham.com/better.html, In the 2003 Spam Conference
28. Sculley D, Wachman GM (2007) Relaxed online SVMs in the TREC spam filtering track. In: TREC2007 Proceedings of the 16th text retrieval conference, National Institute of Standards and Technology, Special Publication 500–274
29. Cormack GV (2008) Email spam filtering: a systematic review. Found Trends Inf Retr 1(4):335–455
30. Kato M, Langeway J, Wu Y, Yerazunis WS (2007) Three non-bayesian methods of spam filtration: CRM114 at TREC 2007. In: TREC2007 Proceedings of the 16th text retrieval conference, National Institute of Standards and Technology, Special Publication 500–274
31. Dietterich TG (2000) Ensemble methods in machine learning. In: MCS2000 Proceedings of the multiple classifier systems, pp 1–15

Author Biographies

Wuying Liu was born in 1980. He received his B.S. degree and M.S. degree in Computer Science and Technology from National University of Defense Technology in 2002 and 2005, respectively. He is currently a Ph.D. candidate at College of Computer, National University of Defense Technology. His research interests include Text Classification, Information Filtering, and Machine Learning. He has published more than 15 research papers in proceedings of international conferences and journals.

Ting Wang was born in 1970. He received his B.S. degree and Ph.D. degree in Computer Science and Technology from National University of Defense Technology in 1992 and 1997, respectively. He is currently a professor and doctoral supervisor at National University of Defense Technology. His research interests include Natural Language Processing and Computer Software. He has published over 50 scientific papers and undertaken two projects of National Nature Science Foundation of China and two projects of National High Technology Research and Development Program of China as principal investigator.