BOOTSTRAP-BASED SVM AGGREGATION FOR CLASS IMBALANCE PROBLEMS

S. Sukhanov, A. Merentitis, C. Debes

AGT International
Darmstadt, Germany
{ssukhanov, amerentitis, cdebes}@agtinternational.com

J. Hahn, A.M. Zoubir

Signal Processing Group
Technische Universität Darmstadt, Germany
{jhahn, zoubir}@spg.tu-darmstadt.de

ABSTRACT

Support Vector Machines (SVMs) are considered to be one of the most powerful classification tools, widely used in many applications. However, in numerous scenarios the classes are not equally represented and the predictive performance of SVMs on such data can drop dramatically. Different methods have been proposed to address moderate class imbalance issues, but there are few methods that can be successful at detecting the minority class while also keeping high accuracy, especially when applied to datasets with significant level of imbalance. In this paper, we consider SVM ensembles that are built by using a bootstrap-based undersampling technique. We target reducing the bias induced by class imbalances via multiple undersampling procedures and then reduce the variance using SVM ensembles. For combining the SVMs, we propose a new technique that deals with class imbalance problems of varying levels. Experiments on several datasets demonstrate the performance of the proposed scheme compared to state-of-the-art balancing methods.

Index Terms— SVMs, Imbalanced dataset, Undersampling, Ensemble learning methods

1. INTRODUCTION

Class imbalance problems occur in many practical machine learning applications where one class has significantly less training instances than other classes. In such scenarios, it is usually more costly to misclassify the instances of the class that is highly underrepresented. Typical examples include rare disease detection, credit card fraud [1], oil spills [2] and generally detection or classification of rare events. The performance of classification algorithms can be strongly affected by imbalanced training datasets. This is due to several reasons [3], with the primary one being that the objective of many algorithms is to maximize the overall accuracy. As the minority class does not significantly contribute to this error a strong decrease of its classification accuracy is observed [4].

Different techniques have been proposed in the literature to deal with class imbalance problems and increase the performance of machine learning algorithms. These techniques can be divided into two groups, namely external and internal methods [5]. Internal approaches comprise the advances of the learning algorithm itself and take into account the imbalance problem while learning. This includes, for example, methods that assign different class costs so that the minority class gets higher costs than the majority class [6]. There are also methods that consider kernel modification in order to account for a bias of either the decision boundary or a fit of the training data [6, 7, 8].

External balancing approaches refer to algorithms that operate on the training data directly before the actual training. The most popular ones are resampling techniques, namely undersampling the majority class [9] and oversampling the minority class [10]. Synthetic Minority Over-Sampling Technique (SMOTE) [11] is practically on of the most popular and successful oversampling method used in many applications. Operating in the feature space, SMOTE balances the dataset by introducing new synthetic examples of the minority class. The main drawback of this approach is the overhead that SMOTE creates and it is still an open question how to determine the amount of oversampling needed. There are many other advanced SMOTE-based approaches that are proposed in the literature e.g. in [12, 13]. The most popular one is Borderline SMOTE [14] which is the standard SMOTE method applied to the borderline samples only, introducing less overhead. SMOTE and other oversampling-based balancing methods brings an overhead that can drastically increase the training time of the classifier. At the same time some methods of oversampling can lead to overfitting [10, 11].

Undersampling, on the other hand, is computationally efficient and an easy way to balance the training set. Since there is a danger of removing potentially useful data, undersampling is often used together with resampling techniques. In [15], several roughly balanced subsets are created by undersampling according to the binomial distribution and then used to learn independent learners. In [16] a method called EasyEnsemble was proposed where undersampling is used to exactly equalize the amount of the majority and minority data examples for every bootstrap sample. Every sample is then used to train AdaBoost ensembles that are combined, eventually, into one ensemble. An advanced version of EasyEnsemble described in [16] is called BalanceCascade where each bootstrapping sample is obtained sequentially by removing

redundant examples from the original majority subset.

