Clustering Based Under-Sampling for Improving Speaker Verification Decisions Using AdaBoost

Hakan Altınçay and Cem Ergün

Advanced Technology Research and Development Institute
Eastern Mediterranean University, Gazi Mağusa KKTC, Mersin 10, Turkey
{hakan.altincay,cem.ergun}@emu.edu.tr

Abstract. The class imbalance problem naturally occurs in some classification problems where the amount of training samples available for one class may be much less than that of another. In order to deal with this problem, random sampling based methods are generally used. This paper proposes a clustering based sampling technique to select a subset from the majority class involving much larger amount of training data. The proposed approach is verified in designing a post-classifier using AdaBoost to improve the speaker verification decisions. Experiments conducted on NIST99 speaker verification corpus have shown that in general, the proposed sampling technique provides better equal error rates (EER) than random sampling.

1 Introduction

Class imbalance where the training data available for some pattern classes is much less than that of the others, naturally occurs in pattern classification problems such as speaker or face verification. Consider a speaker verification experiment where a post-classifier is to be applied on the verification output scores for optimal decision making [1]. In obtaining the training data for the post-classifier, target tests where the tested identity is the same as the claimed is limited to the number of speakers Q, whereas $Q \times (Q-1)$ impostor tests where the tested identity is different than that of the tested can be obtained. It should be noted that the imbalance increases in proportion to the number of identities considered in the verification experiment.

A generally accepted fact which is also observed in other research domains such as text classification is that the developed classifiers may provide much less accuracy for the minority classes having much less amount of training data [2]. Several explanations are already available in the literature. For instance, in Ref. [3] it is argued that this is mainly due to the fact that the a priori probabilities bias the learning procedure in favor of the majority class. Also, due to the insufficiency of the training data available, the minority class models may not be accurate enough.

There are several approaches proposed to deal with the imbalance problem. For instance, *over-sampling* the minority class to make its training data set cardinality same as the majority class or *under-sampling* the majority class. In

A. Fred et al. (Eds.): SSPR&SPR 2004, LNCS 3138, pp. 698–706, 2004.

general, over-sampling is implemented by inserting replicas of the available data points and under-sampling is implemented by taking into account a random subset of the majority class. These techniques have some disadvantages. In the case of over-sampling, the computational load is increased and overtraining may occur due to the replicated samples. Under-sampling does not take into account all available training data which corresponds to loss of available information. Moreover, it is not guarantied that the subset of data includes sufficient amount of critical samples close in the regions where classes overlap. A common drawback of these techniques is that the best test accuracy is not guaranteed for cardinally equal training sets since the class probabilities are highly likely to be different during test phase leading to a larger number of test samples from the majority class. Moreover, a priori probability information is lost after sampling. In fact, using more majority samples than minority during training is shown to provide better test accuracies.

AdaBoost algorithm is an iterative multiple classifier system development tool which is shown to provide improved classification accuracies for many different data sets compared to the best individual classifier. In each iteration, a new classifier is trained on a subset of the training data where the weight of each training sample is taken into account in this process. In fact, this sample selection mechanism corresponds to the simultaneous application of under-sampling and over-sampling.

In this paper, the performance of the AdaBoost algorithm in the class imbalance case is investigated. Having observed its poor performance, under-sampling and over-sampling techniques are applied and a better performance is achieved. The use of *k-means* clustering based centroids in the training set of the AdaBoost algorithm is proposed as alternative under-sampling technique. In the proposed approach, the AdaBoost algorithm is also modified so as to take into account the class a priori probabilities. Each centroid is used as a representative of its neighborhood where the misclassification of one centroid is considered as more costly than another if the number of the training samples in that cluster is more. Experiments on speaker verification which is basically a 2-class classification problem have proven the effectiveness of the proposed approach compared to the random under-sampling.

2 AdaBoost in Class Imbalance

The original AdaBoost (**Ada**ptive **Boost**ing) algorithm is an ensemble creation technique which was introduced in [4]. The sequential structure of the algorithm allows to create new classifiers which are more effective on the training samples that the current ensemble has a poor performance. In order to achieve this, weighting is applied on the training samples where a training sample with a higher weight has a larger probability of being used in the training of the next classifier. The algorithm summarized in Figure 1. $d_m(n)$ denotes the weight of the nth training sample in $\mathcal{S} = \{(x_n, y_n)\}$, $n = 1, \ldots, N$ initialized to $1/N$ and $\mathcal{C}$ denotes the classifier ensemble where $\mathcal{C}_m$ is the classifier obtained at the

mth iteration. At the end of each iteration, the weights of the samples that are correctly classified (misclassified) by the new classifier are decreased (increased). Increasing the weight of a misclassified sample corresponds to increasing the probability of its inclusion in the training set of the next classifier, probably more than once if its weight is high enough.

