Handling Class Overlap and Imbalance to Detect Prompt Situations in Smart Homes

Barnan Das*, Narayanan C. Krishnan†, Diane J. Cook†
School of Electrical Engineering and Computer Science
Washington State University, Pullman, WA 99164-2752
Email: *barnandas@wsu.edu, †{ckn, cook}@eecs.wsu.edu

***Abstract*—The class imbalance problem is a well-known classification challenge in machine learning that has vexed researchers for over a decade. Under-representation of one or more of the target classes (minority class(es)) as compared to others (majority class(es)) can restrict the application of conventional classifiers directly on the data. In addition, emerging challenges such as overlapping classes, make class imbalance even harder to solve. Class overlap is caused due to ambiguous regions in the data where the prior probability of two or more classes are approximately equal. We are motivated to address the challenge of class overlap in the presence of imbalanced classes by a problem in pervasive computing. Specifically, we are designing smart environments that perform health monitoring and assistance. Our solution, ClusBUS, is a clustering-based undersampling technique that identifies data regions where minority class samples are embedded deep inside majority class. By removing majority class samples from these regions, ClusBUS preprocesses the data in order to give more importance to the minority class during classification. Experiments show that ClusBUS achieves improved performance over an existing method for handling class imbalance.**

Keywords*—*Class imbalance; overlapping classes; sampling; automated prompting.

I. Introduction

As machine learning techniques mature and are used to tackle scientific problems, a variety of challenges arise due to the nature of the underlying data. A well-recognized challenge that has attracted growing attention from both academia and industry is the class imbalance problem. The class imbalance problem is concerned with the performance of machine learning classifiers when one or more target classes (the *minority* class(es)) is under-represented in comparison with the other classes (the *majority* class(es)). This problem exists in many problem domains such as cancerous cell identification [1], oil-spill detection [2], fraud detection [3], keyword extraction [4], and text classification [5], where identifying members of the minority class is critical, sometimes more so than achieving optimal overall accuracy for the majority class. In spite of making progress in this area, researchers are facing new emerging challenges that make the problem harder to solve with existing techniques. One such niche yet critical problem is class overlap in which data samples appear as valid examples of more than one class. This situation occurs widely in character recognition [6] and drug design [7], where data samples from different classes have very similar characteristics. The minor differences that are present between the samples of two different classes are usually difficult to capture using only data attributes proposed by the domain experts. As a result, there is a growing need to deal with this issue algorithmically. We are motivated to pursue the challenge of handling data with class overlap in the presence of imbalanced classes by a problem in pervasive computing. Specifically, we are designing smart environments that perform health monitoring and assistance.

The world's population is aging, with the estimated number of individuals over the age of 85 expected to triple by 2050 [8]. Currently, 50% of adults aged 85+ need assistance with everyday activities, with one in three households anticipated to have at least one family member with cognitive decline within the next decade [9]. Because more individuals are living longer with chronic diseases (e.g., Alzheimers disease, heart disease, osteoporosis) there will be an emerging shortage in the care workforce. Therefore, we must consider innovative health care options if we are to provide quality care to our aging population. Moreover, with proactive health care and real-time intervention, we can reduce caregiver burden.

Studies have shown that smart environment technologies can detect errors in activity completion and might be utilized to extend independent living in one's own home without compromising safety [10], [11]. One type of intervention that is valuable for individuals with cognitive impairment is automated prompts that aid with activity initiation and completion. To date, most applications of the smart environment technology for intervention have relied upon partial or full expert design of prompting rules [12], [13]. Our approach to automating prompting-based intervention for activity completion is to emulate the timing of caregiver prompts while an individual performed everyday activities. To automate prompt timing, we are designing machine learning algorithms that use the experimenter prompt delivery situations as ground truth for determining whether a current sensor-based activity step is a prompt or a no-prompt step.

To determine the ability of a machine learning algorithm to generate appropriate activity-aware prompts, we performed a study in our smart home testbed with healthy older adults and individuals with mild cognitive impairment. While the participants performed daily activities, an ambient sensor network monitored motion, door open/shut status, and specific item use throughout the testbed, and the experimenter issued a prompt remotely via a tablet or a smart phone if the participant faced difficulty in activity completion. The raw sensor data collected from this study are converted into a machine readable structured dataset ("prompting" data, as we would refer to it) by adding temporal and spatial features/attributes to individual activity steps. As there were very few situations when the participants needed prompts, there are far fewer prompt samples than no-prompt, and this makes the prompting data inherently imbalanced. Moreover, the current sensor suite does not allow crisp differentiation of some prompt situations from no-prompt. This causes ambiguity of class label for a significant portion of prompting data. This study was primarily focused on proposing a machine learning model that can predict prompt situations in daily activities. Therefore, in this paper we do not perform any analysis of the causal relationship between cognitive health conditions (such as MCI and Alzheimer's) and the variance of prompt types. However, interested readers can take a look at the study done by Dawadi et al. [14] on the correlation of cognitive health and performing complex daily tasks based on activity monitoring in smart homes.

As the accurate classification of prompt samples, which is in the minority, is critically more important than identification of no-prompt samples, which is in the majority, we propose a solution that addresses both class imbalance and overlap. Our solution ClusBUS, is a clustering-based undersampling technique, that identifies data regions where minority class (prompt) samples are embedded deep inside majority class samples. By removing majority class samples from these regions, ClusBUS preprocesses the data in order to give the prompt class more importance during classification.

