Expert Systems With Applications 113 (2018) 531–543

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Feature ranking for enhancing boosting-based multi-label text categorization

Bassam Al-Salemi*, Masri Ayob, Shahrul Azman Mohd Noah

Center for Artificial Intelligence Technology, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, Malaysia

ARTICLE INFO

Article history:
Received 24 January 2018
Revised 9 July 2018
Accepted 10 July 2018
Available online 21 July 2018

Keywords:
RFBoost
Boosting
Multi-label learning
Text categorization
Feature ranking

ABSTRACT

Boosting algorithms have been proved effective for multi-label learning. As ensemble learning algorithms, boosting algorithms build classifiers by composing a set of weak hypotheses. The high computational cost of boosting algorithms in learning from large volumes of data such as text categorization datasets is a real challenge. Most boosting algorithms, such as AdaBoost.MH, iteratively examine all training features to generate the weak hypotheses, which increases the learning time. RFBoost was introduced to manage this problem based on a rank-and-filter strategy in which it first ranks the training features and then, in each learning iteration, filters and uses only a subset of the highest-ranked features to construct the weak hypotheses. This step ensures accelerated learning time for RFBoost compared to AdaBoost.MH, as the weak hypotheses produced in each iteration are reduced to a very small number. As feature ranking is the core idea of RFBoost, this paper presents and investigates seven feature ranking methods (information gain, chi-square, GSS-coefficient, mutual information, odds ratio, F1 score, and accuracy) in order to improve RFBoost's performance. Moreover, an accelerated version of RFBoost, called RFBoost1, is also introduced. Rather than filtering a subset of the highest-ranked features, FBoost1 selects only one feature, based on its weight, to build a new weak hypothesis. Experimental results on four benchmark datasets for multi-label text categorization) Reuters-21578, 20-Newsgroups, OHSUMED, and TMC2007(demonstrate that among the methods evaluated for feature ranking, mutual information yields the best performance for RFBoost. In addition, the results prove that RFBoost statistically outperforms both RFBoost1 and AdaBoost.MH on all datasets. Finally, RFBoost1 proved more efficient than AdaBoost.MH, making it a better alternative for addressing classification problems in real-life applications and expert systems.

© 2018 Elsevier Ltd. All rights reserved.

1. Introduction

According to the International Data Corporation (Gantz & Reinsel, 2012), digital data on the Internet will grow to 40,000 exabytes by 2020, from 130 exabytes in 2005. These huge amounts of data are usually distributed over the World Wide Web in unstructured forms. Managing and organizing these data requires efficient and effective automatic text categorization systems. For this reason, text categorization is still an important research area that receives much attention in the research community and industry.

Text categorization involves automatically assigning texts to the appropriate categories (labels) from a set of predefined categories (Elghazel, Aussem, Gharroudi, & Saadaoui, 2016; Sebastiani, 2002). Many classification algorithms have been investigated for text categorization, such as naïve Bayes, *k* nearest neighbours (*k*NN), support vector machines (SVMs), and decision trees (Farid, Zhang, Rahman, Hossain, & Strachan, 2014; Jiang, Li, Wang, & Zhang, 2016; Onan, Korukoğlu, & Bulut, 2016; Pavlinek & Podgorelec, 2017; Trstenjak, Mikac, & Donko, 2014; Zhang, Liu, Zhang, & Almpanidis, 2017). However, these algorithms are restricted to single-label classification problems, in which each instance (each text, in our case) is assigned to only one class label. Yet by their very nature, texts may belong to more than one class (multi-label classification problem). For example, a news article about "education" may also relate to "economy" and/or "politics". Several multi-label classification algorithms have been proposed which extend the single-label classification algorithms to solve the multi-label problem, such as binary relevance (Boutell, Luo, Shen, & Brown, 2004), classifier chains (Read, Pfahringer, Holmes, & Frank, 2011), label powerset (Tsoumakas & Vlahavas, 2007), ranking by pairwise comparison (Hüllermeier, Fürnkranz, Cheng, & Brinker, 2008), calibrated ranking by pairwise comparison (Fürnkranz, Hüllermeier, Mencía, & Brinker, 2008), hierarchical embedding (Kumar, Pujari, Padmanabhan, Sahu, & Kagita, 2018), clustered intrinsic label correlations (Kumar et al., 2018) and label correlation exploitation algorithms (Yu, Pedrycz, & Miao, 2014).

* Corresponding author.
E-mail addresses: bassalemi@ukm.edu.my (B. Al-Salemi), masri@ukm.edu.my (M. Ayob), shahrul@ukm.edu.my (S.A.M. Noah).

https://doi.org/10.1016/j.eswa.2018.07.024
0957-4174/© 2018 Elsevier Ltd. All rights reserved.

AdaBoost.MH (Freund & Schapire, 1997), the multi-label version of AdaBoost (Schapire, Freund, Bartlett, & Lee, 1998), is accurate and considered to be one of the state-of-the-art multi-label classification algorithms. As a boosting algorithm, AdaBoost.MH iteratively builds a set of weak hypotheses and then combines them as a final classifier which is capable of estimating the multiple labels for a given instance. AdaBoost.MH uses binary features to generate the weak hypotheses of decision stumps. To build a weak hypothesis during a specific boosting round, AdaBoost.MH generates a set of weak hypotheses, equal in number to the training features. The weak hypothesis that minimizes the Hamming loss training error is then selected, and all other hypotheses are eliminated.

AdaBoost.MH's iterative examination of all the training features in its weak learning is time-consuming, particularly when the dataset is large (Esuli, Fagni, & Sebastiani, 2006). To address this limitation, Al-Salemi, Noah, and Ab Aziz (2016) introduced an improved version of AdaBoost.MH, named "RFBoost". RFBoost learns by first ranking the training features and then, during each boosting round, filtering and using a small subset of the top-ranked features to produce a new weak hypothesis. Experimental results show that RFBoost is a fast and accurate algorithm for multi-label text categorization. RFBoost's enhanced performance relative to AdaBoost.MH is due to its ranking of the training features: while AdaBoost.MH uses binary features to build its weak hypotheses, RFBoost uses weighted features. However, Al-Salemi, Ab Aziz, and Noah (2016) only investigated two feature ranking methods for RFBoost. One of these uses the conditional probability of the words across the labels obtained by labelled latent Dirichlet allocation (LLDA; Mcauliffe & Blei, 2007) as the features' weights. The other ranking method uses boosting weights obtained by executing *one* boosting round on the training set. Even though LLDA is an effective method for feature ranking, as a topic model it requires resampling the topics estimation, which may result in increased computation time for large volumes of data.

The aim of the present paper is twofold: to investigate several existing feature weighting methods, namely, information gain, chi-square, GSS-coefficient, mutual information, odds ratio, F1 score, and accuracy (Forman, 2003; Katrutsa & Strijov, 2017; Liu, Lin, Lin, Wu, & Zhang, 2017; Lu et al., 2017; Pascoal, Oliveira, Pacheco, & Valadas, 2017; Qian & Shu, 2015; Song, Jiang, & Liu, 2017), in order to improve RFBoost, and to propose an accelerated variant of RFBoost, named "RFBoost1". Feature weighting allows ranking features based on their weights. The proposed RFBoost1 selects only a single ranked feature, based on its weight, to pass to the base learner for generating a new hypothesis, which eliminates the need to examine all of the training features, as in AdaBoost.MH, or even a subset of the ranked features, as in RFBoost. An empirical analysis was also conducted to validate that RFBoost1 does not penalize the boosting theory; this is described in Section 4.3.

2. Preliminaries and problem statement

Given a training set of labelled documents $\boldsymbol{S} = \{(\mathrm{x}_1, Y_1), \ldots, (\mathrm{x}_n, Y_n)\}$, where each document $\mathrm{x}_i \in \chi$ is assigned to a multiple category (label) Y_i: $Y_i \subseteq \boldsymbol{Y}$, $\boldsymbol{Y} = (\boldsymbol{y_1}, \ldots, \boldsymbol{y_m})$, let $\boldsymbol{T} = \{\boldsymbol{t_1}, \ldots, \boldsymbol{t_v}\}$ be the set of training terms extracted from $\boldsymbol{S}$. AdaBoost.MH infers a strong classifier (the final classifier) $H : \chi \times \boldsymbol{Y} \rightarrow \mathbb{R}$ from $\boldsymbol{S}$ as the combination of a set of weak hypotheses $(h^{(1)}(\mathrm{x}, y), \ldots, h^{(R)}(\mathrm{x}, y))$ with a small Hamming loss in the form $H(\mathrm{x}, y) = \sum_{r=1}^{R} \alpha^{(r)} h^{(r)}(\mathrm{x}, y)$, where R is the number of rounds. A given document x is then assigned to a category y if and only if $H(\mathrm{x}, y)$ is positive.

To produce a new weak hypothesis $h^r(\mathrm{x})$ during a boosting round r, AdaBoost.MH examines all the training terms in $\mathcal{T}$ and determines their absence/presence in each document under each label to build a set of weak hypotheses $(h_1^{(r)}(\mathrm{x}), \ldots, h_v^{(r)}(\mathrm{x}))$, one for each term in $\mathcal{T}$. Then a single hypothesis at a certain term (called the "*pivot term*") that minimizes the Hamming loss is returned, and all other $(v-1)$ hypotheses are discarded. The examination of all training terms in each round increases the training time, especially when the data size is large.

The multi-label boosting algorithm RFBoost (Al-Salemi, Ab-aziz et al., 2016) controls the computational learning cost by first reducing the number of terms to be examined by means of feature ranking. Then, for each boosting round r, only a small subset of the ranked features of a fixed size k, which is a very small number compared to v (the number of training features), are filtered and used to build the weak hypothesis $h^{(r)}$. For the next boosting round $(r+1)$, the pivot term selected in round r is removed and replaced with the next ranked feature in the ranked feature list, and so on. An experimental analysis proved that RFBoost is faster and more accurate than AdaBoost.MH and all of the other algorithms that were examined in the evaluation.

Because the main factor accounting for RFBoost's good performance is its feature ranking, in the present paper we investigate several state-of-the-art feature weighting methods for ranking the features, in an attempt to improve RFBoost's performance. In addition, we propose a variant of RFBoost called "RFBoost1". Rather than filtering a subset of ranked features, as RFBoost does, RFBoost1 selects a single feature to pass to the base learner as a pivot term. This reduces the computational time for building one weak hypothesis from $O(nmv)$ in AdaBoost.MH, where n is the number of training documents, m is the number of labels, and v is the training vocabulary, to $O(nm1) = O(nm)$ in RFBoost1.

