

Constructing support vector machine ensemble with segmentation for imbalanced datasets

Qian Li · Bing Yang · Yi Li · Naiyang Deng ·
Ling Jing

Received: 27 June 2011 / Accepted: 22 June 2012 / Published online: 10 July 2012
© Springer-Verlag London Limited 2012

Abstract A novel method, namely ensemble support vector machine with segmentation (SeEn-SVM), for the classification of imbalanced datasets is proposed in this paper. In particular, vector quantization algorithm is used to segment the majority class and hence generates some small datasets that are of less imbalance than original one, and two different weighted functions are proposed to integrate all the results of basic classifiers. The goal of the SeEn-SVM algorithm is to improve the prediction accuracy of the minority class, which is more interesting for people. The SeEn-SVM is applied to six UCI datasets, and the results confirmed its better performance than previously proposed methods for imbalance problem.

Keywords Imbalance dataset · Classification · Support vector machine · Vector quantization · Ensemble learning

1 Introduction

Many real-world datasets are imbalanced, in which most of the instances belong to one class and far fewer instances belong to another one, yet usually more interesting class, such as fraud detection, risk management, medical diagnosis, bioinformatic, text classification and information retrieval [1]. In the case of binary classification, imbalanced

dataset means that the number of negative samples is much larger than that of positive ones. The classification for imbalanced datasets is a quite pervasive and ubiquitous problem, so it has received a lot of attentions in the machine learning community. This interest gave rise to two important workshops held in 2000 and 2003 at the AAAI [2] and ICML [3] conferences, respectively. A follow-up workshop, PAKDD, was conducted in 2009 [4]. Despite the fact that many workshops have already been held to discuss about the topic, a large number of practitioners plagued by the problem are still working in isolation [5, 6].

One of the main challenges of imbalance problem is that the small classes are often more useful, but standard classifiers tend to be overwhelmed by the large classes and ignore the small ones. To handle the problem, the main idea is to rebalance the data distribution and deflect the decision boundary; so far, a number of approaches have been proposed. Generally, the common methods can be divided into two main directions: at the data level, resampling approaches such as undersampling and oversampling; at the algorithms level, algorithm-based approaches such as cost-sensitive learning, one-class learning and ensemble learning [6].

Undersampling [7] is the simplest way to rebalance a dataset by under-sampling the majority class to match the size of the minority class. Obviously, this method can potentially remove certain important samples and results in information loss for the majority class. In contrast, oversampling duplicates minority instances in the hope of reducing class imbalance, but it can easily lead to overfitting and introduce an additional computational task if the dataset is already large. SMOTE algorithm [8], as one of the successful oversampling methods, is expected to alleviate the over-fitting problem by adding the “new” minority instances. However, just because of this, it may introduce excessive noise.

Q. Li · B. Yang · N. Deng · L. Jing (✉)
College of Science, China Agricultural University,
Beijing 100083, People's Republic of China
e-mail: jingling@cau.edu.cn

Y. Li
Department of Mathematics, Beijing University of Posts and
Telecommunications, Beijing 100876,
People's Republic of China

Algorithm-based approaches are designed to modify the classifiers based on their inherent characteristics in order to adapt them to the datasets. Cost-sensitive learning seems to be the best approach of this kind by assigning distinct costs to the training instances. Usually, this strategy gives higher learning cost to the samples in the minority class and this scheme is generally merged into the common formulation of classification algorithms. Consequently, various experimental studies of this type have been performed using different kinds of classifiers [9–11]. One-class learning is another strategy for imbalance problem, the advantage of which is that discarding the distractive majorities, the “space” where minority data resides could be better determined [12, 13]. But there is a disadvantage of one-class learning: the classifier may over-fit the training minority class.

On the basis of statistical learning theory, support vector machine (SVM) was proposed as computationally powerful tools for supervised learning including both classification and regression [14, 15]. SVM have established themselves as a successful approach for various machine learning tasks. But as for the extremely imbalanced datasets, the decision boundary of SVM obtained from the training data is largely biased toward the minority class. Many researchers have studied and proposed plenty of effective techniques to adjust the decision boundary [16, 17]. Ensemble learning strategies are also employed to construct new algorithms to improve SVM’s performance for the imbalance learning problem [18].