II. Problem Definition

Although, both the problems of class imbalance and overlap can exists in data with multiple classes, due to a general consensus in the literature [1], [2], [3] and our current application objective, in this paper we deal with data that has one *minority* and one *majority* class. A widely accepted rule of thumb is: any dataset that contains less than $\sim 10\%$ minority class samples is considered to be imbalanced. Because the goal of supervised learning algorithms is to optimize prediction accuracy for the entire set of available data points, most approaches ignore performance on the individual class labels. As a result, the highest-performing algorithm will label all points from an imbalanced dataset as members of the majority class despite the fact that all of the minority class samples are incorrectly classified. On the contrary, in many real-life problems accurate classification of minority class samples is critically important. For example, in the mammography dataset where the minority class represents cancerous and majority class represents healthy, a biased prediction would result in the prediction of cancerous patients being healthy. Credit card fraud detection and network security threat detection are some of the other examples where this kind of prediction is not desirable.

The class overlap [15] occurs when there are ambiguous regions in the data where the prior probability for both classes is approximately equal. This makes it difficult or impossible to distinguish between the two classes, primarily because it is difficult to make a principled choice of where to place the class boundary in this region since it is expected that the accuracy would be equal to the proportion of the volume assigned to each class. Figure 1 visually illustrates the difference between normal data with a crisp class boundary and data with class overlap. The combination of imbalanced and overlapping classes makes the resultant problem more difficult than solving them independently.

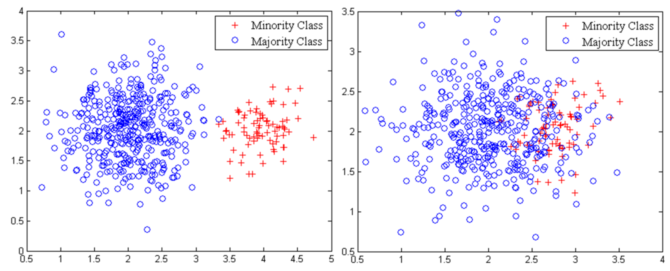

Fig. 1. (left) Data without class overlap, (right) Data with class overlap

Experiments with human participants in our smart home testbed show that there are certain prompt situations where the sensor triggering signature is quite similar to the situations when the participant would probably need no prompt. This kind of situation is prevalent in daily activities that involve object interactions. As our current infrastructure does not have dedicated sensors to track object usage, it is difficult to gauge from the raw sensor data if the participant actually committed an error in such activities. Thus, the absence of sufficient data attributes to differentiate between prompt and no-prompt classes causes class overlap. Our claim is verified by plotting the prompting data in a 3-dimensional space using Principal Component Analysis for dimensionality reduction, which is a well-known technique of data visualization. Figure 2 shows that the minority class samples are highly embedded in the majority class, thus making them inseparable if fed as is to the classifiers. Moreover, due to infrastructural limitations, it is not possible to add new differentiating features to the current dataset, leaving us with the only option of addressing the problem algorithmically.

As a solution to the overlap problem, the literature primarily talks about preprocessing the data before using it to learn a classification model. Preprocessing data to address class overlap is usually a two step process shown in Figure 3. The first and most crucial step is to identify regions of overlap in the data space. This is followed by handling samples in the overlapping region by using methods that mainly fall into three categories [16]: separating, merging and discarding. We discuss these methods in the following section. The approach taken in this paper also uses preprocessing by performing a

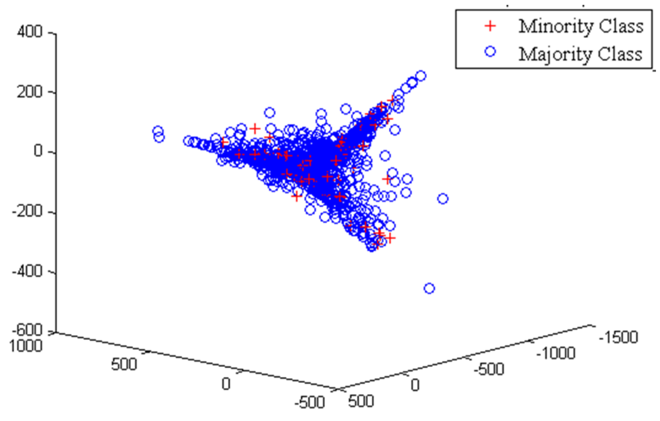

Fig. 2. 3D PCA plot for prompting data

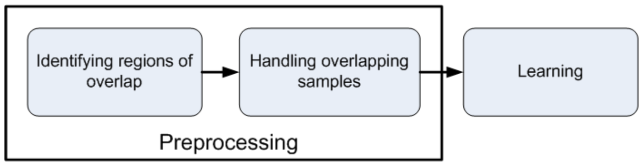

Fig. 3. Steps taken to address class overlap

clustering-based undersampling of the overlapping region to achieve a better learning model.

III. Related Work

While there has not been a significant effort to address class overlap in the presence of imbalanced classes, these two problems have been studied substantially in isolation.

