Boosting Mobile Apps under Imbalanced Sensing Data

Xinglin Zhang, *Student Member, IEEE*, Zheng Yang, *Member, IEEE*,
Longfei Shangguan, *Student Member, IEEE*,
Yunhao Liu, *Senior Member, IEEE*, and Lei Chen, *Member, IEEE*

Abstract—Mobile sensing apps have proliferated rapidly over the recent years. Most of them rely on inference components heavily for detecting interesting activities or contexts. Existing work implements inference components using traditional models designed for balanced data sets, where the sizes of interesting (positive) and non-interesting (negative) data are comparable. Practically, however, the positive and negative sensing data are highly imbalanced. For example, a single daily activity such as bicycling or driving usually occupies a small portion of time, resulting in rare positive instances. Under this circumstance, the trained models based on imbalanced data tend to mislabel positive ones as negative. In this paper, we propose a new inference framework SLIM based on several machine learning techniques in order to accommodate the imbalanced nature of sensing data. Especially, guided under-sampling is employed to obtain balanced labelled subsets, followed by a similarity-based sampling that draws massive unlabelled data to enhance training. To the best of our knowledge, SLIM is the first model that considers data imbalance in mobile sensing. We prototype two sensing apps and the experimental results show that SLIM achieves higher recall (activity recognition rate) while maintaining the precision compared with five classical models. In terms of the overall recall and precision, SLIM is around 12 percent better than the compared solutions on average.

Index Terms—Mobile sensing applications, imbalanced sensing data, machine learning, under-sampling, semi-supervised learning

1 Introduction

Smartphones are proliferating rapidly in recent years and have become ubiquitous mobile sensing units as they are equipped with multiple built-in sensors. Various mobile sensing apps have been investigated and developed based on sensor readings that provide new dimensions to interpret and interact with the living world [1], [2], [3]. The apps range from individual usage to city-scale coverage, by revealing smartphone users' activities, contexts, surrounded environment and social events they involved in.

Inference components (or classifiers), which are responsible for extracting features from the raw sensor data and making inferences accordingly, are the key for many mobile sensing apps [4], [5], [6], [7]. In terms of inference models, there are two classes of sensing data, i.e., positive class and negative class. The sensing data belonging to the positive class represent what the app users are interested in. Thus the aim of a classifier is to correctly identify positive data instances from negative ones. Despite the numerous models for making inferences, most of them include two phases: training and operating. In the training phase, app developers collect labelled training sensing data from specific sensors, and choose a training method to generate an inference model. In the operating phase, the trained model is deployed in user smartphones and starts to make inferences on new sensor data.

A growing number of inference-based apps typically collect a sensing data set and directly use the entire data set to train a classifier, normally with the accuracy and error rate as the performance metrics [6], [8]. However, most of them neglect the imbalanced nature of the data classes. For example, bicycling [6] usually occupies a small portion of time for most people, which results in a small positive data set and a large negative data set from other daily life activities for training a bicycling detection component. A lot more user activities resemble bicycling (such as driving [5] and taking a bus [9]) with respect to the imbalanced distribution of the collected sensing data sets. In summary, with smartphones accompanying users around the clock, the amount of generated sensing data is tremendous, yet the interesting events are rare.

The imbalanced nature of the training sensing data exposes the deficiency of traditional classifiers, which are typically designed for balanced training data. For example, if there is a data set with 10 percent amounts of data belonging to the positive class, and 90 percent negative, then a naive classifier tends to simply put any new observed sample in the negative class, achieving a high classification accuracy of 90 percent. However, the result is obviously unacceptable as the classifier fails to detect the interesting positive samples [10].

On the other hand, training a classifier requires a sufficient number of labelled sensing data instances in order to

- *X. Zhang, Z. Yang and Y. Liu are with the School of Software and Tsinghua National Lab for Information Science and Technology (TNLIST), Tsinghua University, China. E-mail: zhxlinse@gmail.com, {yang, yunhao}@greenorbs.com.*
- *L. Shangguan and L. Chen are with the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong.*

Manuscript received 28 Nov. 2013; revised 18 July 2014; accepted 28 July 2014. Date of publication 30 July 2014; date of current version 1 May 2015.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TMC.2014.2345053

TABLE 1
Daily Activity Statistics

Activities	Eat	Work	Household	Sleep	Leisure
Time	5%	32%	7%	35%	21%

obtain high accuracy and generalization ability. Collecting labelled sensing data is a tedious, expensive, and time-consuming task for app developers [11], [12], [13]. A large amount of inexpensive unlabelled sensing data holds the potential to alleviate the pain of explicit data labelling [14]. The same dilemma occurs when the user needs to personalize a mobile app. The personalization process typically requires each user to manually label instances, which is even more tedious than the labelling process of the app developer. Hence, mining the unlabelled sensing data on the user's smartphone shows a promising direction to alleviate the user's labelling effort. However, those sensing data are also imbalanced and thus may not directly boost inference components as expected.

In this work, we accommodate the above challenges by designing a new inference framework SLIM, Sampling-based semi-supervised learning with imbalanced multi-modal sensing data. SLIM adopts an under-sampling scheme to obtain multiple balanced labelled sensing data subsets. However, using under-sampling further shrinks the size of the training subset, which may lead to inaccurate classification result. Considering this, we may resort to the massive unlabelled sensing data. Yet the unlabelled sensing data are also imbalanced and cannot be directly applied. We hence novelly design a sampling scheme to the unlabelled data according to similarity measurements based on each balanced labelled subset. The outcome is a pseudo-balanced training data subset with a sufficiently large size. Local classifiers can be trained on each balanced sensing data subset, including labelled and unlabelled instances. Finally, a global inference model is achieved by synthesizing the prediction results of all local classifiers.

Furthermore, mobile apps usually adopt multiple types of sensors in order to make reliable inferences. Different sensors may have different confidence levels in making decisions. Given this, SLIM is designed to fully utilize multiple sensors by making inferences on each sensor dimension, followed by a heuristic confidence-based combination of multiple sensor dimensions. As a result, SLIM is highly flexible for any combinations of the built-in sensors.

To the best of our knowledge, SLIM is the first framework that extends under-sampling to semi-supervised learning with a novel similarity-based sampling scheme. Under-sampling performs well considering imbalanced labelled data. Yet it requires a large amount of labelled data, which are expensive to obtain in sensing apps. On the other hand, semi-supervised learning is good at combining a few labelled data and large amounts of cheap unlabelled data. Using these two methods separately will not suffice in sensing apps. Hence we novelly bridge these two methods by designing a similarity-based sampling scheme, such that SLIM can enjoy the power of both methods.

The main contributions of this work are summarized in the following:

- The proposed learning framework SLIM is able to make full use of both imbalanced labelled and unlabelled sensing data by novelly combining under-sampling and semi-supervised learning techniques, resulting in a better inference model for mobile sensing apps.
- SLIM is highly flexible considering the following aspects: it accepts any combination of smartphone built-in sensors for inference; and it adopts diverse traditional learning methods as basic learners.
- Two prototype mobile apps, bicycling detection and backing material detection, are proposed for investigating the effectiveness of SLIM. Bicycling detection aims at examining whether a smartphone user is riding a bike, while backing material detection is able to recognize on what material the smartphone is placed. Through the real experiments, the superiority of SLIM over five representative inference models is verified. SLIM obtains average F-Measures (a widely accepted evaluation metric for imbalanced data classification as explained in Section 4.4) of 79.6 percent in bicycling detection and 76.7 percent in backing material detection, both of which are around 12 percent better than the competitors.