In this paper, we propose a new algorithm, namely ensemble support vector machine with segmentation (SeEn–SVM), for the classification of imbalanced datasets. In particular, vector quantization (VQ) algorithm is used to segment the majority class and hence generate some small datasets that are of less imbalance than original one, and two different weighted functions are proposed to integrate all the results of basic classifiers. The goal of the SeEn–SVM algorithm is to improve prediction accuracy of the minority class, which is more interesting for people. Of course, it will be void if the algorithm decrease specificity too much.

The rest of the paper is organized as follows. In Sect. 2, we provide the background of SVM, ensemble learning as well as VQ principle. The proposed ensemble framework is presented in Sect. 3. In Sect. 4, we report the experiment results, and Conclusion is given in Sect. 5.

2 Background

2.1 Support vector machine (SVM)

Generally, support vector machine is used for classification. The essential idea of SVM is to search a linear

separating hyperplane that maximizes the distance between two classes of data to create a classifier.

For a binary classification problem, the training dataset consists of samples $x_i \in \mathcal{R}^n$, $i = 1, \dots, l$ with corresponding class values $y_i \in \{-1, 1\}$, $i = 1, \dots, l$. The formulation is a linear soft margin algorithm, which is used to solve the following optimization problem:

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l \xi_i \\ \text{s.t.} \quad & y_i((w \cdot x_i) + b) \geq 1 - \xi_i, \quad i = 1, \dots, l \\ & \xi_i \geq 0, \quad i = 1, \dots, l \end{aligned} \quad (1)$$

where the predefined parameter C is a trade-off between training accuracy and generalization, ξ_i is the slack variable, $w \in \mathcal{R}^n$ is a weight vector that defines a direction perpendicular to the hyperplane of the decision function, while b is a bias that moves the hyperplane parallel to itself. The decision function is presented as follows:

$$f(x) = \text{sgn}((w \cdot x) + b) \quad (2)$$

The solution of this optimization problem is given by solving the corresponding dual problem with introduced Lagrange multiplier α

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l y_i y_j (x_i \cdot x_j) \alpha_i \alpha_j - \sum_{j=1}^l \alpha_j \\ \text{s.t.} \quad & \sum_{i=1}^l y_i \alpha_i = 0 \\ & 0 \leq \alpha_i \leq C, \quad i = 1, \dots, l \end{aligned} \quad (3)$$

Generally, the solution α^* of the dual problem is sparse, and the corresponding decision hyperplane depends only on few “support vectors.”

$$f(x) = \text{sgn}((w^* \cdot x) + b^*) = \text{sgn} \left(\sum_{i=1}^l \alpha_i^* y_i (x_i \cdot x) + b^* \right) \quad (4)$$

Actually, not every problem is classified linearly; SVM firstly maps n dimensional input data into a higher dimensional feature space via a nonlinear function $\Phi(\cdot)$. Using the “kernel trick,” a kernel function $K(x_i, x_j)$ is used to substitute the dot product of mapping function $(\Phi(x_i), \Phi(x_j))$. Thus, the explicit computing of $\Phi(\cdot)$ can be avoided, and the decision function is obtained as follows:

$$f(x) = \text{sgn} \left(\sum_{i=1}^l \alpha_i^* y_i K(x_i \cdot x) + b^* \right) \quad (5)$$

2.2 Ensemble learning

Ensemble learning is a machine learning paradigm where multiple learners are trained to solve the same problem. In contrast to individual classifier, ensemble methods try to

construct a set of base classifiers and then classify new samples by combining their outputs in some way. It has been accepted widely that ensemble classification has promising capabilities in improving classification accuracy. Obviously, there exist two steps in constructing ensemble scheme: the first step is to generate a number of base classifiers and the second is to aggregate the base classifiers by weighted or unweighted voting typically.