3. Related work

A simple approach to solving the multi-label classification problem involves transforming the multi-label task into a set of single-label subtasks. A single-label classifier is then used to solve each subtask, and the outputs are combined to solve the original multi-label task. To this end, methods such as binary relevance (Boutell et al., 2004), classifier chains (Read et al., 2011), label powerset (Tsoumakas & Vlahavas, 2007), ranking by pairwise comparison (Hüllermeier et al., 2008), and calibrated ranking by pairwise comparison (Fürnkranz et al., 2008) have been introduced and used to solve many multi-label classification problems. Despite their simplicity, transformation-based methods still depend on the single-label classifiers, and the huge number of single-label classifiers makes it difficult to decide which transformation methods count as state-of-the-art for multi-label classification. Furthermore, the transformation methods have been criticized for being time-consuming and exhaustive in terms of memory resources (Zhang & Zhou, 2014).

An alternative approach to solving the multi-label classification problem is to adapt a single-label algorithm to directly solve the multi-label problem. Several multi-label classifiers have been adapted from single-label classifiers. For example, multi-label *k*NN (ML*k*NN; Zhang & Zhou, 2007) was adapted from the traditional *k*NN algorithm for multi-label classification and uses the maximum posterior principle to assign a given test instance to a label based on the prior and posterior probabilities for labels' frequencies within the k nearest neighbours. Another multi-label classification algorithm adapted from the *k*NN algorithm, BR*k*NN, uses the binary relevance (BR) transformation with the *k*NN algorithm. However, BR*k*NN is more efficient because it reduces the number of the BR pairs for each label. In multi-label instance-based learning by logistic regression (IBLR-ML; Cheng & Hüllermeier, 2009), the *k*NN algorithm is combined with logistic regression and allows the interdependencies between class labels to be captured correctly, so that the multi-label classification is handled appro-

priately. However, all of these algorithms are instance-based algorithms. The main disadvantage of instance-based algorithms (also known as *lazy learning* algorithms) is that they require a large amount of memory to store the data and greater time to predict new examples. Another algorithm, Rank-SVM (Elisseeff & Weston, 2002; Xu, 2012), is an extension of the binary SVM for multi-label classification, with predictions based on a large margin ranking system that shares many properties of SVMs.

AdaBoost.MH is also an adapted multi-label algorithm; it extends the well-known AdaBoost algorithm (for binary classification) to solve multi-label classification tasks. Since its introduction, AdaBoost.MH has received much attention and has been used to solve many real-life classification tasks (Busa-Fekete & Kégl, 2010; Busa-Fekete, Kégl, Éltető, & Szarvas, 2011; Elghazel et al., 2016; T. Zhang, Liu, Xu, & Lu, 2011). Although it is an accurate multi-label classification algorithm, AdaBoost.MH has been criticized for its inefficient processing time (Esuli et al., 2006). The use of single words as elements in the vector space model Bag-Of-Words (BOW) to represent texts generates a high-dimensional feature space. Although feature selection methods can be used to reduce the features' dimensionality and thereby accelerate AdaBoost.MH, AdaBoost.MH's weak learning is still computationally inefficient when the number of training features is large. Al-Salemi, Aziz, and Noah (2015b) attempted to tackle this problem by representing a text as a small set of latent topics using a latent Dirichlet allocation (LDA; Blei, Ng, & Jordan, 2003) topics model. Experimental results established that representing texts with a small number of topics significantly accelerates AdaBoost.MH learning and enhances its categorization performance. This framework, "LDA-AdaBoost.MH," has been extended to other variants of AdaBoost.MH for multi-label text categorization (Al-Salemi, Aziz, & Noah, 2015a): MP-Boost (Esuli et al., 2006), AdaBoost.MH with tree and product (Busa-Fekete & Kégl, 2009), LazyBoosting (Escudero, Mrquez, & Rigau, 2000), and BanditBoost (Busa-Fekete & Kégl, 2009). Although topics-based representation proved to be an effective representation model for improving boosting algorithms in general, it yields poor classification performance on imbalanced and large-scale datasets compared to the BOW representation (Al-Salemi et al., 2015b). Recently, this limitation has been addressed by merging the latent topics with the top-ranked words in a combined representation model, named "BoWT" (Al-Salemi et al., 2016). We therefore use the BoWT representation model for text representation in the present study.

The variants of AdaBoost.MH which aim to accelerate its weak learning by reducing the search space that are most related to our work are LazyBoosting (Escudero et al., 2000) and BanditBoost (Busa-Fekete & Kégl, 2009). LazyBoosting reduces the search space for the pivot terms by randomly selecting a small subset of training features. However, randomly reducing the features can exclude the important features which are candidates for being pivot terms, and this will negatively affect the classification accuracy. Busa-Fekete and Kegl (2009) improved on the LazyBoosting algorithm by proposing the use of multi-armed bandits (MABs) as an inner feature selection method. They employed the upper-confidence bound (UCB) bandit algorithm (Auer, 2002) to choose only the informative features while retaining some consideration of new features. The key behind this algorithm, which is called "BanditBoost", is to associate a bandit arm with each feature and examine the loss reduction as a score. However, a comparative study of boosting algorithms for multi-label text categorization by Al-Salemi et al. (2015a) showed that neither LazyBoosting nor BanditBoost performs well.

Instead of using random feature selection or inner feature selection to decrease the number of features to be examined in each boosting round, as LazyBoosting and BanditBoost respectively do, RFBoost (Al-Salemi, Ab-Aziz et al., 2016) addresses this problem by first ranking the features and then, in each boosting round, passing a small subset of the top-ranked features to the base learner. Because feature ranking is the main idea of RFBoost, in this paper we investigate several feature weighting methods for feature ranking. In addition, with RFBoost1, we investigate accelerating the weak learning by passing only one ranked feature to the base learner in each boosting round.

Algorithm 1 AdaBoost.MH formal description.

Input: a training set $\boldsymbol{S}$, uniform distribution $\boldsymbol{W}$, and number of boosting rounds $\boldsymbol{R}$
Output: final classifier $H(x,1)=\sum_{r=1}^{R}\alpha^{(r)}h^{(r)}(x,1)$
Begin
1: **set** $\boldsymbol{W}^{(1)}\leftarrow\boldsymbol{W}$
2: **for** $r\leftarrow 1$ **to** $\boldsymbol{R}$ **do**
3: **pass** $(\boldsymbol{S}, \boldsymbol{W}^{(r)})$ *to a base learner*
4: **get** *weak hypothesis* $h^{(r)}:\chi\times\mathcal{Y}\rightarrow\mathbb{R}$
5: **choose** $\alpha^{(r)}\in\mathbb{R}^{+}$
6: **update** $\boldsymbol{W}^{(r+1)}$
7: **end for**
end

4. Boosting algorithms

In this section, we describe the boosting algorithms to be evaluated—AdaBoost.MH, RFBoost, and the proposed RFBoost1—and present their mechanisms for performing the weak learning and selecting the weak hypotheses. In addition, we conduct an empirical analysis to confirm that RFBoost1 does not penalize the boosting theory.

4.1. AdaBoost.MH

Let $\boldsymbol{S}=\{(\mathrm{x}_1, Y_1), \ldots, (\mathrm{x}_n, Y_n)\}$ be a training set of labelled documents. AdaBoost.MH (Algorithm 1) induces a composite classifier $H:\chi\times\mathcal{Y}\rightarrow\mathbb{R}$ from $\boldsymbol{S}$ as the combination of a set of weak hypotheses $h^{(1)}(\mathrm{x}), \ldots, h^{R}(\mathrm{x})$ in the form $H(\mathrm{x},y)=\sum_{r=1}^{R}\alpha^{(r)}h^{(r)}(\mathrm{x},y)$ with a small Hamming loss. AdaBoost.MH minimizes the Hamming loss by minimizing the training *exponential margin-based error* as defined in Eq. (1):

$$\hat{R}_{EXP}(\mathrm{H},\mathbf{W})=\sum_{i=1}^{n}\sum_{l=1}^{m}w_{i,l}\exp\left(-H(\mathrm{x}_i,l)\varphi(\mathrm{x}_i,l)\right) \tag{1}$$

where $\mathbf{W}=[w_{i,l}]_{n\times m}$ is a uniform distribution of weights over the training documents and categories, l is the index of a category y in $\mathcal{Y}$, and $\varphi(\mathrm{x}_i,l)$ is the target function. AdaBoost.MH works by iteratively passing $\mathbf{W}$ along with $\boldsymbol{S}$ to a *base learner* and generating a set of weak hypotheses $h^{(r)}:\chi\times\boldsymbol{Y}\rightarrow\mathbb{R}$, $\boldsymbol{r}=\mathbf{1},\ldots,\boldsymbol{R}$. In each boosting iteration r, the goal is to minimize the value of the *base objective* $Z^{(r)}$ as defined in Eq. (2):

$$Z^{(r)}=\sum_{i=1}^{n}\sum_{l=1}^{m}w_{i,l}^{(r)}\exp\left(-\alpha^{(r)}\,h^{(r)}(\mathrm{x}_i,l)\varphi(\mathrm{x}_i,l)\right) \tag{2}$$

where $\alpha^{(r)}$, the *base coefficient*, takes a positive real value.

After selecting the weak hypothesis $h^{(r)}$ with the smallest value for Z, and based on its predictions, the distribution $W^{(r+1)}$ is updated for the next boosting round $(r+1)$ and normalized by Z according to Eq. (3):

$$w_{i,l}^{(r+1)}=\frac{w_{i,l}^{(r)}\exp\left(-\alpha^{(r)}\,h^{(r)}(\mathrm{x}_i,l)\varphi(\mathrm{x}_i,l)\right)}{Z^{(r)}} \tag{3}$$

The same process is repeated for the following iterations until all boosting rounds have been performed. AdaBoost.MH then uses

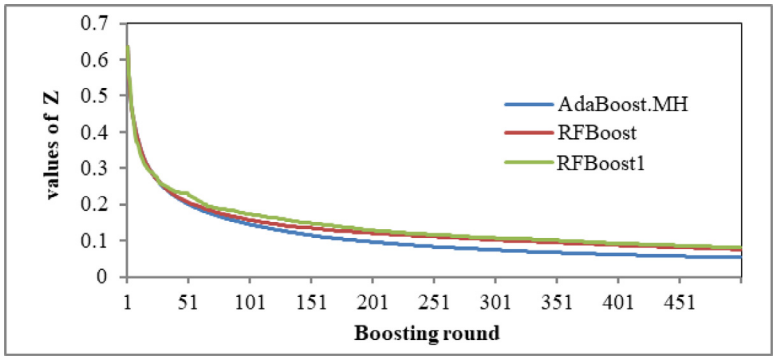

Fig. 1. Base objective values for Reuters21578 dataset.

