Enabling Large-scale Human Activity Inference on Smartphones using Community Similarity Networks (CSN)

Nicholas D. Lane†, Ye Xu*, Hong Lu*, Shaohan Hu*
Tanzeem Choudhury‡, Andrew T. Campbell*, Feng Zhao†

*Dartmouth College
Hanover, New Hampshire, USA

‡Cornell University
Ithaca, New York, USA

†Microsoft Research Asia
Beijing, China

ABSTRACT
Sensor-enabled smartphones are opening a new frontier in the development of mobile sensing applications. The recognition of human activities and context from sensor-data using classification models underpins these emerging applications. However, conventional approaches to training classifiers struggle to cope with the diverse user populations routinely found in large-scale popular mobile applications. Differences between users (e.g., age, sex, behavioral patterns, lifestyle) confuse classifiers, which assume everyone is the same. To address this, we propose *Community Similarity Networks (CSN)*, which incorporates inter-person similarity measurements into the classifier training process. Under CSN every user has a unique classifier that is tuned to their own characteristics. CSN exploits crowd-sourced sensor-data to personalize classifiers with data contributed from other similar users. This process is guided by similarity networks that measure different dimensions of inter-person similarity. Our experiments show CSN outperforms existing approaches to classifier training under the presence of population diversity.

Author Keywords
Mobile Phone Sensing, Activity Recognition, Community Learning.

ACM Classification Keywords
H.5.2 User/Machine Systems; I.5 Pattern Recognition

General Terms
Experimentation, Performance.

INTRODUCTION
The popularity of smartphones with embedded sensors is growing at rapid pace. At the same time, research in the area of mobile phone sensing [17] is expanding the boundaries of mobile applications [12, 25, 22, 21]. Advances in human centric sensing is being fueled by the combination of: i) raw sensor-data, which is now practical to sample from large user populations; and ii) classification models that extract human activities, events and context from low-level sensor-data.

As user populations of mobile sensing applications increase in size the differences between people cause the accuracy of classification to degrade quickly – we call this the *population diversity problem*. In this paper, we demonstrate that the population diversity problem exists and classification accuracy varies widely even as the user population is scaled to up as little as 50 people. To address this problem, we propose *Community Similarity Networks (CSN)*. CSN is a classification system that can be incorporated into mobile sensing applications to address the challenge to robust classification caused by the population diversity problem. The conventional approach to classification in mobile sensing is to use the same classification model for all users. Using CSN, we construct and continuously revise a personalized classification model for each user over time. Typically, personalized models require all users to perform manual sensor-data collection where users provide hand annotated examples of them performing certain activities while their devices gather sensor-data (i.e., labeling data). This is both burdensome to the user and wasteful as multiple users often collect nearly identical data but the training of each model occurs in isolation of each other. The key contribution of CSN is that it makes the personalization of classification models practical by significantly lowering the burden to the user through a combination of crowd-sourced data and leveraging networks that measure the similarity between users.

CSN exploits crowd-sourcing to acquire a steady stream of sensor-data and user input, either corrections to classification errors or the more common hand annotated examples of sensor-data when performing an activity (i.e., labeling). Under CSN training classifiers becomes a networked process where the effort of individual users benefits everyone. However, the use of crowd-sourced data must be done carefully. Crowd-sourced data must only selectively be used during training so the resulting model is optimized for the person using the model. CSN solves this problem by maintaining similarity networks that measure the similarity between people within the broader user population. We do this by proposing three different similarity metrics (i.e., physical, lifestyle/behavior and purely sensor-data driven) that measure different aspects of inter-person diversity which influence classifier performance. The CSN model training phase then utilizes forms of boosting and co-training to allow these different types of similarity to each contribute to improving the accuracy of the personalized classifier.

The contributions of this paper are as follows:

- CSN is the first system to propose embedding inter-person similarity within the process of training activity classifiers. To the best of our knowledge, CSN represents the only activity recognition system designed specifically to cope with the population diversity problem, which would otherwise jeopardize large-scale deployments of mobile sensing systems.
- We propose similarity metrics and a classification training process that support: i) the extraction of similarity networks from raw sensor-data and additional end-user input and ii) a learning process that adapts generic classification models through careful exploitation of crowd-sourced data guided by similarity networks.
- We have evaluated our system with two large-scale mobile sensing datasets, each comprising of approximately 50 people. We measure the robustness of our classifiers and the ability of CSN to cope with population diversity.

COMMUNITY-SCALE CLASSIFICATION

In this section we discuss a key difficulty in realizing large-scale mobile sensing applications. Specifically, we examine how population diversity can cause classification to become unreliable and inaccurate.

One Size Does Not Fit All. As mobile sensing prototype systems are deployed to an increasing number of users their diversity increases as well. These users differ from one another in a variety of ways, a concrete example being physical dissimilarities as measured by sex, weight, height or the level of physical fitness. Beyond these visually obvious differences there are differences based on lifestyle and background. People come from different ethnic and social-economic origins, live and work in different locations and while they may perform the same core collections of activities (e.g., socializing, exercising, working) they may do these activities in significantly different ways.

Inter-personal differences can manifest as differences in the discriminative patterns contained in sensor-data that are used to classify activities, events and contexts. For example, the features from accelerometer data that allow classifiers to distinguish between the basic activities of walking and running can be completely different between a group of older adults (older than 65 years) and a group of people who are in their 20s and 30s. Figure 1 visualizes this difference when these two groups are walking. We plot the first two PCA components on each axis of the figure based on a range of already validated activity recognition accelerometer features [18]. The very clear distinction between sensor-data sourced from these two groups is surprising, particularly given the homogeneity you would expect in a simple activity like walking.

To further quantify this problem we build a LogitBoost classification model [9] and reuse the same previously validated activity recognition features. This model is trained using labeled data from the group of people in their 20s and 30s. Using this model classification accuracy while they walk and climbed stairs ranges between 80% to 90% for each person. However, when this same classifier is used by the group of aged people the average accuracy dropped to nearly 60%. Clearly, a one size fits all approach to classification models will not scale to large user populations which will contain many such groups.