The rest of this paper is organized as follows. Section 2 demonstrates the characteristics of mobile sensing data and verifies the influence of data disproportion by an experiment. Section 3 presents the problem formulation and the design of SLIM in detail. Evaluation results of two prototype apps are illustrated and discussed in Section 4. Finally, related works are discussed in Section 5 and conclusions are drawn in Section 6.

2 Motivation

In this section, the ubiquity of imbalanced data distribution of sensing apps is demonstrated. The imbalanced nature uncovers the necessity of developing more suitable inference components than existing traditional ones. Moreover, the large potential unlabelled set may alleviate the pain of acquiring a large labelled set. Yet directly incorporating them may not help much. As an illustration, a quick experiment is conducted to show the above issues.

Observation 1. Mobile sensing apps usually encounter imbalanced sensing data due to the following reasons. First, the time a user spends in an interesting place or activity only occupies a small portion of time, which means that the corresponding interesting sensing data are rare. Table 1 shows the daily activity statistics of Americans [15]. As can be seen, different categories of activity occupy imbalanced portions of time. Considering that mobile apps are usually designed for even finer categories of activity, the corresponding interesting activities may appear as minor classes. Second, the diversity of user places and activities indicates the imbalanced distribution. Given that each activity or place holds the same amount of sensing data, the data size of a specific place of activity is small compared to the total data size of all places and activities.

TABLE 2
Confusion Matrix

	Predictive Positive Class	Predictive Negative Class
Actual Positive Class	TP (True Positives)	FN (False Negatives)
Actual Negative Class	FP (False Positives)	TN (True Negatives)

Both temporal and spatial dimensions bring the critical issue that is ignored in the literature: the collected labelled data are highly imbalanced for the interesting and non-interesting classes. In this case, using traditional classifiers may result in high classification accuracy, yet it covers the fact that the interesting class may be mis-classified with a high rate.

Observation 2. One intuitive idea to reduce the effort of labelling sensing data is to make use of inexpensive unlabelled sensing data. Then semi-supervised learning techniques can be applied to train app classifiers. However, the unlabelled sensing data are also imbalanced. Therefore, using auxiliary unlabelled data may result in helping a little, or even jeopardizing app classifier training. The massive unlabelled data that actually belong to the negative class will further make negative class overwhelm the interesting positive class. Thus it is not as straightforward as it seems to incorporate unlabelled sensing data for fast and easy development of app classifiers.

Experiment 1. To verify the problems discussed above, we conduct an experiment to examine the performance of two traditional inference models, supervised and semi-supervised learning.

Some mobile apps attempted to record and monitor the exercise level of people [16]. As an example, we would like to design a classifier that can judge whether a smartphone user is riding a bike. Using this information the health monitoring apps can calculate and inform users about their exercise level. We recruited eight graduate students to record the accelerometer and GPS readings of their smartphones when they perform diverse activities for two weeks. The training data have an approximate negative-to-positive class ratio of 10. And we use 10-fold cross validation to train a SVM, both in supervised and semi-supervised paradigms. (More details about the experiments are provided in Section 4.)

The classification accuracies of supervised and semi-supervised methods are 94.8 and 94.5 percent, respectively. The results seem quite encouraging. However, other evaluation metrics are frequently adopted to provide comprehensive assessments by introducing the confusion matrix (Table 2), which records the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). Among these metrics, precision and recall are defined as:

$$Precision = \frac{TP}{TP + FP}, \tag{1}$$

$$Recall = \frac{TP}{TP + FN}. \tag{2}$$

Precision measures exactness (i.e., how many data instances labelled as positive are actually from the positive class), whereas recall measures completeness (i.e., how many data instances belonging to the positive class are actually labelled correctly). The results of traditional supervised and semi-supervised learning methods are shown in Table 3. We can see that the precision of supervised method is high, while the recall is less than 50 percent, which means that less than half of the positive instances are correctly recognized. Considering that the positive class is of the interest and is rare with respect to instance number, the result is far from satisfaction. The recall (or recognition rate) of semi-supervised method is improved compared to the supervised method. However, 61.7 percent is still unsatisfying. We deem that accommodating the imbalanced nature of the sensing data can help to improve the performance of the classifier.

3 SLIM—Boosting App Classifiers

Before delving into the design of SLIM, we first give a few important notations that will be used. For a new app, we use H to represent its inference component, which is in charge of classifying a sensing data instance as positive or negative. The interesting instances are assigned to the positive class $\mathcal{P}$, while the other instances are categorized to the negative class $\mathcal{N}$. We use $\mathcal{L} = \{x_{l,i}, y_i\}(i = 1, \ldots, N)$ to denote the collected labelled training set and $\mathcal{U} = \{x_{u,j}\}(j = 1, \ldots, M)$ to represent the set of unlabelled sensing data, respectively. The objective is to train the classifier H to perform effectively and robustly regarding to the limited number of labelled sensing data and disproportionate distribution of the entire data set. The main notations that will be used in the paper are summarized in Table 4.

In the rest of this section, we first illustrate the framework of SLIM briefly, and then demonstrate the key procedures in detail.

3.1 General Framework of SLIM

The general framework of SLIM is shown in Fig. 1: The negative labelled data and massive shared unlabelled data are under-sampled (Sections 3.2 and 3.3) to generate several subsets. Then on each subset, combining the labelled data, a local classifier will be trained. A final inference model is an integration of all local classifiers (Section 3.4).

Specifically, the work flow of training on each subset is illustrated in Fig. 2. First, the collected negative labelled

TABLE 3
Precision and Recall by Supervised and Semi-Supervised Learning

	Supervised	Semi-supervised
Precision	90.1%	75.0%
Recall	49.0%	61.7%

TABLE 4
Notations

Symbol	Description
$x_{l,i}/x_{u,i}$	the ith labelled/unlabelled sensing data instance
y_i	the label of the ith sensing data instance
$x^k_{l,i}/x^k_{u,i}$	the kth sensor dimension of the ith labelled/unlabelled sensing data instance
N/M	the number of the labelled/unlabelled sensing data instances
K	the number of incorporated sensors
T	the number of sampled subsets
$\mathcal{L}$	the labelled sensing data set
$\mathcal{P}/\mathcal{N}$	the positive/negative labelled class
$\mathcal{U}$	the unlabelled sensing data set
$\mathcal{U}_P/\mathcal{N}_P$	the unlabelled sensing data set with potential positive/negative label
$\mathcal{N}_t/\mathcal{U}_t$	the tth sampled subset with negative/potential negative label
H_t	the classifier trained on the tth sampled data set

sensing data set $\mathcal{N}$ is divided into T subsets by under-sampling, each possessing the same size as the positive labelled sensing data $\mathcal{P}$. Each negative subset $\mathcal{N}_t$, combined with the positive labelled set $\mathcal{P}$ and the large unlabelled sensing data set $\mathcal{U}$, is then used to sample pseudo-balanced unlabelled data set $\mathcal{U}_t$ by performing similarity-based sampling. After acquiring T collection of balanced labelled and unlabelled sensing data set, semi-supervised learning scheme is adopted to train a classifier on each data collection. If multiple sensors are used in the apps, SLIM will automatically evaluate the classification accuracy for each sensor dimension and combine them with relative confidence.