1. for $m = 1, ..., M$

1.1 Build classifier $\mathcal{C}_m$ using sample set $\mathcal{S}_m$ from $\mathcal{S}$ using distribution d_m.

1.2 Compute the weighted error using $\epsilon_m = \sum_{n=1}^{N} d_m(n)(1 - q_{n,m})$ where $q_{n,m} = 1$ if x_n is correctly classified by $\mathcal{C}_m$ and zero otherwise.

1.3 Compute $\alpha_m = \frac{1}{2} \ln(\frac{1-\epsilon_m}{\epsilon_m})$, $\epsilon_m \in (0, 0.5)$ and update the weights using,

$$d_{m+1}(n) = \frac{d_m(n)}{Z_m} \begin{cases} e^{-\alpha_m} & \text{if } C_m(x_n) = y_n \\ e^{\alpha_m} & \text{if } C_m(x_n) \neq y_n \end{cases}$$

where Z_m is a normalization factor so that d_{m+1} is a distribution.

2. The joint output of the classifier ensemble is computed using

$$\mathcal{C}(x) = \sum_{m=1}^{M} \alpha_m \mathcal{C}_m(x).$$

Fig. 1. The AdaBoost algorithm.

In order to evaluate the AdaBoost algorithm in class imbalance case, experiments are conducted on the "phoneme" data set from the ELENA database which involves 3818 and 1586 samples respectively for the first class, w_0 and second class, w_1. 2500 and 100 samples are used for training providing an imbalance ratio of 25 : 1. 1300 samples from each class are used for testing. In the under-sampling case, 100 training samples are selected from w_0 to be considered for model training whereas in the over-sampling case, the training samples of w_1 are replicated for 24 times so that both classes have the same amount of training data. The experiments are repeated for 10 times and the results are averaged. A quadratic discriminant classifier (QDC) from the PRTOOLS toolbox for MATLAB is used as the base classifier [5].

Figure 2 illustrates the training error achieved as a function of the classifiers in the ensemble. As seen in the figure, the performance on the training data in the case of imbalanced classes is much better than the sampling based approaches. However, it is evident from Figure 3 that this is mainly due to training inaccurate models that almost always predict the majority class. The poor test performance of AdaBoost indicates that the algorithm is not well suited for imbalanced data sets. In other words, the inherently available sample weight based under-sampling and over-sampling mechanism in AdaBoost may not be helpful. Moreover, the test performance achieved by the sampling techniques provide their efficiency also for the AdaBoost algorithm where the under-sampling performance is slightly better than over-sampling.

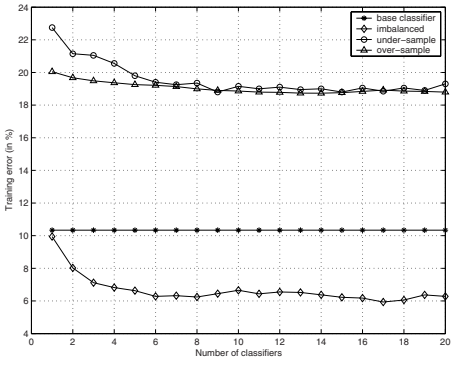

Fig. 2. Training error for different number of base classifiers.

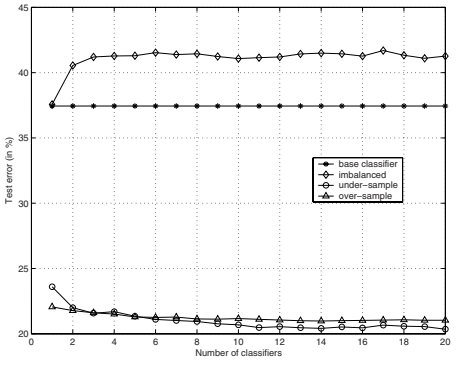

Fig. 3. Test error for different number of base classifiers.

