Contents lists available at ScienceDirect

Decision Support Systems

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

On strategies for imbalanced text classification using SVM: A comparative study

Aixin Sun [a,*], Ee-Peng Lim [b], Ying Liu [c]

[a] School of Computer Engineering, Nanyang Technological University, Singapore
[b] School of Information Systems, Singapore Management University, Singapore
[c] Department of Industrial and Systems Engineering, Hong Kong Polytechnic University, Hong Kong

ARTICLE INFO

Article history:
Received 15 December 2008
Received in revised form 8 July 2009
Accepted 29 July 2009
Available online 8 August 2009

Keywords:
Imbalanced text classification
Support Vector Machines
SVM
Resampling
Instance weighting

ABSTRACT

Many real-world text classification tasks involve imbalanced training examples. The strategies proposed to address the imbalanced classification (e.g., resampling, instance weighting), however, have not been systematically evaluated in the text domain. In this paper, we conduct a comparative study on the effectiveness of these strategies in the context of imbalanced text classification using Support Vector Machines (SVM) classifier. SVM is the interest in this study for its good classification accuracy reported in many text classification tasks. We propose a taxonomy to organize all proposed strategies following the training and the test phases in text classification tasks. Based on the taxonomy, we survey the methods proposed to address the imbalanced classification. Among them, 10 commonly-used methods were evaluated in our experiments on three benchmark datasets, i.e., Reuters-21578, 20-Newsgroups, and WebKB. Using the area under the Precision–Recall Curve as the performance measure, our experimental results showed that the best decision surface was often learned by the standard SVM, not coupled with any of the proposed strategies. We believe such a negative finding will benefit both researchers and application developers in the area by focusing more on thresholding strategies.

1. Introduction

With the rapid development of the Web, huge amount of textual information are now accessible online. Moreover, much more textual documents are being created through Web 2.0 platforms e.g., blogs, wikis and forums, where millions of Web users are now active information providers. This further increases the importance of text classification (i.e., automatically classifying textual documents into topical categories), such that the information can be easily searched and browsed.

In many real-world text classification tasks, a classifier has to learn from imbalanced training examples. That is, the negative training examples overwhelmingly outnumber the positive ones[1] making the classifier training to be imbalanced. Classifying news articles received from multiple news agencies that are interesting to a particular user is one example. Besides text domain, imbalanced classification is also an important problem in medical diagnosis, fraud detection and many other tasks. In the literature, a number of strategies have been proposed to address the imbalanced classification and the commonly used ones are (i) *resampling* that under-samples negative examples or over-samples positive examples so as to re-balance the training examples; (ii) *instance weighting* that assigns different error-classification costs to negative and positive training examples in classifier training; and (iii) *thresholding* that adjusts decision thresholds of a classifier to balance the precision and recall.

1.1. Motivation

Existing works on the effectiveness of these strategies have been mainly conducted on non-text domain (e.g., using UCI datasets[2]) [1,12]. There is a lack of a comparative study on the effectiveness of these strategies in imbalanced text classification. Given the importance of imbalanced text classification in real-world applications and the uniqueness of text classification tasks (e.g., high dimensionality, sparse feature spaces, and linearly separability in most tasks [13]), we believe a comparative study of imbalanced text classification will greatly benefit application developers as well as researchers in Information Retrieval, Machine Learning, and related areas.

Moreover, most existing studies in imbalanced classification used the area under the *Receiver Operating Characteristic* (ROC)-curve for performance evaluation [1,4,10]. A very recent study [7], however, showed that the area under ROC-curve (AUR) could present "an overly optimistic view of an algorithm's performance" in the imbalanced setting and suggested the area under Precision–Recall curve (PR-Curve) instead. Such a finding further motivates this study to evaluate the strategies using the area under the PR-Curve (AUP) as

* Corresponding author.
E-mail addresses: axsun@ntu.edu.sg (A. Sun), eplim@smu.edu.sg (E.-P. Lim), mfyliu@polyu.edu.hk (Y. Liu).

[1] In our discussion, we assume negative training examples are the majority and positive training examples are the minority.

[2] http://mlearn.ics.uci.edu/MLRepository.html.

0167-9236/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.dss.2009.07.011

the performance evaluation metric, to better reflect their effectiveness. Specifically, in this paper, we study the effectiveness of the above-mentioned strategies in imbalanced text classification using Support Vector Machines (SVM) classifiers with AUP. SVM classifier is the interest of this study for three reasons.

- First, SVM has been very successfully applied to text classification and many other supervised learning tasks [3,9,13,24,26,34,36]. Strategies to improve SVM classifiers for imbalanced text classification will therefore benefit existing text classification approaches that use SVM classifiers.
- Second, with SVM being a binary classifier, imbalanced training is almost inevitable when using SVM classifier in multi-category classification tasks. These tasks usually adopt *one-against-all* learning strategy. That is, one SVM classifier is learned for each category, and the positive (negative) training examples are the examples belonging to (not belonging to) the target category. There is therefore a huge number of training examples from the non-target categories.
- Third, studies have shown that SVM can be adversely affected by imbalanced training where negative training examples heavily outnumber positive ones [1]. With imbalanced training examples, SVM often gives high precision but low recall on the target category.

1.2. Contributions

We summarize our research contributions as follows.

- First, we propose a clear taxonomy to describe all strategies for addressing imbalanced classification. Based on the taxonomy, we survey the techniques that have been studied in literature. Although this taxonomy is provided in the context of text classification using SVM classifiers, it can be easily adopted in other imbalanced classification tasks with minimum modification.
- Second, our comparative study systematically evaluated 10 methods best representing the various strategies (and their combinations) on 3 benchmark datasets. The 8 methods materialized with SVM as the underlying classifier are: standard SVM, Stratified RANDom sampling (SRAND), CLuster-based Under-Sampling (CLUS), Synthetic Minority Over-sampling Technique (SMOTE), and the above four methods with instance weighting. The other 2 methods (i.e., SVM_{BEP} and SVM_{F1}) are based on $\text{SVM}^{per\ f}$ where the two methods are formulated for optimizing Precision/Recall Break-Even Point (BEP) and F_1 respectively in training.

Note that, this paper aims to provide a comparative study of existing strategies proposed for imbalanced text classification using SVM through extensive experiments on multiple benchmark datasets. Hence proposing new techniques addressing imbalanced text classification is not the main focus. In our experiments on the three datasets, standard SVM learned either the best or the second best decision surface in almost all experiments. That suggests that finding an appropriate threshold is more worthwhile in imbalanced text classification tasks. We argue that such a negative finding would benefit application developers and researchers to focus more on thresholding strategy when dealing with imbalanced text classification tasks.

1.3. Paper organization

The rest of the paper is organized as follows. In Section 2, we give a brief introduction to SVM and a taxonomy of strategies for handling imbalanced classification. The experiment design and experimental results are reported in Sections 3 and 4 respectively. In Section 5, we study the impact of varying parameters in resampling and instance weighting and the impact of varying imbalance ratio. In Section 6, the performance of SVM and $\text{SVM}^{per\ f}$ is compared. The findings from the experiments are discussed in Section 7. Finally, Section 8 concludes the paper and proposes future works.

2. SVM and imbalanced learning

We first give a brief introduction to SVM and then review the strategies addressing the imbalanced classification. The possible impact of applying these strategies on SVM learning is also discussed.

2.1. Support Vector Machines

The training of a SVM classifier involves finding a hyperplane, as its decision surface, that separates the positive training examples from the negative ones with the largest margin [30]. Fig. 1 illustrates the training of a linear separable SVM. Given training examples represented as pairs ($\vec{x}_i$, y_i), where $\vec{x}_i$ is the weighted feature vector of the ith training example and $y_i \in \{1, -1\}$ is the label of the example. The search for such a hyperplane can be expressed as an optimization problem of minimizing $\frac{1}{2}\|\vec{w}\|^2$ subject to $y_i\,(\vec{w}\cdot\vec{x}_i - b) \geq 1$, $\forall_i$, where $\vec{w}$ is a vector perpendicular to the hyperplane which defines the orientation of the hyperplane, and b defines the position of the hyperplane. The learned hyperplane is defined by a subset of positive and negative training examples, known as positive and negative support vectors respectively (see Fig. 1).

