Multiple Support Vector Machines for Binary Text Classification Based on Sliding Window Technique

Aisha Rashed Albqmi[1,2(✉)], Yuefeng Li[1], and Yue Xu[1]

[1] School of EECS, Queensland University of Technology, Brisbane, QLD, Australia
a.albqmi@hdr.qut.edu.au, {y2.li,yue.xu}@qut.edu.au
[2] Department of CS, Taif University, Taif, Saudi Arabia

Abstract. Supervised machine learning algorithms, such as support vector machines (SVMs), are widely used for solving classification tasks. In binary text classification, linear SVM has shown remarkable efficiency for classifying documents due to its superior performance. It tries to create the best decision boundary that enables the separation of positive and negative documents with the largest margin hyperplane. However, in most cases there are regions in which positive and negative documents are mixed due to the uncertain boundary. With an uncertain boundary, the learning classifier is more complex, and it often becomes difficult for a single classifier to accurately classify all unknown testing samples into classes. Therefore, more innovative methods and techniques are needed to solve the uncertain boundary problem that was traditionally solved by non-linear SVM. In this paper, multiple support vector machines are proposed that can effectively deal with the uncertain boundary and improve predictive accuracy in linear SVM for data having uncertainties. This is achieved by dividing the training documents into three distinct regions (positive, boundary, and negative regions) based on a sliding window technique to ensure the certainty of extracted knowledge to describe relevant information. The model then derives new training samples to build a multiple SVMs based classifier. The experimental results on the TREC topics and standard dataset Reuters Corpus Volume 1 (RCV1), indicated that the proposed model significantly outperforms six state-of-the-art baseline models in binary text classification.

Keywords: Support Vector Machines · Binary text classification · Uncertain boundary · Sliding window technique

1 Introduction

The massive amounts of unstructured data sorted in public resources continue to increase. In order to organize and manage this data, the use of efficient and successful methods must be considered. Text classification is an active technique for information organization and management [1]. Different methods and algorithms have been developed for text classification including Support Vector Machines (SVM) [2], Naive Bayes probabilistic Classifier (NB) [3], Rocchio Similarity [4], K-Nearest Neighbour (KNN) [5], and C4.5 integration Decision Trees [1].

R. Islam et al. (Eds.): AusDM 2018, CCIS 996, pp. 17–29, 2019.
https://doi.org/10.1007/978-981-13-6661-1_2

Binary classification is a key type of text classification with two predefined categories, namely, relevant or irrelevant classes [6], on which our research focuses. A binary text classifier determines a decision boundary to classify documents into two groups: positive and negative classes [7]. However, drawing a clear boundary between the positive and negative classes of text documents is not easy for a classic binary text classifier [8, 9].

The solution of classification issues using SVM, which was proposed by Vapinik in 1995, has gained increasing recognition and popularity among researchers due to its ability to handle high dimensional data such as textual documents [10, 11]. SVM performs classification by finding a decision boundary (separating hyperplane) that partitions the feature space into two distinct classes of data, positive and negative, with the maximum margin and represents the decision boundary using a set of support vectors (SV) generated from the training dataset [12, 13]. However, it is difficult for an SVM classifier to deal with non-separable data because the margin between positive and negative objectives is still unclear. In such situations, due to the uncertainty, an SVM classifier might not be completely effective in providing the optimal classification.

In practical problems, most training datasets include uncertainties. With an uncertain boundary, the learning classifier is more complex and difficult to find the optimal line to classify related objects and a full separation of relevant and irrelevant documents would require a curve. However, it is not easy to achieve the curve in a direct way with high precision because it requires too much computation [8]. Even if this were possible, there is no guarantee that it can be applied to completely classify all unknown testing samples because of the differences between training and testing document sets [9]. Thus, a nonlinear classifier is inefficient for a prediction task where an uncertain boundary exists in the training set. It is, therefore, desirable to design a classifier model able to linearly cope with non-separable data. Therefore, how to cope with data having uncertainties into the learning phase to improve the performance of binary classifier is a challenging problem.