To obtain a good ensemble, the base learners should be as more accurate as possible, and as more diverse as possible [19]. In other words, the generalization performance of ensemble classifiers depends on the accuracy and diversity trade-off of the base classifiers. In practice, the diversity of the base learners can be introduced from different channels, such as subsampling the training samples, manipulating the attributes, manipulating the outputs, injecting randomness into learning algorithms or even using multiple mechanisms simultaneously. The employment of different base learner generation processes and/or different combination schemes leads to different ensemble methods [20].

2.3 Vector quantization (VQ)

Vector quantization, also called “block quantization” or “pattern matching quantization,” is often used in lossy data compression [21]. Given a training dataset consisting of l vectors, $T = \{x_1, x_2, \dots, x_l\}$, the vector quantizer maps T into a finite set of vectors $Code = \{c_1, c_2, \dots, c_M\}$, each vector $c_i, (i = 1, 2, \dots, M)$ is called a codevector, and the set of all the codevectors is called a codebook. Then the whole region is partitioned by the codevectors into a set of subregions, so-called “Voronoi Region,” $V = \{V_1, V_2, \dots, V_M\}$, and it is defined by:

$$V_i = \{x \in T : \|x - c_i\| \leq \|x - c_j\|, \forall j \neq i\}, \quad i = 1, \dots, M \tag{6}$$

$$\bigcup_{i=1}^M V_i = T, \quad \bigcap_{i=1}^M V_i = \Phi \tag{7}$$

Vectors within a region V_i are represented by their codevector $\varphi(x_k) = c_i$, if $x_k \in V_i$. Figure 1 shows the division of VQ.

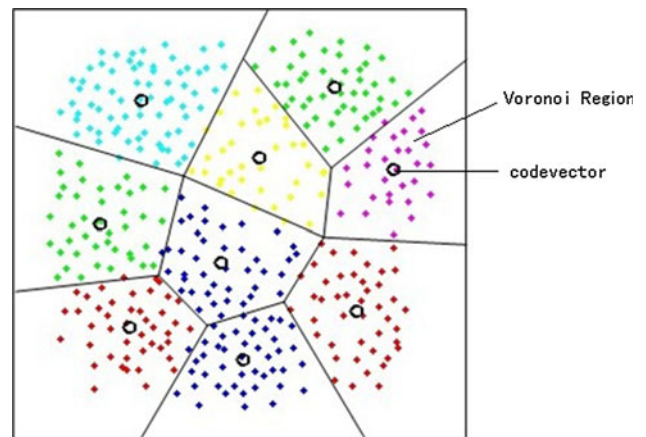


Fig. 1 Division of vector quantization

To find $Code$ and V is to minimize the average distortion which can be given by:

$$D = \frac{1}{Mn} \sum_{i=1}^l \|x_i - \varphi(x_i)\|^2 \tag{8}$$

where n is the dimension of training vectors [22].

3 Proposed algorithm: SeEn-SVM

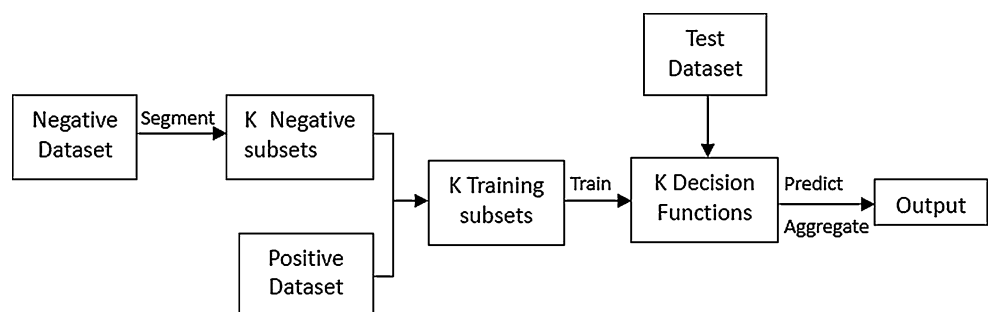
In this paper, we present a new method that focuses on improving classification performance for imbalanced datasets by using an ensemble of SVM integrated with VQ technique. The detailed procedure of the proposed method is given in the following subsections.

