Hindawi Publishing Corporation
Mobile Information Systems
Volume 2016, Article ID 2097243, 10 pages
http://dx.doi.org/10.1155/2016/2097243

Research Article

Representation Learning from Time Labelled Heterogeneous Data for Mobile Crowdsensing

Chunmei Ma,[1] Qing Zhu,[2] Shuang Wu,[3] and Bin Liu[2]

[1]*School of Computer and Information Engineering, Tianjin Normal University, Tianjin, China*
[2]*School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China*
[3]*College of Computer Science, Zhejiang University, Hangzhou, China*

Correspondence should be addressed to Chunmei Ma; mcmxhd@163.com

Received 5 May 2016; Revised 18 July 2016; Accepted 4 August 2016

Academic Editor: Francisco Martínez

Mobile crowdsensing is a new paradigm that can utilize pervasive smartphones to collect and analyze data to benefit users. However, sensory data gathered by smartphone usually involves different data types because of different granularity and multiple sensor sources. Besides, the data are also time labelled. The heterogeneous and time sequential data raise new challenges for data analyzing. Some existing solutions try to learn each type of data one by one and analyze them separately without considering time information. In addition, the traditional methods also have to determine phone orientation because some sensors equipped in smartphone are orientation related. In this paper, we think that a combination of multiple sensors can represent an invariant feature for a crowdsensing context. Therefore, we propose a new representation learning method of heterogeneous data with time labels to extract typical features using deep learning. We evaluate that our proposed method can adapt data generated by different orientations effectively. Furthermore, we test the performance of the proposed method by recognizing two group mobile activities, walking/cycling and driving/bus with smartphone sensors. It achieves precisions of 98.6% and 93.7% in distinguishing cycling from walking and bus from driving, respectively.

1. Introduction

The smartphone has become extremely popular recently. The development of the smartphone with various sensors and powerful capabilities (computing, storage, and communication) motivates a popular computing and sensing paradigm, *crowdsensing*. As a result of the explosion of sensor-equipped mobile phones, we can sense the environment, infrastructure, and even social activities [1]. For example, in [2], the authors proposed using a smartphone with a built-in triaxial accelerometer to recognize physical activities, which can provide valuable information regarding an individual's degree of functional ability and life style. Besides using a single type sensor of smartphones, most often, we use multitypes sensors of the smartphone to obtain more comprehensive sensory data for a variety of applications [3, 4]. However, the sensory data from various data sources are usually heterogeneous, representing different granularity and diverse quality. In addition, the data are usually time labelled. Because of the two characteristics of sensory data, how to "understand" heterogeneous data with time labels correctly becomes a new challenge for data analyzing.

Some existing solutions would prefer to analyze a single type of data sensor by sensor [4–6]. For instance, in [5], authors focus specifically on traffic monitoring by using accelerometer, microphone, GSM radio, and/or GPS sensors of the smartphone to detect potholes, bumps, braking, and honking. They separately analyze data generated from each of these sensors one by one. The disadvantage of these methods is that they can only obtain a unidimensional characteristic of sensory data. Some other researchers proposed sensor fusion methods to learn the sensory data [3, 7]. This is accomplished by a feature extraction approach in which features from each sensor are computed independently. Then, the extracted features are integrated for fusion of information from multisensors. Although these methods derive comprehensive characteristics of sensory data, they cannot denote the internal relations of the heterogeneous

data. Furthermore, all of these methods never consider time labels of sensory data, which could lead to some typical features being neglected. For this reason, some works try to learn the time characteristics of sensory data using Hidden Markov Model (HMM) [8, 9]. But HMM-based algorithm can only obtain features of neighboring time points of sensory data rather than the overall time features.