This paper aims to present an effective binary classification model, called the Multiple-SVMs with Sliding Window model (MSVMs-SW model), in order to overcome the limitations in the existing classifiers and achieve the best performance in linear SVM for data having uncertainties. Different from traditional binary classifiers, the MSVMs-SW model aims to understand uncertainty by partitioning training samples (with two labels) into three regions, namely, positive, boundary, and negative regions in order to understand the decision boundary. Allowing this partitioning of the training set can help to describe relevant and non-relevant information and support to design a multiple-SVMs based classifier. We developed three different SVM classifiers (SVM_P, SVM_N, and SVM_B), each of which is trained using its own training set that is derived by using the three regions. The training set for each classifier was different in order to obtain a greater improvement of the prediction results, to increase the certainty of all objects in positive and negative regions and to resolve the uncertainty in boundary region. The main motivation for using multiple-SVMs to classify new incoming documents is that a problem which requires expert knowledge will be better solved by a committee of experts rather than by a single expert [6]. Therefore, this research made *three innovative contributions* to the fields of text classification: (a) A new and effective model that deals with the uncertain decision boundary for text classification.

Our proposed model uses a training set with only minimal experimental parameters to identify the uncertain boundary, which makes it efficient; (b) An alternative solution for the hard uncertain boundary problem that was traditionally solved by non-linear SVMs; (c) A structure to guide the design of a fusion of multiple classifiers. To measure the effectiveness of the proposed model, extensive experiments were conducted, based on the RCV1 dataset and TREC assessors' relevance judgements. The results show a significant improvement on F_1 and *Accuracy* in the performance of binary text classification.

2 Related Work

Automated binary text classification is a significant research problem in information filtering and information organization fields [15]. It provides a way to determine a decision boundary that classifies textual documents into two distinct classes: relevant or irrelevant. Several approaches to binary text categorization, such as NB, KNN, decision tree, Rocchio, and SVM, have been developed to identify an efficient way to separate all related documents from a large dataset to determine a clear boundary between the classes in the text dataset [1]. However, in practice, the decision boundary includes much uncertainty because of the limitation of traditional machine learning algorithms, the presence of noise in text documents and feature scalability [16, 17].

SVM represents the training dataset as vectors, where each vector comprised of its words with their frequencies, and then tries to locate the linear hyperplane which separates two classes [13]. SVM can solve linear and nonlinear classifications and works well when applied to many practical problems [18, 19]. Although nonlinear SVM is effective when classifying nonlinear data, it has much higher computational complexity than linear SVM when making predictions for sparse data [19]. In addition, linear SVM performs better than nonlinear SVM when the number of features is very high, for example, in document classification [20, 21]. Therefore, if the number of features is extremely large, it is better to select linear SVM, due to the difficulty in finding the optimal parameters of a classifier when using nonlinear SVM [22]. Because linear SVM still has no effective way to deal with the uncertain factors it is, therefore, desirable to have a classifier model with the efficiency of a linear classifier to deal with data having uncertainty. The linear SVM is chosen in this study due to its computational and algorithmic simplicity.

The above limitations can be alleviated by employing the SW technique to divide the training set into three regions based on scores that present their degree of relevance and then to design a multiple-SVMs based classifier in order to derive a linear decision boundary for each classifier. In our proposed model the SW technique can be optimized by using Entropy. The entropy measurement is chosen in this research because it is a commonly understood measure in information theory and it is a fundamental measure for describing randomness and uncertainty of data [14, 23].

3 Description of MSVMs-SW Model

The MSVMs-SW model attempts to use the training dataset effectively to deal with the probable uncertainty and to improve the accuracy of the classifier. Our proposed model uses SVM as a high-performance classifier and generates new training set by dividing a universal set of documents into three disjointed parts (the positive region (POS), the boundary region (BND), and the negative region (NEG)). However, a single SVM may not be sufficient to classify all unknown testing samples. Therefore, we propose to use a multiple-SVMs based classifier. The proposed model contains two stages, a training stage and a testing stage, as shown in Fig. 1.