3.1 The basic idea of SeEn-SVM

There are three main steps in our proposed algorithm, which includes segmentation, training and aggregation. An overview of the procedure is given in Fig. 2.

- (1) First, as mentioned in introduction, undersampling and oversampling have their drawbacks and might result in information loss and over-fitting, respectively. So we start rebalancing the datasets by

Fig. 2 Procedure of SeEn-SVM



segmenting the majority (negative) class; similar to clustering, the idea of VQ technique is applied to divide the majority class into K small data subsets, and the similarity of the data in same subset is much higher than that in two different subsets. Meanwhile, the size of each dataset is allowed to be different and is not necessary to be the same with the size of minority class. What needs us to determine is the number of small data subsets. In SeEn–SVM algorithm, a detailed strategy is constructed for this.

- (2) Then, by combining minority class with each segmented small subset, K less imbalanced training datasets are generated. Subsequently, SVM is used as base learner to train every modified version of the data and K base classifiers can be obtained. Owing to the negative classes in every new training data are distinct completely, the major differences exist among diverse classifiers. This plays an important role in improving generalization ability of an ensemble.
- (3) Finally, after training, we need to aggregate all independent base classifiers into an appropriate combination manner. Instead of majority voting and other combining methods generally used, we formulate two types of functions to determine the weight for every base classifier according to the distance between negative dataset and positive dataset in each training process. For a new test sample, the final prediction is produced by all of the base classifiers.

3.2 Algorithm of SeEn–SVM

Given the minority (positive) training set P and the majority (negative) training set N , there will be two related important problems when implementing our proposed SeEn–SVM: the first is how many small negative subsets should be generated from N ; the second is how to formulate the weighting functions that aggregate all independent base classifiers into an appropriate combination manner. The algorithm of SeEn–SVM is shown in Table 1.

Step 2 of learning process displays a concrete strategy for choosing an appropriate M , which consists of two aspects:

- (1) It is known that SVM may perform well while the imbalance ratio is moderate and some observations have demonstrated that SVM could be robust and self-adjusting, so when $m_1/m_2 > 1 : 5$, that means the ratio between the number of positive samples and negative samples could not be too large to influence their performance. Hence, let $M = 1$, which is equivalent to do nothing for negative training data, and only SVM is utilized to train the original dataset D .
- (2) However, due to extremely skewed data distribution, SVM modeled on the original training dataset is prone to classify most of the samples to be negative. When $m_1/m_2 \leq 1 : 5$, that means the imbalance ratio is larger than the first case, and this may lead to a high false negative rate, so we let $M = 2^{n_0}$, where n_0 can

Table 1 SeEn–SVM algorithm

Input: training set $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_l, y_l)\}$;

Positive training data P ; negative training data N ; (the size of P : m_1 ; the size of N : m_2 ; $m_1 \leq m_2$ and $m_1 + m_2 = l$)

The number of codevectors M in VQ algorithm, M is also the number of negative subsets;

Parameters: C in SVM classifier

Learning Process:

1. Given $M = 1$, compute the codevector c_0 of data P with VQ algorithm;

2. Calculate $q = m_2 / (5 * m_1)$

If $q < 1$, that is $m_1/m_2 > 1 : 5$, let $M = 1$;

If $q \geq 1$, solve $\min_n |q - 2^n|$, ($n = 1, 2, \dots$), and get the optimal solution n_0 , let $M = 2^{n_0}$;

3. Data N is divided into M subsets with VQ algorithm; they are V_i , ($i = 1, \dots, M$), meanwhile, the codevector C_i of each negative subset can be obtained ($i = 1, \dots, M$);

4. **for** $i = 1, \dots, M$

5. Combine minority class P with each segmented small subset V_i , let $D_i = [V_i, P]$

6. Obtain decision hyperplane $g_i(x)$ by solving SVM optimization problem (3) on the dataset $D_i = [V_i, P]$.

7. $d_i = \text{distance}(c_i, c_0)$;

8. **end**

9. $v_i = \exp(-d_i/d_m)$, $i = 1, \dots, M$ (d_m is the median of all distances)

Output: $F(x) = \text{sgn}(\sum_{i=1}^M v_i g_i(x))$

be equal to any positive integer, which is subjected to $\min_n |q - 2^n|$, ($n = 1, 2, \dots$). In this situation, at least one division should be done to the negative class.