Due to the limitations of existing methods of dealing with sensory data separately, we propose a new *representation learning* method of heterogeneous data with time labels to extract typical features using deep learning. In our model, multitype sensors are set to the same sampling frequency. Then, the sensory data are labelled by "sequence tag" according to sequencing the collection of data. Thus, the collected data and their sequence tags can be combined together as an integral feature. Then, such global data integration can be accepted as input by deep learning network. With multiple layers, deep learning is more powerful and flexible. It is able to combine many layers to generate an integrated feature. In crowdsensing, we believe that the integrated feature abstracted from multiple sensory data can well recognize a corresponding context. Besides, due to the sensory data tagged by time labels, we can learn the temporal knowledge from raw data as well in our model. In a word, we not only integrate heterogeneous data from multiple sensors, but also combine it with temporal information. We named the combination as *context fingerprint*.

In this paper, we propose and demonstrate our method to analyze sensory data in an overall view. We group all data collected at time t_0 from multiple sensor sources and their sequence tag T_0 as a vector. Suppose the vector generated at time t_0 is denoted by $\mathbf{x}_1'$ and the length of sampling window is τ. Then, we can get a vector $\mathbf{x}_2'$ with sequence tag T_1 at time $t_0 + \tau$ in the same way. Repeating this sampling process for n times, we can get the sample $\mathbf{X}'$, where $\mathbf{X}' = (\mathbf{x}_1', \mathbf{x}_2', \ldots, \mathbf{x}_n')$ is a matrix with n columns. Since the sensory data are from different granularity data sources, it is necessary to refine the raw sample $\mathbf{X}'$ by data preprocessing. With the preprocessing, we get same size sample $\mathbf{X}$ from $\mathbf{X}'$. The sample $\mathbf{X}$ will be set as an input for our deep learning model. Experimental results show that the context fingerprint reconstructed by deep learning can efficiently represent an invariant feature for a crowdsensing context. Our proposed method has the following innovative features: (i) it integrates the overall feature (*context fingerprint*) of raw data as input, (ii) it captures and learns, in addition to sensory data itself, the tagged time information of data and utilizes both of them for context inferring, and (iii) with the deep learning model, we do not have to do orientation correction; in other words, we need not care about the problem of phone orientation.

The main contributions of this paper are multifold, which include the following:

(1) We propose integrating features of multiple sensors and their sequence tags as an overall fingerprint for data analyzing in crowdsensing.

(2) We consider the factor of time information to improve the efficiency of mobile activities recognizing.

(3) With deep learning model, we make the smartphone data analysis independent of smartphone orientation.

(4) We evaluate our proposed scheme with real data collected from multiple sensors of smartphone.

The rest of the paper is structured as follows. Section 2 presents a brief overview of related works. In Section 3 we introduce the basic architecture of the time-delay multilayer perception model. We explain network training in Section 4. Section 5 evaluates our schemes in human mobile activity inferring by the data we collected in realistic scenarios, and Section 6 summarizes this paper.

2. Related Work

Due to the popularity of the smartphone and multitypes sensors with which it is equipped, there is a growing interest in mobile application researches [10–13]. They leverage the sensors of smartphone to sense our physical environment or individuals' physiological parameters and so forth. The sensory data are always multimodal, representing different granularity and diverse quality. In order to well understand the potential meanings of the collected data, many researchers devote themselves to learning representation of the data that make it easier to extract useful information. In [14], authors proposed calculating resultant vectors of accelerometer, gyroscope, and magnetometer of smartphones, respectively. Then, individual defined thresholds of the three sensors are used for fall detection. The independent representation mechanism of the sensory data of multitypes sensors is used in [15, 16] as well. Although these methods are lightweight, they can only obtain a unidimensional characteristic of sensory data and cannot form a discriminant feature. For example, an accelerometer for downstairs and upstairs has similar change characteristics.

