Two-Phase Resampling for Noisy Imbalanced Multi-class Classification Problems and its Application in Human Activity Recognition

Jianjun Zhang, Wing W. Y. Ng*, Shuai Zhang, and Chris D. Nugent

***Abstract*—Current imbalanced classification research mainly focuses on two-class imbalanced problems. However class imbalance issues are more serious in multi-class problems, e.g. human activity recognition. Multi-minority and multi-majority make multi-class imbalanced problems more challenging. Resampling methods are effective to handle class imbalance issues by rebalancing the class distribution, but current methods tend to apply resampling on all samples including noisy ones, which may participate in classifier training and therefore hinder the performance of the classifier. Hence, in this work, we propose a Two-Phase Resampling (TPRS) method, for multi-class noisy imbalanced problems. In Phase one, a one-vs-one scheme is applied to iteratively remove noisy samples in each class by evaluating the stochastic sensitivities of the training samples. In Phase two, an oversampling method is applied on each minority class to relatively rebalance the class distribution. After completing the TPRS, a multi-class classifier is trained using the noise-filtered rebalanced training dataset. To evaluate the effectiveness of the TPRS, nine UCI datasets and one real-world sensor-based human activity recognition dataset with five levels of noises are employed in the experimental studies. Experimental results show that the TPRS yields relatively stable performances as the noise level increases and significantly outperforms state-of-the-art methods especially in highly noisy environments.**

***Index Terms*—Class imbalance, class noise, resampling, multi-class classification, human activity recognition.**

I. Introduction

PATTERN classification problems have gained a lot of attentions in recent years and are the core components of many different applications, for instances smart home, fitness tracking, medication reminders, and health monitoring [1]. Yet in many real-world pattern classification problems, different issues are often encountered that make the classification problems more difficult and complex, for example class imbalance. Class imbalance problems occur when one class severely out-represents another [2], i.e. the number of samples in at least one class is either much more or less than other classes. When class imbalance occurs, traditional classifiers are often biased to the majority class and yield very low accuracy for the minority class, which is not preferable because misclassifying minority samples are often costly than misclassifying majority ones [3]. Many efforts have been focused on two-class imbalanced problems, but in practice many problem domains contain more than two classes with unbalanced class distributions. In human activity recognition (HAR) problems, the durations of the transitional activities (e.g. walk-to-stand or stand-to-sit) between the main activities (e.g. walking or sitting) are usually very short which results in some classes (e.g. walking and sitting) consist of many more samples than others (e.g. walk-to-stand and stand-to-sit).

Multi-class imbalance problems bring new challenges compared with two-class imbalance problems: decision boundaries between classes may severely overlap [4], and the relations among the classes are no longer obvious [5]. In multi-class imbalance problems, both multi-majority and multi-minority may exist, which will negatively affect the classification performances and simple resampling methods do not yield satisfactory results [6]. Methods designed for binary-class imbalance problems may not be directly applied for multi-class ones or suffer from low performance [7]. Among current literature on multi-class imbalance problems, class decomposition techniques, cost sensitive methods, and resampling methods are often applied. Class decomposition techniques normally consists of two categories, the one-against-one (OAO) and the one-against-all (OAA) [8]. Drawbacks of both methods are obvious: the OAO has higher time complexity for constructing more classifiers while the OAA suffers more severe class imbalance when the dataset is originally imbalanced. A systematic study was carried out and showed that the combination of OAO and OAA with resampling methods and cost sensitive learning produce promising results [9]. Cost-sensitive methods usually modify the loss function of specific classifiers or assign different weights to samples according to their costs in order to minimize the overall costs instead of minimizing the overall error as before [10]. Cost-sensitive methods are mainly specialized for certain types of classifiers, for examples neural networks [7], SVMs [11][12][13], etc. Resampling methods are designed to rebalance the prior probabilities of all classes and are independent of classifiers. In the multi-class imbalance scenario, most resampling methods are focused on modifying existing methods designed for binary-class imbalanced problems by integrating class decomposition techniques, cost sensitive learning or ensemble learning methods. For instances, Lin et al. [14] and Fernandez-Navarro et al. [15] combined resampling and cost sensitive learning to train neural networks.

As pointed out in [3], some problems frequently coexist

This manuscript is submitted to the TNNLS special issue on Recent Advances in Theory, Methodology and Application of Imbalanced Learning.

Jianjun Zhang and Wing W. Y. Ng are with Guangdong Provincial Key Lab of Computational Intelligence and Cyberspace Information, School of Computer Science and Engineering, South China University of Technology, Guangzhou, China 510006 e-mail: wingng@ieee.org.

Shuai Zhang and Chris D. Nugent are with School of Computing, Ulster University, Jordanstown, United Kingdom.

*Corresponding Author.

with class imbalance and further contribute to degrade the performance of predictive models, for example class noise [16]. In class imbalance domains, the least-represented classes (minority class) are more negatively affected by noisy data [17], and more generally, class noise has more significant impact on learning models than class imbalance [18]. The interaction between the levels of imbalance and the levels of noise is a relevant issue and the two aspects should be studied together [3] [19]. Understanding the effects of class noise on learning models in imbalanced data environments may help provide valuable insights into the general problem of class imbalance [20]. Without properly handling the noisy data in the class imbalance domains, classifiers trained using these noisy imbalanced data may yield very poor generalization capability on not only the minority classes but also the majority ones. However, to date, few studies are focused on handling both issues. Among very limited related researches, some works propose to combine the resampling methods and some noise filtering techniques, for example the K-Influential-Neighbor OverSampling (KINOS) [21] and the noise-filtered undersampling (NUS) [22]. These methods mainly utilized a k-nearest neighbors-based noise filter to identify noisy data, but searching optimal k values for different datasets is very time consuming. More importantly, these noise handling methods do not necessarily lead to a more robust classifier because they focus on removing noisy samples only and ignore information from classifier training.

In this work, we propose the Two-Phase Resampling (TPRS) method to deal with the noisy imbalanced data. We mainly focus on multi-class problems in this paper since they are more prevalent in real-world applications. In the first phase of TPRS, a stochastic sensitivity-based noise filtering procedure is firstly applied to identify and remove noisy samples in all classes. Samples are evaluated by computing their stochastic sensitivities using a neural network ensemble. Samples with higher stochastic sensitivities are more likely to be misclassified and more likely yield negative impacts on classifier training, therefore these samples are treated as noises and removed. In the second phase, an oversampling procedure is employed to rebalance the class distribution. After the TPRS, a multi-class classifier is trained using the noise-filtered rebalanced dataset.

The major contributions of this paper are listed as follows:

1) A Two-Phase Resampling (TPRS) method is proposed to handle both class noise and class imbalance problems. Class noise is identified via evaluating the training samples' stochastic sensitivities and class imbalance is handled by employing an oversampling procedure. By removing noisy samples before executing oversampling, the performance of the oversampling is enhanced and classifier trained using the noise-filtered rebalanced dataset yields higher generalization capability.
2) The TPRS serves as a wrapper method and is independent of classifiers. It is feasible to employ different oversampling methods and different types of classifiers, which shows the flexibility of the TPRS.
3) To evaluate the performances of the TPRS, comprehensive experimental studies are carried out based on nine UCI datasets and a real-world human activity recognition dataset. Five levels of class noises are introduced into the datasets to analyze the effects of both the class noise and class imbalance. Experimental results show that the TPRS significantly outperforms the state-of-the-art methods.

The rest of this paper is organized as follows: Section II briefly describes related works. The TPRS is proposed in Section III. A comprehensive experimental study is conducted in Section IV. Section V gives a case study on human activity recognition task using sensor data. Conclusions and future works are given in Section VI.

II. Related Works

In this section, we mainly reviews works related to class noise problems and class imbalance problems, in particular data resampling methods, because this work focuses on applying resampling methods to handle class imbalance problems.