Learning from imbalanced class datasets is a niche, yet critical area in supervised machine learning. Therefore, a wide spectrum of related techniques have been proposed [17], [18] for improved classification of minority class samples. The most common approach is to resample or modify the dataset in a way that balances the class distribution. Naive resampling methods include oversampling the minority class by duplicating existing data points and undersampling the majority class by removing chosen data points. However, random over-sampling and under-sampling increases the possibility of overfitting and discarding useful information from the data, respectively. Therefore, techniques such as SMOTE [19] and Borderline-SMOTE [20] employ intelligent ways of oversampling to synthetically generate new minority class samples in Euclidean space.

Similarly, the problem of overlapping classes or ambiguous data has been studied for over a decade, particularly in the areas of character recognition and document analysis [6], text classification, credit card fraud detection, and drug design [7]. A major hindrance to deal with overlapping class data is the identification of ambiguous or overlapping regions. Tang et al. [21] proposed a k-Nearest Neighbor based approach to extract ambiguous regions in the data. Visa et al. [22] performed a fuzzy set representation of the concept and incorporated overlap information in the fuzzy classifiers. In addition, Xiong et al. [16] used the one-class classification algorithm Support Vector Data Description (SVDD) to capture the overlapping regions in real-life datasets.

Handling samples of overlapping regions is as important as identifying such regions. Xiong et al. [16] proposed that the overlapping regions can be handled with three different schemes: discarding, merging and separating. The discarding scheme ignores the overlapping region and learns on rest of the data that belongs to the non-overlapping region. For example, SMOTE + Tomek Links [23] is one such method. The merging scheme considers the overlapping region as a new class and uses a 2-tier classification model. The upper tier classifier focuses on the entire dataset with an additional class representing the overlapping region. If a test sample is classified as belonging to the overlapping region, the lower tier classifier which works only on the overlapping region is used. For instance, Trappenberg et al. [24] proposed a scheme that refers to the overlapping region class as IDK (I don't know) and predicts the class label of test data only when it is first classified as IDK. The authors argue that by losing prediction accuracy on IDK, a drastic increase in confidence can be gained on the classification of the remaining data. In the separating scheme, the data from overlapping and non-overlapping regions are treated separately to build the learning models. Tang et al. [21] proposes a multi-model classifier named Dual Rough Support Vector Machine (DR-SVM) which combines SVM and kNN under rough set technique. kNN is used to extract boundary patterns or overlapping regions. Two different SVMs are then trained for the overlapping and non-overlapping regions.

There is systematic investigative work on handling class overlap in the presence of imbalanced classes. Prati et al. [25], [26] illustrated the cause of imbalanced class distribution posing a problem in the presence of high degree of class overlap. They showed that overlap aggravates the problem of imbalance and is sufficient to degrade the performance of the classifier on its own. Garcia et al. [27] analyzed the effects of the combined problems on instance-based classification scenario. On the other hand, a category of data cleansing methods tackle the problem by cleaning up unwanted overlapping between classes by removing pairs of minimally distanced nearest neighbors of opposite classes, popularly known as Tomek links [28]. SMOTE+ENN and SMOTE+Tomek [29] utilize the capability of Tomek links to clean the data. However, the cleansing techniques are not desirable for datasets that have inherent class overlap or absolutely rare minority class samples that can cause loss of highly informative data.

In this paper, we take a preprocessing approach similar to the discarding scheme to deal with the overlapping classes. Instead of designating the boundary samples as noise, our approach considers them as crucial for decision making in the overlapping region. The minority class points in the overlapping region are retained and the majority class points are discarded to make a clear distinction.

IV. Smart Home Infrastructure

Our smart home infrastructure is used to replicate near ideal day to day lives of individuals in their homes. The testbed is a two-story apartment equipped with sensors that monitor motion, door open/shut status, and usage of water, burner, and specific items throughout the apartment. Figure 4 gives a high-level understanding of our current infrastructure.

We performed a study in our smart home with 128 volunteer participants, aged 50+, who are healthy older adults or individuals with mild cognitive impairment. Clinically-trained psychologists watch over a web camera as the participants perform 8 different activities. On observing a participant facing difficulty initiating or completing an activity, the psychology experimenters remotely issue an audio/video prompt containing activity specific instructions via a tablet location nearest to the participant or a smart phone. The prompt timings are logged into the database along with the sensor events, denoted by date, time, sensor identifier, and sensor status, that are collected continuously. A human annotator annotates the sensor events with corresponding activity and a sub-step that act as the ground truth. Table I shows a snippet of annotated sensor events. On the basis of ground truth information, the feature vector for every activity sub-step present in the

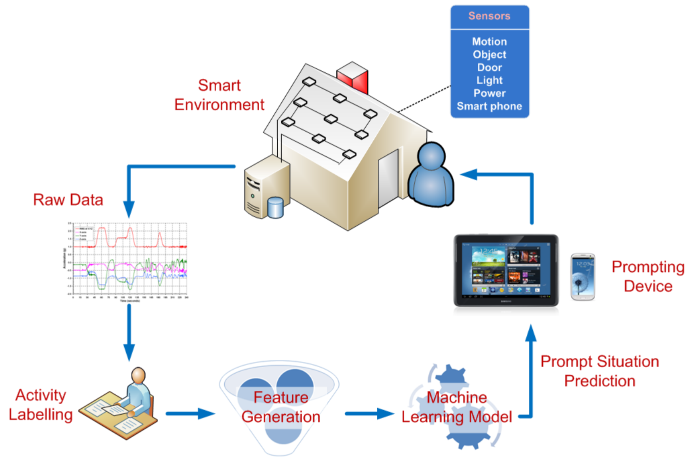

Fig. 4. Smart Home System Architecture

database is generated that consists of temporal, spatial and contextual features. We can view automated prompting as a supervised learning problem in which each activity step is mapped to a "prompt" or "no-prompt" class label. Thus, automated prompting emulates natural interventions provided by a caregiver.