In view of the insufficiency of the independent representation mechanism of the sensory data, some researches put forward sensor fusion based schemes to learn representation of sensory data. Sensor fusion is combining of sensory data or data derived from disparate source such that the resulting information has less uncertainty than what would be possible when these sources were used individually [17]. With a fusion process, we can get more accurate and more dependable result from the disparate raw data source. For example, in [18], in order to improve localization service, the author manipulates at least four sensors including microphone, camera, WiFi radio, and accelerometer. The aim is to combine multiple features for reliable localization service. In [19], authors presented a hierarchical algorithm for the heterogeneous data representation. In the lower level, it extracts feature vectors of accelerometer and microphone for the motion and environment. In the higher level, it combines the extracted two features to get an integration feature for human activity recognition. Similarly, in [20], Zeng et al. proposed a dynamic heterogeneous sensor fusion framework to incorporate various sensory data. It learned the weights of sensors to form an integrated feature for activity recognition. The drawback of these schemes is that they only simply integrate heterogeneous data, which do not consider

the influence of different sensors. In addition, some of the works have to implement coordinate reorientation of sensors to obtain the meaningful sensory data that indicate physical activities of objects [3, 6, 21], which increases the complexity of system implementations.

Since the sensory data may present different temporal characteristics for various sensing events, some studies try to explore the time characteristic in learning sensory data. To the best of our knowledge, the method used to analyze the temporal characteristics of sensory data is Hidden Markov Model-based algorithm [22, 23]. However, HMM-based algorithm requires prior knowledge to define its structure, which limits its feasibility. In addition, it analyzes transfer features of data of neighboring time points such that it cannot extract an integrated time feature of sensory data.

3. Model Review

3.1. Why to Choose Deep Learning. Theoretical results suggest that, for a complicated extraction process, the results can be further improved by applying a "deeper" structure [24, 25]. In this paper, we propose learning representation of the sensory data tagged by time labels to extract typical features using deep learning, which is a generative model that consists of multiple layers of hidden stochastic latent variables of feature. There are two advantages of our method. First of all, we consider temporal information in our algorithm for analyzing the time labelled sensory data. Secondly, different from traditional ways that think each sensor presents one feature (subfeature) separately, we believe that all sensors with which smartphone is equipped will represent a unique feature together corresponding to a context. Namely, we integrate all of the subfeatures as an overall feature, which is the *context fingerprint* we named before. We plan to explain more details of these two considerations as follows.

3.1.1. Time Information of Sensory Data. Usually, the sensory data generated by smartphone is time labelled. For example, if we sample sensory data at the time t_0 with sampling window length of τ, then we can collect data sequences like this: $\{\mathbf{x}_i', y^{(i)},\ i = 1, 2, \ldots, N\}_{[t_0+(i-1)\tau]}$, where $\mathbf{x}_i'$ denotes sample data that is sequenced according to the collecting order. Thus, $\mathbf{x}_i'$ can be time labelled by "sequence tag" T_i. $y^{(i)}$ is the class label that $\mathbf{x}_i'$ belongs to. For mobile crowdsensing, time labels are valuable information that can be used to extract changing characteristics of the sensory data with time. The time labels should be considered in the algorithm design as mucg possible as we can. However, existing methods usually cannot deal with the temporal information effectively. In this paper, we introduce a deep learning model to extract typical features from time labelled sensory data.

3.1.2. Data Integration. In order to achieve typical features extraction using deep learning, it is necessary to determine the data integration which is as input data of our deep learning. Data integration is to combine data residing at different sources and to provide users with a unified view of these data [26]. As discussed before, we combine all kinds of sensory data and sequence tags together to obtain a *context fingerprint*. In our model, rather than considering different sensors as different subfeatures separately, the data integration representation is an invariant feature, namely, the *context fingerprint*. For example, if there are four kinds of sensors, we can manipulate accelerometer, gyroscope, magnetometer, and compass. There must be a special fingerprint vector generated from a special context $\mathbf{c}$ and a time point it corresponds to. For each context $\mathbf{c}$, there must be one and only one $\mathbf{f}$ corresponding to it. And the fingerprint vector $\mathbf{c}$ is orientation invariant.