Once $\vec{w}$ and b are learned, SVM computes a score for an unlabeled document represented by its feature vector $\vec{x}$ using the decision function $f(\vec{x}) = \vec{w}\cdot\vec{x} - b$. The sign of the score is used to predict the label of the document. That is, the document is labeled positive if $f(\vec{x}) \geq 0$, and negative otherwise. In other words, SVM takes 0 as the "default" threshold in its decision function (i.e., default thresholding).

As the hyperplane learned by SVM is defined by support vectors only, it is expected that SVM is less affected by imbalanced training examples [31]. However, it is found that with imbalanced training examples, the hyperplane is often skewed to the minority and the ratio between the positive and negative support vectors is imbalanced (i.e., the hyperplane is defined by more negative support vectors than positive ones) [1,32]. For these two reasons, SVM is more likely to give a negative score when classifying a document in an imbalanced setting.

In this work, we model a SVM classifier with two components: a decision surface $\mathcal{H}$ and a threshold θ. As the score $f(\vec{x})$ is a real number, it is not difficult to introduce a threshold θ, and label a document positively if $f(\vec{x}) \geq \theta$. That is, given a set of documents to be classified, a classifier outputs a score for each document based on H, indicating the document's likelihood of belonging to the target category. The category label of each document is then determined based on a given threshold θ ($\theta = 0$ with default thresholding). A better decision surface $\mathcal{H}$ is the one which better ranks the documents

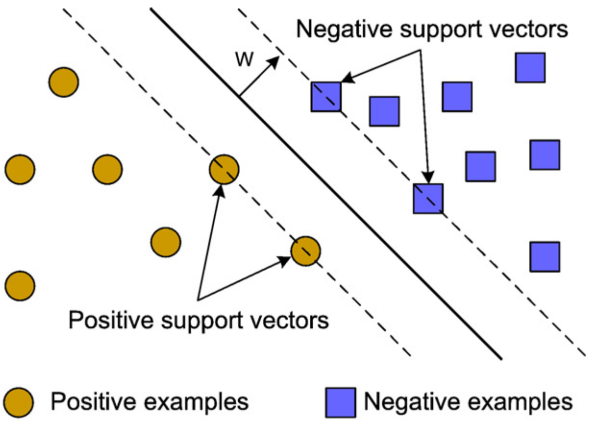


Fig. 1. A linear separable Support Vector Machine.

according to their likelihood of belonging to the target category. To measure the goodness of a decision surface $\mathcal{H}$, we adopt a threshold-independent measure, the area under the Precision–Recall curve (see Section 3.3).

2.2. Strategies for imbalanced classification

Fig. 2 illustrates both the *training* and *classification* processes in a typical text classification task[3]. Both labeled and unlabeled documents are represented by feature vectors according to certain weighting scheme, e.g., *tf·idf*. In the training and classification processes, the strategies for handling imbalanced data, e.g., *resampling*, *instance weighting* and *thresholding*, are applied at different stages, namely, pre-training, in-training, and post-training stages respectively.

2.2.1. Pre-training stage

Resampling is a pre-training strategy that artificially re-balances training examples by either *under-sampling* to select a subset of negative training examples [5,11,16,18,27], or *over-sampling* to (synthetically) generate more positive examples [4].

One typical under-sampling method is random sampling (or undirected sampling) which refers to the process of randomly drawing a subset of training examples from the original set. Many studies have shown that random sampling hurts classifier performance [1]. Directed sampling, on the other hand, aims to select the negative training examples that are expected to be close to the decision surface [5,27]. As the decision surface is defined by both the positive and negative examples, negative training examples close to the decision surface are those that are close to the positive training examples. In [27], the closeness of a negative training example to the positive training examples is computed based on the number of discriminative features it contains. Yoon and Kwek proposed a method to select negative training examples through clustering in [35]. Both negative and positive training examples are first clustered using a supervised clustering algorithm with a class purity maximization function. The clusters containing almost purely negative examples are discarded.

Over-sampling refers to the process of generating more positive training examples. Since studies have shown that over-sampling with replication does not significantly improve the classification accuracy, Chawla et al. proposed Synthetic Minority Over-sampling Technique (SMOTE) to create positive training instances synthetically [4]. For each positive example, its k nearest neighbors among other positive examples are identified. The example and one of its neighbors form a pair which corresponds to two points in the vector space. A new positive example is created by picking up any random point along the line linking these two points (see more detailed discussion in Section 3.2). Despite the effectiveness reported in the literature, it is known that under-sampling involves loss of information and over-sampling does not gain any information but increases the training size [31].

Pre-training methods also include feature selection and term weighting techniques that address class imbalance [6,20,37]. For instance, Zheng et al. proposed a feature selection framework to select positive features that are most indicative of membership of target category and negative features that are most indicative of membership of non-target category separately. The positive and negative features are then combined and used to represent training documents. The proposed technique, however, was not evaluated on SVM classifiers in their experiments. Combarro et al. proposed a family of linear measures for feature selection and evaluated their effectiveness with SVM classifiers on two text datasets (i.e., Reuters-21578 and Ohsumed) and improvement on F_1 was observed. Liu et al. proposed a probability based feature weighting scheme for imbalanced classification [20]. A feature is assigned more weight if it appears more frequently in the positive training examples than negative ones measured by document frequency.

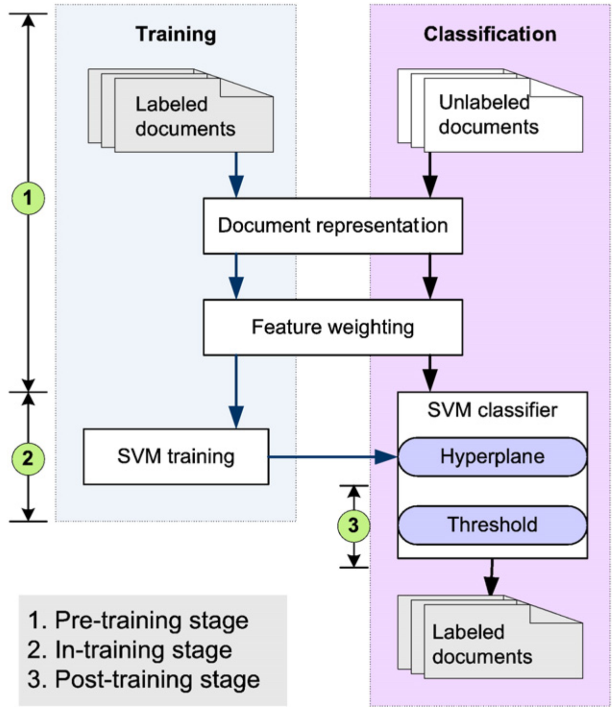


Fig. 2. Training and classification processes.

2.2.2. In-training stage

Instance weighting is a commonly-used in-training strategy that assigns different error-classification costs on the positive and the negative training examples respectively [2]. For instance, in *SVM*^{light} package[4], cost-factor j is used to define by which training errors on the positive examples outweigh errors on negative examples. Examples of more complicated in-training methods include the method that modifies the kernel matrix according to the imbalanced data distribution [32], and SVM formulated to optimize multivariate performance measures (e.g., to optimize the SVM learning for F_1, Precision/Recall break-even point, or other measures) [14]. Recall that with imbalanced training examples, SVM often gives high precision but low recall on the target category. The learning algorithm aiming at optimizing F_1 or Precision/Recall break-even point may therefore achieve more balanced precision and recall values.

Another option to address imbalanced learning is to partition the negative training examples into subsets for training multiple SVM classifiers, each learning from the same set of positive examples and one subset of negative examples [15,19]. Nevertheless, Rifkin and Klautau have shown that simple *one-against-all* learning strategy is as accurate as any other learning strategy, assuming that the underlying binary classifiers are well-tuned regularized classifiers such as SVM [23].

2.2.3. Post-training stage

Thresholding is a post-training strategy that adjusts decision thresholds (see [24] for a good discussion on thresholding). Provost

[3] As feature selection is often not necessary for SVM classifier, it is not shown in the training/classification process.

[4] http://svmlight.joachims.org/.

pointed out that it “may well be a critical mistake” to use classifiers learned from imbalanced data without adjusting the output threshold [22]. When SVM is used in text classification tasks, the default threshold (i.e., $\theta=0$ as discussed in Section 2.1), is commonly adopted. However, depending on the application, a negative threshold may be used and a document may be labeled positively even if it receives a negative score from SVM classifier. Such kind of threshold relaxation has been used in hierarchial text classification to avoid blocking documents at high-level categories in the hierarchy [29].