3.2 Under-Sampling

For imbalanced labelled training set, a lot of techniques have been proposed to train a well-performed classifier [10]. In this paper, we resort to the under-sampling method [17], [18]. Sampling is a class of methods that can obtain balanced positive and negative classes through altering the initial disproportionate labelled training set. Specifically, under-sampling tries to sample a data subset from the negative class, such that the sampled set is comparable to the positive set with respect to the sample size. Then the traditional well-performed learning methods can be employed against the balanced data subset.

However, the data subset generated by under-sampling is only a small portion of the huge original data set. Thus the subset may lose potential useful information of the whole data set. To overcome this deficiency, we propose to perform under-sampling for T rounds to the whole data set, with a heuristic redundancy elimination, i.e., after sampling an instance from the negative class, the corresponding instance is removed from the set. After this multi-round under-sampling, we obtain T subsets of the negative class that can be used separately for training T classifiers.

3.3 Similarity-Based Sampling

The imbalanced nature of sensing data exists not only in the collected labelled training set, but also in the unlabelled

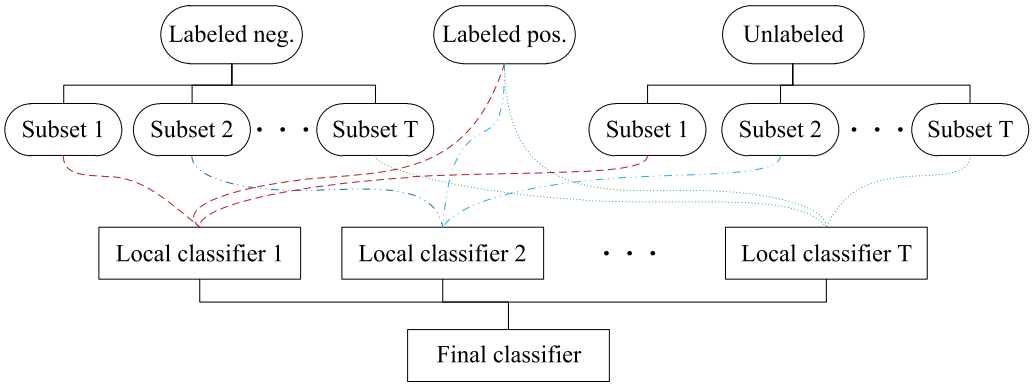


Fig. 1. General framework of SLIM.

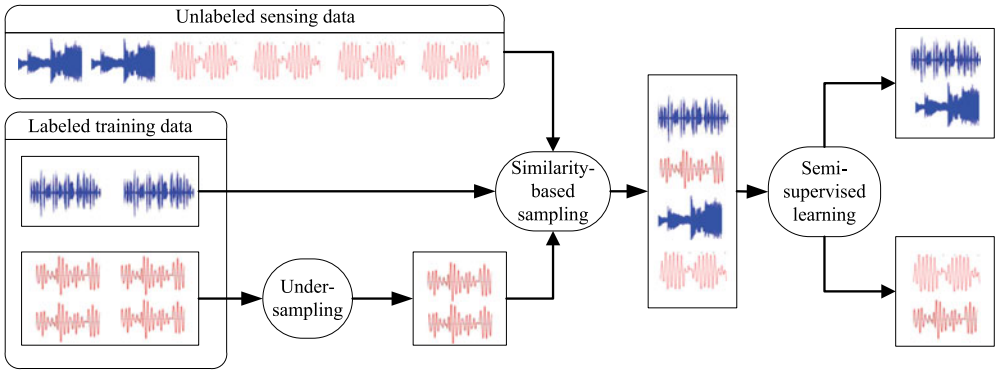


Fig. 2. The work flow of training on each sensing data subset.

sensing data set. In order to employ unlabelled data, the imbalanced nature has to be considered in the first place. Otherwise, training a classifier on the balanced labelled set and the imbalanced unlabelled set using a semi-supervised learning technique will result in an inferior classifier similar to the ones trained on the imbalanced labelled set. As the unlabelled data has no class information, we cannot apply under-sampling directly in this situation to generate a subset that is composed of balanced positive and negative instances. Our intuition is that the sensing data ought to share more similarities among the instances belonging to the same class. Thus we propose to perform unlabelled instance selection in two steps:

- First, for each unlabelled instance $x_{u,j}$, we compute its distance to each labelled training instance $x_{l,i} \in \mathcal{P} \bigcup \mathcal{N}_t$, with respect to a similarity-based distance metric (e.g., Euclidean distance). We assign $x_{u,j}$ a potential label $y_{u,j} = y_i$, such that the distance between $x_{l,i}$ and $x_{u,j}$ is the smallest. After computation, we obtain two labelled sets, positive class $\mathcal{U}_P$ and negative class $\mathcal{U}_N$, with the property that $|\mathcal{U}_P| \ll |\mathcal{U}_N|$.
- Second, perform under-sampling on the negative class $\mathcal{U}_N$ to generate a subset $\mathcal{U}_t$, with the constraint that $|\mathcal{U}_t| = |\mathcal{U}_P|$. Thus we obtain a pseudo-balanced unlabelled data set.

Note that the labels assigned to the unlabelled instances here are not required to be very accurate. Both the correctly and incorrectly labelled instances are helpful for the following semi-supervised learning procedure to train a well-performed classifier. The correctly labelled instances enhance the shared instance properties pertaining to the corresponding class, while the incorrectly labelled instances bring the ambiguity between the classes, thus guiding the training procedure to accommodate this ambiguity. In this sense, the sampled instances may not constitute real balanced positive and negative classes, and hence we term the generated data set pseudo-balanced.

To validate the effectiveness of similarity-based sampling, we investigate the data set collected in Section 2 (Experiment 1), where the ratio of the unlabelled and labelled data is approximately 5. After similarity-based sampling, the size of the positive class increases by 437 percent. The training set is thus effectively enlarged considering both positive and negative classes. On the other hand, the sampled subset has a ground truth positive-to-negative ratio of 0.84. The resultant subset does not badly break the balance of the labelled set once incorporated. Therefore, the enlarged training set, with both labelled and unlabelled data, is nearly balanced and can be fed to classifiers.

3.4 Classifier Ensembles

There are two folds considering classifier ensembles: multiple built-in sensors and multiple balanced training data collections. Smartphones nowadays are equipped with multiple sensors, such as accelerometer, GPS, gyroscope, etc. These sensors can be selected and combined for different mobile apps. Typically, assume that the app developer would like to incorporate K sensors to make inference, then there will be K separate dimensions to describe the characteristics of the app. Usually, some sensor dimensions are more reliable or confident in predicting certain events, thus it is necessary to differentiate the classification ability of multiple sensors. In our framework, we first train classifiers $H_t^{(k)}(x^{(k)})$, $k = 1, \ldots, K$, on each sensor dimension. Then we combine these classifiers with the weights reflecting the confidence of corresponding sensor dimension [19], [20]:

$$H_t(x) = \sum_{k=1}^{K} \alpha_k H_t^{(k)}\big(x^{(k)}\big), \tag{3}$$

where $\alpha_k = \frac{1}{2}\log(\frac{1-\varepsilon_k}{\varepsilon_k})$ represents the prediction confidence of classifier $H_t^{(k)}(x^{(k)})$ (i.e., α_k reflects the confidence of the kth sensing dimension), and ε_k is the error rate of the classifier $H_t^{(k)}(x^{(k)})$.