$$\mathbf{f} = \{Accx, Accy, Accz, Gyrx, Gyry, Gyrz, Magx, Magy, Magz, Com, T\}. \quad (1)$$

3.2. The Deep Learning Model. In order to extract typical features from sensory data with time labels, we use the deep learning model which consists of many layers. Up to now, there are various deep learning architectures, such as convolutional neural networks, recursive neural networks, and deep belief networks. The convolutional neural network (CNN) is suitable for processing visual and other two-dimensional data [27]. The recursive neural network (RNN) uses a tensor-based composition function and its structure is very complex [28]. RNN is suitable for natural language processing [29]. The deep belief networks can be efficiently trained in an unsupervised, layer-by-layer manner, where the layers are made of Restricted Boltzmann Machine (RBM) [30]. Thus, DBN can greatly reduce the training samples. Through the comparative analysis, we select the deep belief network (DBN) as our deep learning model. In this paper, we use four-layer DBN structure which contains a visible layer and three hidden layers. The four layers form three RBM groups as shown in Figure 1. Suppose the input data vector for our network is m-dimensional, which is collected and integrated from accelerometer, gyroscope, magnetometer, compass, and sequence tags (in this paper we only consider 4 sensors at all; for more sensors the network could be enlarged in the same way).

There are l_1 units in the visible layer of our DBN, which is responsible for accepting input samples. The samples data are time labelled. Suppose each sample contains n sampling time points; then it is easy to know that each input sample $\mathbf{X}$ is a matrix with $m \times n$, $\mathbf{X} = [x_{ij}]_{m\times n}$. The visible layer should accept every element of one sample as shown in Figure 1. Until now, the number l_1, which is linearly correlated with both m and n, can be calculated as $l_1 = mn$. As we mentioned before, the sensory data are from different granularity data sources. Thus, the samples we used here are not the raw data collected by smartphone but have been preprocessed. The data preprocessing is further discussed in this paper. The following three layers are hidden layers. The lowest hidden layer has h_1 hidden units, the next one has h_2, and the top layer has h_3 hidden units. The hidden units of the first RBM get inputs from the visible layer and then forward their well-trained outputs to the second RBM. At this time, the units of the first hidden layer become visible units in the second RBM. This process will be repeated until the top layer hidden