Yang studied three thresholding strategies in text classification and found that proportional thresholding (i.e., PCut) performed well in classifying rare categories for multi-category classification task which involves imbalanced classification [33]. With proportional thresholding, it is assumed that the percentage of positive documents in the test data matches the percentage in the training data. Experiments have shown that better SVM classification accuracy can be achieved by adjusting the thresholds when learning from imbalanced data [2,25]. Nevertheless, the effectiveness of PCut heavily depends on the distribution of the data as it assumes that the ratio between positive and negative examples does not change from training data to test data.

Another commonly used approach of determining a reasonable threshold is through validation set (e.g., cross-validation). With this approach, the training data is further split into two sets. The first set is used to learn a classifier and the second set is used to search for a threshold which leads to the best result with respect to the performance evaluation metric (e.g., F_1).

2.2.4. Discussion

Among the above discussed strategies, thresholding does not directly affect the training of a SVM classifier. However, applying strategies in pre- and/or in-training stage (e.g., resampling or instance weighting) could lead to a very different decision surface compared to the decision surface learned by a SVM classifier without applying any strategy. In this paper, we therefore aim to find out through experiments whether or not applying strategies in pre-/in-training stage (or both) leads to a better decision surface in imbalanced text classification. The answer to this question has important implications. For instance, if none of these strategies could learn a better decision surface than the standard SVM, then finding an appropriate threshold is more worthwhile when dealing with imbalanced text classification. On the other hand, if some strategy could lead to a better decision surface than the standard SVM, then whether or not to apply such a strategy heavily depends on the computational cost of applying the strategy and the cost of finding an appropriate threshold.

Among all methods discussed above, we restrict our investigation to the commonly-used ones, namely, random sampling, directed under-sampling, over-sampling, instance weighting, and SVM for multivariate performance measures.

3. Experiment setup

All our experiments were conducted on three benchmark datasets commonly used in text classification tasks, i.e., 20-Newsgroups, Reuters-21578, and WebKB. In total three sets of experiments were conducted. In the first set of experiments, we compare the goodness of the decision surfaces learned by eight methods including standard SVM, random under-sampling, directed under-sampling, over-sampling, and their combinations with instance weighting. In the second set of experiments, we study the impact of varying parameters in resampling and instance weighting and also the impact of varying imbalance ratios. In the third set of experiments, the standard SVM was compared to SVM optimized for F_1 and Precision/Recall break-event point respectively. In summary, 10 methods have been evaluated over 3 datasets.

3.1. Datasets

The three datasets used in our experiments are 20-Newsgroups, Reuters-21578, and WebKB. All these datasets have been commonly used in text classification tasks and the three datasets well represent three types of documents, i.e., UseNet messages, news articles, and personal/project homepages.

20-Newsgroups contains posts collected from 20 UseNet groups with nearly 1000 posts from each group. We used the “bydate” version preprocessed by Ana Cardoso-Cachopo[5], which contains 11,293 training documents and 7528 test documents. All the 20 categories were used as target categories in our experiments. Thus, using *one-against-all* learning strategy, the imbalance ratio (i.e., the ratio between the negative and the positive training examples) is about 19:1 for each category.

Reuters-21578 corpus is one of the most popular datasets used in text classification[6]. The 21,578 documents in this collection are organized in 135 categories. Each document may have zero, one or more category labels. With “ModLewis” split, we had 13,625 training and 6188 test documents respectively. We chose 26 categories as target categories such that each category has at least 50 positive training documents. This is to avoid lack of training examples to confound our study on imbalanced classification[7]. The documents that do not belong to any of the selected 26 target categories were used as negative training/test examples in the experiments. The imbalance ratios range from 4:1 to 272:1 for the 26 categories. Among them, 15 categories have imbalance ratios greater than 100:1.

WebKB dataset contains Web pages collected from Computer Science departments of four universities by the CMU text learning group[8]. The 4162 Web pages collected are classified in 7 categories and the four target categories used in our experiments are *student*, *faculty*, *course* and *project*. All pages from the remaining categories were used as negative training and test pages. As there is no predefined train/test split, we used *leave-one-university-out* cross-validation to conduct training and evaluation. That is, for each category, pages from three universities were used as training examples and the classifier learned was tested with the pages from the remaining university. The imbalance ratios range from 6:1 to 50:1 for WebKB dataset.

The preprocessing of the dataset includes HTML tag removal (for WebKB dataset only), stopword removal, and stemming. Document feature vectors are weighted with $tf \times idf$ scheme and normalized to unit length.

3.2. Methods

The methods evaluated in our experiments are divided into three groups. The first group includes the standard SVM, Stratified Random Sampling (SRAND), CLuster-based Under-Sampling (CLUS), and SMOTE. The second group refers to the above four methods with instance weighting. The third group includes SVM optimized for F_1 and SVM optimized for Precision/Recall break-even point.

SVM: or standard SVM, refers to the SVM classifier with all default setting. We used SVM^{light} (version 5.0)[9] with linear kernel as the underlying classifier in our experiments. We used linear kernel as linear kernel has been commonly used in text classification and the

[5] 20-Newsgroup dataset is available at http://www.gia.ist.utl.pt/~acardoso/datasets/.

[6] Reuters-21578 dataset is available at http://www.daviddlewis.com/resources/testcollections/reuters21578/.

[7] Note that, we distinguish the problem of imbalanced classification from the problem of having limited number of training examples. It has been reported that having 20 or more training examples provides “stable generalization performance” for SVM classifier [8].

[8] WebKB dataset is available at http://www.cs.cmu.edu/~webkb/.

[9] http://svmlight.joachims.org/.

choice of kernel functions do not affect text classification performance much [17].

SRAND: Stratified Random Sampling represents undirected under-sampling method. It selects negative training documents according to a *under-sampling ratio s* with stratified sampling. For a given under-sampling ratio of *s*, one document is randomly chosen in every *s* negative training documents sorted by document id. As there is no guideline on how to set a proper sampling ratio, in the first set of experiments, we simply set $s=2$. That is, half of the negative training documents were selected and used in SVM training for each category. The impact of choosing different *s* is studied in the second set of experiments, reported in Section 5.

CLUS: CLuster-based Under-Sampling is a parameter-free directed under-sampling method. The basic idea is to find those negative examples that are close to any positive example. For each category, the pool of training documents (including both positive and negative) are clustered using *k*-means algorithm, where *k* is the number of positive documents and each cluster centroid is initialized as one positive document. After clustering, the clusters that contain only negative training documents are discarded. Negative documents from the clusters that each contains at least one positive example form the new set of negative training examples[10].

SMOTE: Synthetic Minority Over-sampling Technique, is a method to generate synthetic positive training examples [4]. Given a positive training document, its *k* nearest neighbors among other positive training documents are first identified. Let $\overrightarrow{x_i}$ be the feature vector of document d_i, and $\overrightarrow{x_j}$ be the feature vector of one of d_i's *k* nearest neighbors. The feature vector of a synthetic document is created by $(\overrightarrow{x_i}+g(\overrightarrow{x_j}-\overrightarrow{x_i}))$ where *g* is a random value between 0 and 1. In our experiments, we use *k* as over-sampling ratio where one synthetic positive training example is generated from each of the *k* nearest neighbors of a positive training example. In the first set of experiments, we set $k=5$ as in [4]. The impact of using different *k* values is studied in the second set of experiments in Section 5.

Instance weighting: Instance weighting assigns different error-classification costs to positive and negative training examples. In our experiments, instance weighting was implemented by setting the cost-factor (parameter *j*) in SVM^{light}. Following early works [21], in the first set of experiments, we set *j* to be the imbalance ratio of the target category, e.g., $j=\frac{L^n}{L^p}$, where L^n and L^p refer to the number of the negative and positive training examples respectively for the category. The impact of setting different *j*'s is studied in the second set of experiments. The method where instance weighting is applied to the standard SVM is denoted by SVM_w. Similarly, we use SRAND_w, CLUS_w, and SMOTE_w to denote the other three methods using instance weighting together with resampling (See Table 1).

