SPO: Structure Preserving Oversampling for Imbalanced Time Series Classification

Hong Cao, Xiao-Li Li
Institute for Infocomm Research
Singapore
{hcao, xlli}@i2r.a-star.edu.sg

Yew-Kwong Woon
EADS Innovation Works
Singapore
David.Woon@eads.net

See-Kiong Ng[*†]
[*]Institute for Infocomm Research
Singapore
[*]skng@i2r.a-star.edu.sg

***Abstract*—This paper presents a novel structure preserving oversampling (SPO) technique for classifying imbalanced time series data. SPO generates synthetic minority samples based on multivariate Gaussian distribution by estimating the covariance structure of the minority class and regularizing the unreliable eigen spectrum. By preserving the main covariance structure and intelligently creating protective variances in the trivial eigen feature dimensions, the synthetic samples expand effectively into the void area in the data space without being too closely tied with existing minority-class samples. Extensive experiments based on several public time series datasets demonstrate that our proposed SPO in conjunction with support vector machines can achieve better performances than existing oversampling methods and state-of-the-art methods in time series classification.**

Keywords- Oversampling, learning, imbalance, time series, SVM, structure preserving, eigen regularization

I. INTRODUCTION

Data imbalance is a key source of degraded learning performance [1, 2] since existing learning algorithms often assume a balanced class distribution, as illustrated by the synthetic data in Fig. 1(a). In many real-world applications, the practical learning data can be highly imbalanced as in Fig. 1(b). This imbalance often causes that the learning be biased towards the majority class. However, the minority class often represents the interest to be accurately classified. Existing solutions address the imbalance issue at the data level [3-8], at the algorithm level [9], or at a combination of both levels [10-13]. The data-level methods re-establish the balance through resampling, e.g. oversampling of the minority class, undersampling of the majority class, or both. The algorithm-level methods incorporate or manipulate learning parameters such as dataspace weighting [9, 10, 12], class-dependant cost matrix, and receiver operating characteristics (ROC) threshold [14], into the existing learning paradigms to enforce sufficient emphasis on the minority class. Although several recent algorithm-level methods reviewed in [2] achieved good results on some imbalanced datasets, data-level methods are advantageous of addressing the imbalance issue at the most fundamental level and they can serve as a common preprocessing step for different machine learning algorithms.

Many real-world learning applications in a wide range of domains, such as finance, aerospace, entertainment, network

†: Dr See-Kiong Ng is also a faculty member of Singapore University of Technology and Design. Email: ngseekiong@sutd.edu.sg

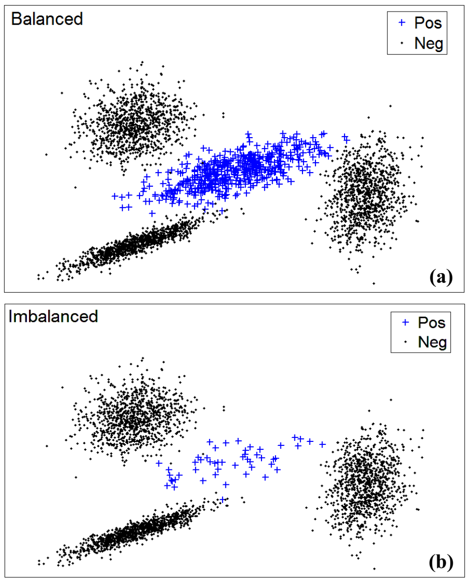

Figure 1. Comparison of the balanced class population in (a) and the practical imbalanced distribution in (b) in 2D feature space.

security, and medicine, involve time series data [15-19]. As defined in [17], a time series instance is an ordered set of real-valued variables that are sampled or extracted on a continuous signal, which can be either in the time or spatial domain. Due to its sequential nature, variables that are close in a time series are often highly correlated. One of the best-known learning methods for time series classification is the one nearest neighbor (1NN) classifier with dynamic time warping (DTW) [15]. The distance between two samples, known as warping distance, is computed by searching for the optimal mapping path to align the two time series sequences. The classification of a test sample is then based on the top-one nearest training neighbor.