In this paper, we compare several techniques that deal with class imbalance issues. We show that combining undersampling together with a bootstrap-based aggregation approach allows achieving higher classification performance compared to traditional SMOTE or other common techniques. The main contributions of the paper are:

- we propose a new ensemble combination technique, the Weighted Balanced Combiner (WBC), that accounts not only for the overall accuracy, but for the accuracy of each class. This allows to obtain better overall classification performance compered to existing bootstrap-based aggregation methods
- we systematically evaluate the performance of the proposed technique and state-of-the-art methods in a variety of datasets with different imbalance levels
- we discuss the effect of imbalance on the ensemble combination scheme and the tradeoff between detecting the minority class while keeping high accuracy, providing guidelines when possible

The paper is structured as follows. In Section 2, we give a brief overview of the SVM and the class imbalance effect. Sections 3 describes the proposed bootstrap-based aggregation approach. Section 4 provides experimental results on various datasets. We conclude the paper with a discussion in Section 5.

2. SVMS AND CLASS IMBALANCE

SVMs [17] are considered to be one of the most powerful classifiers and are widely used in different applications of pattern recognition and machine learning. By utilizing the structural risk minimization concept [17] and the kernel trick [18], SVMs are able to find a linear separating hyperplane in a high dimensional dataspace while achieving good generalization properties. Furthermore, it has been shown that SVMs are not affected by moderate imbalance problem [9] since only a small amount of training samples (support vectors) are responsible for the separating hyperplane positioning. However, SVMs and other discriminative methods are challenged in the presence of high imbalanced training data. In the sequel, we will consider the basic SVM formulation and analyze the reasons for degradation when class imbalance is present.

Consider a binary classification setting and let $\{(\mathbf{x}_i, y_i)\}$, $i = 1, ..., N$ be the training dataset, where $\mathbf{x}_i$ is a training sample and $y_i \in (-1, +1)$ its corresponding label. Given a training set, the SVM finds an optimal hyperplane that separates two classes with maximum margin. The primal formulation is

$$\min \frac{1}{2}(\|\mathbf{w}\|)^2 + C\sum_i \xi_i$$

$$\text{s.t.} \quad y_i(\mathbf{w}\cdot\Phi(\mathbf{x}_i)+b) \geq 1-\xi_i \quad \text{and} \quad \xi_i \geq 0, \forall i$$

where $\mathbf{w}$ is a weight vector, b is the bias of the separating hyperplane, ξ_i is the margin error for the i-th training example and C is a parameter that sets the relative importance of maximizing the margin and minimizing the training error.

The SVM formulation can be adopted to the cost-sensitive case, where different types of errors are assigned different costs. The primal formulation is then

$$\min \frac{1}{2}(\|\mathbf{w}\|)^2 + C\gamma \sum_{i\in I_+} \xi_i + C(1-\gamma)\sum_{i\in I_-} \xi_i$$

where I_- and I_+ are the set of samples that belong to the majority and minority class, respectively.

As has been mentioned, SVMs often suffer from biased decision boundaries toward the minority class, especially when the imbalance level is high. Cost-sensitive extensions of SVMs are a good example of internal methods applied to address class imbalance issues; however, it has been shown that typically only problems with a small amount of class imbalance can be handled using cost-sensitive SVMs [19]. On the other hand, almost all external balancing approaches aim to eliminate the biased decision boundaries effect and can potentially be effective even in cases of high imbalance. For SVMs, it is also of significant importance to have an approach that balances the classes by reducing the size of learning datasets, as non-linear SVMs are sensitive to its size in view of computational complexity.

A common external technique to deal with class imbalances is random undersampling (RU). In RU, the samples of the majority class are being removed randomly, while the samples of the minority class are kept untouched. By doing so, a decrease of bias caused by the effect of imbalance can be achieved. However, RU can discard potentially important information and, as a consequence, different undersampling procedures can lead to different decision boundaries. This implies that RU is a high-variance balancing strategy and cannot be directly used in many practical applications despite its computational attractiveness compared to cost-sensitive SVMs.