A. Methods for Handling Class Noise Problems

Class noise handling aims to train a robust classifier with respect to class noises. One of the common approaches in this area is to eliminate noisy samples via a noise filter so as to produce a dataset with high quality [23] [24]. Numerous noise filtering approaches were proposed, including the Classification Filter (CF) [25], the Ensemble Filter (EF) [26], the Cost-Guided Iterative Classification Filter (CICF) [27] and the Iterative Partitioning Filter (IPF) [28]. The CF identifies noises using a cross validation procedure, where the misclassified samples in each validation set are treated as noises and removed when all validations have finished. In contrast, the EF trains a set of classifiers using each validation set and each of them is evaluated using all samples, where samples that are misclassified by all classifiers or at least half of the classifiers are treated as noises and removed. The CICF extends the CF by integrating a cost-guided rejection sampling procedure [29]. If a misclassified sample yields higher misclassification cost, it has higher probability to be retained in the dataset to avoid removing important informative data. The IPF is the extension of the EF which iteratively removes noisy samples until the number of identified noisy samples is less than a certain threshold in one iteration.

B. Methods for Handling Class Imbalance Problems

Resampling methods are widely applied to handle class imbalance problems by either oversampling the minority classes or undersampling the majority classes to rebalance the class distributions. Resampling methods can be basically categorized into three types, undersampling, oversampling, and hybrid.

Undersampling removes samples from majority classes to relatively rebalance the class distribution, which alleviates the class imbalance issue and lower the computational costs. The simplest undersampling method is to randomly remove some majority samples so as to achieve an equal number of samples among classes. This, however, may lead to severe

information loss and result in insufficient samples for classifier training. To avoid this problem, informed undersampling methods are proposed. Lin et al. proposed to cluster majority classes into several clusters with the number of clusters being the number of minority samples and select those samples located nearest the cluster centers as candidates [30]. These candidates are combined with original minority samples to create a balanced dataset. Similarly, the Diversified Sensitivity Undersampling clusters both classes and iteratively selects a balanced dataset from these clusters to retrain an RBFNN [31]. The clustering procedures aims to preserve the data distributions of both classes to avoid information loss. Recently, a weighted undersampling scheme for SVM [32] was proposed to handle imbalanced classification problems, which groups majority samples into several sub-regions and assigns different weights to these regions by considering their distances to the hyperplane. Samples in the sub-region with higher weights have higher probability to be selected, in which way data distribution information is retained.

Oversampling replicates or generates new minority samples. The random oversampling method replicates some minority samples to rebalance the class distribution, which may easily lead to overfitting. One of the most popular oversampling methods is the Synthetic Minority Oversampling TEchnique (SMOTE) [33], which generates new samples along line segments connecting candidate minority sample and its k-nearest neighbors. The major drawback of the SMOTE is that noisy samples may participate in the data generation procedure, which leads to severe overlapping and new noises introduced. Other minority samples generating methods include the Random Walking Oversampling (RWO) [34] and the Mahalanobis Distance-based Oversampling technique (MDO) [35]. The RWO applies the standard normal distribution to approximate the underlying distribution of numeric features and generates new samples from this distribution, while the MDO generates new samples by maintaining the same Mahalanobis distance from the mean of minority class. These two methods are basically applicable for numeric features. The Adaptive MDO (AMDO) [36] extends the MDO for mix-type features by introducing the Heterogeneous Value Difference Metric (HVDM) [37] as the distance metric to handle nominal features.

Hybrid methods usually combine an ensemble learning method with undersampling or oversampling. For examples the RUSBoost [38] integrates the random undersamping in the boosting iteration, the UnderBagging undersamples the majority class and combines with minority class to create balanced bags for classifier training [39], the UnderOverBagging varies the proportions of original data and the resampled data so that a series of diverse subsets are created [40], and the SMOTEBoost [41] differs from the RUSBoost by employing the SMOTE instead of random undersampling to rebalance the class distribution. Recently a new method combining SMOTE and transfer boosting to rebalance the skewed class distribution is proposed, which integrates the oversampling in the transfer AdaBoost framework [42]. The EasyEnsemble and the BalanceCascade [43] are two effective undersampling-based hybrid methods, which undersamples multiple subsets from the majority class and combines with the minority class. These balanced subsets are used to train several classifiers using AdaBoost and these classifiers are fused together, so the output of these two methods is ensemble of ensemble.

C. Methods for Handling Both Class Noise and Class Imbalance Problems

Existing methods for handling both class noise and class imbalance problems mainly focus on extending the popular oversampling method SMOTE by introducing some kinds of methods to identify noisy samples. For examples the Borderline-SMOTE [44] generates new samples using only the borderline samples to avoid adding new noises and the Majority Weighted Minority Oversampling TEchnique [45] generates new samples inside minority class clusters. In contrast, the Synthetic Minority Oversampling for Multi-class imbalance problems (SMOM) assigns different selection weights for each neighbor direction such that a direction is assigned less weight if generating new samples in this direction would introduce noisy samples or increase the overlapping level [46]. Several methods are proposed to remove the noises introduced by the SMOTE. The SMOTE-ENN [47] employs the Edited Nearest Neighbors method [48] to remove noisy samples in both classes, in which noisy samples are those misclassified by the three-nearest neighbor algorithm. Similarly, the SMOTETOMEK [47] removes the Tomek Links [49] and the SMOTE-IPF [23] removes noises using the IPF after applying the SMOTE, respectively. Recently, a new method KINOS was proposed to firstly under-sample the minority class to reduce the noises in the minority class, then apply the over-sampling method on the noise filtered dataset [21]. Noises, however, are put back to the dataset after the oversampling to avoid information loss.

Few methods are focused on combining noise filter and undersampling, with the NUS [22] being an exception. The NUS claims to be the first method that combine a noise filter and undersampling to boost a classifier's performance, in which a k-nearest neighbors-based noise filter is firstly applied to remove noises from both classes followed by an undersampling procedure.

1) Subsubsection Heading Here: Subsubsection text here.

III. Two-Phase Resampling Method

In this section, the definition of the stochastic sensitivity measure and the details of the proposed TPRS are given in Section III-A and Section III-B, respectively.

A. Stochastic Sensitivity Measure

To evaluate the stability of a classifier, small random perturbations are added to training samples based on which the classifier was trained. If the classifier outputs are severely fluctuated by these small perturbations in inputs, the classifier is sensitive to the training samples or the training samples have high sensitivities with respect to the classifier. Training samples with high sensitivities are more likely to be misclassified by the classifier since a minor perturbation in the input leads to a severe fluctuation of the classifier output. These

samples are more likely to be noisy samples because classifiers are expected to perform well on normal samples. Therefore, sensitivity of a sample could be used as a heuristic to identify noisy samples in a dataset.

In this work, we define the sensitivity of a training sample x as the proportion of randomly perturbed samples with different predicted labels from the true label y of x and name it as the Stochastic Sensitivity Measure (SSM). It is formulated as follows:

$$SSM(x,h) = \frac{\sum_{i=1}^{\beta} |y - h(x^{(i)})|}{\beta} \quad (1)$$

where x, y, β, and $h(\cdot) \in \{0,1\}$ denote a given training sample, the true label of x, the i^{th} perturbed samples around x, the number of perturbed samples, and the predicted label given by the classifier h, respectively. The perturbed samples are created via a small perturbation of the input of the training sample and are located in a region which we name as Q-neighborhood. The Q-neighborhood of a training sample x is defined as follows [50]:

$$S_Q(x) = \{x_p | x_p = x + \Delta x, |\Delta x_i| \leq Q, i = 1, 2, \ldots, n\} \quad (2)$$

where x_p, Δx, Δx_i, Q, and n denote the perturbed sample, the magnitude of perturbation to the training sample, the magnitude of perturbation to the i^{th} feature of the training sample, the maximum magnitude of perturbation, and the number of features, respectively. A $Q = 0.1$ means that a maximum deviation of 10% from the training sample is allowed for perturbations [51].

Samples located within the Q-neighborhood of a training sample are expected to share similar information with the training sample and therefore belong to the same class as the training sample. A training sample yielding a large SSM value indicates that a classifier ~~trained on these data~~ has high possibility to misclassification because small input perturbations lead to large fluctuations of classifier outputs. Therefore, samples that yield high SSM values are more likely to be noisy samples with regard to classifier training.