Imbalanced time series classification is difficult because of its high data dimensionality and inter-variable correlation. Though oversampling is effective for re-balancing the classes, as far as we know, it has not been sufficiently explored for imbalanced time series classification due to the complexity of the problem. Note that the recent booming classification [25] for skewed stream data focuses on solving the issues related with evolving stream, such as concept drift, and the stream data do not necessarily need to be time series data [17]. To achieve the oversampling in general, two existing approaches can be adopted. The first approach interpolates between selected positive samples and their random positive nearest neighbors for generating the synthetic samples. Well-known oversampling methods that adopt this approach are SMOTE [3], Borderline-SMOTE [6] and ADASYN [7]. A recent extension [19] to SMOTE addressed a special scenario that imbalanced data are in the format of pairwise distance matrix. The second oversampling approach is to generate the features of the synthetic samples individually. A representative method is DataBoost [12], which generates each feature based on Gaussian distribution within an empirical range [*min*, *max*]. The two approaches have been shown to work fairly well for various imbalanced non-time series classification datasets. However, we opine they may not be adequate for oversampling largely imbalanced time series datasets. As we have mentioned earlier, the adjacent variables in the time series are usually not independent but highly correlated. The random data variances introduced by both conventional oversampling approaches will weaken or even destroy the inherent correlation structures in the original time series data, resulting in non-representative synthetic training samples with excessive noise that confound the learning.

As such, we propose a novel structure preserving oversampling (SPO) method for a binary time series classification task. Our SPO method is designed to preserve the covariance structure in the training time series data by operating in the corresponding eigen spectrum in two subspaces, a reliable subspace and a unreliable subspace, as follows: 1) The synthetic samples are generated by estimating and maintaining the main covariance structure in the reliable eigen subspace; 2) A regularization procedure is further employed to infer and fix the unreliable eigen spectrum. This helps create some buffer variances of the synthetic data in the trivial eigen subspace to improve the generalization performance on the unseen data. To the best of our knowledge, this is the first oversampling method that preserves the covariance structure in imbalanced learning. In conjunction with Support Vector Machines (SVM), we show that SPO outperforms other oversampling methods and our classification results are better than several state-of-the-art methods for time series classification.

II. The Proposed SPO Framework

Given the positive and the negative learning datasets $P=\{\boldsymbol{x}_{11}, \boldsymbol{x}_{12},\ldots, \boldsymbol{x}_{1|P|}\}$ and $N=\{\boldsymbol{x}_{01}, \boldsymbol{x}_{02},\ldots, \boldsymbol{x}_{0|N|}\}$, where $|P| \ll |N|$, $\boldsymbol{x}_{ij} \in \mathbb{R}^{n\times 1}$ and n denotes the time series length or dimension, our algorithm first computes the covariance matrices of the positive and negative classes using

$$\begin{aligned}\mathbf{W}_P &= \frac{1}{|P|}\sum_{j=1}^{|P|}\left(\boldsymbol{x}_{1j}-\bar{\boldsymbol{x}}_1\right)\times\left(\boldsymbol{x}_{1j}-\bar{\boldsymbol{x}}_1\right)^T \\ \mathbf{W}_N &= \frac{1}{|N|}\sum_{j=1}^{|N|}\left(\boldsymbol{x}_{0j}-\bar{\boldsymbol{x}}_0\right)\times\left(\boldsymbol{x}_{0j}-\bar{\boldsymbol{x}}_0\right)^T\end{aligned} \quad (1)$$

where $\bar{\boldsymbol{x}}_1 = \sum_{j=1}^{|P|}\boldsymbol{x}_{1j}\Big/|P|$ and $\bar{\boldsymbol{x}}_0 = \sum_{j=1}^{|N|}\boldsymbol{x}_{0j}\Big/|N|$ denote the corresponding mean feature vectors. We then perform eigen decomposition on $\mathbf{W}_P$ using

$$\mathbf{D} = \mathbf{V}^T\mathbf{W}_P\mathbf{V} \quad (2)$$

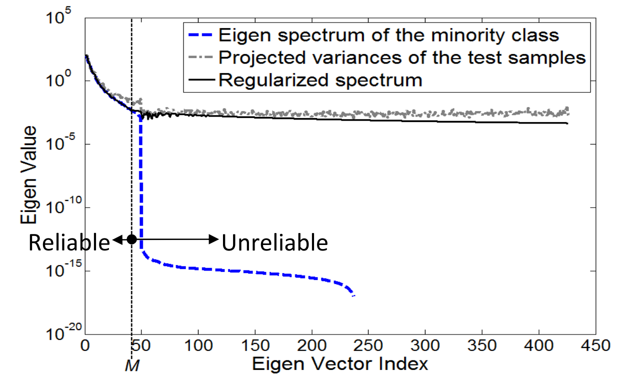

Figure 2. Comparison of estimated minority-class spectrum, projected test variance spectrum and the regularized spectrum for Yoga dataset.