3. BOOTSTRAP-BASED SVM AGGREGATION

To leverage the advantages of RU balancing and handle the effect of non-deterministic decision boundaries, we consider Bootstrap Aggregating (Bagging). Bagging is a variance-reduction method that has been shown to work successfully when applied to different classifiers [20, 21]. First introduced by Breiman [22] to create ensembles of learning algorithms, the key idea of Bagging is to sample from the original dataset several times. By doing this, the variability of the quantities of interest can be assessed [23, 24]. In Bagging, each bootstrap realization is used to train a different model and the results are averaged or otherwise combined to provide the final prediction of the ensemble.

In this paper, we use the bootstrap to handle class imbalances. Sampling is not performed on the whole original

dataset, but only on the majority class subset ensuring that its number of samples is set equal to the number of the minority class samples from the original dataset. The minority class samples are combined with each bootstrap realization to create balanced training subsets. Each subset is then used to independently train K different SVMs.

The most important step in creating SVM ensembles is to define a rule, according to which the result of each individual SVM is combined to obtain a final decision. Combining several independent classifiers can be done in different ways, for example, operating at the class label or at the score (estimated posterior) level [25]. In previous works on bagging for class imbalance problems [15, 16], simple average and majority voting combiners were used. As opposed to these approaches, we consider a new scheme, the Weighted Balanced Combiner (WBC), that is detailed in the sequel.

Let K be the number of base SVMs, $f_k, k \in \{1...K\}$ be a decision function of the k^{th} SVMs and F be the final ensemble aggregating model. The output of the aggregating model $f(\mathbf{x})$ is then $f(\mathbf{x}) = F(f_1(\mathbf{x}), ..., f_k(\mathbf{x}))$. We propose a new method for combining the base classifier outputs on the class label level which is motivated by the poor performance of simple majority voting schemes, such as the Weighted Majority Vote combiner [25] when having a class imbalance problem. We suggest to assign weights to each of the base learners, however, keeping in mind that an overall accuracy-based weight assignment is not an appropriate approach for class imbalance problem. Instead, for each weak learner k, we assign w_k with the following properties:

$w_k \to 0$, if k^{th} weak learner performs bad at least on one class

$w_k \to 1$, if k^{th} weak learner performs good on all classes

To follow this rule, we propose to determine the weights w_k based on the class-specific accuracies as opposed to overall accuracy. We suggest combining the class-specific accuracies in such a way that the final individual SVM weight reflects how well the classifier deals with class imbalance. For an L-class problem and the k-th base learner, we have

$$\frac{1}{w_k} = \frac{1}{L}\left(\frac{1}{\text{acc}_k^{(1)}} + \frac{1}{\text{acc}_k^{(2)}} + ... + \frac{1}{\text{acc}_k^{(L)}}\right)$$

where $\text{acc}^{(l)}, l \in \{1, ..., L\}$ is the class-specific accuracy for the l-th class. For two class problems that are the focus in this paper we have

$$w_k = 2 \cdot \frac{\text{acc}_k^{(-)} \cdot \text{acc}_k^{(+)}}{\text{acc}_k^{(-)} + \text{acc}_k^{(+)}}$$

where $\text{acc}^{(-)}$ and $\text{acc}^{(+)}$ are the accuracies of the majority and minority classes, respectively. The aggregated classifier is then obtained as

$$f(\mathbf{x}) = \text{sgn}\left(\sum_{k=1}^{K} w_k \cdot f_k(\mathbf{x})\right)$$

This formulation assures that a base SVM classifier that performs well only on one class would get a low weight. If an individual SVM classifier performs well on both classes, the proposed combination scheme would assign a high weight to this classifier, leading to a higher contribution to the final decision. The overall procedure is summarized in Table 1.