This effect is not only limited to strictly physical behavior (e.g., walking, running or climbing stairs) but extends to a broader range of behavioral inferences. We investigate the breadth of this problem by performing an experiment on two distinct mobile sensing datasets. The first dataset (obtained from the authors of [29, 30]) contains GPS sensor data for 51 people performing 4 different transportation modes (e.g., driving a car or riding a bike). The second is comprised of multi-modal sensor-data (e.g., microphone and accelerometer data) for 41 people performing a range of everyday activities (e.g., walking up stairs, exercising, brushing teeth). For both datasets we train a single classifier and evaluate the accuracy for each user, by applying the classifier to test data sourced only from that user. Figure 2 is the Cumulative Distribution Function (CDF) of accuracy for these experiments, and shows the spread of accuracy within the user population

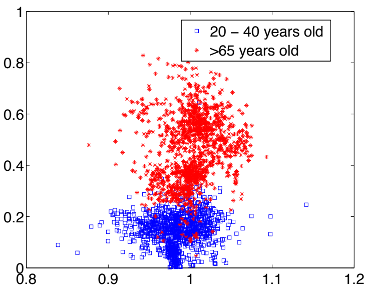


Figure 1. We visualize the differences in features under an identical activity, walking, for two distinct community sub-groups. One group contains people over 65 years old, the other group contains people in their 20s and 30s. Here we show just the first two components of the PCA of these features.

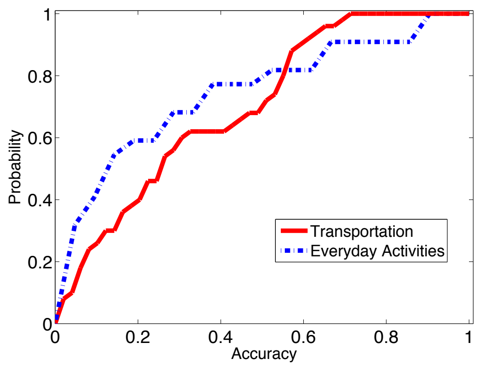


Figure 2. Classification accuracy varies significantly within a large-scale user population for two datasets, one containing everyday activities and the other transportation modes.

under both datasets. Accuracy levels for the transportation mode dataset are as low as approximately 40% for the bottom performing 60% of end-users and as high as 90% for the top 7%. Similarly, we find for the everyday activities dataset for 40% of the users the accuracy is only 12%, even for 80% of users the accuracy raises only marginally to 55%.

Limitations of Current Practice. The de-facto standard practice in incorporating classification into mobile sensing systems centers around a single unchanging classification model which is trained prior to deployment. Due to the reasons of population diversity this model works for some people, but not others; the accuracy of the system remains difficult to predict and increasingly unreliable as the user population grows.

Ideally the classifier would capture the distinctions between certain activities performed by different subgroups in the population as different activities entirely (e.g., walking when performed by two sub-groups could be two different classes); whenever these distinctions impact the classification process. However, this would significantly increase the amount of examples required for the same logical activity. Acquiring these examples is manually intensive (requiring careful labeling of data segments), making this approach impractical as it simply does not scale.

A promising direction being actively explored is the personalization of classification models to improve accuracy (e.g., [19, 26, 20, 15]). These models are tuned to sensor-data generated or encountered by the individual. Typically tuning occurs based on input from the user. For example, the user corrects classification errors or provides additional examples of activities by labeling sensor-data with the ground-truth activity occurring during the sampling of the data. The classifier is then retrained using sensor-data collected and labeled only by the user.

The limitation of such personalization of classification models is that accuracy *only* improves when and if people take the effort to manually provide additional sensor-data examples. Independent of effort it will also take time for people to encounter certain situations that are good discriminative examples to incorporate into the model. The key problem with this type of gradual improvement is that it leads to enormous amounts of redundant effort. Classification models are improved in *isolation* and each user potentially has to repeat steps that have already been done by other users to improve their own personal model.

COMMUNITY SIMILARITY NETWORKS

In this section we describe the system components and the core algorithms used at each stage of CSN. The CSN system is designed to construct and periodically update personalized classification models for each user. A key novelty of CSN is that it achieves personalization by using only a small amount of a specific user's training data and combining it with training data selectively recruited from others with whom the user shares similar traits.

Framework

Figure 3 illustrates stages of the CSN framework that produce personalized classification models for each user. Each of these stages occur either in one of two architectural components, the Mobile Phone Client software or the Mobile Cloud Infrastructure.

The Mobile Phone Client software samples sensor-data to recognize human behavior and contexts by performing inference using classification models. While inference occurs locally the models themselves are downloaded from the Mobile Cloud Infrastructure. The client software also collects training data comprised of both raw sensor-data and data segments that have been labeled by CSN users with the ground-truth activity or context.

The Mobile Cloud Infrastructure is responsible for training classification models. Sensor data is used to construct similarity networks where network edges indicate the level of similarity between two users. CSN employs multiple dimensions of similarity (e.g., sensor-data, physical and lifestyle) to quantify the various ways users can differ. Several similarity networks are generated for each user, one for each of the similarity dimensions. Similarity-sensitive Boosting trains a classifier for each of the different similarity networks (that correspond to a different similarity dimension). Similarity Network Multi-training performs semi-supervised learning and improves every model by recruiting additional labels from the pool of unlabeled data. The final step of multi-training unifies each of the independent classifiers, trained by Similarity-sensitive Boosting, into a single ensemble classifier ready to be installed on the phone of the user.

Mobile Phone Client

In the following subsection we describe the functions performed by the Mobile Phone Client, specifically we detail: i) the classification pipeline, ii) implementation specifics, and iii) the collection of sensor-data and ground-truth labels from users.

Classification Pipeline. The classification pipeline includes sensor-data sampling, feature extraction, and recognition of an activity, event or context.

We use the accelerometer, microphone and GPS sensors to make a variety of proof-of-concept inferences. Our choice of features were based on observations made in prior work [30, 21, 29, 20]. For the accelerometer and microphone we use the same feature set described in [21] which include a variety of time domain and frequency domain features effective for general activity recognition. For the GPS we adopt features that were specifically designed in [30, 29] for transportation mode inference based on time-series GPS readings. Classification is done using a boosted ensemble [23] of naive Bayes classifiers [9]. Inference results are temporally smoothed using a simple Markov model. Although in our client the stages of the classification pipeline (i.e., features and the classification model) remain fixed the parameters of the model are determined, and updated periodically, by the Mobile Cloud Infrastructure.