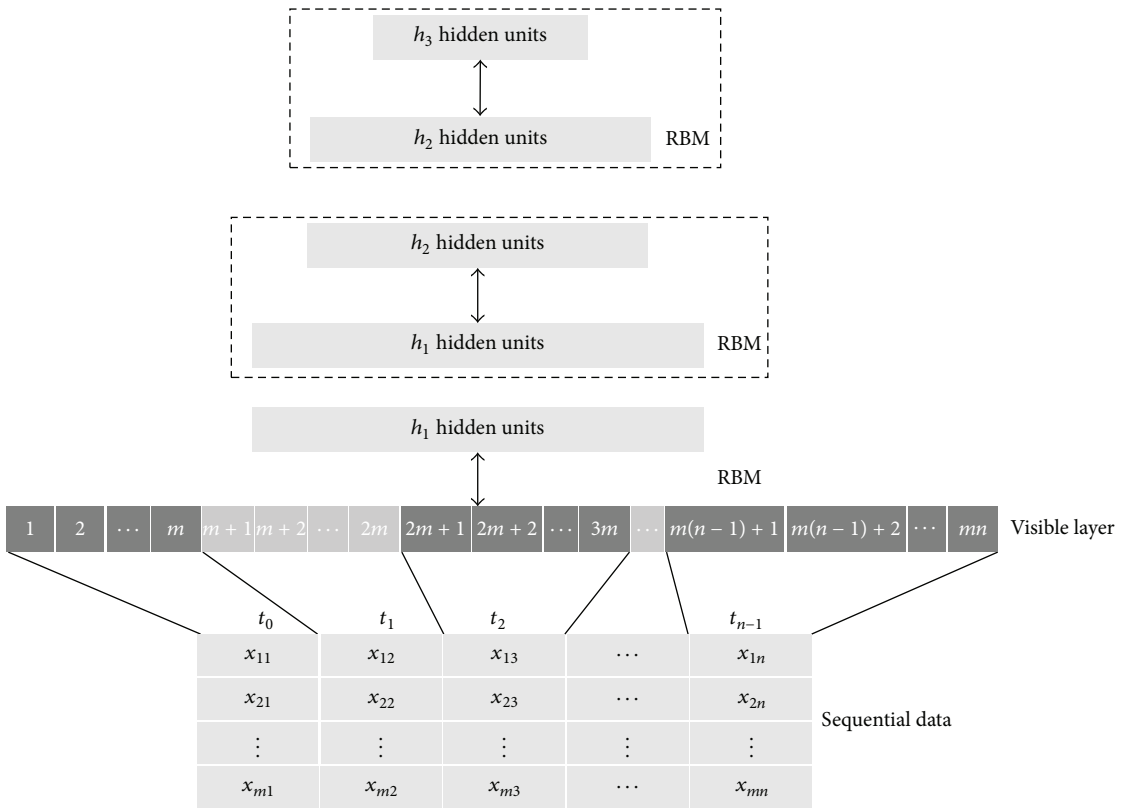


Figure 1: The architecture of the deep belief network for mobile crowdsensing. It contains four layers. The input data is a vector that is collected and integrated from accelerometer, gyroscope, magnetometer, compass, and time sequence.

units are determined. The number of each hidden layer unit should be carefully chosen, and we can tune and search an appropriate one by experiments.

3.3. Data Sampling and Preprocessing. In this subsection, we explain how to define and get samples from the raw data we collected from smartphones. As we discussed before, the raw data is m-dimensional that contains sensory data and sequence tags. For every sampling time point, one kind of an m-dimensional vector would be generated. In our model, we select successive sequences data as our training or test sample rather than only one sampling point, because only long enough sequential data can capture a pattern; in other words, only a successive sampling sequence can represent a special context correctly. Now, the problem is how to explore an appropriate length of time frame of sampling, n, as we discussed before.

In our model, we make the length of sample n related to a specific situation, such as human daily activity recognition [2, 14, 31] or transportation mode recognition [32–34]. For different application purpose, we choose different length of time frame n. For example, in [33], it is necessary to determine whether the people are on the bus. So, we should use longer sensory data to achieve this objective; 20~120 seconds may be an appropriate length for sampling time frame according to our experiments. However, for recognizing human daily activity, such as cycling, 5~8 seconds is enough. A reasonable value of n of different scenarios will be selected from experiments. As we discussed before, there are n times samplings for each raw sample $\mathbf{X}'$; $\mathbf{X}' = [x'_{ij}]_{m\times n}$. Since the granularity of the raw sample is different, we do not input the raw sample $\mathbf{X}'$ into our model directly. Actually, we propose doing preprocessing for $\mathbf{X}'$ and getting the refined sample $\mathbf{X} = [x_{ij}]_{m\times n}$ for training and testing as follows:

$$x_{ij} = \frac{x'_{ij} - \overline{x'_i}}{\sigma_i}, \quad i = 1, 2, \ldots, m, \ \ j = 1, 2, \ldots, n, \tag{2}$$

where $\overline{x'_i} = (1/n)\sum_{j=1}^{n} x'_{ij}$ and σ_i is the variance of the ith row of the raw sample $\mathbf{X}'$. The refined sample $\mathbf{X} = [x_{ij}]_{m\times n}$ is smooth and it can also represent a *context fingerprint*.