TABLE I. HUMAN ANNOTATED SENSOR EVENTS

Date	Time	SensorID	Status	Annotation	Prompt
2009-05-11	14:59:54	D010	CLOSE	cook-3	*prompt*
2009-05-11	14:59:55	M017	ON	cook-4	*no-prompt*
2009-05-11	15:00:02	M017	OFF	none	*no-prompt*
2009-05-11	15:00:17	M017	ON	cook-8	*no-prompt*
2009-05-11	15:00:34	M018	ON	cook-8	*no-prompt*
2009-05-11	15:00:35	M051	ON	cook-8	*no-prompt*
2009-05-11	15:00:43	M016	ON	cook-8	*no-prompt*
2009-05-11	15:00:43	M015	ON	cook-9	*no-prompt*
2009-05-11	15:00:45	M014	ON	cook-9	*no-prompt*

V. DATA REPRESENTATION

A very crucial step in making sense of raw sensor data is in understanding its hidden patterns. In order to do so, appropriate attributes (or features) of the raw data should be exposed to the machine learning algorithms to make accurate and desired classification. We use annotated sensor data to generate relevant and distinguishable temporal and spatial features of the activity steps. Thus, each step of an activity is treated as a separate training sample and pertinent features are defined to describe the step. The annotations are also used to describe if an activity step received a prompt by tagging the data sample corresponding to the step with a "prompt" or "no-prompt" class label. The features used in the current study are summarized in Table II. Machine learning classifiers are applied on the refined dataset to predict if a sample, which is an abstraction of an activity step, belongs to the "prompt" or "no-prompt" class.

The sensor data[1] collected from 128 participants of our study is used to train classification models. Out of 53 pre-defined activity steps, 38 are recognizable from the raw data by the human annotators. The rest of the activity steps are unrecognizable due to their close association with specific object interactions that could not be tracked by the current sensor infrastructure. In our study, participants were issued

[1]Available at: http://ailab.wsu.edu/casas/datasets/prompting.zip

TABLE II. GENERATED FEATURES

Feature	Description
stepLength	Length of activity step in seconds
numSensors	Number of unique sensors involves with the step
numEvents	Number of sensor events associated with the step
prevStep	Previous step ID
nextStep	Next step ID
timeActBegin	Time (seconds) elapsed since the beginning of the activity
timePrevAct	Time (seconds) difference between the last event of the previous step and first event of the current step
stepsActBegin	Number of steps visited since the beginning of the activity
activityID	Activity ID
stepID	Current step ID
location	A set of features representing sensor frequencies in various regions (such as kitchen, dining room, living room, etc.) of a smart home when the current activity step was performed
Class	Binary class attribute representing prompt and no-prompt

prompts in 149 different cases which involved any of the 38 recognizable activity steps. Therefore, approximately 3.74% of the prompting data belongs to the prompt class and the rest to the no-prompt class. Out of the data samples belonging to the prompt class, approximately 60% belong to the overlapping region. The primary contribution of this work is the development of an intelligent data preprocessing technique that handles imbalanced prompting data containing class overlap as well.

VI. CLUSBUS: CLUSTERING-BASED UNDER-SAMPLING

Class imbalance and overlap are strongly coupled with each other when it comes to learning from class imbalanced data. Denil et al. proved [30] this by performing hypothesis tests. They also showed that if overlap and imbalance levels are too high, good performance cannot be achieved regardless of the amount of available training data. We hypothesize that by addressing the overlap problem, we would be able to get rid of the detrimental effects of class imbalance to some extent. Therefore, we employ a **Clus**tering-**B**ased **U**nder-**S**ampling (ClusBUS) technique to get rid of the overlapping classes.

The idea of devising this technique is derived from the use of Tomek links [28] combined with other sampling methods such as Condensed Nearest Neighbor [31] and SMOTE [29]. Given two data samples E_i and E_j belonging to different classes, and $d(E_i, E_j)$ being the distance between E_i and E_j, a (E_i, E_j) pair is called a Tomek link if there is no sample E_k such that $d(E_i, E_k) < d(E_i, E_j)$. Quite naturally, if two samples form a Tomek link, then either one of these samples is noise or both samples are on or near the class boundary. Tomek links are used both as a data cleansing method by removing samples of both classes, and as an under-sampling method by eliminating only majority class samples from the links. For instance, One-sided selection (OSS) [32] is an under-sampling method that applies Tomek links followed by the application of Condensed Nearest Neighbor (CNN). In this method, Tomek links are used to remove noisy and borderline majority class samples. As a small amount of noise can make the borderline samples fall on the wrong side of the decision boundary, borderline samples are considered unsafe. CNN is used to remove samples from the majority class that are far away from the decision boundary. The rest of the majority and minority class samples are used for the purpose of learning. As opposed to using Tomek links in OSS to find minimally-distant nearest neighbor pairs of opposite class samples and