Evaluating the SSM value of a sample using only one classifier may yield a high variance, moreover a classifier trained using an imbalanced dataset may be biased to the majority class because it tends to classify most samples as the majority class. Therefore, a neural network ensemble trained via a balanced bagging method is employed in this work to evaluate the SSM values of the training samples, which is formulated as follows:

$$SSM(x,H) = \frac{\sum_{t=1}^{\|H\|} SSM(x, H^{(t)})}{\|H\|} \quad (3)$$

where $H^{(t)}$ and $\|H\|$ denote the t^{th} base classifier in ensemble H and the number of base classifiers in H, respectively. The mean value of the SSM values of each training sample yielded by all the base classifiers in H is utilized as the final SSM value of each sample. The balanced bagging method is employed to train a neural network ensemble based on several balanced datasets by resampling the same number of samples from each class, which is given in Algorithm 1.

Algorithm 1 Balanced Bagging

Require:
training dataset D, number of base classifier T, learning algorithm L

Ensure:
Ensemble of base classifier H

1: **For** $t = 1$ to T
   1) Set sub-training dataset $U = \emptyset$
   2) Draw $\|D\|/2$ samples from each class randomly with replacement and put them in U
   3) Train a base classifier $H^{(t)}$ based on U using learning algorithm L

   EndFor

2: $H = arg\max_y \sum_{t:H^{(t)}(x)=y} 1$

Algorithm 2 SSM-based Noise Filter

Require:
training dataset D containing K classes, threshold λ, number of base classifiers T

Ensure:
noise-filtered dataset D'

1: **For** $u = 1$ to $K - 1$
   1) **For** $v = u + 1$ to K
      a) Train an ensemble H containing T neural networks based on samples from class u and v using balanced bagging
      b) Compute the average SSM value of each training sample from class u and v via H using formula (3)
      c) Remove samples yielding $SSM(x^{(i)}, H) > \lambda$

      EndFor

   EndFor

2: Set the filtered dataset as D'

Algorithm 3 TPRS

Require:
training dataset D, threshold λ, oversampling method, number of base classifier T, learning algorithm L

Ensure:
noise-filtered rebalanced dataset D^*, and a trained classifeir H^*

1: Remove noises in D using SSM-based noise filter and get a noise-filtered dataset D'
2: Apply oversampling on D' to get noise-filtered rebalanced dataset D^*
3: Train a final classifier H^* using L based on D^*

B. Framework of the Two-Phase Resampling Method (TPRS)

The Two-Phase Resampling (TPRS) method consists of two phases: noise filtering, and oversampling. Details of these phases are given in this section.

The first phase of TPRS employs the SSM-based noise filter to remove noises in all classes. Samples with SSM values higher than a pre-selected threshold λ are identified as noises and removed. In fact, the threshold controls how conservative one treats a sample as noise. The smaller the threshold, the more training samples are regarded as noises. In contrast, a too large threshold leads to very few noisy samples can be recognized and the performance of the oversampling method will be hindered. The threshold used in this work is 0.5, meaning that a training sample is regarded as noise if at least half of the neural networks are likely to misclassify this sample.

Since a dataset contains more than two classes in this work, an OAO scheme is utilized to decompose the multi-class problems into several two-class problems. Individual two-class problem is tackled by an ensemble as introduced in Section III-A. ~~If the balanced bagging is directly applied on the whole dataset, a very complex decision boundary may be learnt which brings negative impacts on the performance of the noise filter especially when the number of classes is large. In an extreme case, all samples from some classes could be wrongly recognized as noises and removed because of the poor decision boundary learnt.~~ The decomposition helps alleviate the negative effects of complex decision boundary because there are only two classes to be learnt. Applying the OAA scheme may produce similar results as that of directly applying balance bagging on the whole dataset, since relations between one class and other individual classes are hidden by merging other classes as one class. The details of the SSM-based noise filter are given in Algorithm 2.

The second phase of TPRS is to apply an oversampling procedure on the noise-filtered dataset obtained from the first phase to rebalance the class distribution. Any type of oversampling methods can be applied here, for instances the random oversampling or the SMOTE. Since the noises have been removed, new noises are not likely introduced in the dataset and the representation of minority classes can be safely enhanced.

After executing the TPRS, a multi-class classifier is trained using the noise-filtered rebalanced dataset obtained from the first and second phase. The classifier is expected to yield high robustness with respect to noises and high generalization capabilities, because both class noise and class imbalance issues have been properly handled.

The pseudocode of TPRS is given in Algorithm 3.

IV. Experimental Studies

In this section, we evaluate the effectiveness of the TPRS. In the experimental studies, Multi-Layer Perceptron Neural Network (MLPNN) has been used as the classifier in all methods for a fair comparison. However, the classifier can be of any chosen classifier. Section IV-A provides the experimental setup, Section IV-B shows the experimental results with discussions, and Section IV-C discusses the reason of the effectiveness of TPRS.

A. Experimental Setup

Nine multi-class datasets from the UCI dataset repository are used in the evaluation [52]. Characteristics of datasets, with respect to the number of features, the number of classes, the number of samples, the imbalance ratio and the class distribution are given in Table I. Imbalance Ratio (IR) is defined as the number of samples from the largest class divided by the number of samples from the smallest class. Majority classes are those classes containing more than average number of samples in a dataset and the rest classes are minority classes. For some of the datasets which are not imbalanced (i.e., "image-segmentation" and "iris"), synthetic imbalanced dataset is generated, as suggested in [14], by random under-sampling of some classes.

The performance metric adopted for this work is the Geometric-mean (G-mean), which is a commonly used metric for imbalanced classification problems. The G-mean takes the accuracy of each class into account and is defined in formula (4), where TPR represents the true positive rate or accuracy of each class.

$$G-mean = \sqrt{\sum_{i}^{K} TPR_i} \tag{4}$$

For the performance evaluation, a ten-time independent runs are employed for each dataset and the average result is recorded for performance comparison. For each repetition, the dataset is randomly split into two halves, one for training and the other for testing.

To compare the difference among methods, as suggested in [53], a ~~one sided Wilcoxon signed ranks test [54] is employed with significance level of 95%~~. The Wilcoxon signed-ranks test is a non-parametric test to detect significant differences between two sample means, which ranks the differences in average performances for each dataset, ignoring the signs and compares the ranks for the positive (the TPRS) and negative (compared method) differences. If the negative differences are lower than a critical value, then the TPRS significantly outperforms the compared method. The ~~Friedman's test with a post-hoc Hochberg's test~~ [55] will be also applied to compare the proposed method with other methods over multiple datasets.

The proposed approach is compared with a number of resampling techniques, namely the standard SMOTE (smote), the Borderline-SMOTE-1 (bsmote1), the Borderline-SMOTE-2 (bsmote2), the SMOTE-TOMEK (stomek), and the KINOS (kinos) with SMOTE as the oversampling method. All methods except the SMOTE are oversampling method combined with some kind of mechanisms to handle noises, where the SMOTE is used as the baseline to see if noise handling is effective. The parameter setting for each method is given in Table II. A single classifier trained using a dataset with no pre-processing (named as "none" in the experiment) is also included as the baseline.

TABLE I
CHARACTERISTICS OF UCI DATASETS

dataset	#features	#samples	#classes	IR	class distribution
cardiotocography	21	2126	10	10.92	384/579/53/81/72/332/252/107/69/197
dermatology	34	358	6	2.31	111/60/71/48/48
image_segmentation	19	630	7	6.6	50/50/50/50/50/50/330
iris	4	110	3	1.67	30/30/50
new_thyroid	5	215	3	5	150/35/30
splice	60	3190	3	2.16	767/768/1655
thyroid_ann	21	7200	3	40.16	166/368/6666
wine	13	178	3	1.48	59/71/48
wine_white	11	4873	5	13.48	163/1457/2198/880/175

Note: The first six classes in dataset "image-segmentation" are undersampled to contain only 50 samples to create an imbalanced class distribution. The first two classes in dataset "iris" are undersampled to contain only 30 samples.