SVM_{BEP} and SVM_{F1}: refer to the SVM classifiers formulated for optimizing Precision/Recall break-even point (BEP) and F_1 respectively. The two methods were based on SVN^{perf} (version 2.1)[11] implementation using the corresponding loss function setting.

3.3. Performance metrics

The commonly-used performance measures are *Precision*, *Recall*, and F_1. Precision for a category, denoted by *Pr*, is the percentage of correct assignments among all the documents assigned to the target category. Recall, denoted by *Re*, is the percentage of correct assignments among all the documents that should be assigned to the target category. $F_1=\frac{2\cdot\text{Pr}\cdot\text{Re}}{\text{Pr}+\text{Re}}$ is the harmonic mean of *Pr* and *Re*.

However, both *Pr*, *Re* (and hence F_1) are threshold dependent. To measure how good a learned decision surface *H* is, performance metrics independent of threshold values are required. Both *Receiver Operating Characteristic* (ROC)-curve and *Precision–Recall curve* (or PR-Curve) have been used in previous works. Although ROC has been used in many studies [1,4,10], a very recent study showed that ROC curve could present "an overly optimistic view of an algorithm's performance" in the imbalanced setting [7]. We therefore adopt PR-Curve to visualize the performance of a classifier and use the Area Under the PR-Curve (or AUP for short) to measure the goodness of a decision surface.

Table 1
List of the eight methods.

Strategy	Without instance weighting	With instance weighting
–	SVM	SVM_w
Undirected under-sampling	SRAND	SRAND_w
Directed under-sampling	CLUS	CLUS_w
Oversampling	SMOTE	SMOTE_w

4. Experimental results

Table 2 reports the macro-averaged imbalance ratio over all categories after resampling with different methods on the three datasets. Note that, for SRAND and SMOTE, the resultant imbalance ratios are purely determined by the parameters given. As a parameter-free method, CLUS selected slightly more than half of negative training documents on Newsgroups and about a quarter on Reuters. OnWebKB dataset, CLUS selected 85% of negative training examples.

In the following pages, we report the experimental results of the 8 methods listed in Table 1 as they are all based on the same underlying classifier (see Section 3.2).

4.1. PR-Curve

The PR-Curves of the eight methods on three datasets are plotted in Fig. 3. Two sets of PR-Curves are plotted for each dataset for better illustration. The figures on the left are for those methods that do not involve instance weighting and the figures on the right are for the methods with instance weighting. On both sets of figures, the PR-Curve for SVM classifier is plotted for easy reference. These PR-Curves are plotted based on macro-averaged precision at each recall value computed using the tool provided by [7]. The dashed line in each plot is provided to identify the break-even point.

As shown in Fig. 3(a) and (b), on Newsgroup dataset, the PR-Curves of all methods are quite similar to each other and hard to distinguish. Nevertheless, among the eight methods, SRAND and SRAND_w performed slightly worse than others. On Reuters dataset, SVM was the method that achieved the best PR-Curve. It is also observed that applying instance weighting hurt the classification performance (see Fig. 3d). On WebKB, without instance weighting, all methods produced similar PR-Curves (see Fig. 3(e)); with instance weighting, SVM was much better than the other methods and CLUS_w was the worst, shown in Fig. 3(f).

4.2. Area under PR-Curve (AUP)

Table 3 reports the macro-averaged AUP for all methods on the three datasets. For each category in a dataset, the AUP is computed using the tool provided by [7]. The value reported for each method is the average over all categories on the dataset. The best value is in bold

Table 2
Macro-averaged imbalance ratio.

Dataset	SVM	SRAND	CLUS	SMOTE
Newsgroups	19.3	9.6	8.8	3.2
Reuters	116.4	58.2	25.4	19.4
WebKB	24.1	12.0	20.3	4.0

[10] CLUS method is similar to the method proposed in [35] with differences in the clustering algorithm and the way of selecting negative training documents.

[11] http://svmlight.joachims.org/.

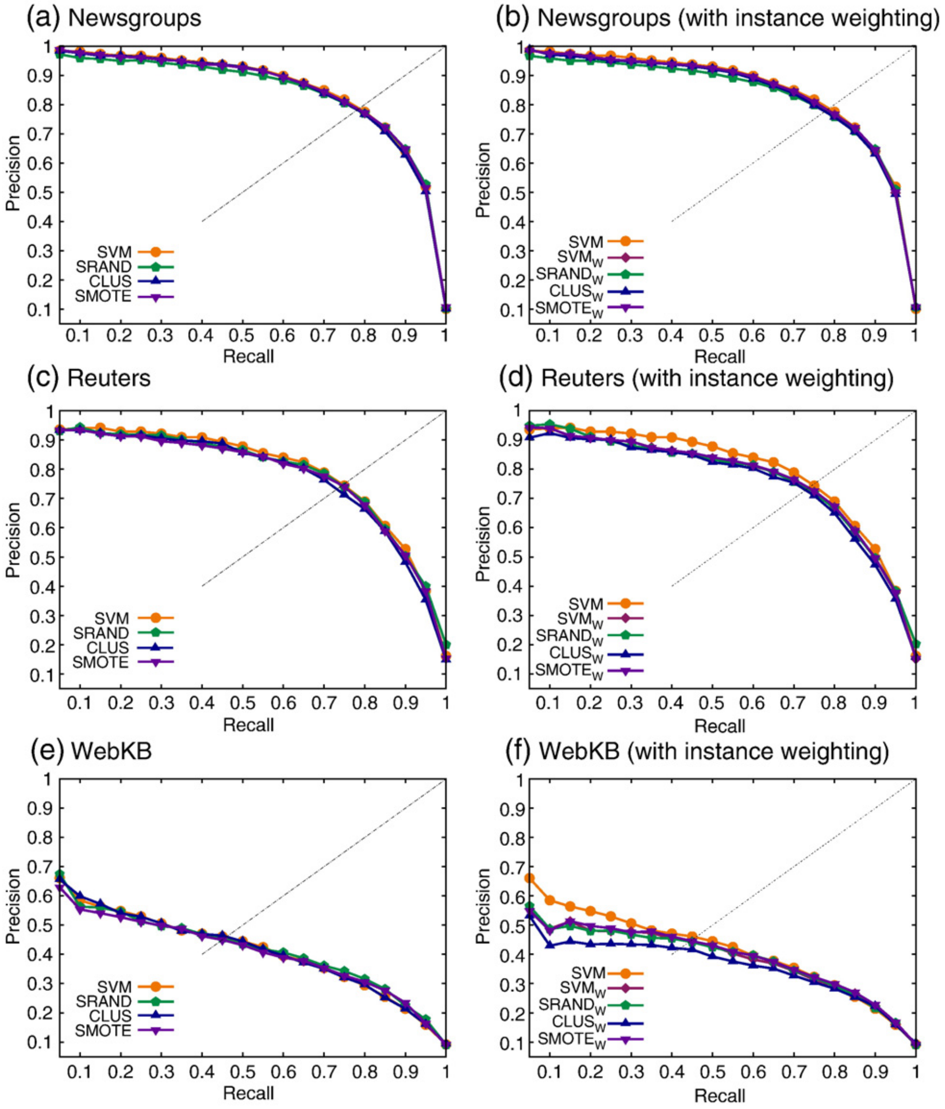


Fig. 3. Precision–Recall curves on Newsgroups, Reuters, and WebKB datasets.

and the second best is underlined. Two observations can be made from the results.

- The standard SVM achieved the best results on Newsgroups and Reuters, and the second best on WebKB. Such an observation suggests that the standard SVM could be the best method among all.
- No method involving instance weighting achieved either the best or the second best. Moreover, each method using instance weighting gave poorer AUP than the same method without instance weighting. That is, method M always delivered better AUP than method M_w, for $M \in$ {SVM, SMOTE, CLUS, SRAND}.

To verify whether the above two observations are statistically significant, we conducted paired t-test on AUP over all categories for each dataset. The p-values are reported in Table 4. Note that, we use 0.001 to indicate that the p-value is either 0.001 or smaller for easy reading, and we use a minus sign ('—') to indicate that the method at the corresponding row is worse than the method at the corresponding column. All those p-values that are smaller than 0.05 are marked with '*'. Based on the significance test, we conclude the following points.