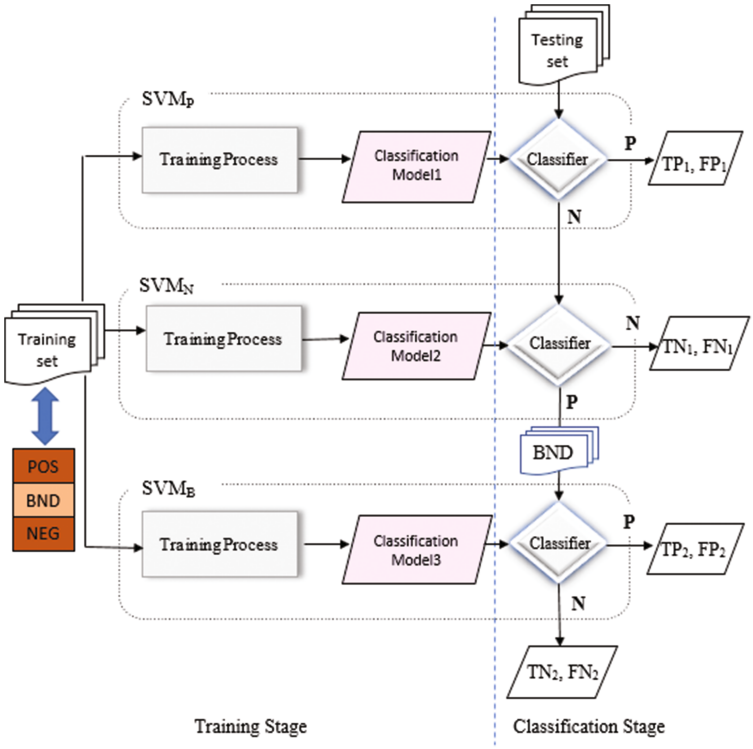


Fig. 1. Architecture of a multiple SVMs classifier

3.1 Training Stage of MSVMs-SW Model

To achieve the best performance in binary classification, the objective is to determine a decision boundary between classes. Our proposed model uses the training set only to set the decision boundary and to explore the uncertainty situation as shown in Fig. 2. It starts with the calculation of the score of training documents, and further regroups the training samples into three regions using the SW technique.

Document Scoring. Scoring documents to indicate their importance is an effective way for ranking relevant information. For a collection of documents in the datasets consisting of two sets (positive document sets, D^+; and negative document sets, D^-), the MSVMs-SW model calculates the weight of terms extracted from D^+ and ranks them to use the *top-k* features based on their values, for example, $T = \{t_1, t_2, t_3, \ldots, t_k\}$. However, identifying the value of k is experimental. In our proposed model, we use the Okapi BM25 as a term weighting function. BM25 is a probabilistic state-of-the-art retrieval model [24], which can be calculated as follows:

$$w(t) = \frac{tf.(k_1+1)}{k_1.\left((1-b)+b\frac{DL}{AVDL}\right)+tf}.\log\frac{\frac{(r+0.5)}{(n-r+0.5)}}{\frac{(R-r+0.5)}{(N-n-R+r+0.5)}} \tag{1}$$

where N is the total number of training documents; R is the number of relevant documents; n is the number of documents which contain the term t; r is the number of relevant documents which contain the term t; tf is the term frequency; DL and $AVDL$ are the document length and average document length, respectively; and k_1 and b are the tuning parameters.

The reason for using the BM25 to calculate term weight is that the BM25 is a probabilistic model and in binary text classification we deal with uncertain information [24]. Probability is the measure used to understand the uncertainty in the information. Therefore, probability theory is the best way to quantify uncertainties. Next, the weighted terms are used to calculate the scores for all training documents $d \in D$ as follows:

$$score(d) = \sum\nolimits_{t\in T} w(t)\cdot\tau(t,d) \tag{2}$$

where $w(t) = \text{BM25}(t, D^+)$; and $\tau\ (t, d) = 1$ if $t \in d$; otherwise $\tau\ (t, d) = 0$.

Once the scores of the documents are calculated, the documents are ranked in descending order based on their scores.

Sliding Window Technique. After ranking the training documents in the previous step, the most related documents will be located at the top of the list, while irrelevant ones will be located at the bottom of the ranked list, as shown in Fig. 2 (step 1). However, in most cases there are regions in which positive and negative documents are mixed due to the uncertain boundary. To find this area with many noisy documents, a sliding window technique and entropy are used to effectively determine the boundary region. Ko and Seo [25] used entropy and a sliding window to remove noisy data and solve the problem of the One-Against-All method. Our proposed model extends this idea to use a sliding window and entropy measurement to construct the decision boundary.