On the other hand, we have generated T data collections that are used to train classifiers H_t in parallel. Borrowing the idea of majority-voting scheme, where the decision is conformed to the most experts' decisions, we construct a synthetic final classifier:

$$H(x) = sgn\bigg(\sum_{t=1}^{T} H_t(x) - \theta\bigg), \tag{4}$$

where $sgn(h)$ is a sign function:

$$sgn(h) = \begin{cases} 1 & h > 0, \\ -1 & h <= 0, \end{cases} \tag{5}$$

and θ is a threshold parameter that indicates how aggressive the classifier is. If θ is small, the classifier tends to label an instance as positive with fewer supports; otherwise the classifier tends to label an instance as positive with more supports. For example, given that $H_t(x) \in \{-1, 1\}$ and $T = 5$, if $\theta = 0$, labelling a new observed sensing instance x as positive requires three or more classifiers $H_t(x)$ agreeing that x belongs to the positive class. If $\theta = 4$, estimating x as a positive instance requires that all five classifiers $H_t(x)$ label the instance x as positive.

3.5 Procedures of SLIM

Algorithm 1 summarizes the main procedures of SLIM. Lines 4 and 5 sample the subset of negative class with elimination operation. Lines 6 to 13 compute similarities of unlabelled and labelled sensing data and generate pseudo-balanced unlabelled subset for training. The combination of different sensor dimensions is achieved from Lines 14 to 19. Finally, The outer for-loop controls how many local classifiers will be trained for the final ensemble. Two key parameters in SLIM are subset number T and threshold value θ, both of which will be investigated and discussed in the evaluation part (Section 4).

3.6 SVM-Based Implementation

The proposed SLIM is a general framework, which can adopt many different specific classifiers. In this sense, the framework can be widely applied to diverse mobile app inferences. In this paper, we implement the framework by using support vector machine (SVM), which has been widely

adopted in mobile sensing [5], [21], [22]. Suppose that we have a labelled training set $\mathcal{L} = \{x_{l,i}, y_i\}(i = 1, \ldots, N)$, viewing each sensing data feature $x_{l,i} \in R^d$, where R^d represents a d-dimensional space, and $y_i \in \{1, -1\}$, where 1 means positive class and -1 means negative class. The classification task is equivalent to finding a hyperplane that can separate the sensing data point from the positive and negative classes. Define the "margin" of a separating plane as the smallest distance from the plane to the closest postive/negative sensing data point. SVM would try to find the hyperplane with the largest margin. In the presence of massive unlabelled sensing data $\mathcal{U} = \{x_{u,j}\}(j = 1, \ldots, M)$, the goal becomes finding a function $f: R^d \rightarrow \{1, -1\}$ such that the following functional is minimized [23]:

$$\min_f \frac{\|f\|_{\mathcal{H}}^2}{2} + C_1 \sum_{i=1}^{N} l\big(y_{l,i}, f(x_{l,i})\big) + C_2 \sum_{j=1}^{M} l_{sym}\big(f(x_{u,j})\big), \quad (6)$$

where $\mathcal{H}$ is a reproducing kernel Hilbert space (RKHS), $l(y, f(x)) = \max\{0, 1 - yf(x)\}$ and $l_{sym}(f(x)) = \max\{0, 1 - |f(x)|\}$ are hinge loss and symmetric hinge loss, respectively. C_1 and C_2 are regularization parameters that control model complexity. For more details of semi-supervised SVM, we refer interested readers to [23], [24].

Algorithm 1. SLIM

1: Compute features x from the raw data for each sensor, forming labelled training set $\mathcal{L} = \{x_{l,i}, y_i\}_{i=1}^{N}$, and unlabelled training set $\mathcal{U} = \{x_{u,j}\}_{j=1}^{M}$;
2: Select the positive class $\mathcal{P} \subset \mathcal{L}$ and negative class $\mathcal{N} \subset \mathcal{L}$;
3: **for** $t = 1$ to T **do**
4: Draw random subset $\mathcal{N}_t \subset \mathcal{N}$, s.t. $|\mathcal{N}_t| = |\mathcal{P}|$;
5: $\mathcal{N} \leftarrow \mathcal{N}/\mathcal{N}_t$;
6: **for** $j = 1$ to M **do**
7: Compute p, s.t.
8: $x_{l,p} = \arg\max_{x_{l,i} \in \mathcal{P} \bigcup \mathcal{N}_t} Sim(x_{u,j}, x_{l,i})$;
9: Assign $y_{u,j} = y_p$;
10: **end for**
11: Let $\mathcal{U}_P = \{x_{u,j} \,|\, y_{u,j} = 1\}, \mathcal{U}_N = \{x_{u,j} \,|\, y_{u,j} = -1\}$;
12: Sample $\mathcal{U}_t \subset \mathcal{U}_N$, s.t. $|\mathcal{U}_t| = |\mathcal{U}_P|$;
13: $\mathcal{U}_t \leftarrow \mathcal{U}_P \bigcup \mathcal{U}_t$;
14: **for** $k = 1$ to K **do**
15: Train $H_t^{(k)}$ on $\mathcal{P} \bigcup \mathcal{N}_t \bigcup \mathcal{U}_t$ by semi-supervised learning;
16: Compute error rate ε_k for $H_t^{(k)}$;
17: Compute weight for sensor k: $\alpha_k = \frac{1}{2}\log(\frac{1-\varepsilon_k}{\varepsilon_k})$;
18: **end for**
19: $H_t = \sum_{k=1}^{K} \alpha_k H_t^{(k)}$;
20: **end for**
21: Compute final classifier, $H = sgn(\sum_{t=1}^{T} H_t - \theta)$;

4 Evaluation

In this section, the performance of SLIM is evaluated by comparing with two traditional inference models for balanced data and three inference models for imbalanced data. Specifically, as in our implementation SVM is adopted as a basic inference component, we will use the following SVM based models as competitors:

- *SVM*. We implement a traditional supervised SVM on the collected imbalanced labelled sensing data.
- *S3VM*. S3VM is a semi-supervised SVM on the collected imbalanced sensing data, including labelled and unlabelled data.
- *UNDER-SVM*. UNDER-SVM is an undersampling based SVM, which randomly draws a subset of negative data instances and train a SVM on the undersampled data [25]. The percentage of undersampled negative data instances is determined by grid search.
- *SMOTE-SVM*. SMOTE-SVM is an oversampling based SVM, which uses the SMOTE algorithm [26] to generate pseudo positive data instances and train a SVM on the oversampled data [25]. The ratio of oversampled positive data is determined by grid search.
- *WEIGHT-SVM*. WEIGHT-SVM is a cost sensitive learning based SVM, which assigns higher penalties to the false negative instances (FNs) than the false positive instances (FPs) [27], [28]. The weight ratio between FNs and FPs is decided by grid search.