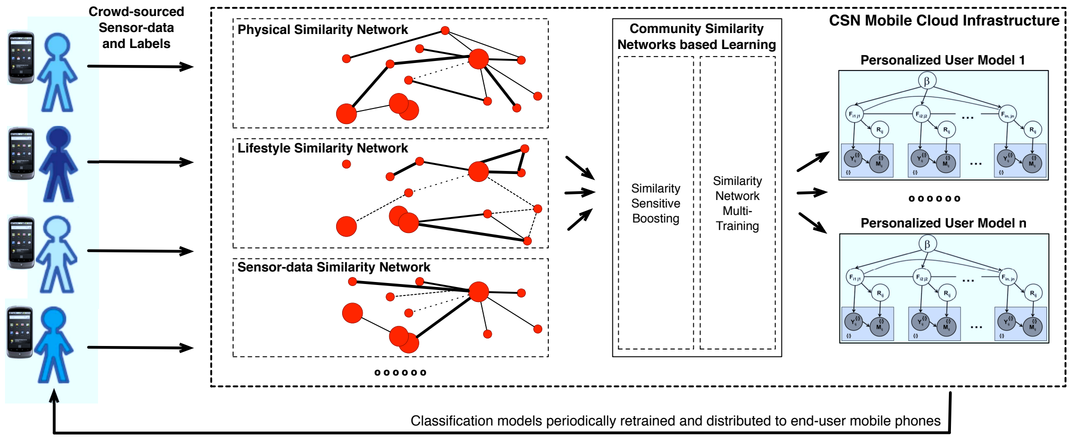

Crowd-sourced Sensor-data and Labels
Physical Similarity Network
Lifestyle Similarity Network
Sensor-data Similarity Network
Community Similarity Networks based Learning
Similarity Sensitive Boosting
Similarity Network Multi-Training
CSN Mobile Cloud Infrastructure
Personalized User Model 1
Personalized User Model n
Classification models periodically retrained and distributed to end-user mobile phones

Figure 3. The processing phases within Community Similarity Networks

Implementation. Our prototype client is implemented on the Google Android Nexus One [2]. The design of the phone client is split between a portable classification pipeline library written in C++ and set of device specific supporting service (e.g., sensor sampling, end-user GUI). The library provides core classification pipeline components, including feature extraction and model inference. The device specific components are written in Java and connected with the library via a JNI bridge.

CrowdSourcing. CSN exploits the crowd-sourcing of both sensor-data and user input to improve classification models. User input provides the ground-truth activity to segments of sensor-data. Two specific types of user input is supported. First, users can be asked to confirm or deny a class inference. For example, asking the user a question – *'Are you currently exercising?'*. Such responses are used later as positive or negative examples of certain activities. Second, users can explicitly label data as being an example of an activity or event. For example, users indicate the ground-truth activity when a segment of sensor-data was sampled by selecting it from a list presented on the phone GUI. These types of interactions with users can be incorporated into applications in various ways. As an example, simple binary yes/no questions can be presented when the user unlocks their phone. Alternatively, more involved interaction, such as when users are selecting activities from a list, can be framed as software configuration or *calibration*. Similar forms of user interaction already occur in real products, for example, reading training sentences into speech recognition software or running for precisely one mile to calibrate a single activity recognition system like Nike+[3].

Mobile Cloud Infrastructure

In this section we describe how the Mobile Cloud Infrastructure: i) computes similarity networks and ii) uses these networks to train personalized classification models that are distributed to all users.

Implementation. Our prototype implementation makes extensive use of Amazon Web Services [5] (AWS) which offer a number of generic components useful in building a distributed system. CSN utilizes the AWS message queues (SQS), binary storage (S3) and the simple query-able hash table service (SimpleDB). Each stage of the model training performed by the cloud is implemented either as python scripts or C++ modules depending on available library support given the required functionality. These stages run on a pool of linux machines as part of the Amazon Elastic Cloud product and interact with the individual AWS services as needed. Once a classification model is trained it is serialized into a JSON-like format and written to the binary storage (S3), ready to be downloaded by the client.

Similarity Networks

Each similarity network within CSN is constructed from the perspective of a single target CSN user. Nodes in the network represent other CSN users and edge-weights measure the degree to which the target user is similar to the other users. The CSN framework is designed to leverage multiple similarity measurements, which capture different dimensions of affinity between people. Depending on the activities or contexts that are to be recognized different dimensions may be utilized. In this paper we propose the use of three dimensions of inter-person similarity: sensor-data similarity, physical similarity and lifestyle similarity. We demonstrate the effectiveness of each of these different dimensions in classifying different categories of activities and contexts in the Evaluation section. However, CSN is agnostic as to the exact similarity dimensions used.

We now describe the different similarity dimensions included within the CSN framework.

Physical Similarity. Physical differences between people (e.g., weight, height, age, level of physical fitness or wellbeing) will vary greatly from person to person within a large user population. Such differences can alter the way people move and perform certain physical activities. For example, as we detailed in the earlier section on community-scale classification differences in age can effect the recognition of seemingly simple activities like walking upstairs or jogging.

To compute a single physical similarity value between a pair of users CSN employes five types of physical information: age, height, weight, and the scores from two well established physical well-being surveys (Yale Physical Activity Survey [13] and SF-36 physical activity score [4]). Each of these five values act as elements in a vector that represents a single user. The physical similarity between users (i, j) is based on the mahalanobis distance between the two vectors for each person, as shown here,

$$sim(i,j)^{phy} = exp(-\gamma(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top \Sigma^{-1}(\boldsymbol{x}_i - \boldsymbol{x}_j)) \quad (1)$$

where, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are the physical vectors for user i and user j, Σ is the covariance matrix and γ is an empirically determined scaling parameter.

Lifestyle Similarity. The lifestyle similarity metric attempts to capture the diversity in how people lives their lives, examples of which include: occupation, diurnal patterns (e.g., are they an early morning person or active late at night), the distribution of activities performed, mobility patterns and significant places [8] (e.g., where they work and live). Occupation and the location of work alter, for instance, the accelerometer and audio patterns occurring during social interactions (e.g., meetings and conversations). The time of day and significant places can effect the background context in which people perform activities. For example, late at night or early in the morning different locations will have different background activities that alter the sampled data (e.g., noise from people or cars). Collectively, these factors can change the distribution of features and shift discriminative boundaries for recognizing classes of activities.