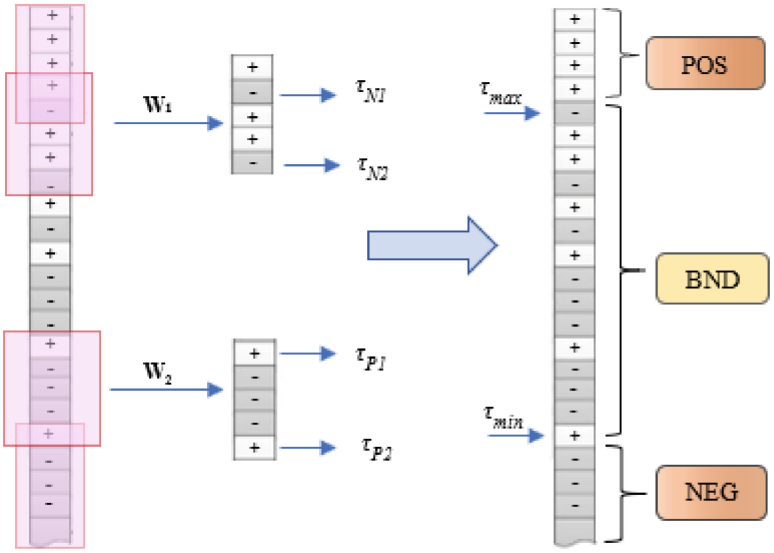
In this research, the sliding window was used to identify the boundary values which denote the region with the highest rate of noisy documents [25, 26]. The window size in this paper was set to 5 documents. The model starts to slide the window from the top documents in the ranked list, and then calculates the entropy value for the window. The window then slides over one document and yields a new entropy value. It continues to

slide and stop when the entropy is greater than the threshold. We chose a high entropy threshold (95%). The same process applies from the bottom of the ranked list as shown in Fig. 2 (step 1).

Entropy Algorithm. Entropy is commonly used to define the uncertainty of variable [23, 26]. In this paper, for each sliding window(s), the entropy value can be calculated using the following function based on the number of positive and negative documents as follows:

$$E(s) = -\left[\frac{P}{P+N}\log_2\left(\frac{P}{P+N}\right) + \frac{N}{P+N}\log_2\left(\frac{N}{P+N}\right)\right] \tag{3}$$

where P and N are the numbers of positive and negative documents in SW, respectively.



Step 1: A sliding window technique over ranked documents

Step 2: Identify the boundary values and three regions.

Fig. 2. Decision Boundary Setting

Next, we select two windows with the greatest degree of entropy value. The first window (W_1) is from the top of the list and the second window (W_2) is from the bottom of the list. For W_1, the irrelevant documents are denoted as τ_N. For W_2, relevant documents are denoted as τ_P. In this study, the values of the boundary are calculated based on the scores of the relevant documents (τ_p) and the irrelevant documents (τ_N);

we selected the highest score of irrelevant documents in W_1 as a maximum threshold (τ_{max}), and the lowest score of relevant documents in W_2 as a minimum threshold (τ_{min}), as shown in Fig. 2 (step 2). Hence, the upper and lower decision boundary values τ_{max} and τ_{min} are calculated as follows:

$$\tau_{\max} = \max_{d_i \in D^- \cap W_1} \{score(d_i)\} \tag{4}$$

$$\tau_{\min} = \min_{d_i \in D^+ \cap W_2} \{score(d_i)\} \tag{5}$$

Three Regions for Partitioning the Training Set. The SVMs-SW model aims to group training sets into three regions rather than two classes. The training set D can be split into three regions based on the document scores and threshold settings in the previous step: the positive region (POS, possible relevant); the boundary region (BND, uncertain); and the negative region (NEG, possible irrelevant). The ranges of these regions are defined as follows:

$$\begin{aligned} POS &= \{d \in D | score(d) > \tau_{\max}\} \\ BND &= \{d \in D | \tau_{\min} \leq score(d) \leq \tau_{\max}\} \\ NEG &= \{d \in D | score(d) < \tau_{min}\} \end{aligned}$$

The boundary region *BND* contains many relevant and irrelevant documents under uncertain decisions which can be divided into two subsets: $B^+ = BND \cap D^+$ and $B^- = BND \cap D^-$.