where $\mathbf{V}=[\boldsymbol{v}_1,\ldots,\boldsymbol{v}_j,\ldots,\boldsymbol{v}_n]$ is the eigenvector matrix and $\mathbf{D}$ is a diagonal matrix with the corresponding eigenvalues $d_1 \geq\ldots\geq d_j \geq\ldots\geq d_n$ in descending order. The eigenvectors in $\mathbf{V}$ are statistically uncorrelated satisfying $\mathbf{v}_i^T\mathbf{W}_P\boldsymbol{v}_j = 0$ for $i \neq j$. Note that $\mathbf{W}_P$ is a symmetric positive semi-definite matrix. In other words, $d_j \geq 0$ must be satisfied for $1 \leq j \leq n$ and d_j represents the projection variance on the j^{th} eigenvector for the positive training data. Suppose we have a large number of positive samples, all the eigenvalues $\{d_j\}$ shall be greater than zero and from Eqn (2), we can derive

$$\mathbf{I}_n = \left(\mathbf{V}\mathbf{D}^{-1/2}\right)^T\mathbf{W}_P\left(\mathbf{V}\mathbf{D}^{-1/2}\right) = \mathbf{F}^T\mathbf{W}_P\mathbf{F} \quad (3)$$

where $\mathbf{F} = \mathbf{V}\mathbf{D}^{-1/2} = [\mathbf{v}_1\big/\sqrt{d_1},\ldots,\mathbf{v}_j\big/\sqrt{d_j},\ldots,\mathbf{v}_n\big/\sqrt{d_n}]$ is a scaled transformation so that the transformed feature vectors $\{\mathbf{F}^T\boldsymbol{x}_{1j}, 1 \leq j \leq |P|\}$ of the positive samples have a covariance of an identity matrix $\mathbf{I}_n \in \mathbb{R}^{n\times n}$. Note that the transformation $\mathbf{F}$ actually turns an arbitrary full-rank covariance matrix into a simple and well-known covariance structure. We can exploit this one-to-one mapping to simplify our goal of oversampling. That is, we can generate the new samples based on a simple identity-matrix covariance structure and then map them into the targeted covariance structure using the inverse transformation of $\mathbf{F}$.

In practice, the number of positive samples ($|P|$) can be significantly less than the time series dimension (n). In this case, the estimated covariance structure is likely to be over-adapted to the small set of positive data and cannot be fully trusted. As can be observed in Fig. 2, the large eigenvalues are often good approximations to the projected variances of the test spectrum, but the remaining portion of very small eigenvalues exhibit a large gap to the projected test variances. Here, the test spectrum is computed by projecting a set of test positive samples onto each of eigenvectors $\{\mathbf{v}_j\}$ and then computing its corresponding projection variance. Given that the inverse of an eigenvalue is commonly used as a multiplicative term in feature scaling, the unreliable portion of the eigen spectrum is a major source of learning instability and poor generalization in discriminant feature extraction [21, 22] and machine learning.

In order to generate random synthetic samples that not only maintain the major covariance structure of positive training dataset but also generalize well on the test dataset (so as to ensure accurate classification results subsequently), we introduce a regularization step to fix the covariance eigen spectrum $\{d_j\}$ of the positive dataset. We divide the eigen spectrum into two regions, namely, the *reliable* and the *unreliable* subspace, and then perform regularization on the *unreliable* spectrum. We perform a c-fold cross validation to determine the division point between the two subspaces (marked as M in Fig. 2). We randomly divide the positive data into c equal partitions. One partition is reserved for testing and the remaining c-1 partitions are used for computing the eigen vectors and the eigen spectrum. The testing partition is then projected onto these eigen vectors to measure the test variance spectrum. We repeat this c times to use each partition as a test partition once. By averaging the c eigen spectrums and the c testing spectrums, a good M is located where the average test spectrum departs from the average eigen spectrum. In this work, as we can have only about a dozen positive samples in the evaluation experiments, we chose a small c=2 to avoid tiny partitions.

With M determined, we compute the regularized eigen spectrum using

$$\hat{d}_j = \begin{cases} d_j, & \text{for } j \le M \\ \min\left(\alpha/(j+\beta), e_j\right), & \text{otherwise} \end{cases} \tag{4}$$

where function $T(j) = \alpha/(j+\beta)$ is a smooth eigen spectrum model suggested in [21] to regularize the *unstable* eigen spectrum. We use $T(1) = d_1$ and $T(M) = d_M$ to ensure the smooth transition and compatibility of the two spectrum regions, and determine the model parameters as follows:

$$\alpha = \frac{d_1 d_M (M-1)}{d_1 - d_M}, \quad \beta = \frac{M d_M - d_1}{d_1 - d_M} \tag{5}$$

The term $e_j = \boldsymbol{v}_j^T \mathbf{W}_N \boldsymbol{v}_j$ represents the projected variance of the negative-class samples on the j^{th} positive-class eigen vector $\mathbf{v}_j$. For the *unreliable* spectrum region, the model $T(j)$ generally provides a smooth regularized spectrum that is greater than the original spectrum d_j and below the spectrum of e_j. In some cases, the regularized eigen values provided by this model may exceed e_j. Since e_j is computed on a large pool of negative-class samples and thus represents a reliable estimation, Eq (4) sets $\hat{d}_j = e_j$ in such cases to ensure $\hat{d}_j \le e_j$ for $j > M$, which minimizes the risk of over regularization.