Of course, M is not only the number of negative subsets but also the number of component members used in ensemble process to achieve a good performance. In order to be more reasonable and feasible, after n_0 is computed, M can be chosen among 2^{n_0-1} , 2^{n_0} and 2^{n_0+1} . Because n_0 is only related to imbalance ratio, but except for this, the size and distribution of data may also influence the performance of classifiers, the capability of SeEn-SVM algorithm would be limited if only $M = 2^{n_0}$ is considered.

SVM assumes that only SVs are informative to classification and other samples can be considered redundant. For imbalanced classification, SVM would be less affected by the negative instances that lie far away from the learned boundary even if there are many of them. In addition, some of the negative SVs may not be the most informative or even noisy. The goal of SeEn-SVM algorithm manages to cope with this problem. Step 9 of the learning process in Table 1 shows the formulation of the weighted function, which decides the importance of the i -th ($i = 1, \dots, M$) base classifier. It is clear that the closer the distance between the codevector of the i -th negative dataset and the codevector of positive dataset, the larger the weight value of the i -th classifier is. And this does not satisfy the linear relationship but the exponential relationship. This strategy can increase the number of SVs especially the negative SVs that are far away from the real decision boundary and hence alleviate the boundary skewness.

3.3 Improvement of SeEn-SVM

For a large and imbalanced dataset, there may be many redundant or noisy negative samples, even there are some positive samples in the negative class and we do not know which ones are false. This phenomenon is common in bioinformatics problems, such as gene function prediction [23], alternative splicing sites identification [24] and horizontal gene transfer detection [25] and so on. It is a well-known fact that the false negative samples may distribute closed to true positive data samples. In this sense, step 9 of

learning process in algorithm might be reformulated as follows:

$$v_i^* = \exp[-|d_i - d_m| / (d_{\max} + d_m - d_i)], \quad i = 1, \dots, M \tag{9}$$

where d_m and d_{\max} are the median and maximum of M values d_i , ($i = 1, 2, \dots, M$) that represent the distances between the codevectors of negative subsets and the codevector of positive dataset, respectively. v_i^* is different from v_i by putting the highest weight to the classifier $g_i(x)$ whose negative dataset is in the “middle” of original negative dataset not to the closest one from the positive dataset. For the rest classifiers, the farther the distance from the positive dataset, the smaller weight the value is. Because negative subset closer to the positive dataset is more useful than farther ones in classification despite some noisy points, the closer subsets’ weights would be larger than the farther ones’. Thus, it is hoped to improve the generalization performance of SeEn-SVM algorithm.

4 Experiments and results

In this section, six UCI imbalanced datasets are used in our experimental study to test our proposed method, namely Glass (7), Segment (1), Abalone (7), Satalog (4), Letter (4) and letter (1). The numbers in the parentheses indicate which classes are selected as minority (positive) class, and all others are used as majority (negative) class.

These datasets are often appeared in related works about imbalanced research. The basic information about these datasets is summarized in Table 2 including the size of every dataset, the number of features, and the number of positive samples and negative samples in each dataset. These datasets are seriously selected to vary in data size (from several hundreds to tens of thousands) and imbalance ratio, the first two datasets are mildly imbalanced, while the rest ones are highly imbalanced as less than 10 % samples are positive.