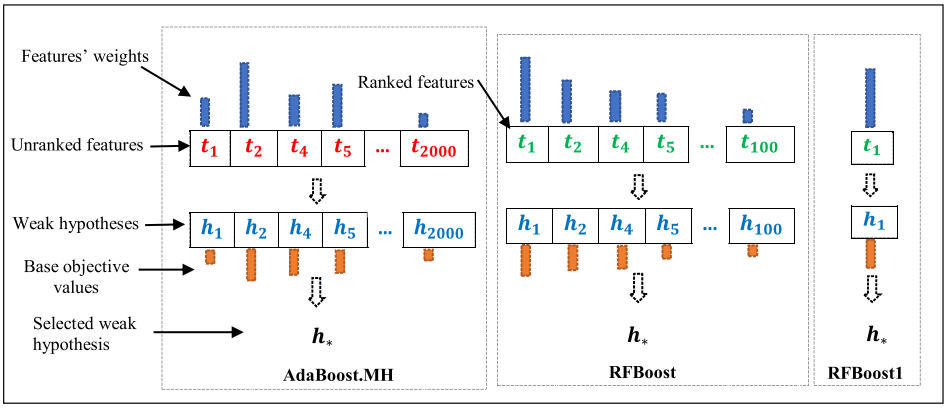

Fig. 2. Illustration of the weak hypothesis selection in the first boosting round.

the selected weak hypotheses to formulate the final classifier according to Eq. (4):

$$H(\mathrm{x}, l) = \sum_{r=1}^{R} h^{(r)}(\mathrm{x}, l) \tag{4}$$

Thus, the correct labels to be assigned to a given text x are those that have obtained a positive value, while those with negative values are considered false labels:

$$l_{\boldsymbol{H}}(\mathrm{x}) = \mathrm{sign}(H(\mathrm{x}, l)), \quad l = 1, \ldots, m \tag{5}$$

To use AdaBoost.MH for text categorization, the single words (terms) that represent the training documents are used to construct the weak hypotheses. Let $\boldsymbol{T} = \{\boldsymbol{t_1}, \ldots, \boldsymbol{t_v}\}$ be the set of all training terms. Each document $\mathrm{x_j}$ is represented as a vector of v binary weights $\mathrm{x} = \langle x_1, \ldots, x_v \rangle$ in which x_i takes the value 1 if t_i appears in x, and 0 otherwise.

For a boosting round r, *AdaBoost*.MH uses each term t_i to build a new weak hypothesis $h_i^{(r)}$. It examines the absence/presence of t_i in x to generate a decision stump of two real numbers as a prediction of x being assigned to a label l, as follows:

$$h_i^{(r)}(\mathrm{x}, l) = \begin{cases} c_{0l} & if\ x_i = 0 \\ c_{1l} & if\ x_i = 1 \end{cases} \tag{6}$$

where c_{0l} and c_{1l} are real-valued constants chosen during iteration r according to the minimization policy of the base objective $Z^{(r)}$. To obtain the values of c_{0l} and c_{1l} for a term t_i, the set of training documents $\mathcal{X}$ is first split into two subsets (X_0, X_1) having the form:

$$X_u = \{\mathrm{x} : x_i = u\}, \quad u = 0, 1 \tag{7}$$

where t_i occurs in each document in X_1 and does not occur in any document in X_0.

Then, for each label l and the current weight distribution $\boldsymbol{W}^{(r)}$, the weights W_{1l}^{+1} for all documents that belong to l and contain t_i, the weights W_{0l}^{+1} for all documents that belong to l and do not contain t_i, the weights W_{1l}^{-1} for all documents that do not belong to l and contain t_i, and the weights W_{0l}^{-1} for all documents that do not belong to l and do not contain t_i are summed according to Eq. (8):

$$W_{ul}^{p} = \sum_{l=1}^{m} \sum_{j=1}^{n} w_{j,l}^{(r)}\ x_i = u\varphi(\mathrm{x}_j,\ l) = p \tag{8}$$

where $u = 0,\ 1$, $\varphi(\mathrm{x}_i, l)$ is the target function and $p = +1, -1$.

Thus, for term t_i, the predictions $h_i^{(r)}(c_{0l})$ and $h_i^{(r)}(c_{1l})$ of document x being assigned to a label l are computed according to Eqs. (9) and (10):

$$c_{0l} = \frac{1}{2} \ln \frac{W_{0l}^{+1} + \epsilon}{W_{0l}^{-1} + \epsilon} : \ x_i = 0 \tag{9}$$

$$c_{1l} = \frac{1}{2} \ln \frac{W_{1l}^{+1} + \epsilon}{W_{1l}^{-1} + \epsilon} : \ x_i = 1 \tag{10}$$

Note that a small value ε is added to both Eqs. (9) and (10) to avoid division by zero. Following Schapire and Singer (2000), $\epsilon = \frac{1}{mn}$. By choosing $\alpha^{(r)} = 1$ (Schapire & Singer, 2000), $Z^{(r)}$ is obtained using Eq. (11):,

$$Z^{(r)} = 2 \sum_{u \in \{0,1\}} \sum_{l=1}^{m} \sqrt{W_{ul}^{+1} \times W_{ul}^{-1}} \tag{11}$$

Among the generated weak hypotheses $h_1^{(r)}, \ldots, h_v^{(r)}$, only one weak hypothesis—denoted as $h_*^{(r)}$—corresponding to the pivot term t_* is selected; this is the hypothesis for which $Z_*^{(r)}$ is the minimum. Then the distribution matrix $\boldsymbol{W}^{(r+1)}$ is updated for the next

Algorithm 2 AdaBoost.MH weak learning.

Input: a training set $\boldsymbol{S}$, uniform distribution $\boldsymbol{W}$, number of boosting rounds $\boldsymbol{R}$, training feature index $\boldsymbol{T}$
Output: final classifier $H(\mathrm{x}, l)$
begin
1. **set** $\mathcal{H}_* \leftarrow ( )$
2. **set** $\boldsymbol{W}^{(1)} \leftarrow \boldsymbol{W}$
3. **for** $r \leftarrow 1$ **to** $\boldsymbol{R}$ **do** ▷ for iteration r
4. **set** $\mathcal{H}^{(r)} \leftarrow ( )$
// Generate a set of weak hypotheses $\mathcal{H}^r$, one for each feature; Eqs. (7)–(10)
5. **for** $i \leftarrow 1$ **to** v **do** ▷ for each training feature in $\boldsymbol{T}$
6. $h_i^{(r)} \leftarrow$ **weak_learner** $(\boldsymbol{S}, t_i, \boldsymbol{W}^{(r)})$ ▷ pass the training examples, the current term, and the weight distribution to a base learner and get a new weak hypothesis
7. $\mathcal{H}^{(r)} \leftarrow \mathcal{H}^{(r)} \cup h_i^{(r)}$ ▷ add $h_i^{(r)}$ to $\mathcal{H}^{(r)}$
8. **end for**
// Select the best weak hypothesis
9. **set** $Z_*^{(r)} \leftarrow$ Real.Max ▷ $Z_*^{(r)}$ Real.Max is the max positive real value
10. **for** $j \leftarrow 1$ **to** v **do**
11. **if** $Z_j^{(r)} < Z_*^{(r)}$ **then** ▷ Z is computed according to Eq. (11)
12. $Z_*^{(r)} \leftarrow Z_i^{(r)}$
13. $h_*^{(r)} \leftarrow h_i^{(r)}$
14. **end if**
15. **end for**
16. $\mathcal{H}_* \leftarrow \mathcal{H}_* \cup h_*^{(r)}$
// Update the weights according to Eq. (3), based on the $h_*^{(r)}$ predictions
17. **update** $\boldsymbol{W}^{(r+1)}$
18. **end for**
// return the final classifier
19. **return** $H(\mathrm{x}, l) \leftarrow \sum_{r=1}^{R} h_*^{(r)}(\mathrm{x}, l)$
end

iteration according to Eq. (3), and the same learning process is performed for the next boosting iterations. AdaBoost.MH's weak learning mechanism for text categorization is presented in Algorithm 2.

4.2. RFBoost

In RFBoost, the training features are first ranked. Then, in each boosting round, only a small subset of the ranked features are used to induce the weak hypothesis corresponding to a pivot term. After being selected in the current boosting round, the pivot term is then removed and replaced with the next ranked feature in the ranked feature index.

RFBoost iteratively generates a set of weak hypotheses in the same manner as AdaBoost.MH. However, RFBoost reduces the number of terms to be passed to the base learner in each iteration, and this accelerates the weak learning. Algorithm 3 describes RFBoost's weak learning procedure. On Line 4, a small subset *RF* of the top k ranked features in the ranked feature index $\mathcal{T}'$ are selected. The weak hypotheses are then generated (Lines 6 – 10), one for each feature in *RF*. To keep *RF* the same size k for the next iteration, the selected pivot term t'_* is replaced by the next highest ranked feature in $\mathcal{T}'$. The weak learning is continued in the following iterations, and the final classifier is finally constructed as a combination of the selected weak hypotheses.

4.3. RFBoost1

As mentioned earlier, this paper proposes a variant RFBoost1 of the RFBoost algorithm. The difference between RFBoost and RFBoost1 is that while RFBoost filters and passes a small subset of the ranked features to the base learner in each boosting round for selecting a weak hypothesis, RFBoost1 selects only one ranked feature. Thus, the size of the search space for the weak hypothesis is reduced from k, the number of ranked features in RFBoost, to one ranked feature in RFBoost1.

Algorithm 3 RFBoost weak learning.

Input: a training set $\boldsymbol{S}$, uniform distribution $\boldsymbol{W}$, number of boosting rounds $\boldsymbol{R}$, training feature index $\boldsymbol{T}$, feature ranking method *Franker*, number of ranked features k
Output: final classifier $H(\mathrm{x}, l)$
begin
1. $\boldsymbol{T}' \leftarrow$ **Rank**$(\boldsymbol{S}, \boldsymbol{T}, \boldsymbol{Fran}\text{ker})$ ▷ rank the feature index $\boldsymbol{T}$ as a new ranked index $\boldsymbol{T}'$
2. **set** $\mathcal{H}_* \leftarrow ( )$
3. **set** $\boldsymbol{W}^{(1)} \leftarrow \boldsymbol{W}$
4. **set** $RF \leftarrow (t'_1, \ldots, t'_k)$ initialize a set *RF* of the top k ranked features in $\boldsymbol{T}'$
5. **for** $r \leftarrow 1$ **to** R **do** ▷ for each iteration r
6. **set** $\mathcal{H}^{(r)} \leftarrow ( )$
// Generate a set of weak hypotheses $\mathcal{H}^{(r)}$, one for each feature; Eqs. (7)–(10)
7. **for** $i \leftarrow 1$ **to** k **do** ▷ for each ranked feature in *RF*
8. $h_i^{(r)} \leftarrow$ **weak_learner**$(\boldsymbol{S}, t'_i, \boldsymbol{W}^{(r)})$ ▷ pass the training examples, the current ranked feature, and the weight distribution to a base learner and get a new weak hypothesis
9. $\mathcal{H}^{(r)} \leftarrow \mathcal{H}^{(r)} \cup h_i^{(r)}$ ▷ add $h_i^{(r)}$ to $\mathcal{H}^{(r)}$
10. **end for**
// Select the best weak hypothesis
11. **set** $Z_*^{(r)} \leftarrow$ Real.Max
12. **for** $j \leftarrow 1$ **to** k **do**
13. **if** $Z_j^{(r)} < Z_*^{(r)}$ **then** ▷ Z is computed according to Eq. (11)
14. $Z_*^{(r)} \leftarrow Z_i^{(r)}$
15. $h_*^{(r)} \leftarrow h_i^{(r)}$
16. **end if**
17. **end for**
18. $\mathcal{H}_* \leftarrow \mathcal{H}_* \cup h_*^{(r)}$
// Update the weights according to Eq. (3) based on the predictions $h_*^{(r)}$
19. **update** $\boldsymbol{W}^{(r+1)}$
// Replace the selected feature in *RF* by the next ranked feature in $\boldsymbol{T}'$ (t'_{k+r})
20. $RF \leftarrow (RF - \{t'_*\}) \cup \{t'_{k+r}\}$
21. **end for**
// return the final classifier
22. **return** $H(\mathrm{x}, l) \leftarrow \sum_{r=1}^{R} h_*^{(r)}(\mathrm{x}, l)$
end

Algorithm 4 RFBoost1 weak learning.