With the regularized positive dataset's covariance structure, we then generate a total of $|N| - |P|$ synthetic positive samples (to balance the population of positive and negative classes) based on multivariate Gaussian distribution (MGD). MGD is chosen here since it is the most natural distribution, and also that summation of a large number of independent distributions obeys the Gaussian distribution [22]. Suppose $\hat{\mathbf{F}} = [\mathbf{v}_1/\sqrt{\hat{d}_1}, \ldots, \mathbf{v}_j/\sqrt{\hat{d}_j}, \ldots, \mathbf{v}_n/\sqrt{\hat{d}_n}]$. Let us assume that $\boldsymbol{z} = \hat{\mathbf{F}}(\boldsymbol{b} - \bar{\boldsymbol{x}}_1)$, the transformed version of the synthetic positive sample $\boldsymbol{b}$ to be generated, follows two separate MGDs of $\mathcal{N}(\mathbf{0}_M, \mathbf{I}_M)$ and $\mathcal{N}(\mathbf{0}_{n-M}, \mathbf{I}_{n-M})$ to cater for the *reliable* and the *unreliable* eigen spectrum regions, respectively. We generate the two portions $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ separately, where $\boldsymbol{z}_1$ is randomly generated from the MGD of $\mathcal{N}(\mathbf{0}_M, \mathbf{I}_M)$ and $\boldsymbol{z}_2$ from $\mathcal{N}(\mathbf{0}_{n-M}, \mathbf{I}_{n-M})$. Here, $\mathbf{0}_M$ denotes a vector of M zeros. The two portions $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are concatenated to form $\boldsymbol{z}$. The synthetic sample is then

$$\boldsymbol{b} = \hat{\mathbf{D}}^{1/2} \mathbf{V}^T \boldsymbol{z} + \bar{\boldsymbol{x}}_1 \tag{6}$$

where $\hat{\mathbf{D}}$ is the diagonal matrix of regularized eigen values $\{\hat{d}_1, \ldots, \hat{d}_n\}$. We also apply a distance constraint to determine whether $\boldsymbol{b}$ is an outlier as follows: First, we find $\boldsymbol{x}_{0k} \in N$, the nearest negative sample to $\boldsymbol{b}$, i.e. $k = \arg\min_j f(\boldsymbol{b}, \boldsymbol{x}_{0j})$. Here $f(\cdot)$ denotes a distance function. As mentioned earlier, as 1NN-DTW has been shown to be one of the best performers for time series classification [15], we would like to use the DTW distance as $f(\cdot)$ ideally. However, because of its high computational load, we use the simple Euclidean distance, which is also known a suitable distance metric for time series classification [17, 18], as our $f(\cdot)$ in this work. To eliminate potential outliers, we require that

$$\min_j \left(f(\boldsymbol{x}_{1j}, \boldsymbol{x}_{0k}) \right) < f(\boldsymbol{b}, \boldsymbol{x}_{0k}) < \max_i \left(f(\boldsymbol{x}_{1i}, \boldsymbol{x}_{0k}) \right) \tag{7}$$

be satisfied in order to include $\boldsymbol{b}$ as one positive synthetic sample. Otherwise, $\boldsymbol{b}$ is discarded. The condition in (7) ensures that the nearest negative neighbor of $\boldsymbol{b}$ in the training data space has at least one closer positive training sample than $\boldsymbol{b}$, and also that $\boldsymbol{b}$ is not located too far away from the existing training population.

The above oversampling process is repeated until $|N| - |P|$ synthetic positive samples are generated. The synthetic set is then included with the existing samples to form a balanced training set.

With the balanced dataset, we choose SVM to learn our classifier for the following reasons: 1) SVM is known to generalize well to unseen data as it minimizes the structural risk instead of the empirical risk [23]; 2) The formulation of SVM allows users to choose different non-linear feature mapping to a high dimensional space so that a good linear separation hyper-plane can be found. Kernel tricks can be employed in this process with acceptable computational load; 3) In our preliminary classification test on balanced time series data, we find that the overall classification accuracy of SVM is comparable to 1NN-DTW, the state-of-the-art time series classification method, if we choose the optimal SVM parameters with radial basis function (RBF) kernel. Our SVM learning follows with what have been suggested in [24] to include the procedures like feature scaling and grid searching for finding the best parameters.