- Standard SVM was significantly better than any method involving resampling and/or instance weighting on both Newsgroups and Reuters datasets (i.e., $p < 0.05$). On WebKB, SVM was comparable with resampling methods (including SRAND, CLUS, and SMOTE),and was significantly better than all methods involving instance weighting.
- Applying instance weighting resulted in significant performance degradation for all methods on all datasets. The only exception was SMOTE (compared to SMOTE$_w$) on WebKB dataset with $p = 0.058$.
- The three resampling methods performed quite differently on the three datasets. On Newsgroups, SMOTE>>CLUS>>SRAND, where >> means significantly better; on Reuters, SRAND>>{SMOTE, CLUS} where SMOTE and CLUS were comparable; on WebKB, all these three methods were comparable.

Table 3
Macro-averaged area under PR-Curve.

Dataset	SVM	SRAND	CLUS	SMOTE	SVM$_w$	SRAND$_w$	CLUS$_w$	SMOTE$_w$
Newsgroups	**0.861**	0.849	0.855	0.858	0.854	0.844	0.851	0.856
Reuters	**0.804**	0.795	0.786	0.788	0.780	0.780	0.765	0.781
WebKB	0.427	**0.429**	0.427	0.420	0.400	0.398	0.368	0.405

For each dataset, the best value is in bold and the second best is underlined.

The first two points well support the two observations made in Section 4.2. Note that SVM was significantly better than all other methods on both Newsgroups and Reuters datasets, but were comparable with SRAND, CLUS and SMOTE on WebKB dataset. One possible reason is that WebKB dataset is relatively small; it is about

Table 4
p-values for paired t-test on AUP.

Method	SRAND	CLUS	SMOTE	SVM_w	$SRAND_w$	$CLUS_w$	$SMOTE_w$
(a) Newsgroups dataset							
SVM	0.001*	0.001*	0.005*	0.001*	0.001*	0.001*	0.001*
SRAND	–	−0.030*	−0.002*	−0.060	0.004*	−0.293	−0.017*
CLUS		–	−0.038*	0.350	0.003*	0.004*	0.377
SMOTE			–	0.003*	0.001*	0.001*	0.007*
SVM_w				–	0.001*	0.006*	−0.009*
$SRAND_w$					–	−0.008*	−0.001*
$CLUS_w$						–	−0.001*
(b) Reuters dataset							
SVM	0.016*	0.001*	0.002*	0.001*	0.001*	0.001*	0.001*
SRAND	–	0.022*	0.022*	0.004*	0.002*	0.001*	0.007*
CLUS		–	−0.331	0.210	0.243	0.001*	0.264
SMOTE			–	0.004*	0.015*	0.001*	0.011*
SVM_w				–	−0.410	0.012*	−0.035*
$SRAND_w$					–	0.012*	−0.404
$CLUS_w$						–	−0.007*
(c) WebKB dataset							
SVM	−0.342	0.406	0.144	0.014*	0.040*	0.002*	0.032*
SRAND	–	0.317	0.143	0.015*	0.013*	0.002*	0.035*
CLUS		–	0.173	0.015*	0.021*	0.002*	0.035*
SMOTE			–	0.021*	0.034*	0.003*	0.058
SVM_w				–	0.335	0.004*	−0.018*
$SRAND_w$					–	0.010*	−0.131
$CLUS_w$						–	−0.001*

* $p<0.05$.

one-fifth of the other two datasets in number of documents, and contains only 4 categories while the other two datasets contains 20 or more categories. With only 4 categories, it is relatively hard for one method to be significantly better than another.

4.3. F_1^M with optimal thresholding

Using AUP as the performance measure, we found that the standard SVM could learn better decision surface than other methods involving resampling and/or instance weighting. That is, the standard SVM could better rank the documents to be classified according to their likelihood of belonging to the target category. This also suggests that, if an appropriate threshold is found, SVM should achieve better F_1 than other methods. To verify, we report the macro-averaged F_1, denoted by F_1^M, using *optimal thresholding*.

With optimal thresholding, all test documents are ranked in descending order according to their scores returned by a classifier. The top ranked d documents are labeled as positive such that the F_1 of the category is maximized. The score of the dth document is the *optimal threshold* for that category. Note that optimal thresholding is not possible in practice as the true labels of test documents are not known a priori. Optimal thresholding however provides the ideal performance of the decision surface learned by a classifier, as our main objective of this study is to measure the goodness of a learned decision surface.

Fig. 4(a), (b), and (c) report F_1^M for eight methods on three datasets respectively. As shown in the figure, with optimal thresholding, SVM achieved the best F_1^M on Newsgroups and Reuters and the second best on WebKB dataset. This is consistent with the results of AUP in Table 3. It is also observed that SMOTE and $SMOTE_w$ achieved slightly better F_1^M than other methods on Newsgroups and Reuters. On WebKB, similar to that of AUP, random sampling was slightly better than SVM.

As mentioned earlier, it is not possible to pre-determine an optimal threshold for a classifier. In reality, many classification tasks simply adopt *default thresholding*. With default thresholding, SVM assigns a document positive label if the score of the decision function is non-negative, i.e., $f(\overrightarrow{x})\geq 0$ (see Section 2.1). For the completeness of the results, we also report F_1^M obtained with default thresholding in Fig. 4. It is interesting to observe that, with default thresholding, SVM became the worst method on all three datasets. Either resampling or instance weighting could further improve F_1^M. This could be the reason why resampling and/or instance weighting are applied in many imbalanced classification tasks as those tasks often adopt default thresholding.

To better explain why SVM became the worst, we plot the optimal thresholds of all methods in Fig. 4(d). It is observed that the difference between the optimal threshold and the default threshold (i.e., 0) for SVM is the largest among all methods. That is, although standard SVM has learnt the best decision surface, the position of the decision surface is far away from its optimal position. To achieve better classification accuracy for standard SVM, one has to find an appropriate threshold to redefine the learned decision surface close to its optimal position.

It is worth noting that finding an appropriate threshold itself is a challenging task [24,25,33] and is out of the scope of this paper.

5. Impact of parameters and imbalance ratio

In our first set of experiments, the over-sampling ratio k in SMOTE, under-sampling ratio s in SRAND and the cost-factor j for instance weighting were pre-defined, for easy comparison among all methods. In this set of experiments, we study the impact of the corresponding parameter for each of the three methods, and also the impact of imbalance ratio.

5.1. Impact of parameters

Over-sampling ratio k determines the number of synthetic documents generated from each positive training document. For example, if $k=1$, one synthetic positive training example is generated from each positive training document. To study the impact of k, we varied k from 1 to 5 and recorded the macro-averaged AUP on the three datasets[12], shown in Table 5. To verify whether the results are statistically significant, the p-values resulted from the paired t-test between SVM and SMOTE (at different k's) are included in Table 5. On Newsgroups, varying k did not affect the AUP much for SMOTE method, and on Reuters, a larger k led to slightly poorer AUP. On both datasets, SVM was significantly better than SMOTE on all k values except $k=2$ on Newsgroups. On WebKB, SMOTE methods at all k values were comparable with SVM.

Under-sampling ratio s determines how many negative samples to select. For instance, if $s=3$, one negative training example is selected among three; hence the imbalance ratio is reduced to the one-third of the original. Similar to the over-sampling ratio k, we evaluated 5 values for s from 2 to 6. Note that $s=1$ means all negative samples are selected, i.e., no change made to the original dataset. Table 6 reports the macro-averaged AUP, together with significance test comparing SVM and SRAND. On both Newsgroups and Reuters, a larger s led to poorer AUP for SRAND. SVM was significantly better than SRAND on almost all s values except $s=3$ on Reuters. On WebKB, no significant different result is observed comparing SVM with SRAND at different s values. For both under-sampling parameter s and over-sampling parameter k, the larger the value, the more the resulted dataset are different from the original training dataset. As the test dataset usually follows the similar distribution as the original training dataset, it is not a surprise that the decision surfaces learned are poorer with larger s and k values.

Cost-factor j defines the weight of training errors on positive examples over negative examples [21]. In our experiments, we compared SVM and SVM_w at different j's defined based on imbalance ratio r, shown in Table 7. Similar to the experiments on k and s, 5 values of j were evaluated from $0.2r$ to r since j has often been set to r [21,28]. On Newsgroups and Reuters, increasing j resulted in slightly poorer AUP delivered by SVM_w. On WebKB, when $j=0.2r$, a better

[12] The setting of parameter k followed that in [4] where SMOTE was evaluated with over-sampling ratio from 1 to 5.