We compute lifestyle similarity using three types of information: mobility and diurnal patterns in combination with the distribution of activities performed by users. Mobility patterns are based on GPS location estimates, which are tessellated into m distinct square tiles of equal size. Diurnal patterns are captured as a series of timestamps that are recorded whenever the user is inferred to be non-stationary by the classification pipeline. These timestamps are rounded, and are represented as the particular hour in the week in which they occur (e.g., they range between hour 0 at the start of the week to hour 167 on the final hour of the final day). The distribution of activities is based on the duration users are inferred to be performing each activity classes (e.g, walking, socializing) detected by the classification pipeline. We construct three histograms for each of these types of lifestyle information for every user, normalizing the frequencies across all histograms. For each pair of users (i, j), we compute the lifestyle based similarity by the following equation:

$$sim(i,j)^{life} = \sum_{f \in \mathcal{F}} \mathbf{T}_f(i)^\top \mathbf{T}_f(j) \quad (2)$$

where $\mathbf{T}_f(i)$ is a histogram vector for user i of type f and $\mathcal{F}$ contains each type of lifestyle histogram. Lifestyle similarity between two users is the sum of the inner product of the histograms for each type of lifestyle information used by CSN.

Sensor-data Similarity. Differences between users lifestyle, behavioral patterns, or location will likely manifest as differences in their sensor-data. Sensor-data similarity, unlike lifestyle similarity, does not require feature engineering to extract particular dimensions from the data. Similarly, unlike physical similarity it does not require additional information to be provided by the user (e.g., age). Instead, measuring inter-person similarity based on sensor-data is inherently purely a data-driven approach. Later, we report its effectiveness across a wide range of classification tasks. However, it requires much larger amounts of computation to determine similarity between users than computing lifestyle or physical similarity.

Computing similarity based on the raw sensor-data will be effected by noise and capture too many insignificant variations in the data. Instead, we compute sensor-data similarity between the features extracted from the raw data. For this purpose CSN employes the same features used by the classification pipeline, described earlier in this section. Individual users will accumulate varying amounts of sensor-data based on how frequently they use their device. Consequently, we compute "set" similarity whereby any duplicate feature vector for a user is ignored and only the unique vectors generated from the data of a person are used. For our similarity measurement we adopt a commonly used formulation [28] where the similarity between two users is,

$$sim(i,j)^{data} = \frac{1}{N_i N_j} \sum_{l=1}^{N_i} \sum_{m=1}^{N_j} sim(\boldsymbol{x}_{il}, \boldsymbol{x}_{jm}) \quad (3)$$

where, $\{\boldsymbol{x}_{il}, l = 1 : N_i\}$ is the data of user i, and $\{\boldsymbol{x}_{jm}, m = 1 : N_j\}$ is the data of user j.

However, this pairwise computation quickly becomes impractical as the number of unique features per user increases. To cope with this problem, we adopt Locality Sensitive Hashing (LSH) [7] to construct a histogram to characterize the "set" of data from each user and then compute the similarity between a pair of users by employing this histogram representation. Our method obviates the need to compute the pairwise relations of data from two users as required by the traditional "set" similarity, which has a linear time complexity with the average data size of each user. The basic idea of the LSH method is that a hashing function family can capture the similarity between data. In other words, similar data have a high probability to share the same value after hash mapping.

$$Pr_{h\in\mathcal{H}}[h(\boldsymbol{x_1}) = h(\boldsymbol{x_2})] = s_{\mathcal{H}}(\boldsymbol{x_1}, \boldsymbol{x_2}) = E_{h\in\mathcal{H}}[s_h(\boldsymbol{x_1}, \boldsymbol{x_2})] \quad (4)$$

Therein, $\boldsymbol{x_1}, \boldsymbol{x_2} \in \mathcal{X}$ are two data, $\mathcal{H}$ is a LSH family, h is the hash function sampled from $\mathcal{H}$, and $s_{\mathcal{H}}$ is a similarity measure of $\mathcal{X}$, which is induced by the LSH family $\mathcal{H}$ [7].

In CSN, we randomly choose B independent $0/1$ valued hashing functions $\{\boldsymbol{h_i}\}$ from the *random projection for $\mathcal{L}_2$ distance LSH family* [7] and form a $B-$bit hash function $f = (\boldsymbol{h_1}, \boldsymbol{h_2}, , ..., \boldsymbol{h_B})$. The number of functions B controls the tradeoff between efficiency and accuracy [7].

We apply the $B-$bit hash function to build histograms for each user, whose size is 2^B. Now, we formalize how to construct a histogram from the features of the user. According to the description, let $\mathcal{X}$ be the data space, $\mathcal{F}$ be the B-bit hash functions family mapping from $\mathcal{X}$ to $\mathcal{D} = \{0, 1, \ldots, 2^B - 1\}$, and $\{\mathbf{e}[i] \mid i \in \mathcal{D}\}$ be the standard basis of the $|2^B|$-dimensional vector space. Hence, given $h \in \mathcal{H}$, the histogram T_f for any user i is defined as follows,

$$\mathbf{T}_f(i) = \sum_{\boldsymbol{x_{il}} \in i} \boldsymbol{e}[f(\boldsymbol{x_{il}})] \quad (5)$$

here, $\{\boldsymbol{x}_{il}, l = 1 : N_i\}$ is data of user i, and $\mathbf{T}_f(i)$ is determined by the hash function f sampled from $\mathcal{F}$.

Thus each element of the histogram vector $\mathbf{T}_f(i)$ can be regarded as a bin to record the frequency at which data from user i is mapped into it. As the value of the hash function indicates the probability that two data share the same value after mapping, two users that have many "matched" values in the corresponding bins of histograms implies a high similarity between them. The inner product of the two histogram vectors is next applied to compute the similarity metric for the two users:

$$sim(i, j)^{data} = \mathbf{T}_f(i)^\top \mathbf{T}_f(j) \quad (6)$$

To estimate the expectation shown in Eq. 5, we construct several histograms $f \in \mathcal{F}$ for each user and compute an average value using Eq. 6.

The time complexity of computing the LSH based similarity metric is linear with the average quantity of data for each user. Compared with the pairwise computing method shown in Eq. 3 which is quadratic, the LSH based similarity metric is very efficient.

Community Similarity Networks based Learning

Learning personalized classification models for each user occurs in two stages under CSN. First, Similarity-sensitive Boosting trains three separate classifiers, one for each type of similarity network that is maintained for every user. Each of the classifiers are personalized to the characteristics of the specific individual who will use them. Further, each have different strengths when recognizing specific categories of activity depending on the similarity network used (e.g., physical similarity performs well with physical activities like climbing stairs). Second, Similarity Network Multi-training occurs which: i) uses a semi-supervised approach to recruit additional labels from the unlabeled pool of sensor-data leveraging the different strengths of each separate classifier; and ii) unifies the three classification models trained by Similarity-sensitive Boosting into a single ensemble classifier, ready to be installed on the phone of the user.