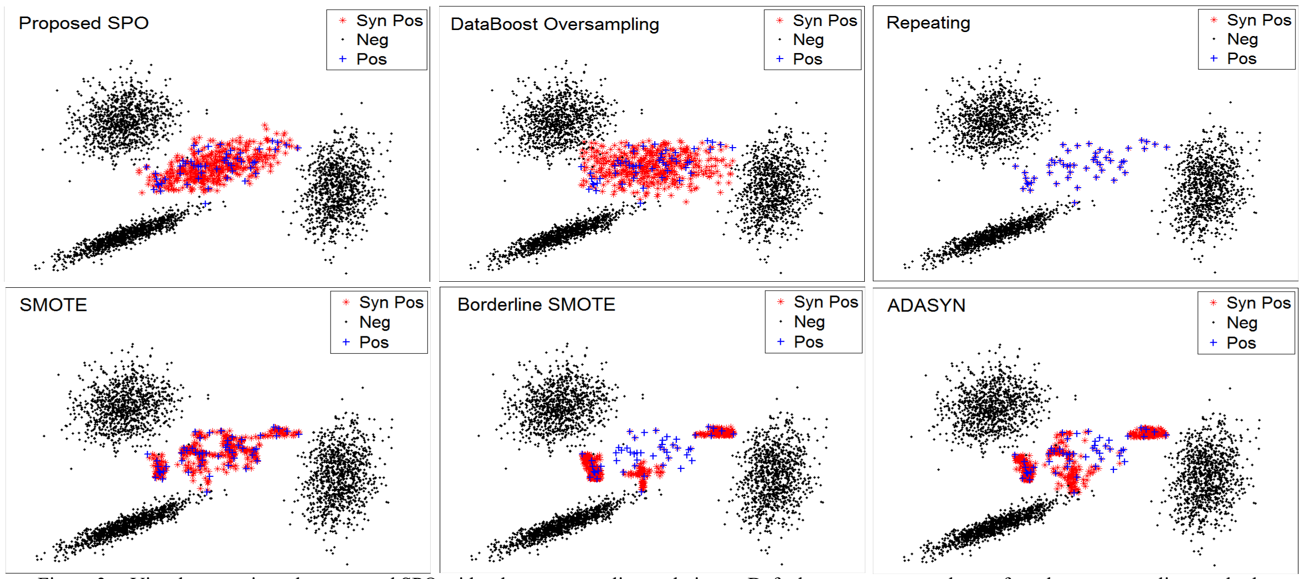

Figure 3. Visual comparison the proposed SPO with other oversampling techniques. Default parameters are chosen for other oversampling methods

TABLE I. IMBALANCED TIME SERIES DATASETS

Datasets	Training			Test		Time series length
	#Pos	*#Neg*	*IM-Ratio*	*#Pos*	*#Neg*	
Adiac	10	350	35	13	408	176
S-Leaf	30	500	17	45	550	128
Wafer	50	3000	60	712	3402	152
FaceAll	40	400	10	72	1738	131
Yoga	50	300	6	1480	1470	426

III. EXPERIMENTAL RESULTS AND DISCUSSION

A. *Visual Comparison*

Based on the example in Fig. 1(b), we visually compare the oversampling effect of our proposed SPO with five existing methods in Fig. 3 in two-dimensional feature space. For each method, we oversample the set of 50 positive samples nine times to have a total of 500 positive samples.

Among all five existing oversampling methods, the synthetic samples generated from DataBoost [12] fills the void areas within the territory of positive samples most competently. However, the synthetic sample distribution appears square-like and looks likely to expand into the territory of the negative class. On the other extreme, the synthetic samples generated by the repeating method overlap exactly with the current positive samples with no additional data variances being created. The synthetic samples generated by the other three nearest-neighbor interpolation methods formed dense clusters with small data variances near the existing samples. Among them, SMOTE creates relatively even synthetic distribution since every existing positive sample is selected for generating roughly the same number of new samples. Borderline-SMOTE creates a synthetic sample distribution where only a small set of existing positive samples, whose neighborhood contains more negative samples than the positive samples, are heavily emphasized. ADASYN strikes a balance between SMOTE and Borderline-SMOTE by adaptively emphasizing more on the samples that are closer to the classification border.

As we can see, the synthetic samples created by our proposed SPO shared the most similar covariance shape with the existing limited positive samples. Similar to DataBoost oversampling, our synthetic sample distribution is not closely tied with the existing positive samples and it fills the void internal spaces within the positive class territory. It also sensibly expands towards the void areas in the vicinity of positive class territory, which are currently not occupied by the abundant negative samples. At the same time, our distance constraint in Eqn (7) ensures that the SPO's synthetic positive distribution does not reach into the negative class territory.

