A Balanced Ensemble Approach to Weighting Classifiers for Text Classification

Gabriel Pui Cheong Fung[1], Jeffrey Xu Yu[1], Haixun Wang[2], David W. Cheung[3], Huan Liu[4]

[1] The Chinese University of Hong Kong, Hong Kong, China, {pcfung, yu}@se.cuhk.edu.hk
[2] IBM T. J. Watson Research Center, New York, USA, haixun@us.ibm.com
[3] The University of Hong Kong, Hong Kong, China, dcheung@cs.hku.hk
[4] Arizona State University, Arizona, USA, hliu@asu.edu

Abstract

This paper studies the problem of constructing an effective heterogeneous ensemble classifier for text classification. One major challenge of this problem is to formulate a good combination function, which combines the decisions of the individual classifiers in the ensemble. We show that the classification performance is affected by three weight components and they should be included in deriving an effective combination function. They are: (1) Global effectiveness, which measures the effectiveness of a member classifier in classifying a set of unseen documents; (2) Local effectiveness, which measures the effectiveness of a member classifier in classifying the particular domain of an unseen document; and (3) Decision confidence, which describes how confident a classifier is when making a decision when classifying a specific unseen document. We propose a new balanced combination function, called Dynamic Classifier Weighting (DCW), that incorporates the aforementioned three components. The empirical study demonstrates that the new combination function is highly effective for text classification.

1 Introduction

Let $\mathcal{U}$ be a set of unseen documents and $\mathcal{C}$ be a set of predefined categories. Automated text classification is the process of labeling $\mathcal{U}$ with $\mathcal{C}$, such that every $d \in \mathcal{U}$ will be assigned to some of the categories in $\mathcal{C}$. Note that d can be assigned to none of the categories in $\mathcal{C}$. If the number of categories in $\mathcal{C}$ is more than two ($|\mathcal{C}| > 2$), it is a multi-label text classification problem. Since every multi-label text classification problem can be transformed to a binary-label text classification problem, we focus on the binary problem in this paper ($|\mathcal{C}| = 2$). Let $c \in \mathcal{C}$. Binary-label text classification is to construct a binary classifier, denoted by $\Phi(\cdot)$, for c such that:

$$\Phi(d) = \begin{cases} 1 & \text{if } f(d) > 0, \\ -1 & \text{otherwise,} \end{cases} \quad (1)$$

where $\Phi(d) = 1$ indicates that d belongs to c and $\Phi(d) = -1$ indicates that d does not belong to it. $f(\cdot) \in \Re$ is a *decision function*. Every classifier, Φ_i, has its own decision function, $f_i(\cdot)$. If there are m different classifiers, there will be m different decision functions. The goal of constructing a binary classifier, $\Phi(\cdot)$, is to approximate the unknown true target function $\check{\Phi}(\cdot)$, so that $\Phi(\cdot)$ and $\check{\Phi}(\cdot)$ are coincident as much as possible [17].

In order to improve the effectiveness, ensemble classifiers (a.k.a classifier committee) were proposed [1, 3, 5, 6, 7, 8, 9, 15, 16, 17, 18, 19]. An ensemble classifier is constructed by grouping a number of member classifiers. If the decisions of the member classifiers are combined properly, the ensemble is robust and effective. There are two kinds of ensemble classifiers: *homogeneous* and *heterogeneous*.

A homogeneous ensemble classifier contains m binary classifiers in which all classifiers are constructed by the *same* learning algorithm. Bagging and boosting [19] are two common techniques [1, 15, 16, 18].