TABLE II
PARAMETER SETTING FOR EACH METHOD

Algorithm	Parameters
none	No parameters to be set
smote	k=3
bsmote1	k=3, m=10
bsmote2	k=3, m=10
stomek	k=3
kinos	smote as oversampling method with k=3, kin_k=15, kin_tau=1
tprs	smote as oversampling method with k=3, Q=0.1, $\lambda = 0.5$,$T = 5$,β=50

To analyze the extent to which different methods handle noisy imbalanced datasets, various amount of additional noise has been introduced to the training datasets, for the datasets which contain little noises. We adopt a pair-wise noise introduction schema as follows: given a pair of classes (y_1, y_2) and a noise level ρ, an instance with label y_1 has a probability of ρ to be incorrectly labeled as y_2, so does an instance with label y_2. This mechanism was proposed by Zhu et al. [24], claiming that in realistic situations, only certain types of classes are likely to be mislabeled. In this work, we only corrupt the training dataset and every majority sample has a probability ρ to be mislabeled as minority class and vice versa. For example, given a training dataset with four classes where class 1 and 2 are majority class and class 3 and 4 are minority class, to corrupt this training dataset with a noise level of 5%, each sample has 5% of chance to be selected to be corrupted. If a majority sample is selected, a minority class (class 3 or 4) is randomly assigned to it. Similarly, if a minority sample is selected, then a majority class (class 1 or 2) is randomly assigned to it. Five levels of noises are introduced in the training datasets in this work: 5%, 10%, 20%, 30%, and 40%.

B. Experimental Results and Discussions

In this section, experiments are conducted to show how different methods handle noises under different noise levels.

The trends of G-mean values yielded by different methods on different datasets by varying the noise levels are given in from Fig. 1 to Fig. 9, where the x-axis represents the noise level and y-axis the G-mean. In these figures, the "none", "smote", "bsomte1", "bsmote2", "stomek", "kinos" and "tprs" are marked in red circle, green cross, blue pentagram, light blue upward-pointing triangle, brown square, yellow diamond, and magenta asterisk, respectively. Fig. 10 gives the average performance of different methods by varying the noise levels. From these figures, several important observations should be addressed:

1) As the noise level increases, all methods are prone to decreasing the performances on all datasets in terms of G-mean no matter what levels of class imbalance occur in these datasets. This is not surprising because as the noise level increases, the learning complexity of the imbalanced datasets grows and results in very poor training of classifiers.
2) Although all methods suffer from performance deterioration as the noise level increases, the "tprs" yields the best performances among all methods in almost all datasets under all noise level settings with very few exceptions. Moreover, the performance gaps between the "tprs" and other methods tend to enlarge as the noise level increases. These show the robustness of the "tprs" with respect to different imbalance ratios and different noise levels.
3) From Fig. 10, the benchmark models "none", marked in red circle usually yields the worst results in all noise level settings in average. This is intuitive because without properly handling both the class noise and class imbalance issues in the dataset, on which a classifier trained is not expected to yield a high generalization capability. This confirms the necessity of techniques to tackle both class imbalance and class noise problems. The "tprs" yields the best average results in all settings. Additionally, the performance gains compared with other methods are prone to getting larger as the noise level increases, which shows the effectiveness of the "tprs" especially in environments with high noise levels.

Table III shows the Wilcoxon signed-ranks test results by giving the p-value of the comparison between each method and