B. *Comparison for Imbalanced Time Series Classification*

As tabulated in Table I, we constructed five imbalanced datasets from the public UCR time series repository [16]. *S-Leaf* here refers to the *"Swedish Leaf"* dataset. One can refer to [16, 17] for description of these datasets. These datasets were selected for containing relatively large numbers of samples to facilitate simulating scenarios of high class imbalance. Out of the five selected datasets, *Adiac*, *S-Leaf* and *FaceAll* originally contained 37, 15 and 14 classes, respectively. We converted them into two-class datasets by selecting one class as the positive class, i.e. Class 2 for *Adiac* (Class 2 is selected as it contains more samples needed than Class 1), Class 1 for *S-Leaf*, Class 1 for *FaceAll*, and using the remaining classes to form the negative class. For each set, the training and test data are apportioned randomly and all available samples are included either in the training or in the test set. The random apportion is repeated five times for each dataset so that we can report the more reliable average testing performances in the following section. As shown in Table I, we choose round numbers for the positive training samples and at the same time, they do not exceed 50% of the total available positive samples. For each training set, the

TABLE II. COMPARISON FOR DIFFERENT OVERSAMPLING METHODS

Eval. metric	**Dataset**	**Oversampling Method**					
		REP	***SMO***	***BoS***	***ADA***	***DB***	***SPO***
F-Value	***Adiac***	.375	.783	.783	.783	.136	**.963**
	S-Leaf	.716	.764	.764	.759	**.796**	**.796**
	Wafer	.962	.968	.968	.967	.977	**.982**
	FaceAll	.935	.935	.935	.935	.890	**.936**
	Yoga	.710	**.729**	.721	.727	.689	.702
	Average	.740	.836	.834	.834	.698	**.876**
G-Mean	***Adiac***	.480	.831	.831	.831	.748	**.999**
	S-Leaf	.800	.861	.861	.849	**.898**	**.898**
	Wafer	.965	.969	.970	.970	.980	**.984**
	FaceAll	.950	.950	.950	.950	.948	**.957**
	Yoga	.741	**.756**	.750	.755	.724	.735
	Average	.787	.874	.872	.871	.859	**.914**

The acronyms are: REP: repeating; SMO: SMOTE; BoS: Borderline SMOTE; ADA: ADASYN; DB: DataBoost oversampling

number of positive samples is kept no more than 50 to simulate the scenario of rare positive instances. We maintained high imbalance ratios (IM-ratios), i.e. the number of negative samples divided by the number of positive samples, in the datasets, with the highest being 60 for the Wafer dataset and the lowest being 6 for the Yoga dataset. Note that for all datasets, the number of positive samples is significantly smaller than the time series length (dimension of the time series). In particular, for *Adiac*, the feature dimension is about 18 times of the number of positive samples. This sparsity of data with respect to the high dimensions in time series classification is another key challenge.

To evaluate our SPO for oversampling, we used six oversampling methods (five different existing methods and our SPO method) to balance the classes in the training dataset separately, and then trained an SVM classifier for each resulting balanced dataset. The classifiers' performance is evaluated using *F-Value* and *G-Mean* [2] in Table II. The results show that our SPO achieved average *F-Value* and *G-Mean* of 0.88 and 0.91, respectively, which are better than all five existing oversampling methods. We found that our good result is largely attributed by excellent recall scores, compared with comparable precisions and true negative rates. The high recall rates indicate that much of the test positive samples can be classified with a good accuracy when using our SPO to supplement the emaciated positive training datasets. In comparison, we also found that the repeating oversampling achieved the largest precision score at the expense of the poorest recall value. This is due to the tendency that a smaller set of positive predictions at a high accuracy rate were made by the SVM when the repeating oversampling is used (i.e. with overfitting).

C. Comparison of Different Learning Frameworks

Finally, we compared our proposed learning framework with two recently developed imbalanced learning methods, EasyEnsemble [8] and BalanceCascade [8], as well as with two other well-known state-of-the-art methods, 1NN-DTW [15] and 1NN [17, 18], for time series classification. Like our SPO, EasyEnsemble and BalanceCascade also address the data imbalance issue at the data level. However, EasyEnsemble and BalanceCascade adopted the approach of *undersampling* the negative class to balance the training datasets. In both methods, a good number of balanced sets are constructed to learn multiple decision-tree weak classifiers. These classifiers are systematically integrated by AdaBoost into a strong ensemble classifier. The key difference between EasyEnsemble and BalanceCascade is that BalanceCascade periodically removes a pre-computed percentage of relatively easier negative samples from the negative training set before the random undersampling takes place. This procedure is to enforce more hard negative samples being included in the balanced training set. 1NN-DTW and 1NN do not explicitly address the data imbalance. As mentioned earlier, 1NN-DTW has been shown to be highly competent for classifying balanced time series data [15]. 1NN classifier with a simple Euclidean distance metric has also been suggested for semi-supervised time series classification [17, 18] involving utilizing unlabelled training samples to improve the learning outcomes.