A heterogeneous ensemble classifier contains m binary classifiers in which all classifiers are constructed by *different* learning algorithms (e.g., one SVM classifier and one kNN classifier are grouped together) [19]. The individual decisions of the classifiers in the ensemble are combined (e.g., through stacking [19]):

$$\Theta(d) = \begin{cases} 1 & \text{if } g\big(\Phi_1(d), \Phi_2(d), \ldots, \Phi_m(d)\big) > 0, \\ -1 & \text{otherwise,} \end{cases} \quad (2)$$

where $\Theta(\cdot)$ is an ensemble classifier; $g(\cdot)$ is a combination function that combines the outputs of all $\Phi_i(\cdot)$. The effectiveness of the ensemble classifier, $\Theta(\cdot)$, depends on the effectiveness of $g(\cdot)$. In this paper, we concentrate on analyzing heterogeneous ensemble classifiers. Our problem is thus to examine how to formulate a good $g(\cdot)$.

Four widely used $g(\cdot)$ are: (1) Majority voting (MV) [8, 9]; (2) Weighted linear combination (WLC) [7]; (3) Dynamic classifiers selection (DCS) [3, 8, 6, 5]; and (4) Adaptive classifiers combination (ACC) [8, 9]. Except for MV, the other three functions assign different weights to the classifiers in the ensemble. The bigger the weight, the more ef-
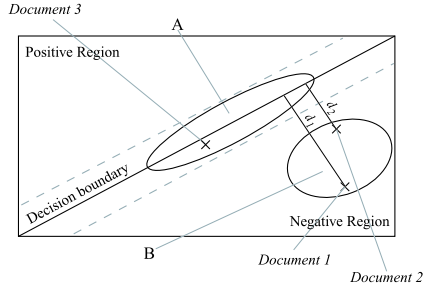
Figure 1. Illustration of local effectiveness and decision confidence.

fective is that classifier. In MV, all classifiers in the ensemble are equally weighted. It can end up with a wrong decision if the minority votes are significant. WLC assigns static weights to the classifiers based on their performance on a validation data. However, a generally well-performed classifier can perform poorly in some specific domains. For instance, the micro-F_1 scores of SVM and Naive Bayes (NB) for the benchmark *Reuters21578* are respectively 0.860 and 0.788. In this sense, SVM excels NB. Yet, for the categories *Potato* and *Retail* in *Reuters21578*, the F_1 scores for NB are both 0.667, but are both 0.0 for SVM. DCS and ACC weight the classifiers by partitioning the validation data (domain specific), they do not *combine* the classifiers' decisions, but *select* one of the classifiers from the ensemble and rely on it solely. We will show in the experiments that this will lead to inferior results.

In this paper, we propose a new combination function called Dynamic Classifiers Weighting (DCW). We consider three components when combining classifiers: (1) Global Effectiveness, which is the effectiveness of a classifier in an ensemble when it classifies a set of unseen documents; (2) Local effectiveness, which is the effectiveness of a classifier in an ensemble when it classifies the particular domain of the unseen document; and (3) Decision confidence, which is the confidence of a classifier in making a decision of the ensemble for a specific unseen document.

2 Motivations

Let $\Phi_1(\cdot), \Phi_2(\cdot), \ldots, \Phi_m(\cdot)$ be m different binary classifiers and $f_1(\cdot), f_2(\cdot), \ldots, f_m(\cdot)$ be their corresponding decision functions. Conceptually, $\Phi_i(\cdot)$ divides the entire domain into two parts according to $f_i(\cdot)$. Figure 1 illustrates this idea. The dashed lines are the decision boundaries. If the unseen document, d, falls into the upper (lower) triangle, it would be labeled as positive (negative). Usually, if d is further away from the decision boundary, the decision of d by $\Phi_i(d)$ is more confident.

Every classifier has different effectiveness. For instance, Support Vectors Machine (SVM) is being regarded as more accurate (effective) than Naive Bayes (NB) [20]). Although it does not imply *all* of the decisions made by SVM must be superior than NB, it does imply that we should value the judgment of SVM higher than that of NB in general. In this paper, we term this kind of effectiveness as *global effectiveness of a classifier*, denoted by α_i (E.g. $\alpha_{SVM} > \alpha_{NB}$). α_i gives us good insight about how to weight the classifiers in an ensemble. Intuitively, if we construct an ensemble classifier by grouping $\Phi_a(\cdot)$ and $\Phi_b(\cdot)$ together, where $\alpha_a > \alpha_b$, then we should value $\Phi_a(\cdot)$ higher than $\Phi_b(\cdot)$.