Similarity-sensitive Boosting. A personalized classification model emphasizes the particular characteristics found in a target user to increase accuracy. CSN accomplishes personalization using a modified online boosting algorithm [23]. Boosting is a common learning technique that builds a model that is a composite of several weak classifiers trained over multiple iterations. At each iteration certain data segments are weighted higher than others. Under conventional boosting these weights are only altered based on the classification performance of the weak learner trained during the previous iteration. Those data segments that were incorrect are weighted higher than others so the weak classifier produced in the next iteration will be better able to classify these previously incorrect segments. CSN modifies this process by imposing an additional term to the weight at the initial iteration. This weight is based on the similarity between the user i, whose personalized model is being trained, and the user which provides the data,

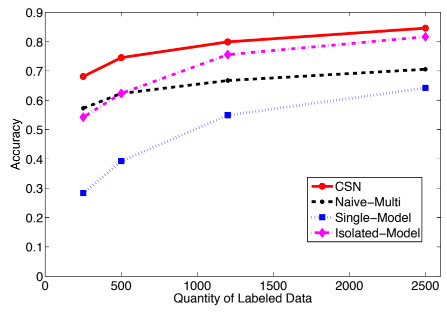
$$weight^{(0)}(\boldsymbol{x}_k) = sim(i, k) \quad (7)$$

where, k indicates the user who produces the data $\boldsymbol{x}_k$. We define $sim(i, k)$ as the edge weight between these two individuals within the similarity network being used during the boosting process. As a consequence only data segments from user i or any users who are highly similar to user i will be weighted highly and so able to have strong influence over the learned classification boundaries. In subsequent iterations the weighting of data segments is left to fluctuate based solely on classification performance. As boosting is an ensemble technique the CSN framework remains flexible, the weak learner can be replaced with any alternative supervised classifier based on the requirements of intended classification task.

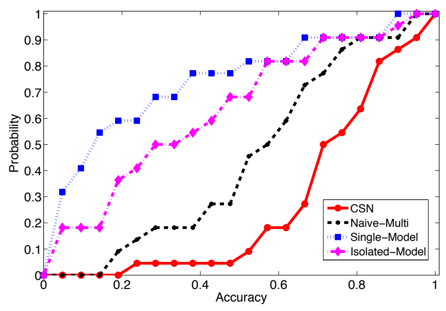
Similarity Network Multi-Training. The three varieties of similarity networks currently used in CSN capture different dimensions of similarity between users. For example, some users being highly similar in terms of physical characteristics but polar opposites when it comes to lifestyle. Using Similarity-sensitive Boosting in conjunction with any of these different networks will result in different classification models. Each network will emphasize different partitions of the training data. This diversity is valuable as the different similarity networks produce models that are highly effective for some classes of activity but not others (see evaluation section). A simple example of this being those activities that are closely connected to the physical characteristics of the person, e.g., running and exercising, benefit from a classification model trained using a physical similarity network.

CSN exploits the strengths of each similarity network by adopting the technique of multi-training (a variation of co-training proposed in [31]). Multi-training is a semi-supervised training algorithm designed to utilize multiple complementary views of the same labeled training data to generate additional labels, which are assigned to data segments within the pool of unlabeled data. This approach is appropriate for CSN given that crowd-sourcing generates large amounts of unlabeled data. People will only infrequently take the time to provide any manual user input; but since simply collecting data is transparent to the user then large pools of unlabeled data quickly can accumulate. Employing a multi-training approach allows CSN to use the diversity provided by the different similarity networks to make use of a plentiful and otherwise wasted resource, unlabeled data.

The multi-training process begins by initially using the three classifiers trained by Similarity-sensitive Boosting. Each of these classification models maintains an independent logical copy of the labeled and unlabeled data. An iterative process is applied whereby the classification models are used to in turn to "label" unlabeled portions in the logical datasets maintained by each of the other models. At the end of each



(a) Accuracy



(b) CDF

Figure 4. Classification accuracy for the Everyday Activities dataset under CSN and three baselines.

iteration the classifiers are then retrained (using Similarity-sensitive Boosting) based on the combination of the labeled data from the previous iteration along with any new additional labels. Acquiring labels in this way can be an error-prone process, as a result labels are only accepted when there is agreement with more than half of the classification models. Judging the quality of a proposed new label, based on a majority decision, is only one of many ways that quality can be assessed. Multi-training continues to iterate for several rounds until a stopping condition is met. CSN uses currently a stopping condition based on how many labels are accepted at each iteration. If the number of recruited labels is too low for too many iterations then multi-training stops.

EVALUATION

In this section we evaluate the effectiveness and design choices of CSN. Our experiments show that by incorporating similarity networks among users into the classification process CSN is better equipped to cope with the population diversity problem, compared to existing techniques.

Experimental Methodology

To evaluate CSN we use two large real-world datasets and three representative baselines.

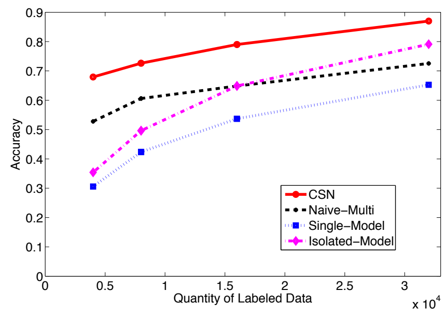
Datasets. Our two datasets require a variety of the activity inferences frequently used in mobile sensing applications. The first dataset, *Everyday Activities*, contains a broad range of routine human activities that have been used to support application domains such as mobile health [12]. The other dataset, *Transportation*, is much more focused on a single category of activity, transportation modes, which are building blocks of various mobile applications, e.g., applications that promote green transportation [14]. We collect the data for *Everyday Activities* as part of a series of internal experiments. The data comprises both simple activities: $\{walk, run, stationary\}$ and high-level behaviors: $\{meeting, studying, exercising, socializing\}$. A total of 41 people contribute to this dataset using a Nexus One smartphone sampling sensor-data from the accelerometer, microphone and GPS. People carry the device for variable lengths of time that range between one and three weeks. For *Transportation* we use an external source [29, 30], with the dataset containing only different transportation modes, specifically: $\{bike, bus, car, walk\}$. This dataset comprises 51 people who carry for three months one of a variety of devices that are equipped with a GPS, including, phones, PDAs and personal navigation devices.