Input: a training set $\boldsymbol{S}$, uniform distribution $\boldsymbol{W}$, number of boosting rounds $\boldsymbol{R}$, training feature index $\mathcal{T}$, feature ranking method *FRanker*
Output: final classifier $H(\mathrm{x}, l)$
begin
1. $\mathcal{T}' \leftarrow$ Rank$(\boldsymbol{S}, \mathcal{T},$ *Franker*$)$ ▷ rank the feature index $\mathcal{T}$ in a new ranked index $\mathcal{T}'$
2. **set** $\mathcal{H}_* \leftarrow ( )$
3. **set** $\boldsymbol{W}^{(1)} \leftarrow \boldsymbol{W}$
4. **for** $r \leftarrow 1$ **to** R **do** ▷ for iteration r
// In each iteration generate only one weak hypothesis using the ranked feature t'_r; Eqs. (7)–(10)
5. $h_*^{(r)} \leftarrow weak_{learner(\boldsymbol{S},\ t'_r,\ \boldsymbol{W}^{(r)})}$
6. $\mathcal{H}_* \leftarrow \mathcal{H}_* \cup h_*^{(r)}$
// Update the weights according to the predictions of $h_*^{(r)}$
7. **for** $i \leftarrow 1$ **to** $|\chi|$ **do** ▷ for each example in $\boldsymbol{S}$
8. **for** $l \leftarrow 1$ **to** $|\mathcal{Y}|$ **do** ▷ for each label in $\boldsymbol{S}$
9. $w_{i,l}^{(r+1)} \leftarrow \frac{w_{i,l}^{(r)} \exp(-\alpha^{(r)} h_*^{(r)}(\mathrm{x}_i, l) \varphi(\mathrm{x}_i, l))}{Z_*^{(r)}}$
10. **end for**
11. **end for**
12. **end for**
// return the final classifier
13. **return** $H(\mathrm{x}, l) = \sum_{r=1}^{R} h_*^{(r)}(\mathrm{x}, l)$
end

Algorithm 4 describes RFBoost1's weak learning. Let $\mathcal{T}' = \{t'_1, \ldots, t'_v\}$ be the ranked term index, sorted in descending order based on the features' weights. Thus, for a boosting round r, the feature term t'_r is used as a pivot term to generate only one weak hypothesis $h_*^{(r)}$. After performing all boosting rounds, the generated weak hypotheses in $\mathcal{H}_*$ are then used to construct the final classifier.

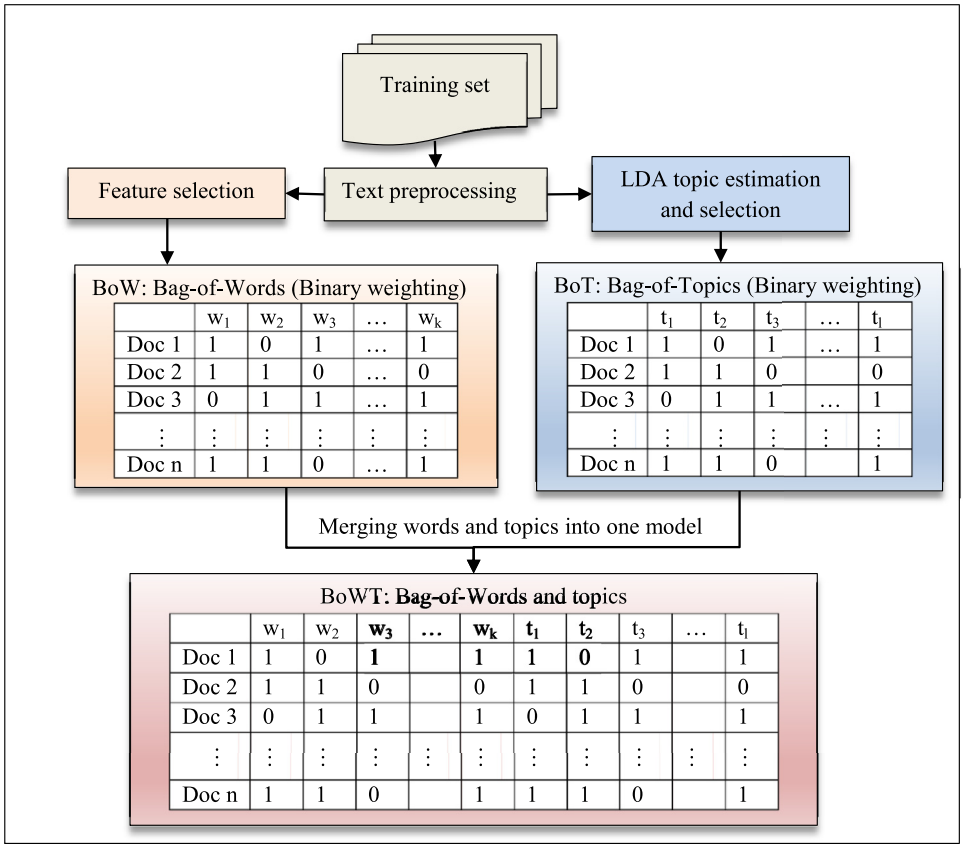

Fig. 3. BoWT text representation model.

RFBoost1 can be seen as a special case of RFBoost in which the number of selected features is exactly one. However, when the only feature passed to the base learner is selected as a pivot term, the following question arises: Does RFBoost1 still adhere to the boosting theory? To answer this question, we performed an empirical validation to ensure that RFBoost1 is as capable of minimizing the empirical Hamming loss of the weak hypotheses as AdaBoost.MH. Schapire and Singer (2000) proved that the empirical Hamming loss of AdaBoost.MH's final classifier is at most

$$\widehat{HLoss(H)} \leq \prod_{r=1}^{R} Z^{(r)} \tag{12}$$

We need to validate that the Hamming loss is also minimized in each boosting round in RFBoost1, such that for iterations r_1 and r_2, $r_1 < r_2$, the product of the base objective $\prod_{r=1}^{r1} Z^{(r)} < \prod_{r=1}^{r2} Z^{(r)}$, that is, we need to prove that the product of the base objective Z is minimized. We performed an empirical analysis to validate that RFBoost1 minimizes the empirical Hamming loss as given in Eq. (12). The benchmarked dataset Reuters21578 was used to perform the empirical validation. After performing 500 boosting rounds, the experimental results shown in Fig. 1 prove that RFBoost1 is as capable of minimizing the Hamming loss as AdaBoost.MH and RFBoost. This is because the values of the base objective Z become smaller when the number of iterations is increased. These findings prove that RFBoost1 does not interfere with the boosting theory.

Fig. 2 illustrates the selection process for the weak hypothesis in the first boosting round of each boosting algorithm. Assume that the number of training features is 2000. According to Algorithm 2 (Lines 5–8), AdaBoost.MH builds 2000 weak hypotheses, one for each feature. Then, of the 2000 generated weak hypotheses ($h_1, \ldots, h_{2000}$), only one weak hypothesis h_* at a certain feature t_* is returned; this weak hypothesis minimizes the value of the base objective Z_*. Assuming that the size of the selected subset of the ranked features to be passed to the weak learning in Algorithm 3 is set to 100 ($k = 100$, provided as the user's input), RFBoost reduces the search space for the weak hypotheses from 2000 in AdaBoost.MH to 100. Thus, the number of weak hypotheses that are generated is reduced to 100. Of the 100 weak hypotheses, only one weak hypothesis h_* is selected for the final classifier; this weak hypothesis minimizes Z_*. In contrast, RFBoost1 (Algorithm 4) reduces the number of generated weak hypotheses to one, with the corresponding feature (t_1) passed to the base learner. Thus, no weak hypothesis selection is performed in RFBoost1, which accelerates the process of weak learning.

5. Feature ranking methods

We evaluate several feature weighting methods for ranking features for RFBoost as well as for RFBoost1; the features will be sorted based on the weights obtained by the feature weighting methods. We evaluate the feature weighting methods information gain, chi-square, mutual information, odds ratio, GSS-coefficient, F1 score, and accuracy. These feature weighting methods have been widely used in many applications (Abualigah, Khader, Al-Betar, & Alomari, 2017; Forman, 2003; Katrutsa & Strijov, 2017; Song et al., 2017; Uysal, 2016).