Yet, a globally effective classifier may sometimes perform poorly on some specific dataset (domain). As an example, consider two classifiers, SVM and NB. According to the benchmark *Reuters21578*, the micro-F_1 scores for SVM and NB are respectively 0.860 and 0.788. Unfortunately, the F_1 score for SVM when classifying *Retail* (*Retail* $\subset$ *Reuters21578*) is 0.0, but it is 0.667 for NB. As a result, an effective classifier may not always perform well in all domains (e.g., SVM performs poorly in *Retail*). This can be further illustrated in Figure 1. The two ovals, A and B, represent two different domains. Oval A covers over the decision boundary, whereas Oval B resides in the lower triangle. All of the documents within the domain of Oval A are aligned near the decision boundary. An unseen document that belongs to this domain may easily be classified wrongly. On the other hand, the documents within the domain of Oval B are well separated by the decision boundary. An unseen document that belongs to this domain will most likely be classified correctly. So, the effectiveness of the classifier also relies on the domain of the unseen data. We term this kind of effectiveness as *local effectiveness of the classifier*, denoted by β_i. β_i helps us to adjust the weights of the classifiers in the ensemble. If the α_i of Φ_i is very high but it is not effective in classifying the domain of the unseen document, we should re-consider its effectiveness.

For every decision a classifier makes, one may ask how confident the classifier is about the decision? Consider the two unseen documents, *document 1* and *document 2*, in the same domain (Oval B) in Figure 1. While both *document 1* and *document 2* reside near the boarder of their domain, *document 2* locates closer to the decision boundary (the dashed line) whereas *document 1* locates far away from it. Since both *document 1* and *document 2* belong to the same domain, the local effectiveness of the classifier upon them are the same. Yet, the confidence in making a correct decision for *document 1* should be higher than that of *document 2*, as *document 1* is further away from the decision boundary ($d_1 > d_2$). In this paper, we term it as *decision confidence*. It is estimated according to the distance between the unseen document and the decision boundary.

We summarize the needs for the above components as follows: if we ignore α_i, over-fitting may result as we neglect the combined influence of all domains. If we ignore

β_i, over-generalization may ensue as it relies on the domain where the unseen document appears. α_i and β_i do not measure the classifier's decision confidence, γ_i is proposed as it indicates how much confidence a classifier has when it classifies the unseen documents.

3 Dynamic Classifiers Weighting (DCW)

In the previous section, we have explained why the three weight components (α_i, β_i and γ_i) are helpful in constructing an effective combination function, $g(\cdot)$. We now describe how they are estimated and how they are combined in an ensemble classifier.

α_i is the effectiveness of the classifier when we use it to classify a set of unseen documents. During the training phase, although we do not have a set of labeled unseen documents, we can estimate α_i from the training data, $\mathcal{D}$: we estimate α_i by 10-folded cross validation. While our experience suggested that estimating the effectiveness of a classifier based on cross validation would always yield an optimistic result than evaluating it from the unseen data, this would not be a problem in our situation, as we are not targeting for evaluating the *real* global effectiveness of the classifiers, but aiming at obtaining the *relative* global effectiveness. We normalize α_i such that $0 < \alpha_i < 1$ and $\sum_{i=1}^{m} \alpha_i = 1$.

β_i is the effectiveness of the classifier when we use it to classify the domain of the unseen document, d. For an unseen document, we would never know what the true domain of d is. As above, we can only estimate its domain according to the training data, $\mathcal{D}$. Let D be a subset of documents in the training data, i.e., $D \subseteq \mathcal{D}$. We can find the domain of the unseen document, d, by using D, to extract the documents in D that are similar to d. Accordingly, the extraction of D is based on a nearest neighbor strategy. We extract the top n documents that are most similar to d from $\mathcal{D}$. The value n can be readily obtained through a validation dataset. The similarities among these n documents are measured by the cosine coefficient [13]. Since D is a subset of the training data ($D \subseteq \mathcal{D}$), we will know precisely the labels of those documents that appear in D. We estimate β_i by evaluating D using the F_1 score. β_i is normalized such that $0 < \beta_i < 1$ and $\sum_{i=1}^{m} \beta_i = 1$.