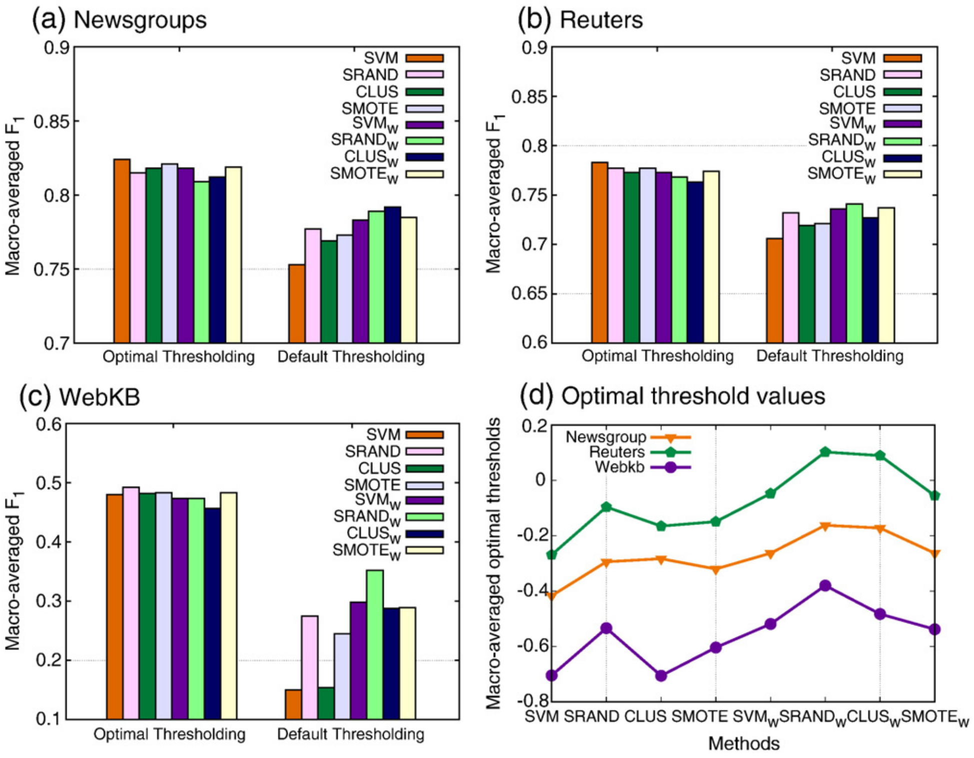


Fig. 4. F_1^M with optimal and default thresholding and optimal threshold values.

AUP than SVM was achieved. However, the AUP achieved was not significantly better than standard SVM. Larger j's on WebKB led to poorer AUP; similar observation holds on the other two datasets.

5.2. Impact of imbalance ratio

Experimental results reported in Sections 4 and 5 are based on three datasets with fixed imbalance ratios. In this section, we design another set of experiments to study the impact of different imbalance ratios on resampling and instance weighting methods, and also the standard SVM classifier. The objective is to answer the question whether resampling and/or instance weighting could be more effective when the imbalance ratio is higher.

We constructed 6 datasets from 20-Newsgroups dataset with different imbalance ratios ranging from 19:1 to 191:1. To construct datasets with different imbalance ratios, for each of the 20 categories, we first derived the category's positive and negative training documents with one-against-all setting. Keeping the negative training documents unchanged, we applied stratified sampling to the positive training examples according to a sampling ratio s. These chosen documents form the positive training documents for that category in the new dataset. Stratified sampling (with the same sampling rate) was also applied to the category's positive test documents to maintain the positive/negative distribution between the training and test documents. Dataset D_s is obtained by applying the same sampling ratio s over the 20 categories. The 6 datasets were obtained with $s=1$, 2, 4, 6, 8, 10. Table 8 reports the averaged positive/negative training/test documents for each category over the 20 categories in each dataset, and the averaged imbalance ratios, where L^p, L^n, T^p, and T^n denote the number of positive training, negative training, positive test and negative test documents respectively. Note that, D_1 refers to the original 20-Newsgroups dataset.

Table 9 reports the area-under PR-Curve of all methods on the 6 datasets. Similar to our earlier results, the best value is in bold and the second best is underlined. From Table 9, we can observe that imbalance ratio has significant impact on all the eight methods

Table 5
Impact of over-sampling ratio k in SMOTE.

Method	Newsgroups		Reuters		WebKB	
Parameter k	AUP	p-value	AUP	p-value	AUP	p-value
SVM	**0.861**	–	**0.804**	–	0.427	–
SMOTE ($k=1$)	0.858	0.023*	0.796	0.001*	0.425	0.356
SMOTE ($k=2$)	0.859	0.063	0.792	0.001*	0.422	0.138
SMOTE ($k=3$)	0.858	0.046*	0.792	0.005*	0.427	−0.472
SMOTE ($k=4$)	0.858	0.011*	0.788	0.001*	**0.428**	−0.403
SMOTE ($k=5$)	0.858	0.005*	0.788	0.002*	0.420	0.144

The best results are in bold. * $p<0.05$.

Table 6
Impact of under-sampling ratio s in SRAND.

Method	Newsgroups		Reuters		WebKB	
Parameter s	AUP	p-value	AUP	p-value	AUP	p-value
SVM	**0.861**	–	**0.804**	–	0.427	–
SRAND ($s=2$)	0.849	0.001*	0.795	0.016*	0.429	−0.342
SRAND ($s=3$)	0.845	0.001*	0.794	0.052	**0.436**	−0.088
SRAND ($s=4$)	0.836	0.001*	0.792	0.020*	0.419	0.131
SRAND ($s=5$)	0.834	0.001*	0.785	0.003*	0.414	0.107
SRAND ($s=6$)	0.830	0.001*	0.783	0.004*	0.423	0.359

The best results are in bold. * $p<0.05$.

Table 7
Impact of cost-factor j in SVM$_w$.

Method	Newsgroups		Reuters		WebKB	
Parameter j	AUP	p-value	AUP	p-value	AUP	p-value
SVM	**0.861**	–	**0.804**	–	0.427	–
SVM$_w$ ($j=0.2r$)	0.856	0.001*	0.781	0.001*	**0.429**	−0.312
SVM$_w$ ($j=0.4r$)	0.855	0.001*	0.780	0.001*	0.403	0.022*
SVM$_w$ ($j=0.6r$)	0.855	0.001*	0.780	0.001*	0.401	0.026*
SVM$_w$ ($j=0.8r$)	0.854	0.001*	0.780	0.001*	0.401	0.014*
SVM$_w$ ($j=r$)	0.854	0.001*	0.780	0.001*	0.400	0.024*

The best results are in bold. * $p<0.05$.

Table 8
Dataset statistics.

Dataset	L^p	L^n	T^p	T^n	Imbalance Ratio
D_1	565	10,728	376	7152	19.3
D_2	283	10,728	188	7152	38.5
D_4	142	10,728	94	7152	76.8
D_6	94	10,728	63	7152	115.2
D_8	71	10,728	47	7152	152.9
D_{10}	57	10,728	38	7152	191.2

including SVM. The higher the imbalance ratio, the poorer the AUP values. Nevertheless, SVM remained the best method which achieved the highest AUP on all 6 datasets. That is, either resampling or instance weighting could not learn a better decision surface than the standard SVM regardless of the imbalance ratio.

6. SVM, SVM$_{BEP}$, and SVM$_{F1}$

In this set of experiments, we compare the performance of SVM, SVM$_{BEP}$, and SVM$_{F1}$ on the three datasets using AUP as performance measure.

Table 10 reports the macro-averaged AUP for the three classifiers on the three datasets. SVM achieved the best AUP on both Newsgroup and Reuters datasets but the worst on WebKB. According to the significance test shown in Table 11, SVM significantly outperformed both SVM$_{BEP}$ and SVM$_{F1}$ on Newsgroup dataset and SVM$_{BEP}$ on Reuters dataset. The WebKB is the only dataset where SVM$_{F1}$ was the best performer. On all three datasets, SVM$_{BEP}$ was always comparable with SVM$_{F1}$. In summary, SVM formulated with optimization for either break-even point or F_1 did not achieve significant performance improvement on AUP compared to standard SVM on two largest datasets out of the three evaluated. Note that the PR-Curves are not reported for this set of experiments as they are very similar to each other as in Fig. 3.