For each label l and feature term t in $\mathcal{T}$, suppose that tp is the number of documents in l that contain t, fp is the number of documents not in l that contain t, fn is the number of documents in l that do not contain t, and tn is the number of documents not in l that do not contain t. Let $gPos = tp + fn$, $gNeg = fp + tn$, $fPos = tp + fp$, and $fNeg = tn + fn$, and let n be the total number of documents in the training set. The score of each of the following feature weight-

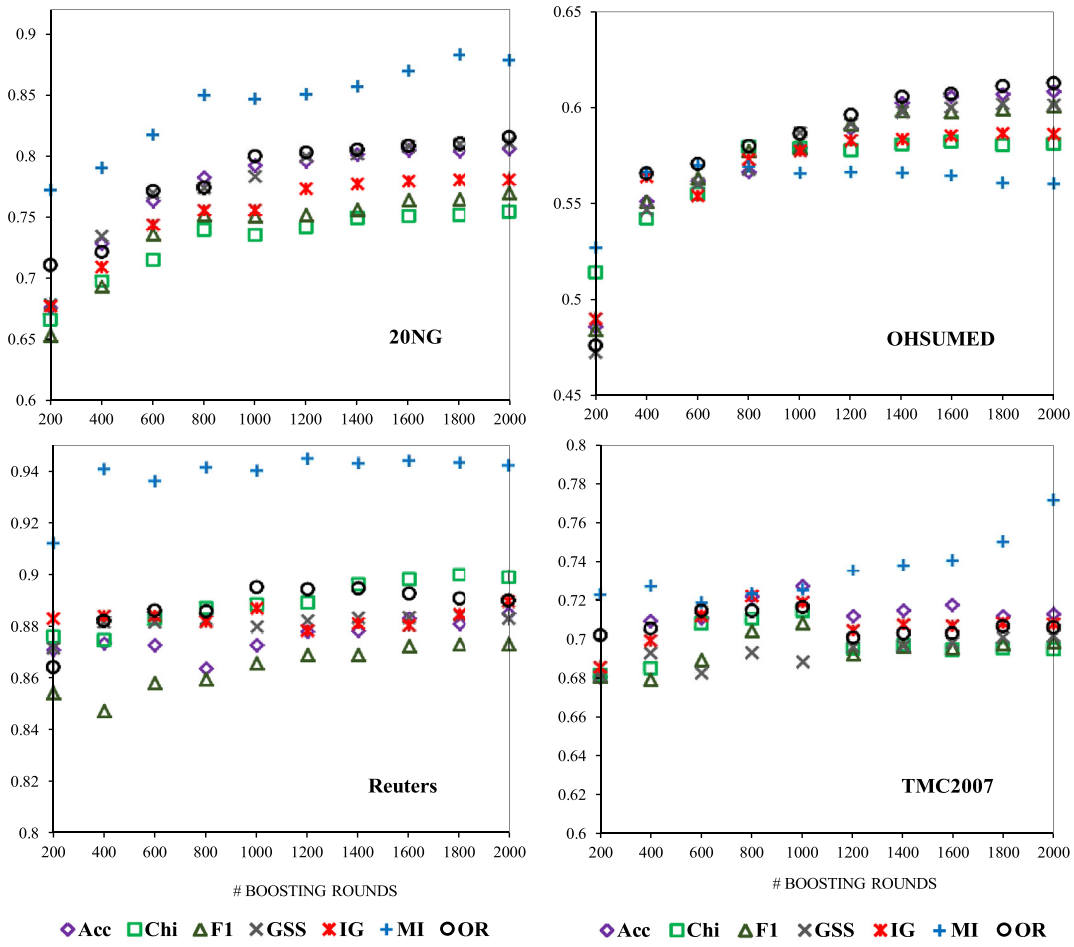

Fig. 4. MacroF1 scores for RFBoost with different feature ranking methods.

ing (selection) measures is considered as a weight for the term t being assigned to the label l.

Information gain (IG) is widely used as a term importance criterion. It is based on information theory (Mitchell, 1997). The approximate information gain of a term t being assigned to a label l is defined in Eq. (13):

$$\begin{aligned}\mathrm{IG}(t,\ l) \cong\ & -\frac{gPos}{n}\log_2\left(\frac{gPos}{n}\right) - \frac{gNeg}{n}\log_2\left(\frac{gNeg}{n}\right) \\ & + \frac{tp}{n}\log_2\left(\frac{tp}{fPos}\right) + \frac{fp}{n}\log_2\left(\frac{fp}{fPos}\right) \\ & + \frac{fn}{n}\log_2\left(\frac{fn}{fNeg}\right) + \frac{tn}{n}\log_2\left(\frac{tn}{fNeg}\right) \end{aligned} \quad (13)$$

Chi-square (Chi) measures the correlation between two variables and evaluates their independence (Rehman, Javed, Babri, & Saeed, 2015). The independence of a term t and a category l using chi-square is defined in Eq. (14):

$$\mathrm{Chi}(t,\ l) \cong\ n \times \frac{(tp\ \times\ tn\ -\ fn\ \times\ fp)^2}{fPos\ \times\ gPos\ \times\ fNeg\ \times\ gNeg} \quad (14)$$

The mutual information (MI) of two variables X and Y measures how much information X and Y share (Datta, Varma, & Singh, 2017; Wang & Lochovsky, 2004). MI has been extensively employed as a feature weighting method. It gauges the goodness of a term t and a category l according to Eq. (15):

$$\begin{aligned}\mathrm{MI}(t,\ l) \cong\ & \frac{tp}{n}\log_2\left(\frac{n\ \times\ tp}{gPos\ \times\ fpos}\right) + \frac{fp}{n}\log_2\left(\frac{n\ \times\ fp}{gNeg\ \times\ fPos}\right) \\ & + \frac{fn}{n}\log_2\left(\frac{n\ \times\ fn}{gPos\ \times\ fNeg}\right) + \frac{tn}{n}\log_2\left(\frac{n\ \times\ tn}{gNeg\ \times\ fNeg}\right) \end{aligned} \quad (15)$$

The odds ratio (OR) measures the odds of a term t occurring in a category l as a positive category normalized by the odds of t occurring in l as a negative category (Zheng, Wu, & Srihari, 2004); it is defined as follows:

$$\mathrm{OR}(t,\ l) \cong \frac{tp}{gPos}\ \times\ \left(1.0\ -\ \frac{fp}{gNeg}\right) \quad (16)$$

The **GSS-coefficient** (GSS) is a simplified chi-square method that was proposed as a feature selection method by Galavotti, Sebastiani, and Simi (2000). The GSS-coefficient of a term t being assigned to a label l is defined by Eq. (17):

$$\mathrm{GSS}(t,\ l) \cong \frac{tp\ \times\ tn - fn\ \times\ fp}{n^2} \quad (17)$$

The **F1 score** (F1) and **accuracy** (ACC) are essentially used for evaluating the performance of classification algorithms. They have been used as feature selection methods (Forman, 2003) based on the positive and negative documents for a category l that contain a term t. The F1 score and accuracy for a term t and a category l are defined by Eqs. (18) and (19):

$$\mathrm{F1}(t,\ l) \cong \frac{2tp}{gPos + fPos} \quad (18)$$

$$\mathrm{Acc}(t,\ l) \cong \left|\frac{tp}{gPos}\ -\ \frac{fp}{gNeg}\right| \quad (19)$$

After obtaining the feature terms' weights for each category, the features are sorted in descending order based on their weights. In accordance with Forman's round-robin technique (Forman, 2004), the features with the highest weights are then chosen for each label and pooled together into the ranked-features index $\mathcal{T}'$. For feature selection, a certain number of the top-ranked features in $\mathcal{T}'$ are chosen as training features.

6. BoWT text representation model

BoW is a typical text representation model in which single words are used to represent texts in the vector space. However, BoW has been criticized for ignoring the order of the words and their relationships in the texts. Moreover, BoW produces a high-dimensional space, which increases the training time for the classification algorithm.

A few recent studies have tried to tackle this problem by using topic modelling to represent each text as a small number of clusters of words instead of single words (Al-Salemi et al., 2015a, 2015b; Pavlinek & Podgorelec, 2017; Shams & Baraani-Dastjerdi, 2017). The LDA topic model is used to estimate the latent topics among the texts, and these topics are then used to represent the texts. Even though representing the texts as a small set of topics reduces the training time and enhances the categorization performance of boosting algorithms for multi-label text categorization, the experimental results in Al-Salemi et al. (2015a, 2015b) show that the topic-based representation is not effective when the data are imbalanced. This is because the number of topics associated with categories that have rare examples is very small and thus cannot characterize those categories perfectly for the learning algorithms. Recently, Al-Salemi et al. (2016) addressed this limitation by proposing BoWT as a hybrid representation obtained by merging the top-ranked words and the topics into one representation model.

In BoWT (as shown in Fig. 3), LDA is used to estimate the topics among the training documents. The topics are then selected based on their probabilities (see Al-Salemi et al., 2015b for further details) and merged with the top-ranked words to produce a new, combined representation model. For the evaluation phase, the topics of the test texts are inferences based on the LDA's outputs in the topics estimation phase and are combined with the selected training features and merged together to represent the test documents to be used for evaluating the categorization performance.

7. Experiments and results

In this section, we first describe the datasets we used for the evaluation. Next, we present and discuss the experimental results of evaluating the feature ranking methods for RFBoost, and we provide a comparative evaluation of the boosting algorithms. Finally, we analyse the computation time for all algorithms.

7.1. Datasets

We utilized four widely used multi-label datasets for text categorization system evaluation in this study:

- **Reuters-21578,** a collection from Reuters news distributed over 135 categories, consists of 12,902 documents divided into 9603 documents for training and 3299 for testing. In this study, out of the 135 categories, we used only the 10 categories which contain the largest numbers of texts.
- **20-Newsgroups** (20NG), a multi-label text dataset, contains 20,000 documents distributed across 20 different newsgroups (categories). In this paper, we used a processed version[1] of 20NG that contains 18,846 documents divided into 11,314 documents for training and 7532 for testing.
- **OHSUMED** is a collection of medical abstracts from the Medical Subject Headings (MeSH) categories for the year 1991. The goal was to classify the abstracts into 23 cardiovascular disease categories. The dataset consists of 13,929 abstracts divided into 6286 abstracts for training and 7643 for testing. For this study, we used a version of this dataset prepared by Moschitti and Basili (2004).
- **TMC2007** is a multi-label text dataset developed for the SIAM Text Mining Competition, 2007. It consists of 28,596 text samples partitioned across 22 categories and divided into 21,519 text samples for training and 7077 for testing.

Table 1 provides summaries of the four datasets.

Table 1
Dataset summaries.

Size/number of:	Reuters	20NG	OHUSMED	TMC2007
training sets	7194	11,314	6286	21,519
test sets	2787	7532	7643	7077
labels	10	20	23	22
features	23,578	117,227	29,422	49,060

7.2. Experimental settings

The typical preprocessing tasks, namely, tokenization, normalization, stemming, and stop-word removal, were performed for each dataset. We represented features using the BoWT representation model. We set the number of estimated topics to 200 topics for each dataset, and the other parameters of LDA were the same as used in Al-Salemi et al. (2016). For all feature ranking methods, the top 3500 most highly weighted features (words and topics) were selected for each dataset. We evaluated the boosting algorithms with varying numbers of boosting rounds, ranging from 200 to 2000 with an increment of 200 rounds. We used the macro-averaged F1 (MacroF1) and micro-averaged F1 (MicroF1) to evaluate the classification performances.

We conducted the experiment in two stages. In the first stage, we evaluated feature ranking methods for RFBoost. We then used the ranking method that yielded the best performance for RFBoost in the second stage, which was a comparative evaluation of AdaBoost.MH, RFBoost, and RFBoost1.

We used the Friedman test (Demšar, 2006) to statistically validate the significant differences between the evaluation boosting algorithms. The Friedman rank test is defined in Eq. (20):

$$X_F^2 = \frac{12N_d}{k(k-1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right] \tag{20}$$

where N_d is the number of datasets, k is the number of evaluated methods, and R_j is the average rank of each method.