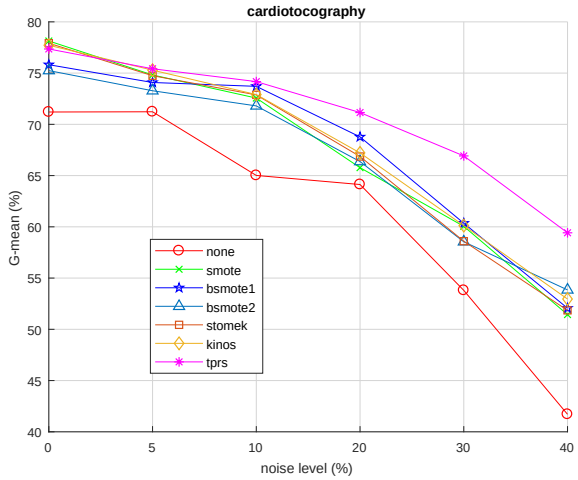

Fig. 1. cardiotocography

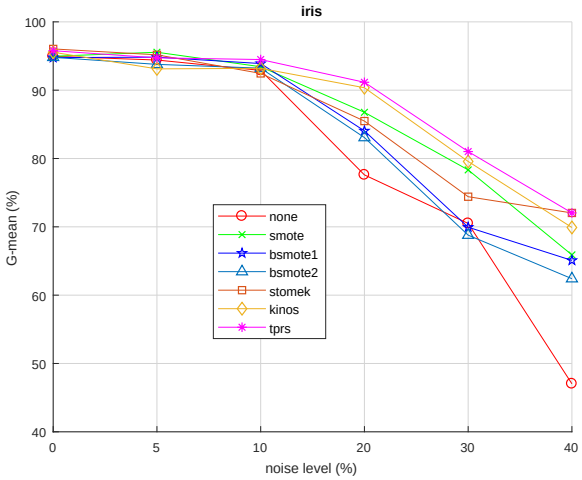

Fig. 4. iris

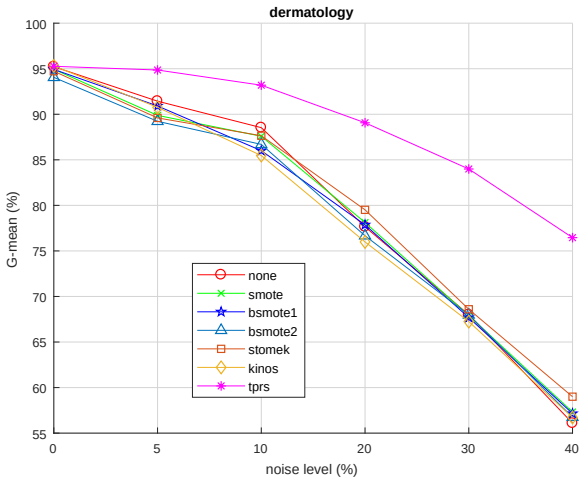

Fig. 2. dermatology

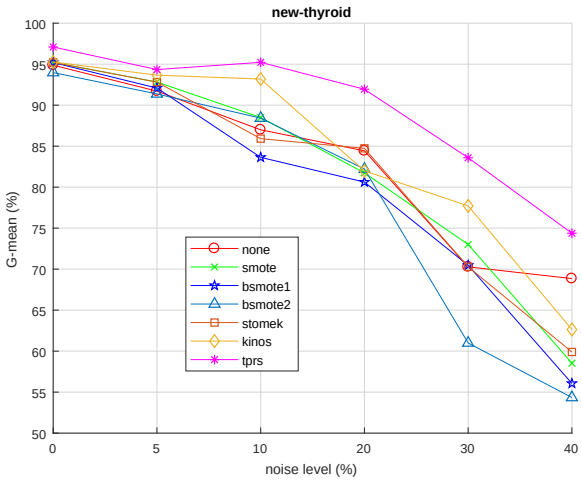

Fig. 5. new-thyroid

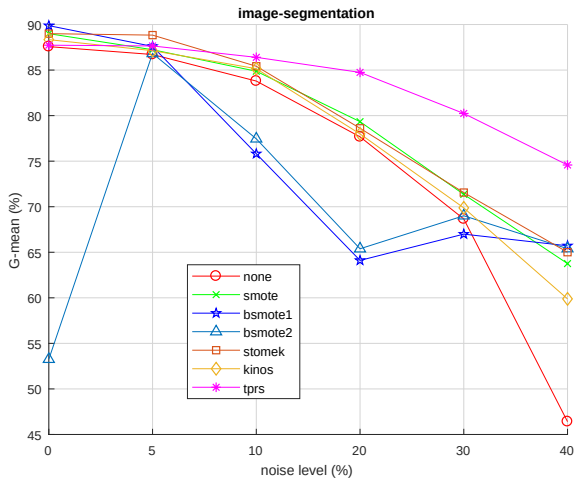

Fig. 3. image-segmentation

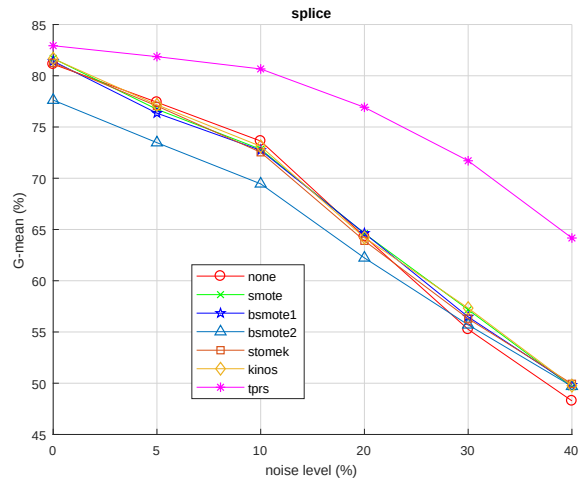

Fig. 6. splice

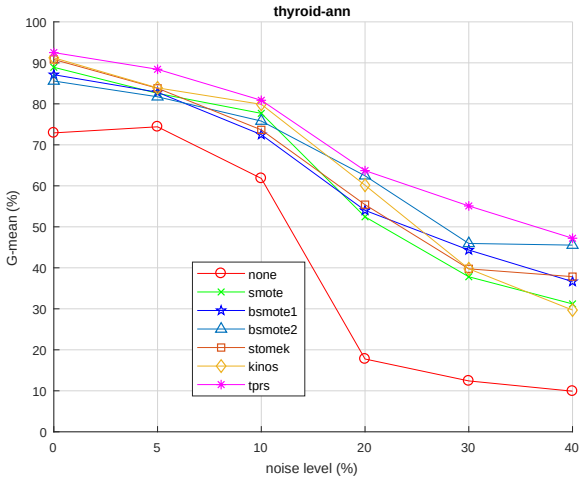

Fig. 7. thyroid-ann

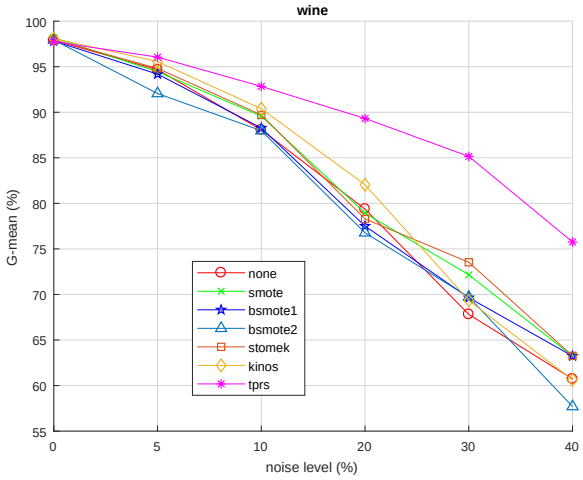

Fig. 8. wine

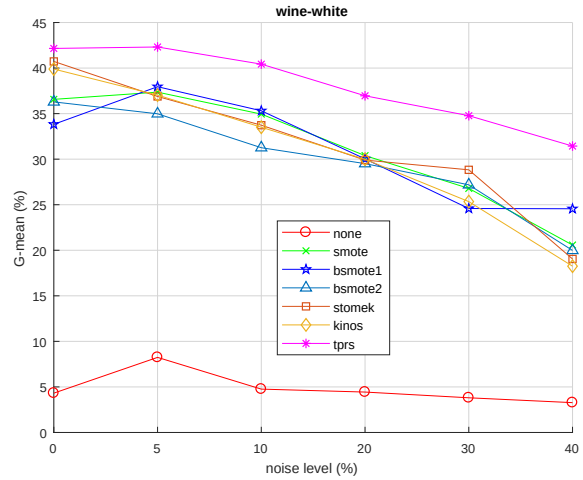

Fig. 9. wine-white

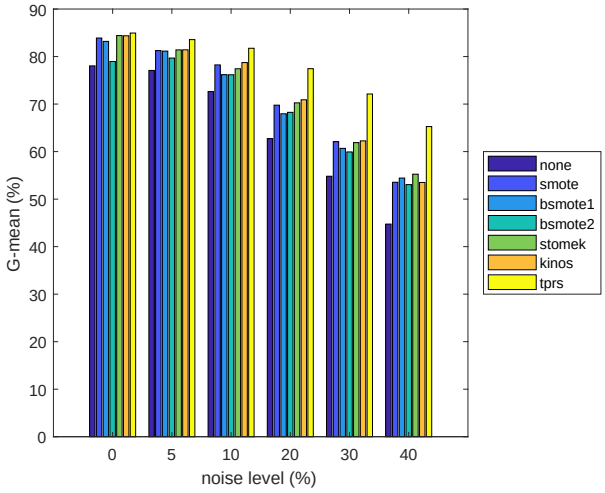

Fig. 10. Average Performances of Different Methods by Varying Noise Levels

the TPRS. In Table III, a method yielding a p-value smaller than 0.05 indicates that the "tprs" significantly outperforms the corresponding method. p-values larger than 0.05 are highlighted in bold face. From Table III, one can see that the "tprs" significantly outperforms the "none" and the "bsmote2" when the noise level is 0%. As the noise level increases, the "tprs" outperforms all other methods significantly, which again shows the effectiveness and robustness of the "tprs" in highly noisy imbalanced environments.

Table IV reports the results of Friedman's test and the post hoc Hochberg's test in terms of G-mean. Values in the row "p-F" indicates the p-value computed by the Friedman's test. A p-F value lower than 0.05 means that there is significant difference between methods in comparisons and the post hoc test is then applied to find out which methods are yielding significantly different performances. The p-H columns give the p-value computed by the post hoc Hochberg's test, a value of which lower than 0.05 indicates that the "tprs" significantly outperform the corresponding method. p-H values higher than 0.05 are highlighted in bold face. From Table IV, the "tprs" always yields the lowest rank in all noise setting, which is why the "tprs" is used as the control method used in Friedman's test. All p-F values are lower than 0.05, which allows the post hoc Hochberg's test to be executed. The "tprs" outperforms the "none" and the "bsomte2" significantly when the noise level is 0% and 5%, given by the facts that their p-H values are lower than 0.05. There are no significant differences between the "tprs" and the other methods when noise level is 0% and 5%, though one should notice that the "tprs" ranks the first in both settings. As for other noise level settings, the "tprs" yields significantly better performance than all other methods in comparisons.

C. Why the TPRS Works

In this work, the oversampling procedure applied in the methods in comparisons is the off-the-shelf SMTOE. The major differences between the TPRS and other resampling

TABLE III
WILCOXON TEST RESULTS BY TAKING THE TPRS AS CONTROL METHOD

method	noise level					
	0%	5%	10%	20%	30%	40%
none	0.009766	0.00586	0.003906	0.003906	0.0019532	0.0019532
smote	>**0.2**	0.0371	0.00586	0.003906	0.0019532	0.0019532
bsmote1	**0.10546**	0.019532	0.009766	0.0019532	0.0019532	0.0019532
bsmote2	0.00586	0.003906	0.003906	0.0019532	0.0019532	0.0019532
stomek	**0.19336**	0.0371	0.00586	0.003906	0.0019532	0.0019532
kinos	>**0.2**	0.013672	0.019532	0.0019532	0.0019532	0.0019532

TABLE IV
RESULTS ON FRIEDMAN'S TEST. THE P-F IS THE P-VALUE COMPUTED BY THE FRIEDMAN'S TEST AND P-H IS THE ADJUSTED P-VALUE COMPUTED BY THE POST HOC HOCHBERG'S TEST

noise level	0%		5%		10%	
methods	rank	p-H	rank	p-H	rank	p-H
tprs	**2.7**	N/A	**2**	N/A	**1.6**	N/A
none	5.85	0.00556	5.1	0.006664	4.8	0.004625
smote	2.75	**0.958724**	3.5	**0.147299**	3.5	0.049219
bsmote1	4.6	**0.196877**	3.9	**0.147299**	4.7	0.005332
bsmote2	5.9	0.005552	6.4	0.000032	5.5	0.000325
stomek	3.4	**0.958724**	3.7	**0.147299**	4.4	0.011257
kinos	2.8	**0.958724**	3.4	**0.147299**	3.5	0.049219
p-F	0.000167		0.000337		0.001835	
noise level	20%		30%		40%	
methods	rank	p-H	rank	p-H	rank	p-H
tprs	**1.3**	N/A	**1**	N/A	**1**	N/A
none	5	0.000641	5.7	0.000007	6.2	0
smote	3.8	0.017279	4	0.00011	4.6	0.000777
bsmote1	4.7	0.001731	4.7	0.000513	3.7	0.010387
bsmote2	5.6	0.000051	5.1	0.003802	4.8	0.000419
stomek	3.6	0.017279	3.5	0.003802	3.2	0.022773
kinos	4	0.015581	4	0.009661	4.5	0.000874
p-F	0.000383		0.000046		0.000007	

methods equipped with a noise filter are twofold. Firstly, the oversampling procedure is applied after the noise filtering procedure. In this way, new noises are not likely be introduced in the dataset and participate in the classifier training. The STOMEK applies the noise filter after the oversampling. This may result in that newly introduced noisy samples form several small clusters, which are difficult for the k-nearest neighbors-based method to identify as noises and thus the de-noising performance is jeopardized. Secondly, unlike the k-nearest neighbors-based method, the noise filter employed in this work is based on evaluating stochastic sensitivities of the training samples, which takes the classifier training into account. The SSM-based noise filter naturally takes the classifier training into account by removing samples that are very hard for classifiers to learn, which makes the boundary between classes more clear and relives the overlapping issues among classes to some extent.