Similar to the results reported in Section 4.3, we also obtained the macro-averaged F_1 values for the three methods on the three datasets (see Table 12) with default and optimal thresholding respectively. The results are consistent with that reported earlier; once a suitable threshold is given, the standard SVM outperformed both SVM$_{BEP}$ and SVM$_{F1}$ on the two largest datasets. Even with default thresholding (e.g., 0), the standard SVM was the best performer on Newsgroups and Reuters. An interesting observation on the optimal threshold values is that the optimal threshold values for standard SVM are always below zero. That is, with default thresholding, SVM would give more False Negatives. However, for both SVM$_{BEP}$ and SVM$_{F1}$, the optimal threshold values were all above zero. With default thresholding, both classifiers led to more False Positives.

7. Discussion

From our experiments, an interesting observation was that resampling and instance weighting strategies were not effective as expected in imbalanced text classification. However, these strategies have been reported to be effective in some other experiments. We believe there are mainly three reasons for their poor performance in our experiments.

- *Performance evaluation metric*. As discussed in Section 1.1, many work involving imbalanced classification adopted area under the ROC-Curve (AUR) as performance measure. With AUR as performance metric used in other experiments, sampling or instance weighting methods may show to be effective. However, a recent study on the relationship between Precision–Recall and ROC curves showed that AUR could present "an overly optimistic view of an algorithm's performance" in the imbalanced setting [7]. This was also the reason we conducted the comparative study.
- *Nature of the classifier*. In other experiments, the methods had been evaluated with classifiers other than SVM including decision tree, Naïve bayes and others. For instance, in [4], where SMOTE algorithm was originally proposed, decision tree, Naïve bayes and Ripper classifiers were evaluated in their experiments. The artificially re-balancing of the dataset through resampling certainly changes the statistical properties of the features. Hence the classifiers that heavily rely on statistical properties of features (e.g., decision tree and Naïve bayes) may give very different classification results. However, for SVM, the decision surface relies on the positive/ negative support vectors, hence SVM is less sensitive to the statistical prosperities of the features.
- *Characteristics of the data*. Compared to data from other domains, text data has its unique characteristics such as high-dimensional feature space, fewer irrelevant features, and sparse feature vectors [13]. The results obtained on datasets from other domains may not necessarily be repeated on text dataset.

In our experiments, we have also observed that the setting of threshold played a critical role in obtaining accurate classification results. However, it is well known that finding optimal thresholding is infeasible in reality in most cases. On the other hand, the setting of the threshold could be heavily application-dependent [24]. Depending on the application, various thresholding techniques maybe adopted. For instance, proportional thresholding has shown its effectiveness when the distribution of the test data (e.g., the ratio between the positive and negative examples) follows that of the training data [33]. Another common approach of finding an appropriate threshold is to use a validation set. In some real-world applications, a classifier may need to classify data objects received along the time, and the threshold could be adjusted during the classification when necessary. In such applications where threshold can be flexibly set, the goodness of the decision surface learned from the training data determines the classification accuracy. In our experiments, we showed that the

Table 9
Area under PR-Curve.

Dataset	SVM	SRAND	CLUS	SMOTE	SVM$_w$	SRAND$_w$	CLUS$_w$	SMOTE$_w$
D_1	**.861**	0.849	0.855	0.858	0.854	0.844	0.851	0.856
D_2	**.784**	0.770	0.778	0.782	0.776	0.761	0.771	0.777
D_4	**.680**	0.658	0.673	0.673	0.669	0.650	0.665	0.669
D_6	**.607**	0.587	0.602	0.603	0.604	0.584	0.598	0.604
D_8	**.563**	0.539	0.560	0.554	0.539	0.524	0.533	0.540
D_{10}	**.487**	0.468	0.481	0.485	0.480	0.465	0.478	0.480

For each dataset, the best values are in bold and the second best are underlined.

Table 10
Macro-averaged area under PR-Curve.

Dataset	SVM	SVM$_{BEP}$	SVM$_{F1}$
Newsgroups	**0.861**	0.821	0.822
Reuters	**0.804**	0.794	0.794
WebKB	0.427	0.430	**0.434**

For each dataset, the best values are in bold and the second best are underlined.

Table 11
p-values for paired *t*-test on AUP.

Dataset	Newsgroup		Reuters		Webkb	
Method	SVM$_{BEP}$	SVM$_{F1}$	SVM$_{BEP}$	SVM$_{F1}$	SVM$_{BEP}$	SVM$_{F1}$
SVM	0.001*	0.001*	0.018*	0.088	− 0.084	− 0.041*
SVM$_{BEP}$	–	− 0.182	–	0.487	–	− 0.151

* $p<0.05$.

Table 12
F_1^M with optimal and default thresholding, and optimal threshold values.

Dateset	F_1^M and Threshold	SVM	SVM_{BEP}	SVM_{F1}
Newsgroup	F_1^M with Default Threshold	**0.753**	0.380	0.611
	F_1^M with Optimal Threshold	**0.824**	0.792	0.791
	Optimal threshold	−0.417	2.103	1.437
Reuters	F_1^M with Default Threshold	**0.706**	0.125	0.467
	F_1^M with Optimal Threshold	**0.783**	0.775	0.773
	Optimal threshold	−0.269	3.479	2.804
WebKB	F_1^M with Default Threshold	0.15	0.196	**0.366**
	F_1^M with Optimal Threshold	0.480	0.494	**0.495**
	Optimal threshold	−0.705	0.739	0.348

The best results are shown in bold.

standard SVM could learn a good decision surface without applying resampling or instance-weighting techniques.

8. Conclusion and future work

In this paper, we give a comparative study on the strategies addressing imbalanced text classification using SVM classifiers. We first summarize the strategies in a taxonomy in the context of text classification. Based on the taxonomy, we give a survey on the techniques proposed for imbalanced classification including resampling and instance weighting and others. Through extensive experiments, we evaluated 10 methods on 3 benchmark datasets using AUP as the performance metric. To the best of our knowledge, this is the first comparative study on imbalanced classification in text domain. Our experimental results showed the standard SVM often learn the best decision surface in most test cases. For the classification tasks involving high imbalance ratios, it is therefore more critical to find an appropriate threshold than applying any of the resampling or instance weighting strategies.

Based on the findings, we suggest two future research directions. One direction is to look deep into thresholding strategies, which may consider the data distribution, the information obtained during the classifier training, and user feedback if available. Another research direction is to improve the SVM learning objective function to consider the data imbalance in learning the decision surface such that the default threshold could be easily adopted.

Acknowledgement

This work was partially supported by A*STAR Public Sector R&D, Singapore, Project Number 062 101 0031.

References

[1] R. Akbani, S. Kwek, N. Japkowicz, Applying support vector machines to imbalanced datasets, *Proc. of ECML'04*, Sep. 2004, pp. 39–50, Pisa, Italy.

[2] J. Brank, M. Grobelnik, N. Milic-Frayling, D. Mladenic, Training text classifiers with SVM on very few positive examples, Technical Report MSR-TR-2003-34, Microsoft Research, Apr. 2003.

[3] M. Chau, H. Chen, A machine learning approach to web page filtering using content and structure analysis, Decision Support Systems 44 (2) (2008) 482–494.

[4] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: synthetic minority over-sampling technique, Journal of Artificial Intelligence Research 16 (2002) 321–357.

[5] C.-M. Chen, H.-M. Lee, M.-T. Kao, Multi-class svm with negative data selection for web page classification, *Proc. of IEEE Int'l Joint Conf. on Neural Networks*, vol. 3, July 2004, pp. 2047–2052, Budapest, Hungary.

[6] E.F. Combarro, E. Montanes, I. Diaz, J. Ranilla, R. Mones, Introducing a family of linear measures for feature selection in text categorization, IEEE Transactions on Knowledge and Data Engineering(TKDE) 17 (9) (Sep. 2005) 1223–1232.

[7] J. Davis, M. Goadrich, The relationship between precision–recall and ROC curves, *Proc. of ICML'06*, ACM Press, Pittsburgh, Pennsylvania, June 2006, pp. 233–240.