Design Multiple SVMs Based Classifier. Building a classifier is achieved by training the SVM using chosen training documents via three regions. As shown on the left side of Fig. 1, we constructed three different SVMs classifiers; SVM_P, SVM_N, and SVM_B. To explain this process, the Algorithm 1 describes the training stage to learn the classifiers. The First classifier, SVM_P (step 8), takes strong positive documents *POS* and all negative documents ($B^- \cup NEG$) as input, and uses the SVM classifier to build a predication model. The SVM_P generates the hyperplane between *POS* and ($B^- \cup NEG$) to maintain the maximum margin between them. However, a potential problem with this approach can arise when the number of training samples in the *POS* part is very low and, in this case, the boundary of class would not be accurate due to insufficient positive training samples provided for text classification. To overcome this issue, we use a *pseudo feedback* technique. We selected the *top-k* scoring documents from the unlabeled testing set U and add them to the *POS* part as shown in step 1 to step 6. Different numbers of *top-k* have been tested and we found that using 5 documents improved the performance compared with using $k > 5$, which reduced the performance.

The second classifier, SVM_N, is constructed from the all positive documents ($POS \cup B^+$) and strong negative documents *NEG*, as in step 9. For SVM_B, it is difficult to construct a classifier from the documents in the boundary region because SVM is

very sensitive to noise, especially when noise is large and, in this case, the classifier will be very poor. Therefore, for even better classification we used the strong positive and negative samples (*POS, NEG*) to build SVM_B in our model, as in step 10.

Algorithm1: Multiple SVMs classifier Learning

Input: POS, NEG, BND; and parameter k;
Unlabelled document in testing set, U;
SVM classification model;

Output: SVM classification models, SVM_P, SVM_N, SVM_B;

// Add *top-k* unlabelled documents to build first classifier

1 let $n = |U|$
2 **for** each $d \in U$ **do**
3 $score(d) = \sum_{t \in T} weight(t).\ \tau(t, d)$
4 **end**
5 let $P = \{d_1, d_2, ..., d_n\}$ in descending ranking order,
6 $D_P = \{d_i \mid d_i \in U, 1 < i \leq k\}$;
// Learn training dataset using SVM classifier, get SVM models.
7 $B^+ = BND \cap D^+$, $B^- = BND \cap D^-$;
8 $SVM_P = Classifier_{SVM}\ (POS \cup D_P, B^- \cup NEG)$;
9 $SVM_N = Classifier_{SVM}\ (POS \cup B^+, NEG)$;
10 $SVM_B = Classifier_{SVM}(POS, NEG)$;

3.2 Testing Stage of MSVMs-SW Model

In this phase, each stage has a different classification model, as shown on the right side of Fig. 1. The SVM_P classification model concentrates on identifying positive documents. In this stage, the documents that are classified as positive are denoted by TP_1 (true positive one) if they are true positive or grouped as FP_1 (false positive one) if they are actually negative. The objective of this stage is to achieve a high precision rate for positive documents and to minimize the *FP* rate, with an acceptable False Negative rate *FN*. The SVM_N classifier, which is generated in stage two, is applied to classify the documents that were predicted as negative in stage one. This stage focuses on increasing the precision rate for negative documents. In this stage, the documents that are classified as negative are denoted by TN_1 (true negative one) if they are negative or grouped into the FN_1 if they actually are positive. However, as the documents that were predicted as positive in this stage are still uncertain, the classifier will collect them into the boundary set *BND*. To classify these documents, we used the final classifier, SVM_B. This classifier can then assign those documents as positive or negative and produce four outputs, namely, TP_2, FP_2, TN_2, *and* FN_2. In our proposed classifier model, true positive $TP = TP_1 + TP_2$, false positive $FP = FP_1 + FP_2$, true negative $TN = TN_1 + TN_2$, and false negative $FN = FN_1 + FN_2$, as listed in Algorithm 2.