4.1 Prototype Apps

4.1.1 Bicycling Detection

Context-aware mobile apps are proliferating rapidly recently. Recognizing smartphone users' context information, such as indoor/outdoor, at home/in office, driving/walking, can help to provide fine-grained mobile computing services. For example, when a smartphone detects that the user is driving, it can automatically block phone calls in the user's hand for the sake of safety. When a user is in a building, it is better to turn the smartphone's GPS off to save energy [29]. In contrast, when a user goes to the open countryside, WiFi is usually unavailable and should be turned off. In summary, classifying users' context lays a foundation for large amounts of mobile apps.

To verify the effectiveness of SLIM in such apps, we prototype a proof-of-concept bicycling detection app, which is able to inference whether a user is riding. Bicycling detection information is useful for multiple mobile apps. For example, recording the biking sensing data can help to monitor the cyclist experience: distance traveled, calories burned, path incline, etc. In short, knowing that a person is riding a bike can be useful for recording the exercise experience of the individual as well as the surrounding environmental conditions [30]. In this sense, for a normal user with only a smartphone and no other special devices, it would be valuable to infer whether the user is riding.

We use two smartphone built-in sensors, accelerometer and GPS, for collecting raw sensing data. Each accelerometer reading contains an x, y, and z value corresponding to three axes. We generate a total of 13 features for training inference model. The features include: average acceleration for each axis, standard deviation for each axis, average absolute difference between the value of each of the readings within the example duration and the mean value of those readings for each axis, average resultant acceleration, and time between peaks in the sinusoidal waves associated with most activities for each axis. For GPS, we generate

commonly used features, including magnitude, variance, first derivative, and most dominant frequency components of the data histogram.

4.1.2 Backing Material Detection

Different from the human-centric context sensing mentioned above, we also investigate environment sensing concerning the smartphone's perspective. Similar to human-centric contexts, being aware of smartphones' environment is directly beneficial to a broad range of mobile apps. For example, if a smartphone is in a bag or pocket, it is useless to light up the screen when receiving a phone call. If a smartphone is placed on a sofa rather than on a desk, it is better to raise the volume of the phone ring to avoid missing calls. In many situations, the environment information is helpful for fine-tuning the smartphone's behavior.

As a proof of concept, we design a mobile sensing app named backing material detection. Backing material detection aims at distinguishing hard/soft material via smartphone-generated vibration patterns: the mechanical motion and the acoustical features. The vibration patterns can be captured by embedded accelerometer and microphone. For accelerometer, we adopt the same set of features as those proposed in bicycling detection. For microphone, we use zero crossing rate (ZCR), spectral centroid (SC), and Mel-frequency cepstral coefficient (MFCC). we select two representative materials with different stiffness for recognition: mattress (soft) and wooden desk (hard). In each scenario, the phone motor is triggered to vibrate for 7 seconds, and the corresponding acceleration readings and sounds are recorded.

4.2 Experiment Device

To give a comprehensive evaluation, we implement SLIM on three different types of smartphones (Samsung Galaxy S2 I9100, Samsung Nexus3 I9250, Motorola MT788). All types of phones are equipped with the necessary sensors, including microphone, accelerometer and GPS. The Samsung S2 Galaxy I9100 has a 1 GB RAM and dual-core 1.2 GHz processor; the Samsung Nexus3 I9250 is equipped with 1 GB RAM and dual-core 1.5 GHz processor; the Motorola MT788 has 1 GM RAM and single-core 2.0 GHz processor.

4.3 Experiment Setting

We recruit eight volunteers (four males and four females) to collect raw sensing data for both apps. The volunteers are required to record the ground truth activities during the day with memo widget [31]. For bicycling detection, the volunteers record data from multiple daily activities on campus for two weeks. For backing material detection, the volunteers are asked to place their smartphones on as many material as possible for one week, including desk, mattress, sofa, chair, bag, and so on. We have collected around 7,000 data instances for bicycling detection and 2,800 data instances for backing material detection. The collected data set is divided into labelled set and unlabelled set (with ratio around 1/5), and is used for training with 10-fold cross validation.

The compared algorithms are implemented in Matlab based on the famous SVM tool LIBSVM [32]. For all competitors, we use the same linear kernel function so that their performances are comparable. For the training/testing process, we have normalized the input data (sensing data features) so that each input feature has 0 mean and 1 standard deviation. All algorithms are executed on the normalized data and the model specific parameters are optimized by grid search.

4.4 Evaluation Metrics

We have already introduced two evaluation metrics in section 2, i.e., precision and recall. Precision and recall are suitable for imbalanced data learning as they are not both sensitive to changes in data distributions. As we can see, precision [Eq. (1)] is sensitive to data distributions, while recall [Eq. (2)] is not. Precision cannot assert how many positive instances are labelled correctly, while recall cannot provide information about how many instances are incorrectly labelled as positive. It is more appropriate to combine both metrics for comprehensively understanding classification performance in imbalanced data scenarios. As a representative, F-Measure (or F-Score) metric [33], [34], as defined in Eq. (7), combines precision and recall as an effectiveness measurement of classification in terms of ratio of the importance on both recall and precision. The parameter β is a coefficient to adjust the relative weight of precision and recall, and is usually set to 1 [10] (We adopt $\beta = 1$ in this paper). As a result, F-Measure provides more insight into the classifying ability than the accuracy metric. In our experiments, we will compare the accuracy, precision, recall and F-Measure of SLIM and two traditional classifiers.

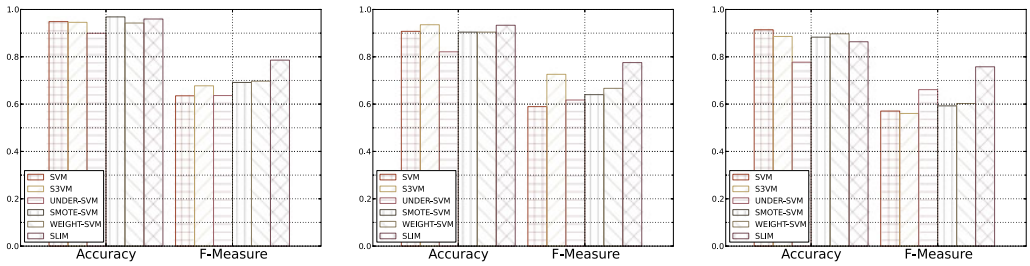
$$F\text{-}Measure = \frac{(1+\beta^2) \cdot Recall \cdot Precision}{Recall + \beta^2 \cdot Precision}. \tag{7}$$

4.5 Performance

4.5.1 Accuracy and F-Measure

As discussed in Section 4.4, the overall performance can be compared by investigating the values of accuracy and F-Measure. Fig. 3 shows that, for all three experiments (bicycling detection, desk detection, mattress detection), the classification accuracy of all inference models, except for UNDER-SVM, are comparable and quite satisfied (around 95 percent for bicycling detection and 90 percent for desk and mattress detection). UNDER-SVM is inferior to the other methods considering accuracy.