To show the performance gains in terms of G-mean of the TPRS by integrating the SSM-based noise filter and oversampling, Fig 11 shows the average gains of the TPRS in different noise level settings over all datasets compared with the SMOTE, STOMEK, and KINOS. The SMOTE is applied in all resampling methods in this work and therefore used as the baseline, the STOMEK is used as a representative for methods that applies noise filtering after oversampling, and the KINOS is used as a representative for methods that handles noises before oversampling. Although the Borderline-SMOTE uses similar procedure as the KINOS, it is shown that when the KINOS is integrated with the Borderline-SMOTE as oversampling method, the KINOS improves the performance of the Borderline-SMOTE [44]. Therefore, the KINOS instead of the Borderline-SMOTE is used here for comparison. From Fig 11, one can see that in all noise level settings, the "tprs" has indeed enhanced the performance of the "smote" by removing noises before executing oversampling. Compared with the "stomek" and the "kinos", the "tprs" still shows quite large performance improvements, from 0.82% to 24.57% for the "stomek" and from 1.03% to 30.95% for the "kinos" as noise level increases. Moreover, the performance gains are prone to enlarging, showing the effectiveness of the "tprs" on handling highly noisy imbalanced environments.

V. CASE STUDY: APPLICATION IN HUMAN ACTIVITY RECOGNITION USING SENSOR DATA

Human activity recognition (HAR) is a fundamental component to a broad range of application areas including ambient assistive living, connected health and pervasive computing [56]. It is commonly used in rehabilitation systems for monitoring the activities of elderly residents to support the management, and also the prevention, of chronic disease. Another common application area is HAR within smart homes, as a key motivation behind HAR research is to monitor the health of smart home inhabitants by tracking their daily activities. In relation to promoting physical activity, HAR is

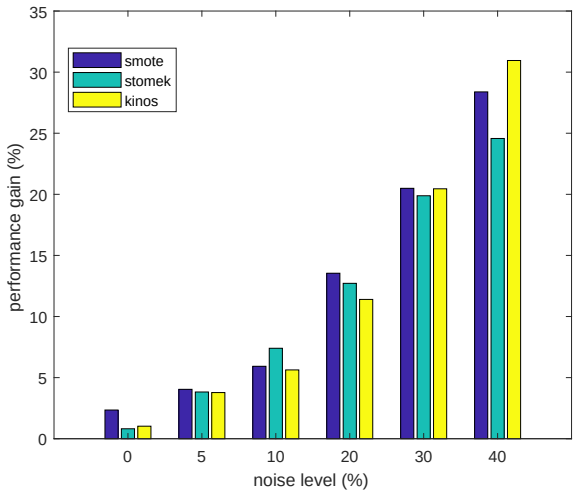

Fig. 11. Average Performance Gains of the TPRS Compared with the SMOTE, STOMEK, and KINOS

applied in rehabilitation centers that focus on stroke rehabilitation and those with motor disabilities [57]. HAR can be generally deemed within two categories: either sensor-based or vision-based activity recognition. Particularly, sensor-based activity recognition has attracted considerable research interest in ubiquitous computing due to advancements with sensor technologies and wireless sensor networks [56]. A frequently utilized wearable sensor for monitoring human activities is the accelerometer, which is particularly effective in observing movements such as walking, standing, sitting, and ascending stairs [58]. However, among these recognized activities, transitional activities are often discarded or the recognition rates are quite low because their occurrence is much lower and the duration is much shorter compared to static and dynamic activities. In general, when transitional activities occur, an online activity recognition system may reduce the classification performance as these activities are not specified in the training dataset or the system has not learnt well on these activities with too few training samples.

There are a few reasons that make the recognition of the transitional activities difficult. On one hand, compared to static and dynamic activities, the occurrences of transitional activities are too low. Severe class imbalance hinders the recognition of the transitional activities as the classifier is biased to the majority classes (static and dynamic activities) and tend to classify most events as static or dynamic activities. On the other hand, people tend to perform consecutive activities which interleave with each other, i.e. people perform activities one by one which are not clearly separated by pauses, e.g. "Stand-to-Walk" interleaves with both "Standing" and "Walking". Additionally, it is difficult to define the exact start and end time of an activity, i.e. noises are inherent in between the activities.

A. Data Collection and Preprocessing

The sensor based human activity recognition dataset is collected by the staff in Ulster University [59]. The sensor data was collected using a tri-axial accelerometer placed on each participant's right wrist. Ten healthy adults (5 males and 5 females) were asked to perform 12 activities in a controlled laboratory environment. Three types of activities are considered in this work, dynamic activities, for instance walking and sweeping, static activities, for instance standing and sleeping, and transitional activities, stand-to-walk and walk-to-stand. The descriptions of the 12 activities are given in Table V. Since transitional activities normally occur within a very short duration, they were allowed to repeat for more times to obtain sufficient data samples.

A total number of 2186855 data samples were recorded and were further preprocessed into a HAR dataset with 15184 data points and 77 features extracted from the time and frequency domain according to the guideline in [59]. The characteristics of this dataset are given in Table VI.

B. Experimental Results on the HAR Task using Sensor Data

Similar to the previous setup, the HAR dataset is randomly divided into two halves, one for training and the other for testing. The process is repeated for ten times to avoid random effects. Five levels of noises are also introduced in this datasets. The numeric results in terms of G-mean on the HAR task using sensor data are given in Table VII. The * symbol indicates that the "tprs" significantly outperforms the corresponding method with 95% confidence according to Student's T-test result. From Table VII, the "tprs" yields significantly better G-mean results than other methods in all comparisons when noise level ranging from 5% to 40%. When noise level is 0%, the "tprs" yields better results than the "none", "bsmote1", "bsmote2", and the "kinos" significantly. The "tprs" yields slightly lower G-mean than the "smote" and higher G-mean than the "stomek", but the differences are not significant.

VI. Conclusions and Future Works

In this work, we propose a Two-Phase Resampling (TPRS) method for multi-class noisy imbalanced classification problems. Noisy samples are identified and removed via a SSM-based noise filter in the first phase to obtain a noise-filtered dataset. The noise-filtered dataset is then oversampled in the second phase to safely enhance the representation of the minority classes to rebalance the class distributions. After the TPRS, a multi-class classifier is trained based on the noise-filtered rebalanced dataset, which is expected to yield a high generalization capability. Experiments on nine UCI multi-class imbalanced datasets with five levels of class noises are carried out to analyze the performances of different methods. Moreover, a real-world human activity recognition task is taken as a case study to show the performance of the proposed TPRS method. Experimental results show that the TPRS significantly outperforms the state-of-the-art methods in terms of G-mean.