3 Proposed Approach

Let N_t and N_i denote the number of training samples from minority and majority classes respectively where $N_i >> N_t$. Let μ_k, $k = 1, \ldots, N_t$ denote the centroids obtained by applying the k-means clustering algorithm to the training samples from the *majority class* so that the same number of training samples as in the minority class is obtained. Assume that c_k denote the number of training samples which are closest to μ_k where $\sum_{k=1}^{N_t} c_k = N_i$ and c_{avg} is the average number of training samples in the clusters. The selection of N_t centroids from the majority training data set balances the training data such that the same number of data points are used for both the minority and majority classes. The use of centroids instead of a random subset as in under-sampling approach has some advantages. For instance, the selection of samples which are very close to each other and have similar classification difficulties is avoided. Also, different costs can be associated with each misclassification. For instance, the misclassification of a centroid with a large number of training data can be considered to have a high cost. Having computed the centroids, the training data, $\mathcal{S}$ involving the N_t centroids from the

majority class and all N_t training data from the minority are considered as the training set, $\mathcal{S} = \{(x_n, y_n)\}$, $n = 1, \ldots, 2N_t$.

There are some drawbacks in applying the AdaBoost algorithm using the centroids. Firstly, the subset selection mechanism in Step 1 does not take into account the relative cardinalities of the training sets of minority and majority classes. Moreover, the cost of misclassifying a highly crowded centroid is not differentiated from a less crowded one in computing the model error, ϵ_m. It should be noted that the term "cost" does not denote the relative importance of correct classification among different classes as used in various cost sensitive boosting algorithms [6]. Instead, it is assumed that a misclassified centroid corresponds to the misclassification of all the majority training vectors closest to it and hence, the term "cost" stands for the contribution to ϵ_m by a misclassified centroid.

In order to avoid the problems specified above, a balancing based AdaBoost algorithm (AdaBoost-B) is proposed as illustrated in Figure 4. The proposed algorithm is based on the SSTBoost algorithm [7]. However, as stated above, the definition of cost and error are not based on relative importance of correct classification among different classes as in SSTBoost. It should be noted that the weight update in AdaBoost-B algorithm is the same as AdaBoost when $cost_n = 1$, $\forall n$. Also, different orders of scaling on the majority class are applied depending on the number of training vectors belonging to the centroids. The weights of the misclassified centroids that are more crowded are increased more than those of less crowded and the weights of the correctly classified centroids that are more crowded are decreased less than those of less crowded. The computation of the weighted error is also modified so as to take into account the fact that each centroid is a representative for a cluster of training data. The contribution to the weighted error by all vectors in a given cluster μ_n is proportional to the number of samples in that cluster. Hence, a weighted distribution should be computed as $w_m(n) = c_n \times d_m(n)/\gamma_m$ to take into account the contribution to the error by all available majority class samples. As a matter of fact, the a priori probabilities are incorporated in the classifier creation which is lost in the under-sampling and over-sampling cases. The scaling factor γ_m is used to make the scaled weights a valid distribution and $c_n = 1$ for the minority class.

For the minority class, $cost_n = 1$, meaning that the standard weighting used in the AdaBoost algorithm is applied where different costs are not associated with correct classification or misclassification.

The performance evaluation of identity verification systems is usually based on the Receiver Operating Characteristic (ROC). The cost of misclassification for different classes determine the operating point on the curve which is generally set using a threshold on the output scores. In summary, the k-means clustering based under-sampling approach helps to select a subset of training samples which represent the underlying distribution more accurately than the random selection approach. Moreover, information about the dense and sparse regions in the input space are included in the iterations so that misclassified centroids corresponding to sparse regions are defined as less costly than those representing the dense regions.

- Define $cost_n = \begin{cases} 1 & \text{if } x_n \in \ minority \\ \frac{c_n}{c_{avg}} & \text{if } x_n \in \ majority \end{cases}$

1. for $m = 1, ..., M$

1.1 Build classifier $\mathcal{C}_m$ using sample set $\mathcal{S}_m$ from $\mathcal{S}$ using distribution W_m.

1.2 Compute $\gamma_m = \sum_{n=1}^{2N_t} d_m(n)c_n$ where c_n is selected as 1 for the minority class.

1.3 Compute the weighted error using $\epsilon_m = \sum_{n=1}^{2N_t} (\frac{c_n}{\gamma_m}) d_m(n)(1 - q_{n,m})$ where $q_{n,m} = 1$ if x_n is correctly classified by $\mathcal{C}_m$ and zero otherwise.