Algorithm 1: ClusBUS

```
1: Let $S$ be the original training set.
2: Form clusters on $S$ denoted by $C_i$ such that $1 < i < |C|$.
3: Find the degree of minority class dominance for all $C_i$ by:
   $r_i = \frac{\text{\# minority class samples in } C_i}{|C_i|}$
4: For clusters that satisfy $0 < r_i < 1$ and $r \geq \tau$ (where, $\tau = f(r)$
   is an empirically determined threshold for $r$ and is uniform over
   all the clusters), remove all the majority class samples and retain
   the minority class samples.
```

Fig. 5. ClusBUS Algorithm

then removing majority class samples, we find clusters which have a *good* mix of minority and majority class samples. A good mix is determined by the degree of minority class dominance in each cluster. The majority class samples from these clusters are then removed.

First, the entire training data is clustered ignoring the class attribute using Euclidean distance as the distance measure. The degree of minority class dominance of each cluster, denoted by r, is calculated as the ratio of number of minority class samples to the size of the cluster. Therefore, $r = 0$ indicates that all the samples of the cluster belong to the majority class, and $r = 1$ indicate that all the samples belong to the minority class. For clusters with $0 \leq r \leq 1$, the majority class samples are removed if r is equal to or greater that an empirically determined threshold τ. Clearly, if the τ is low, more majority class examples would be removed as compared to when τ is high. This method creates a "vacuum" around the minority class samples in each cluster and thus helps the machine learning classifiers learn the decision boundary more efficiently. The ClusBUS algorithm is summarized in Figure 5.

Theoretically, there is no restriction on the choice of clustering algorithm that should be used to identify clusters containing samples from both classes. However, we avoid any form of partitioning-based clustering method in our experiments for a couple of reasons. Firstly, partitioning-based clustering requires user intervention to specify the number of clusters that need to be formed from the data. Secondly, these methods form spherical clusters only, which might not be the ideal cluster shape in many datasets. Therefore, in this study, we use a density-based clustering algorithm, namely, Density-Based Spatial Clustering of Applications with Noise or DBSCAN [33], [34]. DBSCAN forms arbitrary shapes of clusters in the data that are not necessarily spherical (Gaussian). Moreover, as we do not make any assumption on the distribution of the data, the notion of density on which DBSCAN is based is more meaningful than specifying the number of clusters and forcing the data to be partitioned accordingly. In the following, we provide a brief background description of DBSCAN.

DBSCAN is a density based clustering technique that treats clusters as dense regions of objects in the data space that are separated by regions of low density, mostly representing noise. An object that is not contained in any cluster is considered as noise. In other words, DBSCAN defines a cluster as a maximal set of density-connected points. The neighborhood of an object or data point is defined by a parameter ϵ. If the ϵ-neighborhood of a data point contains at least a minimum number of other points denoted by **MinPts**, then the point is called a core point, and the ϵ-neighboring points are directly density-reachable from the core point. A point p is density-reachable from point q with respect to ϵ and **MinPts**, if there is a chain of objects $p_1, \ldots, p_n$, where $p_1 = q$ and $p_n = p$ such that $p_i + 1$ is directly density-reachable from p_i with respect to ϵ and **MinPts**. In a similar way, a point p is density-connected to q if there is a point o in the same data space such that both p and q are density-reachable from o with respect to ϵ and **MinPts**. The algorithm is summarized in Figure 6.

Algorithm 2: DBSCAN

```
1: Search for clusters by checking $\varepsilon$-neighborhood of each point.
2: If $\varepsilon$-neighborhood of a point $p$ contains more than MinPts, a new
   cluster with p as core point is created.
3: Iterate collection of directly density-reachable points from the
   core points.
4: Terminate when no new point can be added to any cluster.
```

Fig. 6. DBSCAN Algorithm

The degree of minority class dominance is a very crucial factor for the selection of candidate clusters to perform undersampling. An appropriate choice of threshold τ could directly affect the performance of the classifiers. We take an empirical approach (described in the following section) to determine τ by choosing a value between min and median of r.

VII. Experiments and Results

We use four commonly known machine learning classifiers to perform our experiments and thus validate the effectiveness of our proposed data preprocessing approach: C4.5 decision tree (C4.5), k-nearest neighbor (kNN), naive Bayes classifier (NBC) and a Sequential Minimal Optimization version of support vector machines (SVM). We perform a 5-fold cross validation on the data where the training and test data in each fold retains the ratio of class imbalance and overlap, and the training data is preprocessed using ClusBUS.