γ_i is a measure about how confident the classifier is when it makes a decision upon d. From Eq. (1), the classification decision of the classifier, $\Phi_i(\cdot)$, is based on the decision function, $f_i(\cdot)$. For most cases, if not all, the higher the magnitude of $f_i(\cdot)$, the more confident are their decisions. Consequently, we can compute γ_i by using the decision function, $f_i(\cdot)$. Unfortunately, the range of $f_i(\cdot)$ varies among different algorithms. For example, $\Phi_i(\cdot)$ may have $f_i(\cdot)$ in the range of $[-1, 1]$, whereas $\Phi_j(\cdot)$ may have another $f_j(\cdot)$ in the range of $(-\infty, +\infty)$. Since different decision functions have different ranges, a direct comparison among them is

inappropriate. We solve the problem as follows: Let D be the domain of the unseen document. D is obtained by the technique described previously. We compute γ_i as follows:

$$\gamma_i = \left| \frac{f_i(d)}{\mu_i} \right|, \quad (3)$$

$$\mu_i = \frac{1}{|D|} \sum_{d' \in D} f_i(d'), \quad (4)$$

where μ_i is the average confidence of the decisions made by $f_i(\cdot)$ among the documents in D. Since $D \subseteq \mathcal{D}$, we can presume that μ_i is non-zero. When $\gamma_i > 1$, $f_i(d)$, has more than average confidence to make a correct classification on d, where d will be far away from the decision boundary (e.g., *document 1* in Figure 1). When $\gamma_i < 1$, the decision function, $f_i(d)$, has less than average confidence to make a correct classification on d, where d will be closer to the decision boundary (e.g., *document 2* in Figure 1). We normalize γ_i such that $0 < \gamma_i < 1$ and $\sum_{i=1}^{m} \gamma_i = 1$.

We now present how α_i, β_i and γ_i are combined. Assume that there are m classifiers in the ensemble. In the most simplest form, the combination function, $g(\cdot)$ is:

$$g(\cdot) = \sum_{i}^{m} \text{decision}_i, \quad (5)$$

where $\text{decision}_i = \Phi_i(d) \in \{1, -1\}$ (Eq. (eq:c)). Here, all classifiers in the ensemble are equally weighted (i.e. MV). In DCW, since a confidence (γ_i) is associated with each decision_i, therefore:

$$g(\cdot) = \sum_{i}^{m} \text{decision}_i \times \gamma_i. \quad (6)$$

Yet, even for a confident decision, we need to review whether the classifier, which makes this decision, is effective in the ensemble. Consequently:

$$g(\cdot) = \sum_{i}^{m} \text{decision}_i \times \gamma_i \times \text{effectiveness}_i. \quad (7)$$

Since there are two kinds of effectiveness for each of the classifier (α_i and β_i), we have:

$$g(\cdot) = \sum_{i}^{m} \left(\Phi_i(d) \times \alpha_i \times \beta_i \times \gamma_i \right), \quad (8)$$

4 Experimental Study

The purpose of the experiments is twofold. (1) We want to examine how effective the Dynamic Classifiers Weighting (DCW) is, when it is compared with the other kinds of heterogeneous ensemble classifiers. As such, we implemented four existing ensemble classifiers for comparison:

No.	Combination	Reuters21578					Newsgroup20				
		MV	WLC	DCS	ACC	DCW	MV	WLC	DCS	ACC	DCW
1	S+N	–	0.874	0.859	0.852	**0.876**	–	0.817	0.761	0.794	**0.817**
2	S+R	–	0.883	0.862	0.874	**0.885**	–	0.800	0.762	0.793	**0.800**
3	S+K	–	0.862	0.862	0.843	**0.863**	–	0.813	0.763	0.780	**0.815**
4	R+N	–	0.833	0.831	0.821	**0.834**	–	0.762	0.738	0.759	**0.765**
5	K+N	–	0.824	0.827	0.820	**0.829**	–	0.780	0.746	0.769	**0.780**
6	K+R	–	0.825	**0.832**	0.821	0.831	–	0.762	0.764	0.760	**0.765**
7	S+K+R	0.872	0.879	0.862	0.876	**0.882**	0.776	0.815	0.763	0.812	**0.816**
8	S+K+N	0.855	0.874	0.859	0.865	**0.873**	0.783	0.819	0.762	0.809	**0.821**
9	S+R+N	0.852	0.872	0.861	0.856	**0.874**	0.777	**0.815**	0.761	0.801	**0.815**
10	K+R+N	0.857	0.825	0.837	0.823	**0.830**	0.775	0.782	0.750	0.775	**0.784**
11	S+K+R+N	–	0.851	0.861	0.859	**0.853**	–	0.720	0.762	0.761	**0.763**

Table 1. The results of the micro-F_1 for different ensemble classifiers.

Majority voting (MV) [8, 9], Weighted linear combination (WLC) [7], Dynamic classifiers selection (DCS) [3, 8, 6, 5], and Adaptive classifiers combination (ACC) [8, 9]. We report the results in Section 4.1. (2) We want to understand how significant the results are whenever one of the ensemble classifiers outperforms the others. As such, we performed a pairwise significant test in Section 4.2.

In the experiments, two benchmarks are used: *Reuters21578* and *Newsgroup20*. For *Reuters21578*, we separate the dataset into training data and testing data using the ModApte split [2]. For *Newsgroup20*, for each of the categories, we randomly select 80% of the postings as training data, and the remaining as testing data.

For the data preprocessing, punctuation, numbers, web page addresses, and email addresses are removed. All features are stemmed and converted to lower cases, and are weighted using the standard $tf \cdot idf$ schema [14]. Features that appear in only one document are ignored. All features are ranked based on the NGL Coefficient[12], and the top X features are selected. This X is tuned for different classifiers and for different benchmarks.

For creating the ensemble classifiers, different combinations of four kinds of classifiers are used: (1) Support Vectors Machine (SVM); (2) k-Nearest Neighbor (kNN); (3) Rocchio (ROC); (4) Naive Bayes (NB). Their default settings are as follows: For SVM, we use linear kernel with $C = 1.0$. No feature selection is required [4]. For kNN, we set $k = 50$ and select 2,750 and 4,900 features for *Reuters21578* and *Newsgroup20*. For ROC, we implement the version in [11] and selects 2,750 and 7,500 features for *Reuters21578* and *Newsgroup20*. For NB, we implement the multinomial version [10] and selects 2,750 and 9,500 features for *Reuters21578* and *Newsgroup20*.

4.1 Effectiveness Analysis

Table 1 shows the results of the micro-F_1 score for all ensemble classifiers (MV, WLC, DCS, ACC and DCW) when they are created using different combinations of the binary

classifiers for both benchmarks. The left most column denotes which of the binary classifiers are used for creating the corresponding ensemble classifier. We use S, K, R and N to denote SVM, kNN, Rocchio and Naive Bayes, respectively. For example, S+K+R represents an ensemble classifier which is comprised of SVM, kNN and Rocchio. Note that MV cannot be created if the number of binary classifiers in the ensemble is an even number, hence the "–" entries in Table 1.

At the first glance, the results are promising. DCW, the proposed approach, dominates over all other approaches when they are being created using the same set of binary classifiers. Similar results are obtained when we use the macro-F_1 score. The only case where DCW performs inferior is case 6 when DCW is created by kNN and Rocchio (K+R), meanwhile it is evaluated using *Reuters21578*. Its micro-F_1 is 0.831, which is 0.001 lower than DCS (Dynamic Classifiers Weighting). Nevertheless, such a difference can be negligibled.