Algorithm 2: Multiple SVMs Classifier Testing

Input: Incoming document without label for testing, U;
SVM classification models, SVM_P, SVM_N, SVM_B;

Output: Positive documents *POS*, and negative documents *NEG*;

1 $POS := NEG := BND := \emptyset$;
2 **for** each $d \in U$ **do**
// Predict label of new documents($d_{unlabeled}$) using SVM_P.
3 $d_{labeled} = SVM_P\ (d_{unlabeled}, Model_1)$;
// If SVM_P label it as positive, the label of document is positive
4 **if** $d_{labeled}$ *is positive* **then**
5 $POS = POS \cup \{d\}$;
6 **else**
7 $d_{labeled} = SVM_N\ (d_{unlabeled}, Model_2)$;
// If SVM_N label it as negative, the label of document is negative
8 **if** $d_{labeled}$ *is negative* **then**
9 $NEG = NEG \cup \{d\}$;
10 **else**
11 $BND = BND \cup \{d\}$;
end
end
// Predict label the rest documents in *BND* using SVM_B.
12 **for each** $d \in BND$ **do**
13 $d_{labeled} = SVM_B\ (d_{unlabeled}, Model_3)$
14 **if** $d_{labeled}$ *is positive* **then**
15 $BND = BND - \{d\}$; $POS = POS \cup \{d\}$;
16 **else**
17 $BND = BND - \{d\}$; $NEG = NEG \cup \{d\}$;
end

4 Experiments and Evaluation

4.1 Dataset and Evaluation Metrics

To evaluate the performance of our proposed model, the RCV1 dataset, which consists of 100 topics, was used. Each topic has been divided into training and testing sets with relevance judgements. The RCV1 corps has more than 804,000 documents which are news stories in English published by Reuters journalists [27]. These documents are grouped into 100 collection with 100 different topics. However, in our experiments in this study, we used the first 50 topics where the experiments are more reliable.

Three evaluation metrics were used to measure the effectiveness of the MSVMs-SW model and the baselines. The measures are the F_1-score and *Accuracy*. These evaluation metrics are widely used in text classification research. For more details of these measures refer to [6]. We also used the t-test *p-values* to analyse the significance of the difference between the results of the MSVMs-SW model and the baselines.

4.2 Baseline Models and Settings

In order to make an extensive evaluation, we compared our proposed model with six different baseline models. These models are the state-of-the-art influential models, which include statistical methods *libSVM,* SVMperf [28], J48 [29], NB [3], IBk (Instance-Based Learning), and Rocchio. All six models were trained and tested with the same dataset to conduct the experiments. They were also run with their best settings obtained through experimental practice. For *libSVM*, some default setting were utilized because the F_1-scores of the classifier are low when using the default setting. Different types of kernel functions and values of C were conducted, and we found that if we set $k = 0$ (linear kernel) and $C = 1$, we could get better results. In addition, we set $C = 10$ in SVMperf as it is the best value recommended in [9]. For our proposed model we used the linear kernel because it is quick and efficient with very large numbers of features as in document classification. For the experimental parameters of the BM25, k_1 and b values were set at 1.2 and 0.75, respectively.

4.3 Experimental Results

The experimental results of the MSVMs-SW and the baseline models are presented in Table 1. These results are the average of the 50 collections of the RCV1 dataset. The comparison between the proposed model, MSVMs-SW model, and other six baseline models was completed using two measures, F_1 and *Accuracy*. The results in Table 1 have been categorized into two groups. The first group includes two *SVM* models (*libSVM* and SVMperf); the second group includes a popular influential classifier.

Table 1 shows that our proposed model outperformed all baseline models for text classification. Compared to the SVM models, the MSVMs-SW was significantly better

Table 1. Evaluation results of our model compared with the baselines.

Models	F_1	Accuracy
MSVMs-SW model	**0.4157**	**0.8621**
libSVM	0.3271	0.8557
SVMperf	0.2864	0.8001
improvement%	**+36.1%**	**+4.3%**
J48	0.3449	0.8263
Naïve Bayes	0.1851	0.8131
IBk	0.2970	0.8404
Rocchio	0.3681	0.5646
improvement%	**+49.5%**	**+16.4%**

on average with a minimum improvement of 4.3% and a maximum improvement of 36.1%. Compared to the IBK model, which has the highest *Accuracy* value in the second group, F_1 and *Accuracy* of the MSVM-SW model were significantly improved by 40.1% and 2.6%, respectively.