Considering the F-Measure, the proposed SLIM is the highest in all three cases, with around 12 percent average improvement over the other five models. The imbalanced models (i.e., UNDER-SVM, SMOTE-SVM, and WEIGHT-SVM) achieve higher F-Measures than the traditional SVM, which conforms with the design theory of these models. Yet their performances are inferior to SLIM, since these models do not make use of the abundant unlabelled sensing data to facilitate learning. On the other hand, S3VM incorporates both labelled and unlabelled sensing data for training. S3VM performs better than SVM and imbalanced models in desk detection (Fig. 3b), yet it loses when comparing to SLIM. In fact, in bicycling detection (Fig. 3a) and mattress detection (Fig. 3c), S3VM performs even worse than imbalanced models. The reason is that, though unlabelled sensing



(a) Bicycling detection (b) Desk detection (c) Mattress detection

Fig. 3. Accuracy and F-measure.

data increase the size of training set, the data are also imbalanced in nature. Hence unlabelled data may improve the performance in semi-supervised model in some cases, and may enhance the distribution skew so severely that it has canceled the benefit brought by more training data in other cases. To summarize, the proposed SLIM performs consistently better than the imbalanced models and semi-supervised model.

4.5.2 Precision and Recall

To investigate the detail performance of the six models, we can refer to the precision and recall results in Fig. 4. The precision of SVM is the highest among the six models, while its recall is much lower than the other five models. Considering that precision represents exactness measurement (i.e., how many data instances labelled as positive by the classifier are actually from the positive class) and recall represents completeness measurement (how many data instances belonging to the positive class are actually labelled correctly), SVM is highly skewed that it sacrifices the chances to classify an instance as positive in order to get a high precision value. UNDER-SVM gives the other extreme performance. It achieves the highest recall at the cost of lowest precision. The result is reasonable as the undersampling scheme adopted by UNDER-SVM loses much valuable information about the negative class.

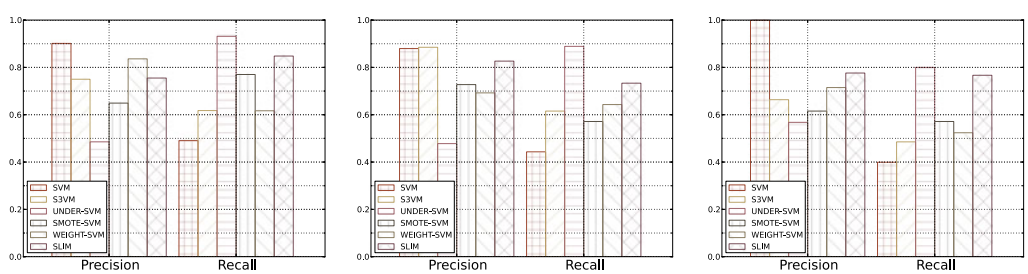
For a practical application, these two extreme performances are unacceptable. Hence the other four models inherently try to balance the precision and recall values. S3VM benefits from the auxiliary unlabelled data, while SMOTE-SVM and WEIGHT-SVM make use balancing techniques. The proposed SLIM makes good use of both unlabelled data and balancing techniques, and hence performs best among these models.

4.5.3 The Effect of Sampling Subset Number T

The number of sampling subset T is an important parameter in SLIM, as it reflects the coverage degree of the sensing data. Typically, if the distribution of sensing data is balanced for both positive and negative class, the large difference of instance numbers of training data will not be of great influence. For example, if the two classes are generated from two Gaussian distributions with standard deviation that are far away from each other, the different number of samples generated from these distributions will not make a classifier perform poorly. However, in reality, the sensing data are not distributed symmetrically. Sampling few instances from the negative class can only represent a small coverage. From Fig. 5 we can see that, when the subset number is small, the precision and recall of SLIM have a large gap, with recall extremely high and precision extremely low. With the increment of the subset number, the gap of precision and recall gets smaller and smaller. And finally, when the subset number is large enough, SLIM achieves better performance, with both precision and recall getting stable.

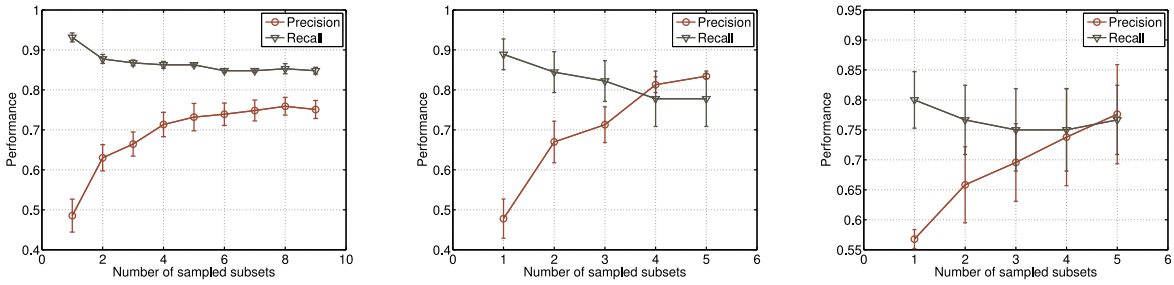
4.5.4 The Effect of Threshold θ

The threshold parameter θ controls the aggressiveness of the final classifier [Eq. (4)]. When θ is small, SLIM will recognize a positive instance with a low standard, i.e., only a few of the classifiers trained on subsets are needed to agree that the instance is positive. This behavior results in low precision and high recall, as shown in Fig. 6. With the increment of θ, the precision and recall become more balanced. The result is expected, as each classifier trained on different subsets only reflects the data features of the corresponding subset. If one of the classifier considers that the instance is positive, it can only be induced that the instance is a



(a) Bicycling detection (b) Desk detection (c) Mattress detection

Fig. 4. Precision and recall.



(a) Bicycling detection (b) Desk detection (c) Mattress detection

Fig. 5. Classification performance with different number of subsets T.

potential real positive instance. With more and more classifiers voting on the instance, the confidence gets higher. The results of all experiments indicate that a stricter SLIM will generate more balanced performance considering precision and recall.

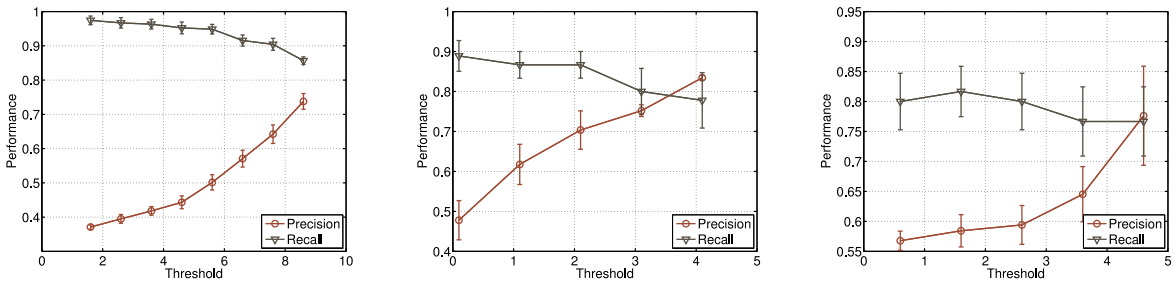
5 Related Work

Context sensing. Context sensing applications are related to our work as many of them require inference components. SoundSense [4] models sound events on mobile phones to achieve context recognition. Jigsaw [35] constructs a general-purposed pipeline-based engine to support continuous sensing applications on mobile phones. IODetector [29] provides an indoor/outdoor detection service via the collaboration of phone sensors. CarSafe [5] detects and alerts drivers to dangerous driving conditions and behavior by using duel cameras of smartphones. These works provide application scenes for our work when the characteristic of sensing data distribution is incorporated.