Concerning DCW, the best combination of binary classifiers in the ensemble is SVM and Rocchio (case 2) for *Reuters21578*. The micro-F_1 score is 0.885. It is also the best results obtained among all of the ensemble classifiers that we have evaluated. For *Newsgroup20* the best result is obtained by comprising SVM, kNN and Rocchio together (case 8). The micro-F_1 score is 0.821. It is also the best result obtained among all approaches.

For MV, its philosophy is to take the majority agreement among the binary classifiers in the ensemble. Hence, the number of binary classifiers must be an *odd number*. So we can only create MV using three different binary classifiers. Interestingly, all combinations perform similarly.

Concerning WLC, the best combination for *Reuters21578* (case 2), its micro-F_1 score is 0.883, which is higher than all ensemble classifiers (*except* DCW). For *Newsgroup20*, similar observations are made, where its best combination is case 8. Although the idea of WLC is very simple – assigns static weights to the classifiers in the ensemble according to their global effectiveness and

combines them linearly – it performs surprisingly well. Another interesting finding is that when SVM is included in the ensemble, the effectiveness of WLC would be increased dramatically. This suggests that the choice of the classifiers in WLC is particularly important.

Concerning DCS, its best micro-F_1 score for *Reuters21578* (case 2) is 0.862 only. It is far lag behind all the other approaches. For *Newsgroup20*, none of the F_1 score is higher than 0.77. We believe that the reasons of why DCS performs poorly are because: (1) It does not *combine* the classifiers' decisions. Rather, it *selects* one of the classifier in the ensemble and relies on it completely. (2) It neither pays attention to the global effectiveness of the classifiers nor the decision confidence.

ACC performs slightly better than DCS. This may be because the decision strategy for ACC is more sophisticated that DCS. The best ensembles for *Reuters21578* and *Newsgroup20* are both case 7. However, these results are all inferior than both WLC and our DCW.

4.2 Significant Test

In this section, we conduct a pairwise comparison among them using the significant test [20]. Given two classifiers, $\Phi_A(\cdot)$ and $\Phi_B(\cdot)$, the significant test determines whether $\Phi_A(\cdot)$ performs better than $\Phi_B(\cdot)$ based on the errors that $\Phi_A(\cdot)$ and $\Phi_B(\cdot)$ made. Let N be the total number of the unseen documents, and $a_i = \{0,1\}$ ($b_i = \{0,1\}$) indicate whether $\Phi_A(\cdot)$ ($\Phi_B(\cdot)$) makes a correct classification upon the ith unseen document. $a_i = 0$ means $\Phi_A(\cdot)$ makes an incorrect classification whereas $a_i = 1$ means $\Phi_A(\cdot)$ makes a correct one. Similar definition is also applied to b_i. Let d_a be the number of times that $\Phi_A(\cdot)$ performs better than $\Phi_B(\cdot)$, and d_b be the number of times that $\Phi_B(\cdot)$ performs better than $\Phi_A(\cdot)$. In this test, the null hypothesis is that both classifiers perform the same ($H_0 : d_a = d_b$). The alternative is that $\Phi_A(\cdot)$ and $\Phi_B(\cdot)$ performs differently ($H_1 : d_a \neq d_b$).