Future works concerning the TPRS may include the following. For example, the MLPNN is used as the classifier for all methods in comparisons. Different classifiers, for examples RBFNN, SVM, and decision trees can be employed to analyze

TABLE V
DETAILED DESCRIPTIONS OF THE TWELVE ACTIVITIES [59]

Activity	Activity details
Static Activities	
Class 1, Standing	Standing still for 5 mins, the participants could stand still or stand still and talk, it is better not to move the arms and legs too much.
Class 2, Watching TV	Watching TV at smart home, with sitting on the sofa in whatever posture the participant feels comfortable for 5 mins, changing sitting posture is allowed.
Class 3, Sleeping	Lying on the sofa while doing nothing for 5 mins, small movements such as changing the lying posture are allowed.
Dynamic activities	
Class 4, Walking	5 mins walking on treadmill with the set speed.
Class 5, Running	5 mins running on the treadmill.
Class 6, Sweeping	5 mins sweeping with the vacuum cleaner in the home area.
Transitional activities	
Class 7, Stand-to-walk	In order to capture the whole transition, the participant is told to perform the stand-to-walk-to-stand, standing still for 15s then start to walk, keep walking for 15s, then standing still for 15s, repeat for 15 times. The first foot step movement and the last foot step movement would be labeled. This transition will be divided into stand-to-walk and walk-to-stand in the data analysis phase.
Class 8, Walk-to-stand	
Class 9, Stand-to-sit	The participant is told to stand still for 15s and then sit on the chair. The start point is backward movement after standing, the end point is sitting on the sofa (record finish point), repeat for 15 times.
Class 10, Sit-to-stand	The participant is told to sit on the chair for 10s and then stand up. The start point is forward movement after sitting, the end point is standing still (record finish point), repeat for 15 times.
Class 11, Sit-to-lie	The participant is told to sit on the sofa for 15s and then lie down. The start point is backward movement after sitting, the end point is lying on the sofa (record finish point), repeat for 15 times.
Class 12, Lie-to-sit	The participant is told to lie on the sofa for 15s and then sit on the sofa. The start point is forward movement after lying on the sofa, the end point is sitting on the sofa (record finish point), repeat for 15 times.

TABLE VI
CHARACTERISTICS OF THE PREPROCESSED HAR DATA

Dataset	#samples	#features	Class Distribution	IR
HAR	15184	77	2380/2378/2374/2328/2281/2340/182/182/185/188/180/186	13.22

TABLE VII
MEAN AND STANDARD DEVIATION VALUES OF G-MEAN PRODUCED BY DIFFERENT METHODS UNDER DIFFERENT NOISE LEVELS

noise level (%)	none	smote	bsmote1	bsmote2	stomek	kinos	tprs
0	64.62±3.54*	**75.91±1.29**	73.73±1.67*	73.23±1.53*	75.29±1.33	74.63±1.64*	75.71±1.84
5	65.06±2.07*	71.03±1.60*	71.46±0.94*	70.46±1.88*	71.03±1.42*	70.86±2.20*	**73.09±1.59**
10	61.67±1.97*	66.95±1.52*	68.73±2.14*	69.36±1.57*	66.28±1.84*	67.52±2.18*	**71.53±1.52**
20	56.36±2.17*	59.95±2.49*	62.33±1.51*	61.50±2.12*	60.14±1.69*	61.04±2.74*	**67.80±1.43**
30	50.32±4.19*	52.60±2.67*	57.32±3.19*	56.56±3.53*	53.67±2.51*	54.58±2.36*	**62.45±2.99**
40	45.25±4.40*	48.97±3.87*	50.65±4.10*	47.57±3.60*	47.76±2.88*	45.70±3.07*	**59.77±1.42**

the effectiveness of the TPRS. In addition, the TPRS serves as a wrapper method to deal with noisy imbalanced classification problems, but only the off-the-shelf SMOTE is employed in this work. Different oversampling methods can be applied here to analyze the effects of different oversampling methods, for examples the RWO method, Borderline-SMOTE, etc. One limitation of the TPRS is that the Q-neighborhood in the calculation of the SSM is only defined in real feature spaces. Researches can be carried out to extend the definition of the Q-neighborhood to binary and nominal feature space. Moreover, the Q value is fixed in this work, how to determine the optimal value for different datasets automatically remains an open research problem. Another limitation of the TPRS is that the OVO scheme is applied here to handle multi-class problems, which has high time complexity. Effects of different schemes can be analyzed and researched to lower the time complexity. Finally, only oversampling is considered in this work to handle imbalanced classification problems. The combination of the undersampling and the SSM based noise filtering seems to be a promising research area.

ACKNOWLEDGMENT

This work was supported by National Natural Science Foundation of China under Grant 61572201, Guangzhou Science and Technology Plan Project 201804010245, and Fundamental Research Funds for the Central Universities 2017ZD052.

REFERENCES

[1] S. C. Mukhopadhyay, "Wearable sensors for human activity monitoring: a review," in *IEEE Sensors Journal*, vol. 15, no. 3, pp. 1321-1330, 2014.
[2] H. He, and E. A. Garcia, "Learning from imbalanced data," in *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263-1284, 2009.
[3] P. Branco, L. Torgo, and R. P. Ribeiro, "A survey of predictive modeling on imbalanced domains," in *ACM Computing Surveys*, vol. 49, no. 2, article 31, 2016.
[4] H. Guo, Y. Li, J. Shang, M. Gu, Y. Huang, and B. Gong, "Learning from class-imbalanced data: review of methods and applications," in *Expert Systems with Applications*, vol. 73, pp. 220-239, 2016.
[5] B. Krawczyk, "Learning from imbalanced data: open challenges and future directions," in *Progress in Artificial Intelligence*, vol. 5, no. 4, pp. 1-12, 2016.
[6] S. Wang, and X. Yao, "Multi-class imbalance problems-analysis and potential solutions," in *IEEE Transactions on Systems Man and Cybernetics Part B*, vol. 42, no. 4, pp. 1119-1130, 2012.

[7] Z.-H. Zhou, and X.-Y. Liu, "Training cost-sensitive neural network with methods addressing the class imbalance problem," in *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 1, pp. 63-77, 2006.

[8] P. Jeatrakul, and K. W. Wong, "Enhancing classification performance of multi-class imbalanced data using the OAA-DB algorithm," in *International Joint Conference on Neural Networks*, pp. 1-8, 2012.

[9] A. Fernandez, V. Lopez, M. Galar, M. J. D. Jesus, and F. Herrera, "Analyzing the classification of imbalanced data-sets with multiple classes: binarization techniques and ad-hoc approaches," in *Knowledge-Based Systems*, vol. 42, no. 2, pp. 97-110, 2013.

[10] C. L. Castro, and A. P. Braga, "Novel cost-sensitive approach to improve the multilayer perceptron performance on imbalanced data," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 6, pp. 888-899, 2013.

[11] Y. Tang, Y.-Q. Zhang, N. V. Chawla, and S. Krasser, "SVMs modeling for highly imbalanced classification," in *IEEE Transactions on Systems Man and Cybernetics Part B*, vol. 39, no. 1, pp. 281-288, 2009.

[12] J. Mathew, C. K. Pang, M. Luo, and W. H. Leong, "Classification of imbalanced data by oversampling in kernel space of support vector machines," in *IEEE Transactions on Neural Networks and Learning Systems*, doi: 10.1109/TNNLS.2017.2751612.

[13] B. Gu, V. S. Sheng, K. Y. Tay, W. Romano and S. Li, "Cross validation through two-dimensional solution surface for cost-sensitive SVM," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1103-1121, June 1 2017.

[14] M. Lin, K. Tang, and X. Yao, "Dynamic sampling approach to training neural networks for multiclass imbalance classification," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 4, pp. 647-660, 2013.

[15] F. Fernandez-Navarro, C. Hervas-Martinez, and P. Antonio-Gutierrez, "A dynamic oversampling procedure based on sensitivity for multi-class problems," in *Pattern Recognition*, vol. 44, no. 8, pp. 1821-1833, 2011.

[16] R. Wang, T. Liu and D. Tao, "Multiclass learning with partially corrupted labels," in *IEEE Transactions on Neural Networks and Learning Systems*, doi: 10.1109/TNNLS.2017.2699783.

[17] G. M. Weiss, "Mining with rarity: a unifying framework," in *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 7-19, 2004.

[18] C. Seiffert, T. M. Khoshgoftaar, J. V. Hulse, and A. Folleco, "An empirical study of the classification performance of learners on imbalanced and noisy software quality data," in *Information Sciences*, vol. 259, pp. 571-595, 2014.