In order to evaluate the classifiers on imbalanced datasets, we use G-mean instead of prediction accuracy to measure the performance of algorithm since it combines the values of both sensitivity and specificity. Because

Table 2 Characteristics of datasets used in this paper

Dataset	# of samples	# of features	# pos/# neg	Ratio of pos (%)
Glass7	214	9	29/185	13.55
Segment1	2,310	19	330/2,280	14.28
Abalone7	4,177	7	391/3,786	9.37
Satalog4	6,435	36	626/5,809	9.73
Letter4	20,000	16	805/19,185	4.03
Letter1	20,000	16	789/19,211	3.95

of samples, the total number of instances; # of features, the number of attributes of instances; # pos, the number of positive instances; # neg, the number of negative instances

Table 3 G-mean values of five methods on six datasets

Dataset	SVM	Under-sampling	SMOTE	Cost-sensitive learning	SeEn–SVM (parameters)
Glass [16]	0.8666	0.8801	0.8771	0.9199	0.9465 ± 0.016 ($M = 2, v$)
Segment [16]	0.9792	0.9918	0.9765	0.9950	0.9943 ± 0.005 ($M = 2, v^*$)
Abalone [28]	0	0.765	0.742	0.412	0.7755 ± 0.017 ($M = 4, v$)
Statlog [28]	0.7678	0.871	0.862	0.761	0.8648 ± 0.004 ($M = 2, v^*$)
Letter1 [28]	0.9712	0.994	0.995	0.988	0.9951 ± 0.001 ($M = 4, v^*$)
Letter4 [18]	0.8876	0.5354	0.7078	0.8890	0.9660 ± 0.006 ($M = 4, v^*$)
Mean	0.7454	0.8396	0.86	0.8247	0.9292

For each dataset, the best result is shown in boldface

overall accuracy is not sufficient any more in evaluating the classifier with highly skewed dataset, sensitivity and specificity are then usually adopted to monitor classification performance on two classes separately and they are defined as

$$\text{sensitivity} = \frac{\#\text{true_positive}}{\#\text{true_positive} + \#\text{false_negative}} \quad (10)$$

$$\text{specificity} = \frac{\#\text{true_negative}}{\#\text{true_negative} + \#\text{false_positive}} \quad (11)$$

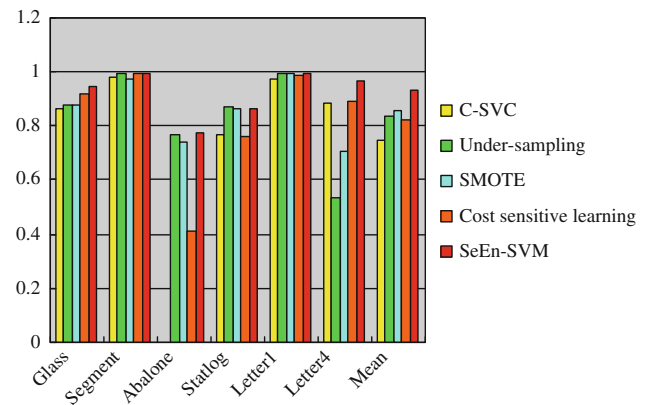
Based on these two metrics, G-mean metric suggested by Kubat et al. [26] is defined as follows:

$$\text{G-mean} = \sqrt{\text{sensitivity} \cdot \text{specificity}} \quad (12)$$

G-mean metric indicates balanced classification ability between positive class accuracy and negative class accuracy by taking the geometric mean of sensitivity and specificity, and obviously, if a classifier is highly biased toward one class, the G-mean value would be low. Hence, G-mean metric is used to compare the classification performance of models with imbalanced datasets in our experiments.

In the following, SeEn–SVM algorithm is compared with four other popular methods, which are SVM, under-sampling, SMOTE and cost-sensitive learning. In our experiments, we use 10-fold cross-validation to train our classifier since it provides more realistic results. Each dataset in Table 2 is trained respectively by all methods, while finding the optimal parameters the final 10-fold cross-validation results can be obtained. SVM is implemented in LIBSVM [27], and SeEn–SVM is implemented by writing procedure in Matlab, but the results of other three methods on different datasets are referred to different literatures as shown in Table 3. Our G-mean measure is gained by running the experiment 10 times with different training and test datasets; besides the measurement of G-mean, the standard error of the G-mean is also given to our classifier. Table 3 also illustrates the comparison of the G-mean value by our method and the rest ones.