1.4 Compute $\alpha_m = \frac{1}{2} \ln(\frac{1-\epsilon_m}{\epsilon_m})$, $\epsilon_m \in (0, 0.5)$ and update the weights using,

$$d_{m+1}(n) = \frac{d_m(n)}{Z_m} \begin{cases} e^{-\alpha_m(2-cost_n)} & \text{if } C_m(x_n) = y_n \\ e^{\alpha_m cost_n} & \text{if } C_m(x_n) \neq y_n \end{cases}$$

where Z_m is a normalization factor so that d_{m+1} is a distribution.

2. The joint output of the classifier ensemble is computed using

$$\mathcal{C}(x) = \sum_{m=1}^{M} \alpha_m \mathcal{C}_m(x).$$

Fig. 4. The balancing based AdaBoost algorithm, AdaBoost-B.

4 Speaker Verification and Experimental Setup

In Speaker Verification (SV), the aim is to decide whether the tested speech utterance belongs to the claimed identity or it is an impostor [8]. In the state-of-the-art SV systems, the output is composed of two likelihood scores where the decision is based on the likelihood ratio obtained as the difference of the log likelihoods of the outputs,

$$\mathcal{L}_{\mathcal{R}} = \mathcal{L}(X|\lambda_i) - \mathcal{L}(X|\lambda_B) \tag{1}$$

where λ_i denotes the claimant model, λ_B denotes the reference model and X denotes the tested utterance. The decision to accept or reject is based on comparing the likelihood ratio to a threshold, Θ such that

$$\mathcal{L}_{\mathcal{R}} \geq \Theta \implies \text{target speaker} \tag{2}$$

$$\mathcal{L}_{\mathcal{R}} < \Theta \implies \text{impostor} \tag{3}$$

Universal Background Models (UBM) are generally used as the reference models where each UBM is a Gaussian Mixture Model (GMM) having a large number of mixtures trained to represent speaker-independent distribution of the feature vectors [9]. The claimant models, λ_i are also GMMs which are trained using Bayesian adaptation from the UBM. The short-time spectral information extracted from the speech utterances, Mel-frequency cepstral coefficients (MFCC) are used as the feature vectors. Sixteen MFCCs and their Delta's [8] are computed in every 80 samples for the spectral representation of each Hamming windowed speech frame of length 160 samples.

A subset of the corpus is excluded from verification tests and is used for training the reference model. Approximately one hour of speech is used to train a UBM for male and another one hour of speech for training a female UBM. Each UBM involves 1024 mixtures. Then, these UBMs are combined to obtain a single 2048 mixture joint UBM to be used as a reference model. Excluding the speakers used for UBM training, 396 speakers (245 female and 151 male) are considered during the verification experiments. The training data of each speaker is split into 6 equal parts for 6-fold cross validation to obtain the training data for the multiple classifier based decision boundary. During the cross-validation, the speech segment which is not included in the model training and kept outside for validating the models had impostor attacks on all the other speakers. During the testing phase, non-overlapping speech segments of length 10s are used. The target tests and impostor attacks are performed using the standard setup defined for this corpus.

The output scores corresponding to the tested speaker and the joint UBM are the treated as the inputs for the classifier ensemble to be created using AdaBoost. The number of training and test samples for the minority class (target tests) and the majority class (impostor attacks) are given in Table 1. Due to 396 speakers involved in the *SV* experiment, the ratio of impostor to target training samples is high as 395 : 1. This ratio represents a rather high imbalance ratio. However, it naturally occurs in practice for *SV* problem.

Table 1. The number of training and test samples for the minority and the majority classes.

class	number of training samples	number of test samples
majority	938520	22071
minority	2376	1479

5 Results

In the experiments, two independent multiple classifier systems are implemented. The first one, *Ar* corresponds to the application of the original AdaBoost algorithm on a random subset of the majority class which involves the same number of training samples as the minority class. The second system, *ABc* corresponds to the use of cluster centroids of the majority class equal to the number of minority samples and the AdaBoost-B algorithm. Using *Ar*, the performance of AdaBoost algorithm in improving the verification decision for SV systems is investigated. Using *ABc*, it is aimed to examine whether alternative sampling techniques can improve the verification accuracy or not. Two different versions of the boosting algorithm are considered, aggressive and conservative. In [10], the given form of the boosting algorithm in Figure 1 is referred as *Aggressive boosting* since the weights of both correctly and incorrectly classified samples are modified. Alternatively, in *Conservative boosting*, either the weights of misclassified samples are increased or the weights of correctly classified samples are

decreased. In the conservative implementations in this study, only the weights of correctly classified samples are updated in both *Ar* and *ABc*. The over-sampling approach is not considered due to the heavily increased computational load after over-sampling in our case.