Ensemble of classifiers. Many prior context sensing engines largely aimed at optimizing individual classifiers [35], [36]. Recently, some works try to integrate multiple classifiers. A context querying engine for mobile phones is proposed in [37]. The engine is able to plug in different classifiers. CQue [38] allows individual classifiers to be easily integrated without worrying about how to leverage other contexts to improve performance and efficiency. Our work also considers combining different classifiers. However, the combinations of classifiers are represented in two folds: combination of multiple sensing dimensions and combination of multiple training subsets.

Semi-supervised learning. Recent works also considered employing semi-supervised schemes [13], [14], such that the unlabelled sensing data can be incorporated for training robust classifiers with a few labelled sensing data. In [13], semi-supervised learning techniques are used to sample additional sensing data from diverse operating environments and multiple devices. In [14], the authors make use of unlabelled data to bootstrap new apps with a small initial training data set. These works haven't considered the imbalanced nature of sensing data and employ traditional semi-supervised paradigms. Our work, on the contrary, explicitly investigates the difficulty of the imbalanced sensing data and designs a sampling-based approach.

Under-sampling. Under-sampling schemes have been studied for imbalanced learning in machine learning [17], [18], [39], [40]. In [17], two ensemble algorithms (EasyEnsemble and BalanceCascade) are proposed. EasyEnsemble samples independent subsets from majority class, while BalanceCascade uses trained classifiers to guide the sampling process for subsequent classifiers. In [18], four KNN under-sampling methods are proposed based on the characteristics of the given data distribution. One-sided selection (OSS) method is proposed in [40], where a representative subset of the majority class is selected and combined with the minority class, forming a preliminary set. Further refinement using data cleaning technique is then applied to the preliminary set. GSVM-RU [39] takes advantage of granular support vector machines (GSVM) by using an iterative learning process that uses SVM itself for under-sampling. All of these under-sampling schemes are designed for supervised learning framework. In this work, we adapt under-sampling to mobile sensing apps with both labelled and unlabelled data based on feature similarity measurement.



(a) Bicycling detection (b) Desk detection (c) Mattress detection

Fig. 6. Classification performance with different threshold θ.

6 Conclusion

Having well-performed inference components provides great opportunities for developing diverse mobile apps. In this paper, we have presented SLIM, a new inference framework that overcomes several challenges in training inference components. Firstly, the imbalanced nature of sensing data renders traditional training models inappropriate. SLIM resolves this problem by resorting to under-sampling schemes, such that several balanced data subsets can be used to train reliable classifiers. Secondly, acquiring sufficient labelled sensing data to train a classifier is expensive and time-consuming. SLIM makes use of abundant unlabelled sensing data shared by existing mobile apps to facilitate model training, such that only a small set of labelled data is needed. Lastly, SLIM is able to incorporate and differentiate multiple built-in sensors. Thus it is flexible for diverse mobile apps.

Two proof-of-concept mobile apps, bicycling detection and backing material detection, are designed to investigate the performance of SLIM. We compare SLIM with representative supervised and semi-supervised learning paradigms on several evaluation metrics (including accuracy, precision, recall, F-Measure). The results show that even though all three methods perform well considering accuracy, SLIM performs much better than the other two classical methods considering F-Measures. Therefore, SLIM is more promising to train inference models for mobile sensing apps.

As sensing data from other applications are employed in SLIM, the data privacy and security may be of concern in practice. In the future work, we will apply SLIM to more mobile sensing applications and design mechanisms for protecting data privacy and security.

Acknowledgments

This work was supported in part by the NSFC Major Program 61190110, NSFC under Grants 61171067 and 61133016, and the NSFC Distinguished Young Scholars Program under Grant 61125202.

References

[1] A. T. Campbell, S. B. Eisenman, N. D. Lane, E. Miluzzo, R. A. Peterson, H. Lu, X. Zheng, M. Musolesi, K. Fodor, and G.-S. Ahn, "The rise of people-centric sensing," *IEEE Internet Comput.*, vol. 12, no. 4, pp. 12–21, Jul./Aug. 2008.

[2] N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell, "A survey of mobile phone sensing," *IEEE Commun. Mag.*, vol. 48, no. 9, pp. 140–150, Sep. 2010.

[3] M. Srivastava, T. Abdelzaher, and B. Szymanski, "Human-centric sensing," *Philosoph. Trans. R. Soc. A: Math., Phys. Eng. Sci.*, vol. 370, no. 1958, pp. 176–197, 2012.

[4] H. Lu, W. Pan, N. D. Lane, T. Choudhury, and A. T. Campbell, "Soundsense: scalable sound sensing for people-centric applications on mobile phones," in *Proc. ACM Int. Conf. Mobile Syst., Appl., Serv.*, 2009, pp. 165–178.

[5] C.-W. You, N. D. Lane, F. Chen, R. Wang, Z. Chen, T. J. Bao, Y. Cheng, M. Lin, L. Torresani, and A. T. Campbell, "Carsafe app: Alerting drowsy and distracted drivers using dual cameras on smartphones," in *Proc. ACM Int. Conf. Mobile Syst., Appl., Serv.*, 2013, pp. 13–26.

[6] A. Zhan, M. Chang, Y. Chen, and A. Terzis, "Accurate caloric expenditure of bicyclists using cellphones," in *Proc. ACM Conf. Embedded Netw. Sens. Syst.*, 2012, pp. 71–84.

[7] W. Hu, G. Cao, S. V. Krishanamurthy, and P. Mohapatra, "Mobility-assisted energy-aware user contact detection in mobile social networks," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst.*, 2013, pp. 155–164.

[8] B. Liu, Y. Jiang, F. Sha, and R. Govindan, "Cloud-enabled privacy-preserving collaborative learning for mobile sensing," in *Proc. ACM Conf. Embedded Netw. Sens. Syst.*, 2012, pp. 57–70.

[9] P. Zhou, Y. Zheng, and M. Li, "How long to wait? Predicting bus arrival time with mobile phone based participatory sensing," *IEEE Trans. Mobile Comput.*, vol. 13, no. 6, pp. 1228–1241, Jun. 2014.

[10] H. He, and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Jul. 2009.

[11] H.-T. Cheng, F.-T. Sun, M. Griss, P. Davis, J. Li, and D. You, "Nuactiv: Recognizing unseen new activities using semantic attribute-based learning," in *Proc. ACM Int. Conf. Mobile Syst., Appl. Serv.*, 2013, pp. 361–374.

[12] M. Stikic, D. Larlus, S. Ebert, and B. Schiele, "Weakly supervised recognition of daily life activities with wearable sensors," *IEEE Trans. Pattern Anal. Mach. Intel.*, vol. 33, no. 12, pp. 2521–2537, Dec. 2011.

[13] E. Miluzzo, C. T. Cornelius, A. Ramaswamy, T. Choudhury, Z. Liu, and A. T. Campbell, "Darwin phones: The evolution of sensing and inference on mobile phones," in *Proc. ACM Int. Conf. Mobile Syst., Appl. Serv.*, 2010, pp. 5–20.