Table 2 shows the results of comparing the performance of DCW with the other ensemble classifiers. $A \gg B$ means A performs significantly better than B (P-Value ≤ 0.01). $A > B$ means A performs slightly better than B. $A \sim B$ means no evidence indicates A and B has any differences in terms of the errors they made. A summary is given below:

Reuters21578: {DCW, WLC} > {MV, ACC} $\gg$ DCS

Newsgroup20: DCW > WLC > ACC $\gg$ MV $\gg$ DCS

5 Conclusions

In order to formulate an effective combination function for heterogeneous ensemble classifier, three weight components are necessary: Global Effectiveness, Local Effectiveness, and Decision Confidence. We compare DCW with

A	B	Reuters21578	Newsgroup20
MV	WLC	<	≪
MV	DCS	≫	≫
MV	ACC	∼	≪
MV	DCW	≪	≪
WLC	DCS	≫	≫
WLC	ACC	>	>
WLC	DCW	∼	<
DCS	ACC	≪	≪
DCS	DCW	≪	≪
ACC	DCW	<	≪

Table 2. Results of the significant test.

four other kinds of heterogeneous ensemble classifiers using two benchmarks. The results indicated that DCW can effectively balance the contributions of the three components and outperforms the existing approaches.

References

[1] W. W. Cohen and Y. Singer. Context-sensitive learning methods for text categorization. *ACM Transactions on Information Systems (TOIS)*, 17(2):141–173, 1999.

[2] F. Debole and F. Sebastiani. An analysis of the relative hardness of Reuters-21578 subsets. *Journal of the American Society for Information Science and Technology*, 56(6):584–596, 2004.

[3] G. Giacinto and F. Roli. Adaptive selection of image classifiers. In *Proceedings of the 9th International Conference on Image Analysis and Processing (ICIAP'97)*, pages 38–45, Florence, Italy, 1997.

[4] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of 10th European Conference on Machine Learning (ECML'98)*, pages 137–142, Chemnitz, Germany, 1998.

[5] K. B. Kevin Woods, W. Philip Kegelmeyer. Combination of multiple classifiers using local accuracy estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 19(4):405–410, 1997.

[6] W. Lam and K.-Y. Lai. A meta-learning approach for text categorization. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01)*, pages 303–309, New Orleans, Louisiana, USA, 2001.

[7] L. S. Larkey and W. B. Croft. Combining classifiers in text categorization. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'96)*, pages 289–297, Zurich, Switzerland, 1996.

[8] Y. H. Li and A. K. Jain. Classification of text documents. *The Computer Journal*, 41(8):537–546, 1998.

[9] R. Liere and P. Tadepalli. Active learning with committees for text categorization. In *Proceedings of 14th National Conference on Artificial Intelligence (AAAI'97)*, pages 591–596, Providence, Rhode Island, 1997.

[10] A. McCallum and K. Nigam. A Comparison of Event Models for Naive Bayes Text Classification. In *The 15th National Conference on Artificial Intelligence (AAAI'98) Workshop on Learning for Text Categorization*, 1998.

[11] A. Moschitti. A study on optimal parameter tuning for rocchio text classifier. In *Proceedings of the 25th European Conference on Information Retrieval Research (ECIR'03)*, pages 420–435, Pisa, Italy, 2003.

[12] H. T. Ng, W. B. Goh, and K. L. Low. Feature selection, perception learning, and a usability case study for text categorization. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'97)*, pages 67–73, Philadelphia, PA, USA, 1997.

[13] E. Rasmussen. Clustering algorithm. In W. B. Freakes and R. Baeza-Yates, editors, *Information Retrieval Data Structures & Algorithms*, pages 419–442. Prentice Hall PTR, 1992.

[14] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management (IPM)*, 24(5):513–523, 1988.

[15] R. E. Schapire and Y. Singer. BoosTexter: a boosting-based system for text categorization. *Machine Learning*, 39(2–3):135–168, 2000.

[16] R. E. Schapire, Y. Singer, and A. Singhal. Boosting and Rocchio applied to text filtering. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98)*, pages 215–223, Melbourne, Australia, 1998.

[17] F. Seabastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[18] S. M. Weiss, C. Apte, F. J. Damerau, D. E. Johnson, F. J. Oles, T. Goetz, and T. Hampp. Maximizing text-mining performance. *IEEE Intelligent Systems*, 14(4):63–69, 1999.

[19] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, second edition, 2005.

[20] Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*, pages 42–49, Berkeley, California, USA, 1999.