Benchmarks. We compare the performance of CSN against three benchmarks, *single*, *isolated* and *naive-multi*. Our benchmarks use the same features and apply the same classification model as CSN but differ significantly in how they approach classifier training. The benchmarks of *single* and *isolated* correspond with the two types of common practice we detailed in the earlier section on community-scale classification. In *single* the same generic model is provided to all users. Unlike CSN after the release of the system the model does not change and new training data is not collected. Under *isolated* every user has their own model. Each user model is personalized by using training data sourced directly from the user. The weakness is that each classification model is considered in isolation of one another. No co-operation or sharing of training data occurs among users. Finally, *naive-multi* allows us to demonstrate the benefit of CSN solely attributable to the use of similarity networks. During training *naive-multi* performs boosting and multi-training, the same techniques used in CSN. However, *naive-multi* uses the conventional versions of these learning techniques without the use of similarity networks. Specifically, the differences are: i) during boosting the weighting of training data at each iteration only changes based on classification performance instead of inter-personal similarity and ii) during multi-training the classifiers used are not based on dimensions of similarity but based on classifiers trained with equally-sized random subgroups of the training data.

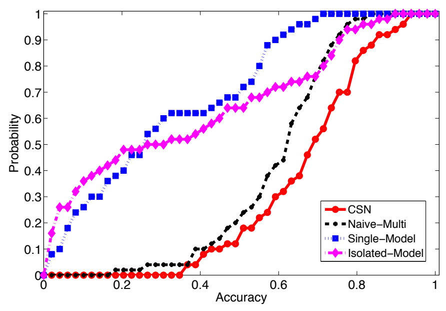
Robust Classification with Low User Burden

Our first set of experiments finds, under both datasets, CSN provides more robust classification than any of the benchmarks. Not only is CSN able to achieve higher classification accuracy but we observe classification accuracy is also more evenly distributed throughout the user population. Under CSN the burden to provide training data is lowered, thus, accuracy numbers comparable to the benchmarks can be achieved with smaller quantities of data. In what follows, we report accuracy values resulting from five fold cross-validation.

Figures 4(a) and 5(a) show the results of experiments where we assume users contribute different amounts of labeled data. For each quantity of labeled data we measure the average

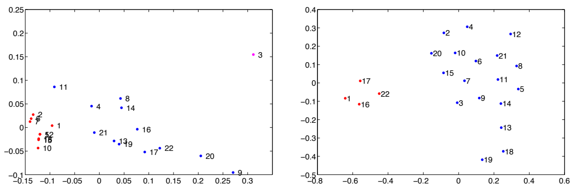


(a) Accuracy



(b) CDF

Figure 5. Classification accuracy for the Transportation dataset under CSN and three baselines.



(a) Physical (b) Lifestyle

Figure 6. MDS projection of physical and lifestyle similarity networks used by CSN.

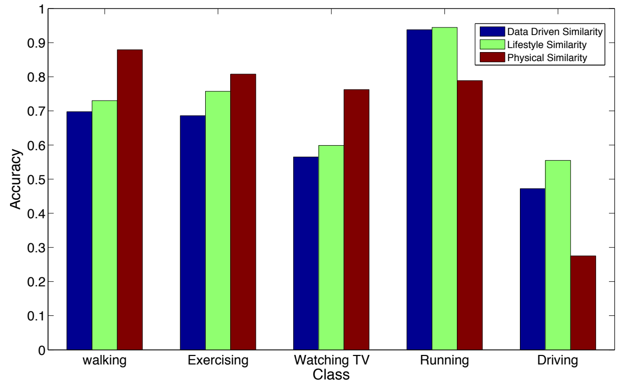


Figure 7. The classification accuracy of each activity class under different dimensions of similarity using the Everyday Activities dataset. It shows different dimensions of similarity are effective for different activities.

per person accuracy of classification for models trained under CSN and the three other benchmarks. Figure 4(a) uses *Everyday Activities*, and Figure 5(a) repeats the experiment using *Transportation*. In both figures the accuracy of CSN outperforms all baselines for each quantity of training data tested. For example, Figure 4(a) shows if 500 labeled data segments are used (approx. 15 minutes of training data per user) then CSN outperforms *naive-multi* and *isolated* by 22%. Similarly, from Figure 5(a) we see if 1.6 x 10^4 labeled data segments are used (approx. 137 minutes of training data from each user) the accuracy of CSN exceeds *single* by 47% and *naive-multi* by 32%.

From Figures 4(a) and 5(a) we also learn that CSN is able to lower the user burden of contributing training data. As an example, Figure 4(a) shows *isolated* requiring 36 minutes of training data from a user to achieve 74% accuracy. CSN can provide approximately this same accuracy for only 15 minutes of training data, a data reduction of 58%. Alternatively, if we consider Figure 5(a) *isolated* is able to perform with 77% accuracy but requires 270 minutes of training data. Again, CSN can provide approximately this level of accuracy but with 49% less data, only needing 137 minutes of data per user. Under CSN users are better rewarded for contributing data due to it having a higher ratio of classification accuracy to crowd-sourced training data, than the other benchmarks.

Figures 4(b) and 5(b) present CDFs of per user accuracy. We illustrate the fraction of the user population who experience different classification accuracy under CSN and all benchmarks. Figure 4(b) uses *Everyday Activities* and assumes users provide 15 minutes of training data each, while Figure 5(b) assumes users provide 137 minutes of labeled data and uses the *Transportation*. Ideally all users should receive the same level of accuracy, otherwise classification performance will be unpredictable when deployed. Better performance is indicated in these figures by curves that are furtherest to the right. We observe from each figure CSN has the most even distribution of accuracy compared to all benchmarks. For example, Figure 4(b) shows for 75% of users that CSN provides 82% accuracy compared to just 65% for *isolated*, 48% for *single* and 52% for *naive-multi*. Figure 5(b) reinforces this finding and indicates for again 75% of users CSN provides 77% accuracy instead of the 68%, 53% and 66% accuracy offered by *isolated*, *single* and *naive-multi* respectively.

Benefits of Leveraging Similarity Networks

With the following experiments we investigate the effectiveness of the similarity networks used by CSN.

To test if the similarity networks used by CSN are capturing meaningful differences between people, we collected additional demographic information from 22 individuals who contributed to *Everyday Activities*. Figures 6(a) and 6(b) plot the result of applying multidimensional scaling (MDS) to two similarity matrices for these people using physical and lifestyle similarity. Distances between points in these figures are proportional to differences in similarity. Figure 6(a) shows two clear groupings. We find these groups correspond to people with similar physical characteristics. The people in the tight cluster near the left of the figure are all over 30 years old, all male, and have similar physical fitness levels.