Table 1. The Bootstrap procedure

Step 0.	*Dataset partitioning*. Split the training dataset S into minority and majority sets S_{min} and S_{maj}
Step 1.	*Bootstrapping*. Sample K times from S_{maj} to get bootstrap realizations $S^k_{\text{bootstr}}, k = 1, \ldots, K$ ensuring that the sample number in each realization is equal to the sample number in S_{min}
Step 2.	*Merging*. Combine each bootstrap realization S^k_{bootstr} with the minority class subset S_{min} to obtain K new balanced training sets S^k_{balanced} (i.e. $S^k_{\text{bootstr}} \cup S_{\text{min}} = S^k_{\text{balanced}}, k = 1, \ldots, K$).
Step 3.	*SVM training*. Train K individual SVMs independently on the balanced training sets S^k_{balanced} to obtain K decision functions $f_k, k = 1, \ldots, K$
Step 4.	*Assigning weights*. Calculate the class-specific accuracies $\text{acc}_k^{(-)}$ and $\text{acc}_k^{(+)}$ on the validation set and assign weights w_k to every base learner as $w_k = 2 \cdot \frac{\text{acc}_k^{(-)} \cdot \text{acc}_k^{(+)}}{\text{acc}_k^{(-)} + \text{acc}_k^{(+)}}$
Step 5.	*SVM combining*. In the operation (or testing) mode combine K outputs to obtain a final decision as $f(\mathbf{x}) = \text{sgn}(\sum_{k=1}^{K} w_k \cdot f_k(\mathbf{x}))$

4. EXPERIMENTAL RESULTS

We conduct two sets of experiments to study the effectiveness of the proposed approach. In our first experiment, we use a synthetic dataset and study the performance of bootstrap-based SVM aggregation with the proposed WBC combiner (BWBC) while varying the level of imbalance. Results of SMOTE are also reported for comparison. In our second experiment, we compare a set of representative state of the art balancing methods with BWBC, using real-world datasets. The performance of the classifiers is measured by the G-mean [10], which is a widely used performance measure when having class imbalance problems. For completeness, tradeoffs on prediction accuracy are also reported.

4.1. Synthetic dataset

In this experiment, we study the classification performance of BWBC in the presence of imbalance using synthetic datasets with different imbalance ratios ($\frac{\#positive\_instances}{\#negative\_instances}$). For that we create twelve synthetic datasets drawn from the same family of functions, however, with different majority-to-minority

class examples ratio. For BWBC we create $K = 100$ bootstrap realizations. This particular K allows us to reach a constraint on the upper bound of standard deviation of G-mean (std $< \varepsilon$), where $\varepsilon = 0.05$ in this paper. Further, 1000 Monte Carlo iterations are conducted. The results of this experiment are depicted in Figure 1. As can be seen, the average G-mean

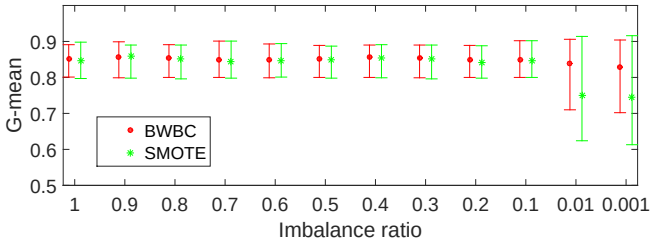

Fig. 1. G-mean of BWBC and SMOTE on a synthetic dataset

value for BWBC over 1000 Monte Carlo iterations stays almost constant, independently of the imbalance degree. The obtained results indicate that the proposed BWBC is robust with respect to the degree of imbalance. SMOTE is performing similarly to BWBC when having moderate imbalance. However, BWBC outperforms SMOTE on the imbalance levels of 0.01 and 0.001.

4.2. Real datasets

To compare existing balancing methods with the BWBC approach, we have chosen five binary classification problems with imbalanced datasets (haberman, cmc, yeast, ecoli, balance) from the UCI [26] repository with different imbalance levels. During the experiment, we average the evaluated measures over 5-fold cross validation runs. For this experiment, we test SVMs without imbalance fixing (SVM), SMOTE, Bagging with Averaging Combiner (BAVC), EasyEnsemble, BalanceCascade and the proposed BWBC method. Table 2 presents G-mean and accuracy measures with their standard deviation for all five datasets and six methods.