[8] S.T. Dumais, J. Platt, D. Heckerman, M. Sahami, Inductive learning algorithms and representations for text categorization, *Proc. of ACM CIKM'98*, Nov. 1998, pp. 148–155, Bethesda, Maryland.

[9] W. Fan, M.D. Gordon, P. Pathak, An integrated two-stage model for intelligent information routing, Decision Support Systems 42 (1) (2006) 362–374.

[10] T. Fawcett. Roc graphs: Notes and practical considerations for data mining researchers. Technical Report HPL-2003-4, HP Laboratories, Jan. 2003. http://www.hpl.hp.com/techreports/2003/HPL-2003-4.html.

[11] D. Fragoudis, D. Meretakis, S. Likothanassis, Integrating feature and instance selection for text classification, *Proc. of ACM SIGKDD'02*, Edmonton, Alberta, Canada, July 2002, pp. 501–506.

[12] N. Japkowicz, S. Stephen, The class imbalance problem: a systematic study, Intelligent Data Analysis 6 (5) (2002) 429–449.

[13] T. Joachims, Text categorization with support vector machines: learning with many relevant features, *Proc. of ECML'98*, Apr. 1998, pp. 137–142, Springer-Verlag.

[14] T. Joachims, A support vector method for multivariate performance measures, *Proc. of ICML'05*, Aug. 2005, pp. 377–384, Bonn, Germany.

[15] U.H.-G. Krebel, Pairwise classification and support vector machines, in: B. Schölkopf, C. Burges, A. Smola (Eds.), *Advances in kernel methods: support vector learning*, MIT Press, 1999, pp. 255–268.

[16] M. Kubat, S. Matwin, Addressing the curse of imbalanced training sets: one-sided selection, *Proc. of ICML'97*, July 1997, pp. 179–186.

[17] E. Leopold, J. Kindermann, Text categorization with support vector machines how to represent texts in input space? Machine Learning 46 (1–3) (2002) 423–444.

[18] H. Liu, H. Motoda, On issues of instance selection, Data Mining and Knowledge Discovery 6 (2002) 115–130.

[19] X.-Y. Liu, J. Wu, Z.-H. Zhou, Exploratory under-sampling for class-imbalance learning, *Proc. of ICDM'06*, Dec. 2006, pp. 965–969, Hong Kong, China.

[20] Y. Liu, H.T. Loh, A. Sun, Imbalanced text classification: a term weighting approach, Expert System with Applications 36 (1) (2009) 690–701.

[21] K. Morik, P. Brockhausen, T. Joachims, Combining statistical learning with a knowledge-based approach — a case study in intensive care monitoring, *Proc. of ICML'99*, 1999, pp. 268–277, Bled, Slowenien.

[22] F. Provost, Machine learning from imbalanced data sets 101, *Proc. of Workshop on Learning from Imbalanced Data Sets (AAAI'00)*, 2000, pp. 1–3, Menlo Park, California.

[23] R. Rifkin, A. Klautau, In defense of one-vs-all classification, Journal of Machine Learning Research 5 (2004) 101–141.

[24] F. Sebastiani, Machine learning in automated text categorization, ACM Computing Surveys 34 (1) (2002) 1–47.

[25] J.G. Shanahan, N. Roma, Boosting support vector machines for text classification through parameter-free threshold relaxation, *Proc. of CIKM'03*, 2003, pp. 247–254, New Orleans, LA.

[26] D. Song, R.Y.K. Lau, P.D. Bruza, K.-F. Wong, D.-Y. Chen, An intelligent information agent for document title classification and filtering in document-intensive domains, Decision Support Systems 44 (1) (2007) 251–265.

[27] A. Sun, E.-P. Lim, B. Benatallah, M. Hassan, FISA: Feature-based instance selection for imbalanced text classification, *Proc. of PAKDD'06*, 2006, pp. 250–254, Singapore.

[28] A. Sun, E.-P. Lim, W.-K. Ng, Web classification using support vector machine, *Proc. of WIDM'02*, ACM, McLean, Virginia, USA, 2002, pp. 96–99.

[29] A. Sun, E.-P. Lim, W.-K. Ng, J. Srivastava, Blocking reduction strategies in hierarchical text classification, IEEE Transactions on Knowledge and Data Engineering (TKDE) 16 (10) (Oct. 2004) 1305–1308.

[30] V.N. Vapnik, The nature of statistical learning theory, Springer Verlag, Heidelberg, DE, 1995.

[31] S. Visa, A. Ralescu, Issues in mining imbalanced data sets — a review paper, *Proc. of Midwest Artificial Intelligence and Cognitive Science Conference (MAICS'05)*, 2005, pp. 67–73, Dayton.

[32] G. Wu, E.Y. Chang, KBA: kernel boundary alignment considering imbalanced data distribution, IEEE Transactions on Knowledge and Data Engineering (TKDE) 17 (6) (June 2005) 786–795.

[33] Y. Yang, A study of thresholding strategies for text categorization, *Proc. of SIGIR'01*, 2001, pp. 137–145, New Orleans, USA.

[34] Y. Yang, X. Liu, A re-examination of text categorization methods, *Proc. of ACM SIGIR'99*, Aug. 1999, pp. 42–49, Berkeley, USA.

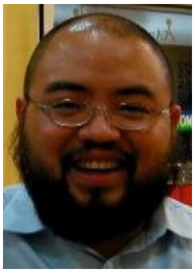
[35] K. Yoon, S. Kwek, An unsupervised learning approach to resolving the data imbalanced issue in supervised learning problems in functional genomics, *Proc. of International Conference on Hybrid Intelligent Systems*, 2005, pp. 303–308.

[36] Y. Zhang, Y. Dang, H. Chen, M. Thurmond, C. Larson, Automatic online news monitoring and classification for syndromic surveillance, Decision Support Systems 47 (4) (2009) 508–517.

[37] Z. Zheng, X. Wu, R. Srihari, Feature selection for text categorization on imbalanced data, SIGKDD Explorations Newsletter 6 (1) (2004) 80–89.

Aixin Sun is an Assistant Professor with School of Computer Engineering, Nanyang Technological University (NTU), Singapore. He received his B.A.Sc with First Class Honours and Ph.D. in 2001 and 2004 respectively, both in Computer Engineering from NTU. His research interests include information retrieval, text/web mining, and digital libraries. He has published more than 40 papers in major international conferences and journals including SIGIR, WSDM, CIKM, ACM/IEEE JCDL, IEEE ICDM, PAKDD, IEEE TKDE, JASIST, and KAIS. Aixin is serving as a PC member of various data mining/information retrieval conferences and reviewer for various journals. He is a member of ACM and a member of IEEE.

Ee-Peng Lim is a professor at the School of Information Systems of the Singapore Management University (SMU). He received Ph.D. from the University of Minnesota, Minneapolis in 1994. His research interests include information integration, data/text/web mining, and digital libraries. He is currently an Associate Editor of the ACM Transactions on Information Systems (TOIS), Journal of Web Engineering (JWE), International Journal of Digital Libraries (IJDL) and International Journal of Data Warehousing and Mining (IJDWM). He is a member of the ACM Publications Board. He is also on the Steering Committees of the International Conference on Asian Digital Libraries (ICADL), and Pacific Asia Conference on Knowledge Discovery and Data Mining (PAKDD). He is a member of ACM and a senior member of IEEE.



Dr. Ying Liu is presently an Assistant Professor with the Department of Industrial and Systems Engineering at the Hong Kong Polytechnic University. He obtained his Bachelor and Master from Chongqing University in 1998 and 2001 respectively, and M.Sc. and Ph.D. from the Singapore MIT Alliance (SMA) at the Nanyang Technological University and the National University of Singapore in 2002 and 2006 respectively. His current research interests focus on design informatics, data mining and text mining, intelligent information processing and management, machine learning, and their joint research and applications in engineering design, manufacturing and medical and healthcare industry for knowledge discovery and management purpose. He is the lead editor for the book “Advances of Computational Intelligence in Industrial Systems” Springer 2008 and he has served as guest editor for several special issues with the Journal of Intelligent Manufacturing, Information. Systems Frontiers and Advanced Engineering Informatics. He is a member with ACM, IEEE, ASME and the Design Society.