The t-test *p-values* evaluation in Table 2 also indicated that the proposed model is extremely statistically significant with a $p\text{–}value < 0.05$, compared with other baseline models on both F_1 and *Accuracy* for both one-tail and two-tails.

In order to test the effectiveness of using multiple-SVMs in our proposed model, we performed the same experiments with a single SVM classifier which used the original training set. The aims of using multiple-SVMs was to provide a way to make the decision boundary better. Therefore, we tried to separate the uncertain boundary to identify a clear boundary for both relevant and irrelevant parts. Table 3 shows the results of the performance of a single SVM classifier and multiple-SVMs on the RCV1 dataset. We used the *precision*, F_1-measure and *Accuracy* as measures for comparison.

Table 2. The p-values (one/two-tails) of the baseline models in comparison with MSVMs-SW model on RCV1.

No	Models	Tail(s)	F_1	Accuracy
1	libSVM	one	0.001908	0.337803
		two	0.0038162	0.675605
2	SVMperf	one	1.96E-05	2.42E-07
		two	3.91E-05	4.85E-07
3	J48	one	1.08E-14	1.24E-16
		two	2.15E-14	2.49E-16
4	Naïve Bayes	one	4.84E-09	0.000343
		two	9.67E-09	0.000686
5	IBk	one	7.85E-05	0.003296
		two	0.000157	0.006591
6	Rocchio	one	0.041785	4.71E-15
		two	0.083571	9.42E-15

In Table 3 we found that using multiple-SVMs achieved an average increase of 30.4% for F_1. When considering precision value, multiple-SVMs showed the best performance, especially for the relevant part (Precision$^+$) with 7.8% improvement on average. It is clear that using multiple-SVMs instead of a single one can lead to better classification and improve the overall accuracy with data having uncertainty.

Based on the results presented earlier, the MSVMs-SW model improved the binary classification with the highest score in both F_1 and *Accuracy* (and particularly in F_1) that best expresses the real situation in text classification.

Table 3. Multiple-SVMs Results compared with single SVM on RCV1.

Models	$Precision^{+}$	$Precision^{-}$	F_1	*Accuracy*
MSVMs-SW model (multiple-SVMs)	**0.5407**	**0.8767**	**0.4157**	**0.8621**
Single SVM	0.5016	0.8623	0.3187	0.8543
Improvement %	+7.8%	+2.0%	+30.4%	+1.0%

5 Conclusion

The MSVMs-SW model was proposed to deal with data having uncertainty, a situation in which is difficult to obtain good results when using non-linear SVM. This model uses the training set effectively to achieve super machine learning with high classification accuracy and to improve the performance of binary text classification. It tries to understand uncertainty by dividing the training set into three regions, namely, positive, negative, and boundary, in order to improve the certainty of both relevant and irrelevant parts and to reduce the impact of uncertainty in the boundary part. The partition of training sets was achieved by applying an effective SW technique and threshold setting and then organizing training samples to generate new training sets. After the boundary region was identified, we used multiple-SVMs instead of a single one to learn the classifiers and to classify new incoming documents. The experimental results using the standard RCV1 dataset show that the proposed model achieved significant improvements in F_1 and *Accuracy*, especially F_1, and outperforms existing classifiers, including state-of-the-art classifiers.