Table III shows the comparison of our SPO cum SVM with these other learning methods in terms of *F-Value* and *G-Mean*. Except for the *FaceAll and Yoga* datasets that SPO was the second best, SPO achieved the best results consistently for the remaining three datasets. On average, the *F-Value* of SPO is 9% higher than the second best 1NN method and its *G-Mean* is 4% higher than the second best BalanceCascade method. Amongst the existing methods, we noticed that 1NN-DTW gave the best results for the *FaceAll and Yoga* datasets. However, its performance is much worse than 1NN for *S-Leaf* and *Wafer* datasets. To circumvent this instability issue, we have tried to define a warping window and vary its size from 1 to 11 as suggested in [15]. However, we did not obtain better results than what we reported in Table IV. 1NN gave more consistent performance and better average *F-Value* and *G-Mean* than 1NN-DTW. BalanceCascade provided significantly better average *F-Value* and *G-Mean* than its counterpart EasyEnsemble. This is in agreement with the discussion in [8] that BalanceCascade is more suitable for highly imbalance data than EasyEnsemble.

Currently, the number of synthetic samples to be generated by our SPO is dependent on size of the negative class. The oversampling procedure may not be practical when the negative training class is extremely large, e.g. with millions of samples. In such a situation, we can reduce the negative class to a manageable yet representative size through undersampling, which is similar to other methods [3, 4]. Our proposed method can then be applied to the modified training set in a smaller scale. Using our most populous time series dataset, *Wafer*, our current MATLAB SPO implementation takes an average of 4.0×10^{-2} second to create a synthetic sample about 150 dimensions on an ordinary computer with Intel 2.53-GHz CPU. This small time requirement suggests that our proposed algorithm can be used for many practical time series classifications.

IV. CONCLUSION

In this paper, we have proposed a novel structure preserving oversampling method for the challenging learning

TABLE III. COMPARISON OF SEVERAL CONVENTIONAL METHODS

Eval. metric	Dataset	Learning methods				
		Easy	Bal.	1NN	1NN DTW	*SPO*
F-Value	***Adiac***	.534	.348	.800	.917	**.963**
	S-Leaf	.521	.578	.716	.429	**.796**
	Wafer	.795	.954	.949	.857	**.982**
	FaceAll	.741	.625	.802	**.959**	.936
	Yoga	.356	.689	.652	**.710**	.702
	Average	.589	.639	.784	.766	**.876**
G-Mean	***Adiac***	.782	.897	.875	.920	**.999**
	S-Leaf	.712	**.898**	.798	.572	**.898**
	Wafer	.817	.970	.953	.870	**.984**
	FaceAll	.792	.918	.983	**.985**	.957
	Yoga	.464	.688	.695	**.741**	.735
	Average	.713	.874	.860	.810	**.914**

The acronyms are: Easy: EasyEnsemble; Bal: BalanceCascade; 1NN: One nearest neighbor classifier using Euclidean distance; 1NN DTW: One nearest neighbor classifier using dynamic time warping distance; SPO: Proposed structural preserving oversampling with support vector machine classifier

problem of imbalanced time series classification. In the current work, SPO addresses the imbalanced learning issue by oversampling the minority class. The synthetic samples are generated in eigen decomposed subspace and based on regularized eigen spectrum. This allows the resulting sample data to preserve the major covariance structure of the original minority-class samples and at the same time, to intelligently incorporate some protective variances in the trivial eigen dimensions. As a result, our synthetic samples are not tied closely with the existing samples. They fill up the gaps in between the minority-class samples and sensibly expand into the vicinity of minority-class territory without introducing outliers.

Based on five public sets of highly imbalanced UCR time series data, our SPO with SVM achieved good average *F-Value* and *G-Mean* of 0.88 and 0.91, respectively. It outperformed an array of existing oversampling methods as well as state-of-the-art learning methods for time series data. The results are particularly significant given that many real-world data mining applications are afflicted with data imbalance and involve time series data, but there have been few if any work on imbalanced time series learning. Our results with SPO showed that by taking into careful consideration the specific issues related to time series data, such as covariance structure preservation, the oversampling approach can be employed to effectively address the challenging problem of data imbalance in time series classification.