Unlike most classification scenarios, classification performance for imbalanced and overlapping datasets cannot be evaluated using conventional performance measures such as accuracy and error rate. For instance, a random algorithm that classifies all the samples in the prompting data as *no-prompt* class would yield 96% accuracy ($\sim$3.94% of the data belongs to the *prompt* class), although all predictions for prompt samples are incorrect. Therefore, it is critical to evaluate the classifiers with performance metrics that capture classification performance for both classes. We report the True Positive (TP) rate (prompt class being the positive class) that represents the ratio of activity steps that are correctly classified as requiring a prompt, and the False Positive (FP) rate that represents the ratio of no-prompt steps that are classified as prompt steps. The TP rate is thus capable of measuring the performance of the classifiers separately for the prompt class. An ROC curve plots a classifier's FP rate on the x-axis and the TP rate on the y-axis. Thus it illustrates the classifier's performance without taking into account class distribution or error cost. We report area under the ROC curve (AUC) to evaluate the performance over the costs and distributions. We also report the geometric mean of true positive and true negative rates (G-mean = $\sqrt{TPRate \times TNRate}$) to measure the effectiveness of a classifier on both of the classes together. In the current application, false positives are more acceptable than false negatives. While a prompt that is delivered when it is not needed is a nuisance, that type of mistake is less costly than not delivering a prompt when one is needed, particularly for individuals with dementia. However, considering that the purpose of this research is to assist people by delivering fewer number of prompts, there should be a trade-off between the correctness of predicting a prompt step and the total accuracy of the entire system.

Due to the unavailability of any implementation of approaches that provide a unified solution to address class imbalance and overlap, we compare the performance of ClusBUS with a well-known oversampling technique, known as known as SMOTE [19]. SMOTE oversamples the minority class by creating "synthetic" samples based on spatial location of the samples in the Euclidean space. Oversampling is performed by considering each minority class sample and introducing synthetic examples along the line segments joining any or all of the k-minority class nearest neighbors. The k-nearest neighbors are randomly chosen depending upon the amount of oversampling that is required. Synthetic samples are generated in the following way: First, the difference between the minority class sample under consideration and its nearest neighbor is computed. This difference is multiplied by a random number between 0 and 1, and it is added to the sample under consideration. This results in the selection of a random point in the Euclidean space, along the line segment joining two specific data samples. Consequently, by adding diversity to the minority class, this approach forces the decision boundary between the two regions to be crispier. However, SMOTE does not guarantee that the generated samples would belong to the minority class, especially when the samples from the majority and minority classes overlap. We use SMOTE to produce a 50:50 distribution of minority and majority class samples, which is considered near optimum [35].

As mentioned in the previous section, candidate clusters for undersampling are chosen on the basis of an empirically determined threshold τ on the degree of minority class dominance r. In order to determine τ, we assume real values of r as q-quantile ordered data. Quantiles are points taken at regular intervals from the cumulative distribution function of a random variable which in our case is r. Dividing ordered data into q essentially equal-sized data subsets is the motivation for q-quantiles. We vary τ from the boundary values of first quantiles for q-quantile values of r where $2 < q < 10$. Based on these threshold values, we preprocess the prompting data using ClusBUS and feed it to a C4.5 decision tree classifier. The plots for TP rate, FP rate and AUC obtained by applying C4.5 on prompting data using different values of τ are shown in Figure 7. A uniform trend in the performance measures indicate that there is no further improvement of a classifier after 1st quantile boundary value of a 4-quantile model of ordered r values. Therefore, further experiments with other classifiers are performed with $\tau = 0.25$.

We compare the performance of the four classifiers on the prompting data using the following preprocessing techniques: (a) no preprocessing (represented as *Original* in the plots), (b) SMOTE, and (c) ClusBUS. The performance results are plotted in Figure 8. It is quite clear from the figure that

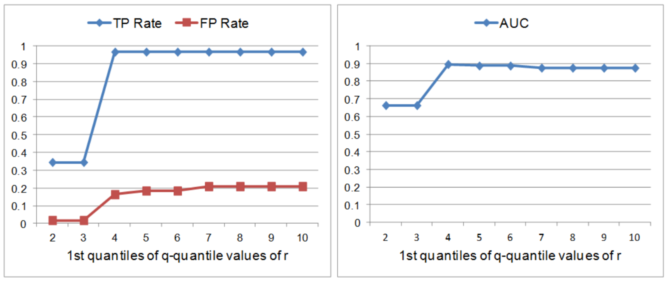

Fig. 7. TP Rate, FP Rate and AUC obtained from C4.5 for different 1st quantile boundary values of q-quantile r values

all four classifiers perform better when preprocessed using ClusBUS than with no preprocessing or when preprocessed by SMOTE. TP rate and G-means show statistically significant improvement over *Original* and SMOTE ($p < 0.05$). An interesting thing to note is that the improvement varies across different classifiers which can be attributed to the underlying principle of the classifiers and how SMOTE and ClusBUS effect those principles. For example, the performance of SMOTE is worse when used with C4.5 and kNN than when used with SVM. This is because SVM uses a kernel function that helps in separating overlapping classes to some extent. The consistent improvement of ClusBUS across all four classifiers is a strong evidence that it can be used as a reliable preprocessing techniques irrespective of the classifier.

ClusBUS results in better AUC over *Original* for all classifiers except NBC. However, there is no notable improvement in AUC when compared with SMOTE. This is because ClusBUS induces minor increase in FP rate. ROC curves are plotted as FP rate versus TP rate. Therefore, a minor increase in FP rate can cause the curve to deviate away from the top-left corner which is the ideal target for any classifier. Although this can be considered as the only limitation of ClusBUS from a theoretical perspective, it is found in the literature that most sampling techniques often cause increase in FP rate. However, the minor increase in FP rate does not demean the advantages and improvement of ClusBUS on other methods. This is because ClusBUS does not entail any costly data synthesis technique like SMOTE, but performs undersampling of majority class training samples that belong to the overlapping region. Moreover, from our application perspective achieving a low FP rate does not justify classifying prompt situations as no-prompt.