By carrying out the Friedman test on the performance ranks of the boosting algorithms for the different datasets, we obtained the distribution using Eq. (20) with $k-1$ degrees of freedom, and we computed the p-values at a 5% significance level. Following Demšar (2006), after securing a rejection of the null hypothesis that the classification algorithms have the same performance, we performed a two-tailed Bonferroni-Dunn test following the Friedman test to compare each pair of methods.

7.3. Results and discussion

In this section, we present and discuss the experimental results of the evaluation. The evaluation is divided into two parts: (a) an evaluation of feature ranking methods with respect to RFBoost's

[1] http://qwone.com/~jason/20Newsgroups/.

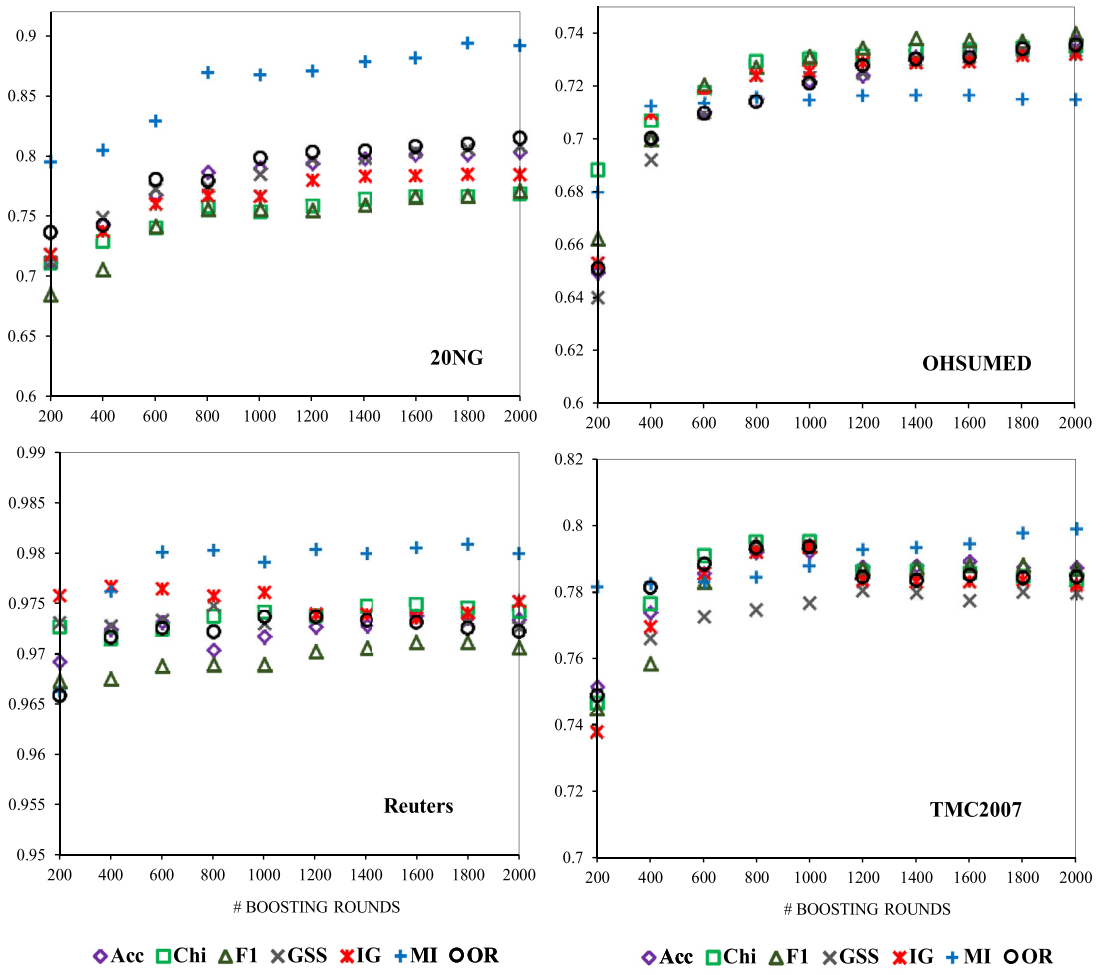

Fig. 5. MicroF1 values of RFBoost for different feature ranking methods.

performance and (b) an empirical comparison and statistical analysis of the evaluated boosting algorithms.

Evaluation of feature ranking methods for RFBoost

Fig. 4 depicts RFBoost's performance measured in terms of MacroF1 on all datasets using the different feature ranking methods. The MI feature ranking method yielded the best performance on all datasets except for the OHSUMED dataset. The MicroF1 values shown in Fig. 5 support these findings, as MI yields the best performance for RFBoost in general. There is no significant difference between the performances of the other feature ranking methods. The accurate performance of MI might be due to the fact that it computes the mutual dependence between each training term and category. It measures how much information the appearance of a term contributes to making the accurate assignment. However, the poor performance of MI for the OHSUMED dataset is due to the nature and the structure of the dataset.

The best results obtained for RFBoost as measured by both MacroF1 and MicroF1 for all ranking methods and all datasets are summarized in Table 2. It is clear that MI achieved the best MacroF1 and MicroF1 values on all datasets except for the OHSUMED dataset, where it had the worst value. The OR feature ranking method came in second place, with the second average rank after MI. OR achieved the best MacroF1 value on the OHSUMED dataset. The worst performance overall was exhibited by the GSS ranking method.

Comparative evaluation of boosting algorithms

MI was used for the feature ranking and selection in all of the boosting algorithms evaluated in this experiment, since it showed the best performance in the previous experiment. Fig. 6 represents the MacroF1 results for all evaluated boosting algorithms on the four datasets with different numbers of boosting rounds. RFBoost outperformed both AdaBoost.MH and RFBoost1 on all datasets except for OHSUMED, where AdaBoost.MH performed slightly better than RFBoost for all numbers of boosting rounds exceeding 400. However, at 200 and 400 boosting rounds, RFBoost1 achieved the best performance. It is also clear that RFBoost1 outperformed AdaBoost.MH on all datasets except for TMC2007, where AdaBoost.MH performed slightly better.

Turning to the MicroF1 results shown in Fig. 7, it is clear that RFBoost1 outperformed AdaBoost.MH on the 20NG and OHSUMED datasets, while AdaBoost.MH performed better on the Reuters and TMC2007 datasets. Further, RFBoost outperformed both AdaBoost.MH and RFBoost1 on all datasets except for OHSUMED, where RFBoost was slightly less effective than the other two boosting algorithms. The best MacroF1 and MicroF1 values for all boosting algorithms on all datasets are summarized in Table 3.

To statistically validate the significant differences between the boosting algorithms, we used the Friedman test at a 5% significance level. Having rejected the null hypothesis that the methods have the same performance, we conducted the two-tailed Bonferroni-Dunn test for multiple pairwise comparisons between the methods. However, the best experimental results for a boosting algorithm cannot be used to analyze the general performance of the

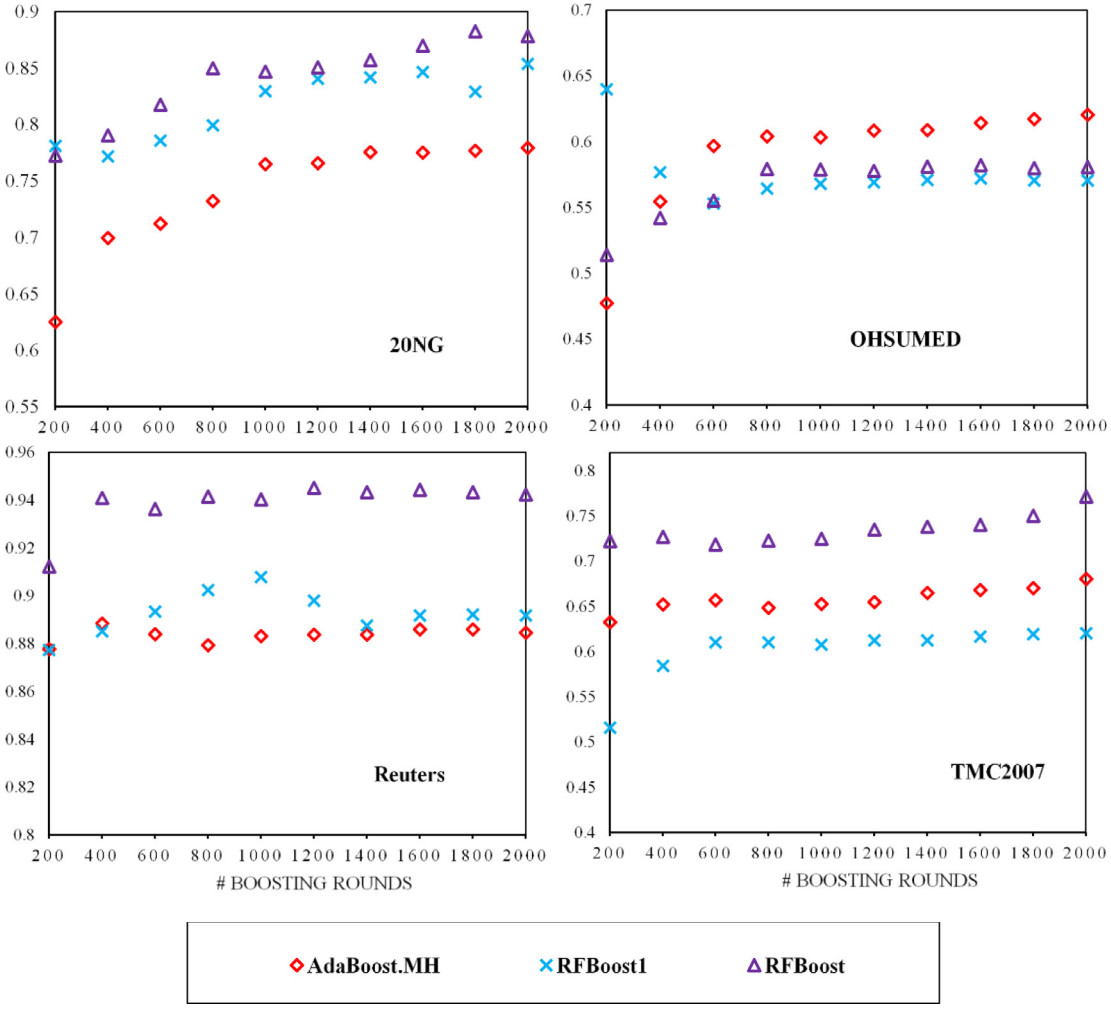

Fig. 6. MacroF1 values for boosting algorithms with different numbers of boosting rounds.

Table 2
The best MacroF1 and MicroF1 values (%) for RFBoost for all ranking methods.