REFERENCES

[1] N.V. Chawla, N. Japkowicz and A. Kolcz, "Editorial: Special issue on learning from imbalanced data sets," ACM SIGKDD Explorations Newsletter, vol. 6(1), 2004, pp. 1-6.

[2] H. He and E.A. Garcia, "Learning from imbalanced data," IEEE Trans. on Knowledge and Data Engineering, vol. 21-9, Sept. 2009, pp. 1263-1284.

[3] N.V. Chawla, K.W. Bowyer, L.O. Hall and W.P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," Journal of Artificial Intelligence, vol. 16, 2002, pp. 321-357.

[4] A. Estabrooks, T. Jo and N. Japkowicz, "A multiple resampling method for learning from imbalanced data sets," Computational Intelligence, vol. 20, 2004, pp. 18-36.

[5] G.E.A.P.A. Batista, R.C. Prati and M.C. Monard, "A Study of the behavior of several methods for balancing machine learning training Data," ACM SIGKDD Explorations Newsletter, vol. 6(1), 2004, pp. 20-29.

[6] H. Han, W.Y. Wang and B.H. Mao, "Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning," Proc. Int. Conf. Intelligent Computing, 2005, pp. 878-887.

[7] H. He, Y. Bai, E.A. Garcia and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," Proc. Int. Conf. Neural Networks, 2008, pp. 1322-1328.

[8] X.-Y. Liu, J. Wu and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," IEEE Trans. on System, Man and Cybernetics, vol. 39 (2), Apr. 2009, pp. 539-550.

[9] Y. Sun, M.S. Kamel, A.K.C. Wong and Y. Wang, "Cost-sensitive boosting for classification of imbalanced data," Pattern Recognition, vol. 40 (12), Dec. 2007, pp. 3358-3378.

[10] N.V. Chawla, A. Lazarevic, L.O. Hall and K.W. Bowyer, "SMOTEBoost: Improving prediction of the minority class in boosting," Proc. European Conf. Principles and Practice of Knowledge Discovery in Databases, 2003, pp. 107-119.

[11] B. Raskutti and A. Kowalczyk, "Extreme re-balancing for SVMs: a case study," ACM SIGKDD Explorations Newsletter, vol. 6(1), 2004, pp.60-69.

[12] H. Guo and H.L. Viktor, "Learning from imbalanced data sets with boosting and data generation: The DataBoost-IM approach," ACM SIGKDD Explorations Newsletter, vol. 6(1), 2004, pp. 30-39.

[13] S. Chen, H. He and E.A. Garcia, "RAMOBoost: Ranked minority oversampling in boosting," IEEE Trans. on Neural Networks, vol. 21 (10), Oct. 2010, pp. 1624-1642.

[14] M.A. Maloof, "Learning when data sets are imbalanced and when costs are unequal and unknown," Proc. Int. Conf. Machine Learning, Workshop Learning from Imbalanced Data Sets II, 2003.

[15] X. Xi, E. Keogh, C. Shelton and L. Wei, "Fast time series classification using numerosity reduction," Proc. Int. Conf. on Machine Learning, 2006, pp. 1033-1040.

[16] E. Keogh, X. Xi, L. Wei and C.A. Ratanamahatana (2006), UCR time series classification/clustering page, www.cs.ucr.edu/~eamonn/time_series_data.

[17] L. Wei and E.J. Keogh, "Semi-supervised time series classification," Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, Aug. 2006, pp. 748-753.

[18] M.N. Nguyen, X.-L. Li and S.-K. Ng, "Positive unlabeled learning for time series classification," Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI 2011), Jul. 2011.

[19] S. Koknar-Tezel and L.J. Latecki, "Improving SVM classification on imbalanced time series data sets with ghost points," Knowledge and Information Systems – KAIS, vol. 24(2), 2010, DOI: 10.1007/s10115 -010-0310-3.

[20] R. Pearson, G. Goney and J. Shwaber, "Imbalanced clustering for microarray time-series," Proc. Int. Conf. Machine Learning, Workshop Learning from Imbalanced Data Sets II, 2003.

[21] X. Jiang, B. Mandal and A.C. Kot, "Eigenfeature regularization and extraction in face recognition," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 30(3), Mar. 2008, pp. 383-394.

[22] X. Jiang, "Linear subspace learning-based dimensionality reduction," IEEE Signal Processing Magazine, vol. 28(2), Mar. 2011, pp. 16-26.

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

[24] C.-W. Hsu, C.-C. Chang and C.-J. Lin, "A pratical guide to support vector classification," 2008.

[25] J. Gao, W. Fan, J. Han and P.S. Yu, "A general framework for mining concept-drifting data streams with skewed distributions," Proc. of the SIAM Int. Conf. on Data Mining, 2007.