VIII. Conclusion

This paper proposes ClusBUS, a preprocessing technique to deal with class overlap problem in the presence of imbalanced classes in data. The proposed method helps us in better identification of potential prompt situations in daily human activities performed in smart homes. The effectiveness of ClusBUS is established by an improved performance over a widely known technique to handle class imbalance, SMOTE. We plan to further test our approach on datasets from other problem domains such as network intrusion detection, credit card fraud detection, etc. We will also compare our approach with other unified solutions that address class imbalance and overlap such as SMOTE+ENN and SMOTE+Tomek.

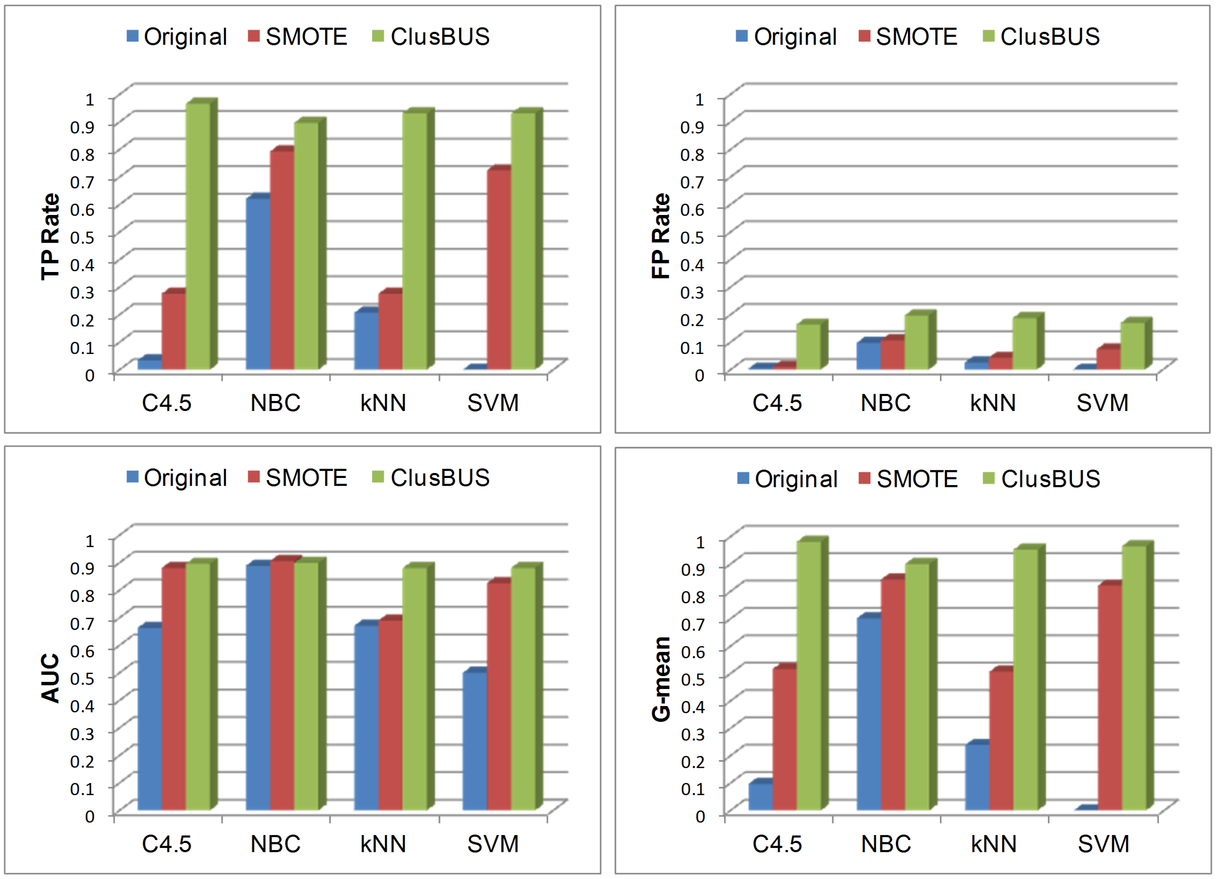

Fig. 8. Comparison of performance measures: (top-left) TP Rate, (top-right) FP Rate, (bottom-left) AUC, and (bottom-right) G-means

References

[1] K. Woods, C. Doss, K. Bowyer, J. Solka, C. Priebe, and K. W., "Comparative evaluation of pattern recognition techniques for detection of microcalcifications in mammography," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 7, no. 6, pp. 1417–1436, 1993.

[2] M. Kubat, R. Holte, and S. Matwin, "Machine learning for the detection of oil spills in satellite radar images," *Machine learning*, vol. 30, no. 2, pp. 195–215, 1998.

[3] C. Phua, D. Alahakoon, and V. Lee, "Minority report in fraud detection: classification of skewed data," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 50–59, 2004.

[4] P. Turney *et al.*, "Learning algorithms for keyphrase extraction," *Information Retrieval*, vol. 2, no. 4, pp. 303–336, 2000.