References

1. Lata, S., Loar, M.R.: Text clustering and classification techniques using data mining. Int. J. on Futur. Revolut. Comput. Sci. Commun. Eng. **4**(4), 859–864 (2018)
2. Joachims, T.: Transductive inference for text classification using support vector machines. In: ICML 1999, pp. 200–209. ACM, San Francisco (1999)
3. John, G.H., Langley, P.: Estimating continuous distributions in Bayesian classifiers. In: UAI 1995, pp. 338–345. ACM, Canada (1995)
4. Aggarwal, C.C., Zhai, C.: A Survey of Text Classification Algorithms. In: Mining Text Data, pp. 163–222. Springer, Boston, (2012). https://doi.org/10.1007/978-1-4614-3223-4_6
5. Cover, T., Hart, P.: Nearest neighbor pattern classification. IEEE Trans. Inf. Theory **13**(1), 21–27 (1967)
6. Sebastiani, F.: Machine learning in automated text categorization. ACM Comput. Surv. **34** (1), 1–47 (2002)
7. Forman, G.: An extensive empirical study of feature selection metrics for text classification. J. Mach. Learn. Res. **3**, 1289–1305 (2003)
8. Zhang, L., Li, Y., Bijaksana, M.A.: Decreasing uncertainty for improvement of relevancy prediction. In: Proceeding of the Twelfth Australasian Data Mining Conference, AusDM 2014, Brisbane, pp. 157–162 (2014)
9. Li, Y., Zhang, L., Yue, X., Yiyu, Y., Raymond, L., Yutong, W.: Enhancing binary classification by modeling uncertain boundary in three-way decisions. IEEE Trans. Knowl. Data Eng. **29**(7), 1438–1451 (2017)

10. Wardaya, P.D.: Support vector machine as a binary classifier for automated object detection in remotely sensed data. In: IOP Conference Series: Earth and Environmental Science, vol. 18, no. 1. IOP Publishing, Bristol (2014)
11. Wei, L., Wei, B., Wang, B.: Text classification using support vector machine with mixture of Kernel. J. Softw. Eng. Appl. **5**(12), 55–58 (2012)
12. Joachims, T.: Text categorization with support vector machines: learning with many relevant features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0026683
13. Burges, C.J.: A tutorial on support vector machines for pattern recognition. Data Min. Knowl. Disc. **2**(2), 121–167 (1998)
14. Shannon, M.: Forensic relative strength scoring: ASCII and entropy scoring. Int. J. Digit. Evid. **2**(4), 1–19 (2004)
15. Lau, R.Y., Bruza, P.D., Song, D.: Towards a belief-revision-based adaptive and context-sensitive information retrieval system. ACM Trans. Inf. Syst. (TOIS). **26**(2), 1–38 (2008)
16. Bekkerman, R., Gavish, M.: High-precision phrase-based document classification on a modern scale. In: KDD 2011, pp. 231–239. ACM, San Diego (2011)
17. Li, Y., Algarni, A., Zhong, N.: Mining positive and negative patterns for relevance feature discovery. In: KDD 2010, pp. 753–762. ACM, New York (2010)
18. Fu, Z., Robles-Kelly, A., Zhou, J.: Mixing linear SVMs for nonlinear classification. IEEE Trans. Neural Netw. **21**(12), 1963–1975 (2010)
19. Rodriguez-Lujan, I., Cruz, C.S., Huerta, R.: Hierarchical linear support vector machine. Pattern Recogn. **45**(12), 4414–4427 (2012)
20. Gao, Y., Sun, S.: An empirical evaluation of linear and nonlinear kernels for text classification using support vector machines. In: FSKD 2010, pp. 1502–1505. IEEE, Yantai (2010)
21. Lan, M., Tan, C.L., Low, H.B.: Proposing a new term weighting scheme for text categorization. In: AAAI 2006, pp. 763–768. ACM, Boston (2006)
22. Hsu, C.W., Chang, C.C., Lin, C.J.: A practical guide to support vector classification. Technical Report, Department of Computer Science, National Taiwan University, Taipei (2003)
23. Du, L., Song, Q., Jia, X.: Detecting concept drift: an information entropy based method using an adaptive sliding window. Intell. Data Anal. **18**(3), 337–364 (2014)
24. Robertson, S., Zaragoza, H.: The Probabilistic Relevance Framework: BM25 and Beyond. Now Publishers Inc., Breda (2009)
25. Ko, Y.J., Seo, J.Y.: Issues and empirical results for improving text classification. J. Comput. Sci. Eng. **5**(2), 150–160 (2011)
26. Hall, G.A.: Sliding window measurement for file type identification. Technical Report, ManTech Security and Mission Assurance (2006)
27. Lewis, D.D., Yang, Y., Rose, T.G., Li, F.: Rcv1: a new benchmark collection for text categorization research. J. Mach. Learn. Res. **5**, 361–397 (2004)
28. Joachims, T.: A support vector method for multivariate performance measures. In: ICML 2005, pp. 377–384. ACM, Germany (2005)
29. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Francisco (1993)