4.3. Discussion

As can be observed from Table 2, the proposed method achieves the highest G-mean value in most of considered datasets loosing typically less than 2-5 percentage points of accuracy for a much larger gain in G-mean, even for cases of relatively high imbalance. Comparing BWBC with other bagging-based methods, we see that it outperforms them either in G-mean (having higher or comparable accuracy) or in accuracy (having comparable G-mean), indicating the importance of the combining scheme in ensemble methods. It should be noted that ecoli dataset has too few training samples (only 35 minority class examples) and seven continuous features. For that reason all ensemble-based methods show lower performance than SMOTE for this dataset. Finally, BWBC has also less variance than SMOTE, both in the synthetic and the real datasets. This is mainly due to two reasons: on the one hand, SMOTE has strong randomization effects due to the random interpolation procedure, while on the other hand, our method benefits from the variance reduction effect that is inherent in Bagging. This benefit comes at some computational cost (from Bagging) but the use of undersampling often compensates for the cost, at least for non-linear SVMs.

5. CONCLUSIONS

We have addressed the class imbalance problem by applying a bootstrap-based SVM aggregation method. Furthermore, we proposed a new ensemble combination technique that takes into consideration the accuracies for each class. We have applied this method on synthetic as well as real datasets, achieving better class discrimination performance compared to other state-of-the-art balancing methods, especially for datasets with a high class imbalance level. Finally, we have explained why the proposed method yields lower variance compared to SMOTE and we have experimentally demonstrated this effect across several real world datasets.

		SVMs	SMOTE	BAVC	EasyEnsemble	BalanceCascade	BWBC
haberman	G	0.306 ± 0.069	0.564 ± 0.056	0.623 ± 0.015	0.626 ± 0.024	0.621 ± 0.017	$\mathbf{0.643 \pm 0.015}$
	Acc	$\mathbf{0.736 \pm 0.010}$	0.658 ± 0.032	0.682 ± 0.018	0.627 ± 0.017	0.629 ± 0.014	0.704 ± 0.014
cmc	G	0.320 ± 0.043	0.601 ± 0.021	0.623 ± 0.009	0.632 ± 0.009	0.647 ± 0.009	$\mathbf{0.672 \pm 0.008}$
	Acc	$\mathbf{0.778 \pm 0.005}$	0.713 ± 0.012	0.676 ± 0.008	0.654 ± 0.008	0.673 ± 0.008	0.687 ± 0.008
balance	G	0.000 ± 0.000	0.000 ± 0.000	0.483 ± 0.023	0.456 ± 0.028	$\mathbf{0.534 \pm 0.037}$	0.471 ± 0.039
	Acc	$\mathbf{0.922 \pm 0.000}$	0.869 ± 0.000	0.683 ± 0.021	0.498 ± 0.015	0.494 ± 0.013	0.758 ± 0.023
ecoli	G	0.745 ± 0.021	$\mathbf{0.896 \pm 0.016}$	0.814 ± 0.012	0.824 ± 0.030	0.824 ± 0.019	0.892 ± 0.003
	Acc	$\mathbf{0.934 \pm 0.005}$	0.903 ± 0.007	0.827 ± 0.011	0.852 ± 0.020	0.817 ± 0.021	0.857 ± 0.006
yeast	G	0.665 ± 0.005	0.783 ± 0.007	0.788 ± 0.013	0.791 ± 0.008	0.792 ± 0.008	$\mathbf{0.809 \pm 0.004}$
	Acc	$\mathbf{0.887 \pm 0.002}$	0.811 ± 0.006	0.817 ± 0.004	0.800 ± 0.005	0.801 ± 0.005	0.821 ± 0.003

Table 2. Accuracy (Acc) and G-mean (G) for different methods evaluated on various datasets

6. REFERENCES

[1] P. K. Chan and S. J. Stolfo, "Toward scalable learning with nonuniform class and cost distributions: A case study in credit card fraud detection," *In Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pp. 164–168, 1998.

[2] M. Kubat, R.C. Holte, and S. Matwin, "Machine learning for the detection of oil spills in satellite radar images," *Machine Learning - Special issue on applications of machine learning and the knowledge discovery process*, vol. 30, pp. 195–215, 1998.