In contrast, the looser clump of people near the middle are in the same age range (22 - 26) but are all more diverse in terms of sex, and fitness. The outlier in Figure 6(a) is a 50 year old woman and is distinct due to her sex and exceptional fitness. The clusters in Figure 6(b) also correspond to our interview ground truth. The tight cluster to the left are a small group of people who live off campus and maintain regular 9 a.m. to 5 p.m. working hours. They are in sharp contrast to the very loose cluster on the right of the figure. This cluster contains students who, although they live very close to each other, also have erratic sleeping and activity patterns which results in them being grouped but not as tightly as the nearby cluster.

In Figure 7 we can see the value of using multiple similarity dimensions. The figure illustrates the different levels of classification accuracy achieved when using each of our three similarity dimensions to classify classes found in *Everyday Activities*. None of the three similarity metrics has the highest accuracy across all the activities. We find a similar pattern exists within *Transportation*. By exploiting all of these forms of similarity CSN is able to better handle a wide range of classification tasks. This result supports the design choice to use multiple dimensions of similarity and leverage them all when training classification models.

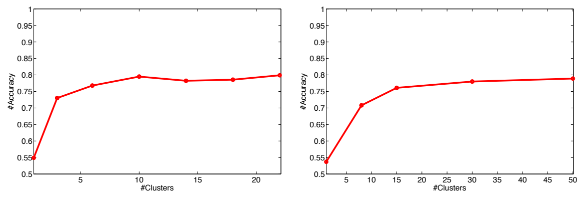
Cloud Scalability with Low Phone Overhead

Our remaining results report on the overhead to smartphones in adopting CSN, along with the ability for CSN to scale to large user populations.

We profile the computation and energy consumption of our CSN client on the Android Nexus One. We find resource consumption comparable to prior implementations of classification pipelines on phones (e.g., [22, 21]). As this overhead is not specific to CSN but found in any mobile sensing application we do not report further details. Overhead specific to CSN includes the transmission of sensor-data and the downloading of classifiers trained in the cloud. We find typical file sizes for our classification models are on the order of 1 $\sim$ 2 KBs, which means the cost of downloading classification models is minor. However, a significant cost to the phone can accrue when uploading sensor-data. To eliminate this cost our client implements an uploading strategy that waits until the phone is recharging before uploading data, effectively removing any burden to the battery.

The computational demands of computing the three CSN similarity dimensions range from being light-weight to very demanding. We quantify this by profiling the computational overhead for computing similarity networks for all people within *Everyday Activities*. This raw dataset is more than 400GB (mainly due to audio data). Using our CSN Mobile Cloud Infrastructure, configured with only one linux machine in the node pool the computational time for each variety of similarity is, $\approx$200 minutes, $\approx$9 minutes and $\approx$3 minutes respectively for sensor-data, lifestyle and physical similarity. The sensor-data similarity is the most costly of these three as it requires pairwise calculations between users.

Personalized models are trained by CSN for each user, however, this can become a bottle-neck. The workload of the Mobile Cloud Infrastructure increases with population size due to: i) the pairwise calculation of similarity between users and ii) each new user requires a new model to be trained. For this reason we designed our Mobile Cloud Infrastructure to effectively leverage a variable sized pool of cloud nodes, so additional nodes could be added when required. However, we experiment with an alternative approach that requires a simple extension to CSN. Instead of training a model for each user, users are first grouped together by clustering. Similarity networks are then built not between people but between these groups, with a model trained for each group. We investigate this trade-off and cluster people with k-means using the least computationally costly similarity dimensions, lifestyle and physical. By lowering the number of groups we can reduce the Mobile Cloud Infrastructure workload. This trade-off is seen in Figure 8. These figures illustrate how accuracy falls as the cluster size (the k in k-means clustering procedure) is reduced. Reducing the number of models dilutes the similarity between people in the cluster. Consequently, the model used by the entire group is less appropriate for everyone. Still, as the cluster number decreases the overhead to the mobile cloud is reduced, since fewer models need to be maintained. This approach allows us to regulate resource consumption by CSN irrespective of the size of the user population.



(a) Everyday Activities (b) Transportation

Figure 8. The accuracy of CSN when we group the users into different number of clusters under both datasets.

RELATED WORK

Applications that use mobile phone sensors have been steadily rising (e.g., [10, 12, 11, 6, 22]) and accurate classification of sensor-data is becoming increasingly important.

Researchers investigating sensor-enabled mobile phone applications frequently encounter the limits of activity classification. It is becoming obvious that conventional approaches that rely on supervised learning and carefully controlled training experiments are not suitable. In recognition researchers are considering alternatives. Current research directions point towards models that are adaptive and incorporate people in the process. Automatically broadening the classes recognized by a model is studied in [19] where active learning (where the learning algorithm selectively queries the user for labels) is investigated in the context of heath care. In SoundSense [20] a supervised classification process for a fixed category of sounds is augmented with a human-in-the-loop guided unsupervised process for learning novel sounds.

Research, such as [19, 20], focuses primarily on the individual to assist with classification. CSN leverages the user but also exploits communities of people (rather than just isolated individuals). Researchers are beginning to explore this new direction. [16] proposes a community-based technique that groups people based on their social clique and partitions their data accordingly to improve learning. However, the grouping is based on self-reported social-networks and is evaluated only under a location-based classification scenario. Community-guided Learning (CGL) [24] builds models of human behavior with training data provided by non-expert low-commitment mobile device users. CGL overcomes the challenge presented by noisy labels being introduced to the training process by using data similarity in combination with the crowd-sourced labels.

The potential for crowd-sourcing has been long recognized with interest in the area being established by Luis Von Ahn [27]. Now, commercially available systems including Amazon's Mechanical Turk [1] have made it simple to exploit the power of using thousands of people. The use in CSN of crowd-sourcing builds directly on these existing directions. We see CSN as part of an exciting area of hybrid systems that intelligently combine the effort of the masses towards a task that neither computers nor humans can perform on their own.

CONCLUSION

In this paper, we have proposed Community Similarity Networks (CSN), a classification system designed to address the population diversity problem. We demonstrated that the population diversity problem appears when using conventional techniques with as few as 50 users. CSN combines the crowd-sourcing of labels and sensor-data with multiple similarity networks that capture user similarities across different dimensions. The similarity networks guide the process of selectively merging data from different individuals to produce personalized classifiers at a much lower per-user cost. Finally, the generality, flexibility, and effectiveness of CSN are demonstrated using two distinct mobile sensing datasets.