[14] X. Bao, P. Bahl, A. Kansal, D. Chu, R. R. Choudhury, and A. Wolman, "Helping mobile apps bootstrap with fewer users," in *Proc. ACM Conf. Ubiquitous Comput.*, 2012, pp. 491–500.

[15] Daily activity statistics—statistics brain. (2012). [Online]. Available: http://statisticbrain.com/average-daily-activities/

[16] T. Denning, A. Andrew, R. Chaudhri, C. Hartung, J. Lester, G. Borriello, and G. Duncan, "Balance: Towards a usable pervasive wellness application with accurate activity inference," in *Proc. ACM Int. Workshop Mobile Comput. Syst. Appl.*, 2009, pp. 5:1–5:6.

[17] X.-Y. Liu, J. Wu, and Z.-H. Zhou, "Exploratory undersampling for class-imbalance learning," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 2, pp. 539–550, May 2009.

[18] I. Mani and I. Zhang, "knn approach to unbalanced data distributions: A case study involving information extraction," in *Proc. Int. Conf. Mach. Learn. Workshop Learning Imbalanced Datasets*, 2003.

[19] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, 1997.

[20] Z.-H. Zhou, "Ensemble learning," in *Encyclopedia of Biometrics*. New York, NY, USA: Springer, 2009, pp. 270–273.

[21] E. Miluzzo, A. Varshavsky, S. Balakrishnan, and R. R. Choudhury, "Tapprints: Your finger taps have fingerprints," in *Proc. ACM Int. Conf. Mobile Systems, Appl., and Serv.*, 2012.

[22] J. Manweiler, N. Santhapuri, R. R. Choudhury, and S. Nelakuditi, "Predicting length of stay at wifi hotspots," in *Proc. IEEE Int. Conf. Comput. Commun.*, 2013, pp. 323–336.

[23] Y.-F. Li and Z. H. Zhou, "Towards making unlabeled data never hurt," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 1081–1088.

[24] K. Bennett and A. Demiriz, "Semi-supervised support vector machines," in *Proc. Adv. Neural Inf. Process. Syst.*, 1998, pp. 368–374.

[25] R. Akbani, S. Kwek, and N. Japkowicz, "Applying support vector machines to imbalanced datasets," in *Proc. Eur. Conf. Mach. Learning*, 2004, pp. 39–50.

[26] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artif. Intel. Res.*, vol. 16, pp. 341–378, 2002.

[27] E. Osuna, R. Freund, and F. Girosi, "Support vector machines: Training and applications," Tech. Rep.144, 1997.

[28] K. Veropoulos, C. Campbell, and N. Cristianini, , "Controlling the sensitivity of support vector machines," in *Proc. Int. Joint Conf. Artif. Intel.*, 1999, pp. 55–60.

[29] P. Zhou, Y. Zheng, Z. Li, M. Li, and G. Shen, "Iodetector: A generic service for indoor outdoor detection," in *Proc. ACM Conf. Embedded Netw. Sens. Syst.*, 2012, pp. 113–126.

[30] S. B. Eisenman, E. Miluzzo, N. D. Lane, R. A. Peterson, G.-S. Ahn, and A. T. Campbell, "Bikenet: A mobile sensing system for cyclist experience mapping," *ACM Trans. Sens. Netw.*, vol. 6, no. 1, p. 6, 2009.

[31] Memo widget. (2013). [Online]. Available: https://play.google.com/store/apps/details? id=com.about jsp.memowidget

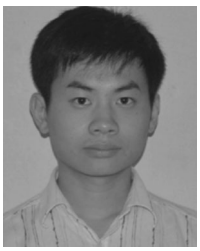
[32] C.-C. Chang, and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM Trans. Intell. Syst. and Technol.*, vol. 2, no. 3, pp. 27:1–27:27, 2011.

[33] N. Chinchor, "Muc-4 evaluation metrics," in *Proc. Conf. Message Understanding*, 1992, pp. 69–78.

[34] C. J. van Rijsbergen, *Information Retrieval*. London, U.K.: Butterworths, 1979.

[35] H. Lu, J. Yang, Z. Liu, N. D. Lane, T. Choudhury, and A. T. Campbell, "The jigsaw continuous sensing engine for mobile phone applications," in *Proc. ACM Conf. Embedded Netw. Sens. Syst.*, 2010, pp. 71–84.
[36] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," *ACM SIGKDD Explorations Newsletter*, vol. 12, no. 2, pp. 74–82, 2011.
[37] D. Chu, N. D. Lane, T. T.-T. Lai, C. Pang, X. Meng, Q. Guo, F. Li, and F. Zhao, "Balancing energy, latency and accuracy for mobile sensor data classification," in *Proc. ACM Conf. Embedded Netw. Sens. Syst.*, 2011, pp. 54–67.
[38] A. Parate, M.-C. Chiu, D. Ganesan, and B. M. Marlin, "Leveraging graphical models to improve accuracy and reduce privacy risks of mobile sensing," in *Proc. ACM Int. Conf. Mobile Systems, Appl. Serv.*, 2013, pp. 83–96.
[39] Y. Tang and Y.-Q. Zhang, "Granular SVM with repetitive undersampling for highly imbalanced protein homology prediction," in *Proc. IEEE Int. Conf. Granular Comput.*, 2006, pp. 457–460.
[40] M. Kubat and S. Matwin, "Addressing the curse of imbalanced training sets: one-sided selection," in *Proc. Int. Conf. Mach. Learning*, 1997, pp. 179–186.

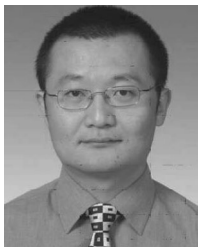
Xinglin Zhang received the BE degree in software engineering from Sun Yat-sen University in 2010, and the PhD degree in computer science from Hong Kong University of Science and Technology in 2014. He is currently doing research at Tsinghua University. His main research interests include wireless ad-hoc/sensor networks, mobile computing, and crowdsourcing. He is a student member of the IEEE and the ACM.



Zheng Yang received the BE degree in computer science from Tsinghua University in 2006, and the PhD degree in computer science from Hong Kong University of Science and Technology in 2010. He is currently a faculty member at Tsinghua University. His main research interests include wireless ad-hoc/sensor networks and mobile computing. He is a member of the IEEE and the ACM.

Longfei Shangguan received the BS degree from the School of Software from Xidian University, China, in 2011. He is currently working toward the PhD degree in the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. His current research interests include pervasive computing, wireless sensor networks and RFID system. He is a student member of the IEEE and ACM.

Yunhao Liu received the BS degree in automation from Tsinghua University, China, in 1995, and the MS and PhD degrees in computer science and engineering from Michigan State University, in 2003 and 2004, respectively. He is currently a professor with Tsinghua University. His current research interests include wireless sensor network, peer-to-peer computing, and pervasive computing. He is a senior member of the IEEE.



Lei Chen received the BS degree in computer science and engineering from Tianjin University, China, in 1994, the MA degree from the Asian Institute of Technology, Thailand, in 1997, and the PhD degree in computer science from the University of Waterloo, Canada, in 2005. He is currently an associate professor in the Department of Computer Science and Engineering, Hong Kong University of Science and Technology. His current research interests include crowdsourcing on social networks, uncertain and probabilistic databases, web data management, multimedia and time series databases, and privacy. He is a member of the IEEE.