Dataset	Acc		Chi		F1		GSS		IG		MI		OR	
	MacroF1	MicroF1	MacroF1	MicroF1	MacroF1	MicroF1	MacroF1	MicroF1	MacroF1	MicroF1	MacroF1	MicroF1	MacroF1	MicroF1
20NG	80.65	80.35	75.51	76.9	77.03	77.14	81.16	80.88	78.14	78.53	**88.28**	**89.38**	81.58	81.51
OHSUMED	60.85	73.67	58.24	73.5	60.10	**73.97**	60.19	73.34	58.68	73.21	57.01	71.67	**61.27**	73.56
Reuters	88.49	97.38	90.00	97.49	87.31	97.12	88.42	97.48	88.95	97.67	**94.52**	**98.09**	89.5	97.37
TMC2007	72.72	79.21	71.45	79.51	70.83	79.45	70.22	78.06	72.2	79.36	**77.19**	**79.91**	71.64	79.36
Rank average	3.25	4.25	5	4	5.75	4.25	4.75	4.75	4.25	4.25	**2.5**	**2.5**	**2.5**	3.75

algorithm for all boosting rounds. That is, each boosting algorithm was evaluated for 10 different numbers of boosting rounds, and the final classifier constructed at a certain number of boosting rounds was somewhat independent of the final classifier built at a different number of boosting rounds. Accordingly, instead of analyzing significant differences between the performances of the boosting algorithms in terms of the best overall results, we treat each classification result for a certain number of boosting rounds as an independent observation for validating the significant differences.

To validate the significant differences between the evaluated boosting algorithms, we first ranked the classification performances as measured by MacroF1 at each boosting round and on

Table 3
The best MacroF1 and MicroF1 results for all boosting algorithms on all datasets.

Dataset	MacroF1 (%)			MicroF1 (%)		
	AdaBoost.MH	RFBoost1	RFBoost	AdaBoost.MH	RFBoost1	RFBoost
20NG	80.65	85.37	**88.28**	79.15	87.1	**89.38**
OHSUMED	60.85	**63.99**	57.01	**74.05**	73.26	71.67
Reuters	88.49	90.8	**94.52**	97.33	96.95	**98.09**
TMC2007	72.72	62.09	**77.19**	75.78	72.72	**79.91**

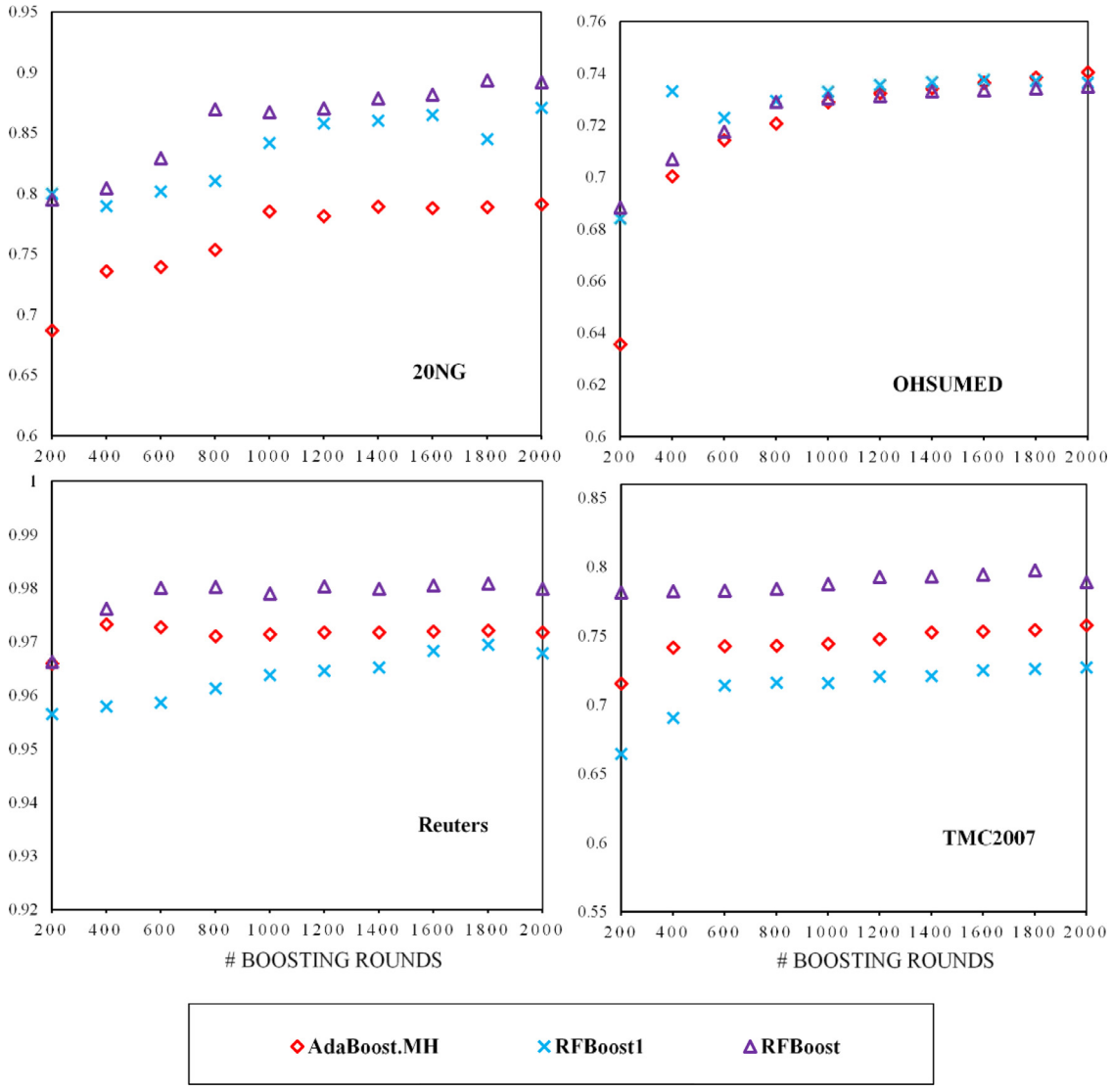

Fig. 7. MicroF1 values for the boosting algorithms with different numbers of boosting rounds.

Table 4
Multiple pairwise comparisons between the boosting algorithms.

	AdaBoost.MH	RFBoost1
RFBoost1	0.781	
RFBoost	**< 0.0001**	**< 0.0001**

Two-tailed Bonferroni-Dunn test after Friedman test with $\alpha = 0.05$, critical value = 5.991, p-value (two-tailed) = 0.0001, and Bonferroni corrected significance level = 0.0167.

all datasets. Then we conducted the Friedman test and obtained the distribution according to Eq. (20). The obtained p-value was 0.0001, which is smaller than the significance level (0.05). Thus, there was a significant difference between the methods' performances, and the null hypothesis that the methods have the same performance was rejected. Having rejected the null hypothesis, we conducted the two-tailed Bonferroni-Dunn test. The multiple pairwise comparisons between boosting algorithms (Table 4) showed that RFBoost performs significantly better than both RFBoost1 and AdaBoost.MH. Further, there is no significant difference between the performances of RFBoost1 and AdaBoost.MH; that is, RFBoost1 is a strong competitor to AdaBoost.MH in terms of the classification performance. However, RFBoost1 is much faster than AdaBoost.MH in terms of the training cost, which makes it a better classifier than AdaBoost.MH.

7.4. Computational cost

Suppose that the number of training examples is n, the number of categories is m, and the number of training features (after feature selection) is v. The time for performing one boosting iteration in AdaBoost.MH is linear to n, m, and v; that is, the time complexity is $O(mnv)$. RFBoost reduces v to a smaller number k (the number of top-ranked features, provided by the user). Thus, the time for running one boosting round in RFBoost is $O(mnk)$. RFBoost1 passes only one feature to the base learner, so that $k = 1$. Therefore, the time complexity for RFBoost1 is $O(mn1) = O(mn)$, that is, RFBoost1's computational time is linear to the number of categories and the training set size. The size of the training feature set has no effect on the learning time.

Fig. 8 presents the computational learning cost for the boosting algorithms in seconds for the Reuters dataset with different numbers of boosting rounds. The system used for performing all experiments was developed in Java, and all experiments were conducted on a PC with a 3.00 GHz Intel CORE-i5 processor with 8.00GB of RAM using the Windows 10 64-bit operating system. It is clear that

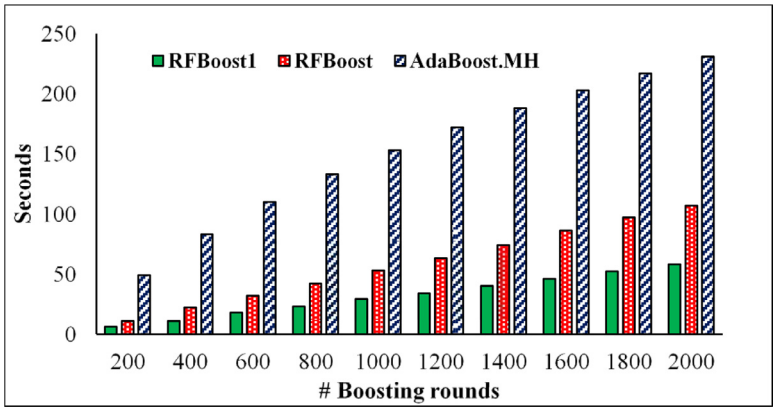

Fig. 8. The computational learning time for the boosting algorithms on the Reuters dataset.

RFBoost1 was the fastest algorithm for all cases, followed by RFBoost, while AdaBoost.MH was the slowest. RFBoost1 was approximately four times faster than AdaBoost.MH. RFBoost1's fast performance makes it a good alternative choice for boosting-based text categorization when the learning time is problematic.

8. Conclusion

RFBoost is an improved and accelerated version of AdaBoost.MH which proves to be effective and efficient for multi-label text categorization. Because feature ranking is known to be crucial for RFBoost's accurate and fast performance, this study aimed to investigate and evaluate the use of different feature weighting methods for feature ranking to improve RFBoost's performance. Among the evaluated feature ranking methods, MI proved to be a good choice for improving RFBoost's performance. However, the results show that there is no best feature ranking method in general, as the performances of the feature ranking methods basically depend on the nature of the datasets.

In addition, we proposed an accelerated version of RFBoost named "RFBoost1". Instead of filtering a set of highly weighted features for generating a new weak hypothesis, as RFBoost does, RFBoost1 passes only one of the top-ranked features to the base learner for generating a new weak hypothesis. The experimental results show that RFBoost1 accelerates the weak learning without penalizing the classification performance. The results also show that RFBoost1's performance is not significantly different than that of AdaBoost.MH, while RFBoost achieves the best performance overall. However, the main feature of RFBoost1 is its fast performance. It is approximately four times faster than AdaBoost.MH, making it a more efficient algorithm and a good choice when computation time is an issue. In the future, we will improve RFBoost1 so that it is more reliable and applicable to many real-life applications. We will also investigate the use of other existing feature ranking methods for improving both RFBoost and RFBoost1.