REFERENCES

1. Amazon Mechanical Turk. `http://www.mturk.com`.
2. Google Nexus One. `http://www.google.com/phone/detail/nexus-one`.
3. Nike+. `http://www.apple.com/ipod/nike/run.html`.
4. SF-36.org. A Community for Measuring Health Outcoming using SF Tools. `http://www.sf-36.org/tools/SF36.shtml`.
5. Amazon Elastic Cloud Computing. `http://aws.amazon.com/ec2`.
6. T. Abdelzaher, Y. Anokwa, P. Boda, J. Burke, D. Estrin, L. Guibas, A. Kansal, S. Madden, and J. Reich. Mobiscopes for Human Spaces. *IEEE Pervasive Computing*, 6(2):20–29, 2007.
7. A. Andoni and P. Indyk. Near-optimal Hashing Algorithms for Approximate Nearest Neighbor in High Dimensions. *Communications of the ACM*, 51(1):117–122, 2008.
8. D. Ashbrook and T. Starner. Using GPS to Learn Significant Locations and Predict Movement Across Multiple Users. *Personal and Ubiquitous Computing*, 7(5):275–286, 2003.
9. C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, August 2006.
10. J. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava. Participatory Sensing. In *Workshop on World-Sensor-Web: Mobile Device Centric Sensor Networks and Applications*, WSW '06, pages 117–134, 2006.
11. A. T. Campbell, S. B. Eisenman, N. D. Lane, E. Miluzzo, and R. A. Peterson. People-centric Urban Sensing. In *Proceedings of the 2nd Annual International Wireless Internet Conference*, WICON '06, pages 18–es., 2006.
12. S. Consolvo, D. W. McDonald, T. Toscos, M. Y. Chen, J. Froehlich, B. Harrison, P. Klasnja, A. LaMarca, L. LeGrand, R. Libby, I. Smith, and J. A. Landay. Activity Sensing in the Wild: A Field Trial of UbiFit Garden. In *Proceedings of the 26th Annual SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1797–1806, 2008.
13. L. Dipietro, C. Caspersen, A. Ostfeld, and E. Nadel. A Survey for Assessing Physical Activity Among Older Adults. *Medicine and Science in Sports and Exercise*, 25:628–628, 1993.
14. J. Froehlich, T. Dillahunt, P. Klasnja, J. Mankoff, S. Consolvo, B. Harrison, and J. A. Landay. UbiGreen: Investigating A Mobile Tool for Tracking and Supporting Green Transportation Habits. In *Proceedings of the 27th International Conference on Human Factors in Computing Systems*, CHI '09, pages 1043–1052, 2009.
15. A. Kapoor and E. Horvitz. Experience Sampling for Building Predictive User Models: A Comparative Study. In *Proceedings of the 26th Annual SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 657–666, 2008.
16. N. D. Lane, H. Lu, S. B. Eisenman, and A. T. Campbell. Cooperative Techniques Supporting Sensor-based People-centric Inferencing. In *Proceedings of the 6th International Conference on Pervasive Computing*, Pervasive '08, pages 75–92, 2008.
17. N. D. Lane, E. Miluzzo, H. Lu, D. Peebles, T. Choudhury, and A. T. Campbell. A Survey of Mobile Phone Sensing. *Comm. Mag.*, 48:140–150, September 2010.
18. J. Lester, T. Choudhury, N. Kern, G. Borriello, and B. Hannaford. A Hybrid Discriminative/generative Approach for Modeling Human Activities. In *Proceedings of the International Joint Conference on Artificial Intelligence*, IJCAI '05, pages 766–772, 2005.
19. B. Longstaff, S. Reddy, and D. Estrin. Improving Activity Classification for Health Applications on Mobile Devices using Active and Semi-Supervised Learning. In *Proceedings of ICST Conference on Pervasive Computing Technologies for Healthcare*, PervasiveHealth '10, pages 1–7, 2010.
20. 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 *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services*, Mobisys '09, pages 165–178, 2009.
21. 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 *Proceedings of the 8th International Conference on Embedded Networked Sensor Systems*, Sensys '10, pages 71–84, 2010.
22. M. Mun, S. Reddy, K. Shilton, N. Yau, J. Burke, D. Estrin, M. Hansen, E. Howard, R. West, and P. Boda. PEIR, the Personal Environmental Impact Report, as a Platform for Participatory Sensing Systems Research. In *Proceedings of the 7th International Conference on Mobile Systems, Applications, and Services*, MobiSys '09, pages 55–68, 2009.
23. N. C. Oza and S. Russell. Online Bagging and Boosting. In *Proceedings of the 8th International Workshop on Artificial Intelligence and Statistics*, AISTAT '01, pages 105–112, 2001.
24. D. Peebles, H. Lu, N. Lane, T. Choudhury, and A. Campbell. Community-guided Learning: Exploiting Mobile Sensor Users to Model Human Behavior. In *Proceedings of the 24th National Conference on Artificial Intelligence*, AAAI '10, pages 1600–16006, 2010.
25. K. K. Rachuri, M. Musolesi, C. Mascolo, P. J. Rentfrow, C. Longworth, and A. Aucinas. Emotionsense: A Mobile Phones based Adaptive Platform for Experimental Social Psychology Research. In *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*, Ubicomp '10, pages 281–290, 2010.
26. M. Stikic, K. Van Laerhoven, and B. Schiele. Exploring Semi-supervised and Active Learning for Activity Recognition. In *Proceedings of the 12th IEEE International Symposium on Wearable Computers*, ISWC '08, pages 81–88, 2008.
27. L. von Ahn, B. Maurer, C. Mcmillen, D. Abraham, and M. Blum. reCAPTCHA: Human-Based Character Recognition via Web Security Measures. *Science*, pages 1465–1468, 2008.
28. C. T. Y. Rubner and L. J. Guibas. The Earth Movers Distance as a Metric for Image Retrieval. *IJCV*, 40(2):99–121, 2000.
29. Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma. Understanding Mobility based on GPS Data. In *Proceedings of the 10th ACM International Conference on Ubiquitous Computing*, Ubicomp '08, pages 312–321, 2008.
30. Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma. Mining Interesting Locations and Travel Sequences from GPS Trajectories. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 791–800, 2009.
31. X. Zhu and A. B. Goldberg. *Introduction to Semi-Supervised Learning*. Morgan and Claypool, 2009.