Two different base classifiers are considered, namely quadratic discriminant classifier (QDC) and an MLP neural network consisting of one hidden layer with 10 neurons trained for 300 iterations using fast backpropagation algorithm (NNET). The experiments are conducted for 10 times and the results are averaged. In the experiments, the total number of classifiers in each ensemble is selected as 20.

The Equal Error Rate (EER) provided by the baseline *SV* system based on a linear Bayes decision boundary is 15.28%. The EER's obtained using AdaBoost are presented in Table 2. As seen in the table, both of the systems considered in this study provide significant improvements in the verification accuracy. *ABc* provides better accuracies in both conservative and aggressive types of the boosting for the QDC type of base classifier. *ABc* provides better accuracy also in the conservative boosting of the NNET classifiers. However, the aggressive boosting of NNET classifiers in *Ar* yields a better performance than *ABc*.

There are some points that should be emphasized. Firstly, the imbalance ratio may be so large that, irrespective of the problems that may occur during the learning process, training an ensemble of classifiers may be infeasible from the computational point of view. Secondly, under-sampling based ensemble creation is observed to provide significant improvements. Moreover, taking into account the distribution of the impostor scores during sampling is valuable for further improvement. In fact, we mainly observe that the clustering based sampling and cluster dependent scaling of the training samples provide better accuracies than random selection in majority of the cases.

Table 2. Experimental results for two different base classifiers (in %).

	base classifier: QDC		base classifier: NNET	
MCS	Conservative	Aggressive	Conservative	Aggressive
Ar	13.87	14.00	13.46	13.33
ABc	13.72	13.86	13.39	13.40

6 Conclusions

In this study, the class imbalance problem is addressed and it is observed that the under-sampling approach is a simple but effective method where the AdaBoost algorithm based multiple classifier approach trained using the under-sampled training data provided significant improvements. The use of *k-means* clustering based centroids in the training set of the AdaBoost algorithm is proposed as an alternative under-sampling technique. In the proposed approach, the AdaBoost algorithm is also modified so as to take into account the class a priori probabilities. Each centroid is used as a representative of its neighborhood where the

misclassification of one centroid is considered as more costly than another if the number of the training samples in that cluster are more. Experimental results have shown that the proposed approach may be effective in majority of the cases.

References

1. S. Bengio and J. Mariethoz. Learning the decision function for speaker verification. *IEEE-ICASSP Proceedings*, 2001.
2. M.C. Monard and G.E.A.P.A. Batista. Learning with Skewed Class Distribution. In J.M. Abe and J.I. da Silva Filho, editors, *Advances in Logic, Artificial Intelligence and Robotics*, pages 173–180. IOS Press, 2002.
3. G.M. Weiss and F. Provost. The effect of class distribution on classifier learning: An empirical study. *Technical Report ML-TR-44, Department of Computer Science, Rutgers University*, August 2001.
4. Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Second European Conference on Computational Learning Theory*, March 1995.
5. R.P.W. Duin. PRTOOLS (version 3.0). A Matlab toolbox for pattern recognition. *Pattern Recognition Group, Delft University, Netherlands*, January 2000.
6. Kai Ming Ting. A comparative study of cost-sensitive boosting algorithms. In *Proc. 17th International Conf. on Machine Learning*, pages 983–990. Morgan Kaufmann, San Francisco, CA, 2000.
7. S. Merler, C. Furlanello, B. Larcher, and A. Sboner. Automatic model selection in cost-sensitive boosting. *Information Fusion*, 4(1):3–10, March 2003.
8. D.A. Reynolds. Speaker identification and verification using Gaussian mixture speaker models. *Speech Communication*, 17:91–108, 1995.
9. D.A. Reynolds, T.F. Quateri, and R.B. Dunn. Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing*, 10:19–41, 2000.
10. L.I. Kuncheva and C.J. Whitaker. Using diversity with three variants of boosting: Aggressive, conservative, and inverse. *MCS2002*, 2002.