[3] V. Ganganwar, "An overview of classification algorithms for imbalanced datasets," *International Journal of Emerging Technology and Advanced Engineering*, vol. 2, pp. 42–47, April 2012.

[4] G. Wu and E. Chang, "Class-boundary alignment for imbalanced dataset learning," *ICML 2003 Workshop on Learning from Imbalanced Data Sets II, Washington, DC.*, pp. 49–56, 2003.

[5] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas, "Handling imbalanced datasets: A review," *GESTS International Transactions on Computer Science and Engineering*, vol. 30, 2006.

[6] K. Veropoulos, C. Campbell, and N. Cristianini, "Controlling the sensitivity of support vector machines," *Proceedings of the International Joint Conference on AI*, pp. 55–60, 1999.

[7] N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylo, "On kernel-target alignment," *Advances in Neural Information Processing Systems 14*, pp. 367–373, 2002.

[8] Z. Yong, Z. Yan-Hong, and L. Zheng-Ding, "A new method to improve the sensitivity of support vector machine based on data optimization," *IEEE International Conference on Robotics, Intelligent Systems and Signal Processing, Proceedings*, vol. 2, pp. 892–896, 2003.

[9] N. Japkowicz and S. Stephen, "The class imbalance problem: a systematic study," *Intelligent Data Analysis 6*, pp. 429–450, 2002.

[10] M. Kubat and S. Matwin, "Addressing the curse of imbalanced training sets: One sided selection," *Proceedings of the Fourteenth International Conference on Machine Learning, Tennesse*, pp. 179–186, 1997.

[11] N. V. Chawla, L. O. Hall, K. W. Bowyer, and W. P. Kegelmeyer, "Smote: Synthetic minority oversampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[12] C. Bunkhumpornpat and S. Subpaiboonkit, "Safe level graph for synthetic minority over-sampling techniques," *Communications and Information Technologies (ISCIT), 2013 13th International Symposium*, pp. 570–575, 2013.

[13] T. Maciejewski and J. Stefanowski, "Local neighbourhood extension of smote for mining imbalanced data," *Computational Intelligence and Data Mining (CIDM), 2011 IEEE Symposium*, pp. 104–111, 2011.

[14] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-smote: A new over-sampling method in imbalanced data sets learning," *Advances in Intelligent Computing. Lecture Notes in Computer Science*, vol. 3644, pp. 878–887, 2005.

[15] S. Hido, H. Kashima, and Y. Takahashi, "Roughly balanced bagging for imbalanced data," *Statistical Analysis and Data Mining*, vol. 2, no. 5-6, pp. 412–426, 2009.

[16] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 39, pp. 539–550, 2008.

[17] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag New York, Inc., 1995.

[18] B. Schölkopf and A. J. Smola, *Learning with Kernels. Support Vector Machines, Regularization, Optimization, and Beyond*, The MIT Press, Cambridge, 2002.

[19] B. X. Wang and N. Japkowicz, "Boosting support vector machines for imbalanced data sets," *Knowledge and Information Systems*, vol. 25, no. 1, pp. 1–20, 2009.

[20] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: a statistical view of boosting," *Annals of Statistics*, vol. 28, no. 2, pp. 337–374, 2000.

[21] C. Chen, A. Liaw, and L. Breiman, "Using random forest to learn imbalanced datag," *Statistics Technical Reports, ID: 666. The University of California*, 2004.

[22] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, pp. 123–140, 1996.

[23] A.M. Zoubir and D.R. Iskander, "Bootstrap methods and applications," *IEEE Signal Processing Magazine*, vol. 24, no. 4, pp. 10–19, 2007.

[24] B. Efron and R.J. Tibshirani, *An Introduction to the Bootstrap*, Chapman and Hall, 1993.

[25] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*, Wiley, New York, 2004.

[26] K. Bache and M. Lichman, "UCI machine learning repository [http://archive.ics.uci.edu/ml]," 2013.