In executing SeEn–SVM algorithm, for every dataset, radial basis functions (RBF) are used as kernel functions,

**Fig. 3** Bar graph of compared methods' results

and we choose the same cost parameter C in every component SVM. But for SeEn–SVM algorithm, the segmentation number M and weighted function should be determined. In last column of Table 3, the results of selected M and weighted function are given in brackets; v or v_i^* represents which weighted function is used in decision-making process.

Table 3 shows obviously that the performance of SeEn–SVM algorithm is better than the previously proposed methods, and the G-mean value of SeEn–SVM is higher than that of other methods on most of the experiment datasets. There are only two datasets on which SeEn–SVM does not perform best, but the G-mean value of SeEn–SVM is much close to the best result. The last line of Table 3 displays the average G-mean values over all datasets of each methods, the result of SeEn–SVM is higher overwhelmingly than other methods, and the standard errors show that SeEn–SVM is stable too. Figure 3 gives the bar graph of compared methods' results, which shows the results gained by all of the methods more visually, and the relative improvements can be gotten from the bar graph.

Table 4 displays the sensitivity and specificity of SVM and SeEn–SVM algorithm on all datasets. Note that SVM has almost perfect specificity, but poor sensitivity because it tends to classify all samples as negative. Any proposed

Table 4 Sensitivity and specificity values of SVM and SeEn–SVM on six datasets

Dataset	SVM		SeEn–SVM	
	Sensitivity	Specificity	Sensitivity	Specificity
Glass	0.75	1	0.9064	0.9884
Segment	0.9589	1	0.9973	0.9914
Abalone	0	1	0.8484	0.7090
Statlog	0.6	0.9826	0.8505	0.8792
Letter1	0.9438	0.9995	0.9987	0.9914
Letter4	0.7892	0.9982	0.9987	0.9344
Mean	0.5487	0.9967	0.9333	0.9156

For each dataset, the best result of sensitivity is shown in boldface

algorithm for imbalanced datasets sacrifices inevitably some specificity in order to improve the sensitivity and SeEn–SVM has no exception. But SeEn–SVM decreases specificity in a less extent and simultaneously increases sensitivity greatly.

5 Conclusions

This paper introduces a novel approach for learning from imbalanced datasets through making an ensemble of SVM classifiers with VQ techniques. Although SVM has shown an outstanding performance in many research areas and it can adjust itself well to some degree of data imbalance, the performance of SVM can still be influenced by data imbalance.

To cope with the problem associated with imbalanced datasets, SeEn–SVM segments the original negative data into some small subsets firstly and then trains M new small training datasets to learn base classifiers by SVM, and finally makes decision by aggregating all the base learners with appropriate weighted function. Through theoretical analysis and empirical studies, we demonstrate this algorithm effective and applicable. In the experiment studies, the new algorithm was applied to six UCI datasets, and the results confirmed its better performance than previously proposed methods for imbalance problem.

Acknowledgments This work is supported by the National Natural Science Foundation of China (No. 10971223, No. 11071252) and Chinese Universities Scientific Fund (2011JS039, 2012YJ130). The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