[19] T. M. Khoshgoftaar, J. V. Hulse, and A. Napolitano, "Supervised neural network modeling: an empirical investigation into learning from imbalanced data with labeling errors," in *IEEE Transactions on Neural Networks*, vol. 21, no. 5, pp. 813-830, 2010.

[20] T. M. Khoshgoftaar, J. V. Hulse, and A. Napolitano, "Comparing boosting and bagging techniques with noisy and imbalanced data," in *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 41, no. 3, pp. 552-568, 2011.

[21] R. F. A. B. de Morais, and G. C. Vasconcelos, "Under-sampling the minority class to improve the performance of over-sampling algorithms in imbalanced data sets," in *Proceedings of International Joint Conference on Artificial Intelligence*, 2017.

[22] Q. Kang, X.-S. Chen, S.-S. Li, and M.-C. Zhou, "A noise-filtered under-sampling scheme for imbalanced classification," in *IEEE Transactions on Cybernetics*, vol. 47, no. 12, pp. 4263-4274, 2017.

[23] J. A. Saez, J. Luengo, J. Stefanowski, and F. Herrera, "SMOTECIPF: addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering," in *Information Sciences*, vol. 291, pp. 184-203, 2015.

[24] X. Zhu, X. Wu, and Q. Chen, "Eliminate class noise in large datasets," in *Proceedings of the International Conference on Machine Learning*, pp. 920-927, 2003.

[25] D. Gamberger, N. Lavrac, and C. Groselj, "Experiments with noise filtering in a medical domain," in *Proceedings of the International Conference on Machine Learning*, pp. 143-151, 1999.

[26] T. M. Khoshgoftaar, V. Joshi, and N. Seliya, "Detecting noisy instances with the ensemble filter: a study in software quality estimation," in *International Journal of Software Engineering and Knowledge Engineering*, vol. 16, no. 1, pp. 53-76, 2006.

[27] X. Zhu, and X. Wu, "Class noise handling for effective cost-sensitive learning by cost-guided iterative classification filtering," in *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 10, pp. 1435-1440, 2006.

[28] T. M. Khoshgoftaar, and P. Rebours, "Improving software quality prediction by noise filtering techniques," in *Journal of Computer Scinece and Technology*, vol. 22, no. 3, pp. 387-396, 2007.

[29] J. Von Neumann, "Various techniques used in connection with random digits," in *National Bureau of Standards Applied Math Series*, vol. 12, pp. 36-38, 1951.

[30] W.-C. Lin, C.-F. Tsai, Y.-H. Hu, and J.-S. Jhang, "Clustering-based undersampling in class-imbalanced data," in *Information Sciences*, vol. 409-410, pp. 17-26, 2017.

[31] W. W. Y. Ng, J. Hu, D. S. Yeung, S. Yin, and F. Roli, "Diversified sensitivity-based undersampling for imbalance classification problems," in *IEEE Transactions on Cybernetics*, vol. 45, no. 11, pp. 2402-2412, 2014.

[32] Q. Kang, L. Shi, M. Zhou, X. Wang, Q. Wu and Z. Wei, "A distance-based weighted undersampling scheme for support vector machines and its application to imbalanced classification," in *IEEE Transactions on Neural Networks and Learning Systems*, doi: 10.1109/TNNLS.2017.2755595.

[33] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," in *Journal of Artificial Intelligence Research*, vol. 16, no. 1, pp. 321-357, 2002.

[34] H. Zhang, and M. Li, "RWO-sampling: a random walk over-sampling approach to imbalanced data classification," in *Information Fusion*, vol. 20, no. 1, pp. 99-116, 2014.

[35] L. Abdi, and S. Hashemi, "To combat multi-class imbalanced problems by means of over-sampling techniques," in *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 1, pp. 238-251, 2015.

[36] X. Yang, Q. Kuang, W. Zhang and G. Zhang, "AMDO: an over-sampling technique for multi-class imbalanced problems," in *IEEE Transactions on Knowledge and Data Engineering*, doi: 10.1109/TKDE.2017.2761347.

[37] D. R. Wilson, and T. R. Martinez, "Improved heterogeneous distance functions," in *Journal of Artificial Intelligence Research*, vol. 11, no. 1, pp. 1-34, 1997.

[38] C. Seiffert, T. M. Khoshgoftaar, J. V. Hulse, and A. Napolitano, "RUSBoost: a hybrid approach to alleviating class imbalance," in *IEEE Transactions on Systems, Man, and Cybernetics C Part A: Systems and Humans*, vol. 40, no. 1, pp. 185-197, 2010.

[39] R. Barandela, R. M. Valdovinos, and J. S. Sanchez, "New applications of ensemble of classifiers," in *Pattern Analysis and Applications*, vol. 6, no. 3, pp. 245-256, 2003.

[40] S. Wang, and X. Yao, "Diversity analysis on imbalanced data sets by using ensemble models," in *IEEE Symposium on Computational Intelligence and Data Mining*, pp. 324-331, 2009.

[41] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "SMOTEBoost: improving prediction of the minority class in boosting," in *Proceedings of 7th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pp. 107-119, 2003.

[42] X. Zhang, Y. Zhuang, W. Wang, and W. Pedrycz, "Transfer boosting with synthetic instances for class imbalanced object recognition," in *IEEE Transactions on Cybernetics*, vol. 48, no. 1, pp. 357-370, 2018.

[43] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," in *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 2, pp. 539-550, 2009.

[44] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning," in *Proceedings of International Conference on Advances in Intelligent Computing*, pp. 878-887, 2005.

[45] S. Barua, Md. M. Islam, X. Yao, and K. Murase, "MWMOTE-majority weighted minority oversampling technique for imbalanced data set learning," in *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 2, pp. 405-425, 2013.

[46] T. Zhu, Y. Lin, and Y. Liu, "Synthetic minority oversampling technique for multiclass imbalance problems," in *Pattern Recognition*, vol. 72, pp. 327-340, 2017.

[47] 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," in *Proceedings of ACM SIGKDD Explorations*, vol. 6, no. 1, pp. 20-29, 2004.

[48] D. L. Wilson, "Asymptotic properties of nearest neighbor rules using edited data," in *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 2, no. 3, pp. 408-421, 1972.

[49] I. Tomek, "Two modifications of CNN," in *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-6, no. 11, pp. 769-772, Nov. 1976.

[50] D. S. Yeung, W. W. Y. Ng, D. Wang, E. C. C. Tsang, and X.-Z. Wang, "Localized generalization error model and its application to architecture selection for radial basis function neural network," in *IEEE Transactions on Neural Networks*, vol. 18, no. 5, pp. 1294-1305, 2007.

[51] D. S. Yeung, J.-C. Li, W. W. Y. Ng, and P. P. K. Chan, "MLPNN training via a multiobjective optimization of training error and stochastic

sensitivity," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 5, pp. 978-992, 2016.

[52] M. Lichman, "UCI machine learning repository," in *Irvine, CA: University of California, School of Information and Computer Science*, http://archive.ics.uci.edu/ml, 2013.

[53] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," in *Journal of Machine Learning Research*, vol. 7, no. 1, pp. 1-30, 2006.

[54] F. Wilcoxon, "Individual comparisons by ranking methods," in *Biometrics bulletin*, vol. 1, no. 6, pp. 80-83, 1945.

[55] S. Garcia, A. Fernandez, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining- experimental analysis of power," in *Information Sciences*, vol. 180, pp. 2044-2064, 2010.

[56] T. Gu, L. Wang, Z. Wu, X. Tao and J. Lu, "A pattern mining approach to sensor-based human activity recognition," in IEEE Transactions on Knowledge and Data Engineering, vol. 23, no. 9, pp. 1359-1372, 2011.

[57] O. C. Ann and L. B. Theng, "Human activity recognition: a review," in *Proceedings of IEEE International Conference on Control System, Computing and Engineering*, Batu Ferringhi, pp. 389-393, 2014.

[58] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook and Z. Yu, "Sensor-based activity recognition," in *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 790-808, 2012.

[59] Q. Ni, T. Patterson, I. Cleland, and C. D. Nugent, "Dynamic detection of window starting positions and its implementation within an activity recognition framework," in *Journal of Biomedical informatics*, vol. 62, pp. 171-180, 2016.