CRediT author statement

Bassam Al-Salemi: Conceptualization, Methodology, Software, Investigation, and Writing – Original Draft.

Masri Ayob: Validation, Supervision, Writing – Review & Editing, Project Administration, and Funding Acquisition.

Shahrul Azman Mohd Noah: Validation, Formal Analysis, Supervision, Writing – Review & Editing.

Acknowledgment

This work was supported by Universiti Kebangsaan Malaysia [Grant number DIP-2014-039].

References

Abualigah, L. M., Khader, A. T., Al-Betar, M. A., & Alomari, O. A. (2017). Text feature selection with a robust weight scheme and dynamic dimension reduction to text document clustering. *Expert Systems with Applications, 84*, 24–36.

Al-Salemi, B., Ab Aziz, M. J., & Noah, S. A. M. (2016). BOWT: A hybrid text representation model for improving text categorization based on AdaBoost.MH. In *Multi-disciplinary trends in artificial intelligence: proceedings of MIWAI 2016, December 7–9, 2016* (pp. 3–11). Springer International Publishing.

Al-Salemi, B., Aziz, M. J. A., & Noah, S. A. (2015a). Boosting algorithms with topic modeling for multi-label text categorization: A comparative empirical study. *Journal of Information Science, 41*(5), 732–746.

Al-Salemi, B., Aziz, M. J. A., & Noah, S. A. (2015b). LDA-AdaBoost.MH: Accelerated AdaBoost.MH based on latent Dirichlet allocation for text categorization. *Journal of Information Science, 41*(1), 27–40.

Al-Salemi, B., Noah, S. A. M., & Ab Aziz, M. J. (2016). RFBoost: An improved multi-label boosting algorithm and its application to text categorisation. *Knowledge-Based Systems, 103*, 104–117.

Auer, P. (2002). Using confidence bounds for exploitation–exploration trade-offs. *Journal of Machine Learning Research, 3*(Nov.), 397–422.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research, 3*(Jan.), 993–1022.

Boutell, M. R., Luo, J., Shen, X., & Brown, C. M. (2004). Learning multi-label scene classification. *Pattern Recognition, 37*(9), 1757–1771.

Busa-Fekete, R., & Kégl, B. (2009). Accelerating AdaBoost using UCB. *Journal of Machine Learning Research—Proceedings Track, 7*, 111–122.

Busa-Fekete, R., & Kégl, B. (2010). *Fast boosting using adversarial bandits.* Paper presented at the 27th international conference on machine learning (ICML 2010), Haifa.

Busa-Fekete, R., Kégl, B., Éltető, T., & Szarvas, G. (2011). A robust ranking methodology based on diverse calibration of AdaBoost. In W. Daelemans, & K. Morik (Eds.), *Machine learning and knowledge discovery in databases* (pp. 263–279). Cham, Switzerland: Springer International Publishing.

Cheng, W., & Hüllermeier, E. (2009). Combining instance-based learning and logistic regression for multilabel classification. *Machine Learning, 76*(2–3), 211–225.

Datta, D., Varma, S., & Singh, S. K. (2017). Multimodal retrieval using mutual information based textual query reformulation. *Expert Systems with Applications, 68*, 81–92.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research, 7*, 1–30.

Elghazel, H., Aussem, A., Gharroudi, O., & Saadaoui, W. (2016). Ensemble multi-label text categorization based on rotation forest and latent semantic indexing. *Expert Systems with Applications, 57*, 1–11.

Elisseeff, A., & Weston, J. (2002). A kernel method for multi-labelled classification. In *Proceedings of the 14th international conference on neural information processing systems: Natural and synthetic* (pp. 681–687). Cambridge, MA: MIT Press.

Escudero, G., Mrquez, L., & Rigau, G. (2000). *Boosting applied to word sense disambiguation.* In *Proceedings of the 11th European conference on machine learning* (pp. 129–141). London: Springer-Verlag.

Esuli, A., Fagni, T., & Sebastiani, F. (2006). *MP-Boost: A multiple-pivot boosting algorithm and its application to text categorization,* Paper presented at the string processing and information retrieval conference, Glasgow.

Farid, D. M., Zhang, L., Rahman, C. M., Hossain, M. A., & Strachan, R. (2014). Hybrid decision tree and naïve Bayes classifiers for multi-class classification tasks. *Expert Systems with Applications, 41*(4), 1937–1946.

Forman, G. (2003). An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research, 3*(7–8), 1289–1305.

Forman, G. (2004). A pitfall and solution in multi-class feature selection for text classification. In *Proceedings of the twenty-first international conference on machine learning* (p. 38). New York, NY: ACM.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line

learning and an application to boosting. *Journal of Computer and System Sciences, 55*(1), 119–139. doi:10.1006/jcss.1997.1504.
Fürnkranz, J., Hüllermeier, E., Mencía, E. L., & Brinker, K. (2008). Multilabel classification via calibrated label ranking. *Machine Learning, 73*(2), 133–153.
Galavotti, L., Sebastiani, F., & Simi, M. (2000). *Experiments on the use of feature selection and negative evidence in automated text categorization.* Paper presented at the international conference on theory and practice of digital libraries, Berlin.
Gantz, J., & Reinsel, D. (2012). The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the Far East. *IDC iView: IDC Analyze the Future 2007*, 1–16.
Hüllermeier, E., Fürnkranz, J., Cheng, W., & Brinker, K. (2008). Label ranking by learning pairwise preferences. *Artificial Intelligence, 172*(16), 1897–1916.
Jiang, L., Li, C., Wang, S., & Zhang, L. (2016). Deep feature weighting for naive Bayes and its application to text classification. *Engineering Applications of Artificial Intelligence, 52*, 26–39.
Katrutsa, A., & Strijov, V. (2017). Comprehensive study of feature selection methods to solve multicollinearity problem according to evaluation criteria. *Expert Systems with Applications, 76*, 1–11.
Kumar, V., Pujari, A. K., Padmanabhan, V., Sahu, S. K., & Kagita, V. R. (2018). Multi-label classification using hierarchical embedding. *Expert Systems with Applications, 91*, 263–269.
Liu, J., Lin, Y., Lin, M., Wu, S., & Zhang, J. (2017). Feature selection based on quality of information. *Neurocomputing, 225*, 11–22.
Lu, H., Chen, J., Yan, K., Jin, Q., Xue, Y., & Gao, Z. (2017). A hybrid feature selection algorithm for gene expression data classification. *Neurocomputing, 256*, 56–62.
Mcauliffe, J. D., & Blei, D. M. (2007). Supervised topic models. In J. C. Platt, D. Koller, Y. Singer, & S. T. Roweis (Eds.), *Advances in neural information processing systems 20: proceedings of the 21st annual conference on neural information processing systems* (pp. 121–128). New York: Current Associates.
Mitchell, T. M. (1997). *Machine learning*. Maidenhead, UK: McGraw-Hill.
Moschitti, A., & Basili, R. (2004). Complex linguistic features for text classification: A comprehensive study. In S. McDonald, & J. Tait (Eds.), *Advances in information retrieval* (pp. 181–196). Berlin: Springer.
Onan, A., Korukoğlu, S., & Bulut, H. (2016). Ensemble of keyword extraction methods and classifiers in text classification. *Expert Systems with Applications, 57*, 232–247.
Pascoal, C., Oliveira, M. R., Pacheco, A., & Valadas, R. (2017). Theoretical evaluation of feature selection methods based on mutual information. *Neurocomputing, 226*, 168–181.
Pavlinek, M., & Podgorelec, V. (2017). Text classification method based on self-training and LDA topic models. *Expert Systems with Applications, 80*, 83–93.
Qian, W., & Shu, W. (2015). Mutual information criterion for feature selection from incomplete data. *Neurocomputing, 168*, 210–220.
Read, J., Pfahringer, B., Holmes, G., & Frank, E. (2011). Classifier chains for multi-label classification. *Machine Learning, 85*(3), 333–359.
Rehman, A., Javed, K., Babri, H. A., & Saeed, M. (2015). Relative discrimination criterion: A novel feature ranking method for text data. *Expert Systems with Applications, 42*(7), 3670–3681.
Schapire, R. E., Freund, Y., Bartlett, P., & Lee, W. S. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. *The Annals of Statistics, 26*(5), 1651–1686.
Schapire, R. E., & Singer, Y. (2000). BoosTexter: A boosting-based system for text categorization. *Machine Learning, 39*(2), 135–168.
Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys (CSUR), 34*(1), 1–47.
Shams, M., & Baraani-Dastjerdi, A. (2017). Enriched LDA (ELDA): Combination of latent Dirichlet allocation with word co-occurrence analysis for aspect extraction. *Expert Systems with Applications, 80*, 136–146.
Song, Q., Jiang, H., & Liu, J. (2017). Feature selection based on FDA and F-score for multi-class classification. *Expert Systems with Applications, 81*, 22–27.
Trstenjak, B., Mikac, S., & Donko, D. (2014). KNN with TF-IDF based framework for text categorization. *Procedia Engineering, 69*, 1356–1364.
Tsoumakas, G., & Vlahavas, I. (2007). Random k-labelsets: An ensemble method for multilabel classification. In J. Koronacki, R. L. de Mantaras, S. Matwin, D. Mladenič, & A. Skowron (Eds.), *Machine Learning: ECML 2007* (pp. 406–417). Berlin: Springer.
Uysal, A. K. (2016). An improved global feature selection scheme for text classification. *Expert Systems with Applications, 43*, 82–92.
Wang, G., & Lochovsky, F. H. (2004). Feature selection with conditional mutual information maximin in text categorization. In *Proceedings of the thirteenth ACM international conference on information and knowledge management* (pp. 342–349). Washington, DC: ACM.
Xu, J. (2012). An efficient multi-label support vector machine with a zero label. *Expert Systems with Applications, 39*(5), 4796–4804.
Yu, Y., Pedrycz, W., & Miao, D. (2014). Multi-label classification by exploiting label correlations. *Expert Systems with Applications, 41*(6), 2989–3004.
Zhang, C., Liu, C., Zhang, X., & Almpanidis, G. (2017). An up-to-date comparison of state-of-the-art classification algorithms. *Expert Systems with Applications, 82*, 128–150.
Zhang, M.-L., & Zhou, Z.-H. (2007). ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognition, 40*(7), 2038–2048.
Zhang, M.-L., & Zhou, Z.-H. (2014). A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering, 26*(8), 1819–1837.
Zhang, T., Liu, S., Xu, C., & Lu, H. (2011). Boosted multi-class semi-supervised learning for human action recognition. *Pattern Recognition, 44*(10), 2334–2342.
Zheng, Z., Wu, X., & Srihari, R. (2004). Feature selection for text categorization on imbalanced data. *ACM Sigkdd Explorations Newsletter, 6*(1), 80–89.