References

- Chawla NV, Japkowicz N, Kotcz A (2004) Editorial: special issue on learning from imbalanced data sets. *SIGKDD Explor* 6(1):1–6
- Japkowicz N (2000) Learning from imbalanced data sets: a comparison of various strategies. In: *Proceedings of the AAAI'2000 workshop on learning from imbalanced data sets*, pp 10–15
- Chawla NV, Japkowicz N, Kolcz A (Eds.) (2003) In: *Proceedings of the ICML'2003 workshop on learning from imbalanced data sets*
- Chawla NV, Japkowicz N, Zhou ZH (2009) In: *PAKDD'2009 workshop: data mining when classes are imbalanced and errors have costs*, Thailand
- Nguwi YY, Cho SY (2010) An unsupervised self-organizing learning with support vector ranking for imbalanced datasets. *Expert Syst Appl* 37(12):8303–8312
- Tian J, Gu H, Liu WQ (2011) Imbalanced classification using support vector machine ensemble. *Neural Comput Appl* 20(2):203–209
- Kubat M, Matwin S (1997) Addressing the curse of imbalanced training sets: One-sided selection. In: *Proceedings of the fourteenth international conference on machine learning*, pp 179–186
- Chawla NV, Bowyer K, Hall L, Kegelmeyer WP (2002) SMOTE: synthetic minority over-sampling technique. *J Artif Intell Res* 16:321–357
- Domingos P (1999) MetaCost: a general method for making classifiers cost sensitive. In: *Proceedings of the fifth international conference on knowledge discovery and data mining*, pp 155–164
- Elkan C (2001) The foundations of cost-sensitive learning. In: *Proceedings of the seventeenth international joint conference on artificial intelligence*. Morgan Kaufmann, San Francisco, pp 973–978
- Drummond C, Holte RC (2003) C4.5, class imbalance and cost sensitivity: why under-sampling beats over-sampling. In: *Workshop on Learning from Imbalanced Datasets II*, held in conjunction with ICML 2003
- Manevitz LM, Yousef M (2001) One-class SVMs for document classification. *J Mach Learn Res* 2(2):139–154
- Raskutti B, Kowalczyk A (2003) Extreme re-balancing for SVMs: a case study. In: *Workshop on learning from imbalanced data sets II*, international conference on machine learning
- Cortes C, Vapnik V (1995) Support-vector networks. *Machine Learn* 20:273–297
- Deng NY, Tian YJ, Zhang CH (2012) Support vector machines: theory, algorithms, and extensions. CRC Press (in press)
- Akbani R, Kwok S, Japkowicz N (2004) Applying support vector machines to imbalanced datasets. In: *Proceedings of ECML 2004*. LNCS (LNAI), 3201, pp 39–50
- Yang CY, Wang JJ, Yang JS and Yu GD (2008) Imbalanced SVM learning with margin compensation. In: *Proceedings of ISNN 2008, Part I*, LNCS 5263, pp 636–644
- Benjamin X, Wang, Japkowicz N (2008) Boosting support vector machines for imbalanced data sets. In: *Proceedings of ISMIS 2008*, LNAI 4994, pp 38–47
- Krogh A, Vedelsby J (1995) Neural network ensembles, cross validation, and active learning. *Advances in neural information processing systems 7*. MIT Press, Cambridge, MA, pp 231–238
- Zhou ZH, Wu J, Tang W (2002) Ensembling neural networks: many could be better than all. *Artif Intell* 137(1–2):239–263
- Gersho A, Gray RM (1992) Vector quantization and signal compression. Kluwer, Dordrecht
- Yu T, Debenham J, Jan T, Simoff S (2006) Combine vector quantization and support vector machine for imbalanced datasets. In: *TFTP international federation for information processing, 2006*, pp 217–227
- Zhao XM, Wang Y, Chen LN, Kazuyuki A (2008) Gene function prediction using labeled and unlabeled data. *BMC Bioinform* 9:57–62
- Dror G, Sorek R, Shamir R (2005) Accurate identification of alternatively spliced exons using support vector machine. *Bioinformatics* 21(7):897–901

25. Ning H, Yang B, Cui J, Jing L (2009) Detection of horizontal gene transfer in bacterial genomes. In: Proceedings of the third international symposium on optimization and systems biology, pp 229–236
26. Kubat M, Holte R, Matwin S (1997) Learning when negative examples abound. In: Proceedings of the 9th European conference on machine learning. London: Springer, Heidelberg, 1224, pp 146–153
27. Hsu C-W, Chang C-C, Lin C-J (2008) A practical guide to support vector classification. <http://www.csie.ntu.edu.tw/~cjlin>
28. Liu XY, Wu JX, Zhou ZH (2009) Exploratory undersampling for class-imbalance learning. *IEEE Transact Syst Man Cybern Part B Cybern* 39(2):539–550