[5] D. Lewis and W. Gale, "A sequential algorithm for training text classifiers," in *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Springer-Verlag New York, Inc., 1994, pp. 3–12.

[6] C. Liu, "Partial discriminative training for classification of overlapping classes in document analysis," *International Journal on Document Analysis and Recognition*, vol. 11, no. 2, pp. 53–65, 2008.

[7] S. Andrews, *Learning from ambiguous examples*, 2007, vol. 68, no. 07.

[8] G. K. Vincent and V. A. Velkoff, *The next four decades: The older population in the United States: 2010 to 2050*. US Department of Commerce, Economics and Statistics Administration, US Census Bureau, 2010, no. 1138.

[9] A. Association, "2013 alzheimer's disease facts and figures." *Alzheimer's & Dementia*, vol. 9, no. 2, 2013.

[10] D. Cook, M. Schmitter-Edgecombe *et al.*, "Assessing the quality of activities in a smart environment," *Methods of Information in Medicine*, vol. 48, no. 5, p. 480, 2010.

[11] F. Doctor, H. Hagras, and V. Callaghan, "A fuzzy embedded agent-based approach for realizing ambient intelligence in intelligent inhabited environments," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 35, no. 1, pp. 55–65, 2005.

[12] M. Pollack, L. Brown, D. Colbry, C. McCarthy, C. Orosz, B. Peintner, S. Ramakrishnan, and I. Tsamardinos, "Autominder: An intelligent cognitive orthotic system for people with memory impairment," *Robotics and Autonomous Systems*, vol. 44, no. 3-4, pp. 273–282, 2003.

[13] H.-H. Hsu, C.-N. Lee, and Y.-F. Chen, "An rfid-based reminder system for smart home," in *Advanced Information Networking and Applications (AINA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 264–269.

[14] P. N. Dawadi, D. J. Cook, and M. Schmitter-Edgecombe, "Automated cognitive health assessment using smart home monitoring of complex tasks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2013.

[15] M. Denil, "The effects of overlap and imbalance on svm classification," Ph.D. dissertation, Dalhousie University, 2010.

[16] H. Xiong, J. Wu, and L. Liu, "Classification with class overlapping: A systematic study."

[17] H. He and E. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[18] N. Chawla, "Data mining for imbalanced datasets: An overview," *Data Mining and Knowledge Discovery Handbook*, pp. 875–886, 2010.

[19] N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, June 2002.

[20] H. Han, W. Wang, and B. Mao, "Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning," *Advances in Intelligent Computing*, pp. 878–887, 2005.

[21] Y. Tang and J. Gao, "Improved classification for problem involving overlapping patterns," *IEICE Transactions on Information and Systems*, vol. 90, no. 11, pp. 1787–1795, 2007.

[22] S. Visa and A. Ralescu, "Learning imbalanced and overlapping classes using fuzzy sets," in *Proceedings of the ICML*, vol. 3, 2003.

[23] G. Batista, A. Bazan, and M. Monard, "Balancing training data for automated annotation of keywords: a case study," in *Proceedings of the Second Brazilian Workshop on Bioinformatics*, 2003, pp. 35–43.

[24] T. Trappenberg and A. Back, "A classification scheme for applications with ambiguous data," in *Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on*, vol. 6. IEEE, 2000, pp. 296–301.

[25] R. Prati, G. Batista, and M. Monard, "Class imbalances versus class overlapping: an analysis of a learning system behavior," *MICAI 2004: Advances in Artificial Intelligence*, pp. 312–321, 2004.

[26] G. Batista, R. Prati, and M. Monard, "Balancing strategies and class overlapping," *Advances in Intelligent Data Analysis VI*, pp. 741–741, 2005.

[27] V. García, R. Alejo, J. Sánchez, J. Sotoca, and R. Mollineda, "Combined effects of class imbalance and class overlap on instance-based classification," *Intelligent Data Engineering and Automated Learning–IDEAL 2006*, pp. 371–378, 2006.

[28] I. Tomek, "Two modifications of CNN," *IEEE Transaction on Systems, Man and Cybernetics*, vol. 6, pp. 769–772, 1976.

[29] G. Batista, R. Prati, and M. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 20–29, 2004.

[30] M. Denil and T. Trappenberg, "Overlap versus imbalance," *Advances in Artificial Intelligence*, pp. 220–231, 2010.

[31] P. Hart, "The condensed nearest neighbor rule," *IEEE Transaction on Information Theory*, vol. 3, pp. 515–516, 1968.

[32] M. Kubat and S. Matwin, "Addressing the curse of imbalanced training sets: one-sided selection," in *Machine Learning-International Workshop then Conference*. Morgan Kaufmann Publishers, Inc., 1997, pp. 179–186.

[33] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the 2nd International Conference on Knowledge Discovery and Data mining*, vol. 1996. AAAI Press, 1996, pp. 226–231.

[34] J. Han and M. Kamber, *Data mining: concepts and techniques*. Morgan Kaufmann, 2006.

[35] G. Weiss and F. Provost, "Learning when training data are costly: The effect of class distribution on tree induction," *Journal of Artificial Intelligence Research (JAIR)*, vol. 19, pp. 315–354, 2003.