4. The Deep Belief Network Training

After preprocessing, the samples with size of $m \times n$ could be accepted by the visible layer of DBN. However, different from image data, which is pixel matrix, our samples are time-delay data sequences. In order to integrate the sensory data, forming a typical feature, the deep belief network should be well trained. Therefore, our purpose is to find the parameters of DBN to minimize the network errors. This procedure is divided into two phases: (1) pretraining phase and (2) fine-tuning phase. In the following sections, we will describe the two phases in detail.

4.1. Pretraining Phase. As shown in Figure 1, our DBN consists of three RBM groups, which are separated from

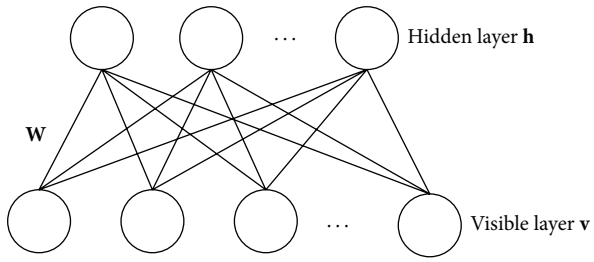


Figure 2: The model of Restricted Boltzmann Machine (RBM).

each other. Therefore, we train each RBM group individually. For each RBM, it is an undirected graph that consists of two layers: visible layer used to denote the observations and hidden layer used to denote the feature detectors. **W** is the weight of the connection between the visible layer and hidden layer. The structure of RBM is shown in Figure 2.

Let vectors **v** and **h** denote the state of visible unit and hidden unit, in which v_i denotes the state of the ith visible unit and h_j denotes the state of the jth hidden unit. For a given state of $(\mathbf{v}, \mathbf{h})$, the energy of the joint configuration in RBM is

$$E(\mathbf{v}, \mathbf{h} \mid \theta) = -\sum_{i \in \mathbf{v}} a_i v_i - \sum_{j \in \mathbf{h}} b_j h_j - \sum_{i \in \mathbf{v}} \sum_{j \in \mathbf{h}} v_i w_{ij} h_j, \tag{3}$$

where $\theta = \{w_{ij}, a_i, b_j\}$ is the parameter that needs to be trained in RBM. w_{ij} is the weight of the connection between the ith visible unit and jth hidden unit and a_i and b_j are their bias. Based on the energy function, the joint probability distribution of $(\mathbf{v}, \mathbf{h})$ is given as

$$P(\mathbf{v}, \mathbf{h} \mid \theta) = \frac{e^{-E(\mathbf{v}, \mathbf{h} \mid \theta)}}{Z(\theta)}, \tag{4}$$

where $Z(\theta) = \sum_{\mathbf{v}, \mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h} \mid \theta)}$ is the partition function. For a practical problem, the aim of the pretraining algorithm is to determine the distribution of the observation data $P(\mathbf{v} \mid \theta)$, that is, the marginal probability of $P(\mathbf{v}, \mathbf{h} \mid \theta)$. It can be given as

$$P(\mathbf{v} \mid \theta) = \frac{1}{Z(\theta)} \sum_{\mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h} \mid \theta)}. \tag{5}$$

Since the energy of a training sample could be lowered by raising the probability of the sample, the optimal parameter θ can be computed by maximizing the likelihood function of $P(\mathbf{v} \mid \theta)$. It can be computed by taking the derivation of the likelihood function of $P(\mathbf{v} \mid \theta)$ with respect to the parameters:

$$\begin{aligned} \frac{\partial \log P(\mathbf{v} \mid \theta)}{\partial w_{ij}} &= \left\langle v_i h_j \right\rangle_{\text{data}} - \left\langle v_i h_j \right\rangle_{\text{model}} \\ \frac{\partial \log P(\mathbf{v} \mid \theta)}{\partial a_i} &= \left\langle v_i \right\rangle_{\text{data}} - \left\langle v_i \right\rangle_{\text{model}} \\ \frac{\partial \log P(\mathbf{v} \mid \theta)}{\partial b_j} &= \left\langle h_j \right\rangle_{\text{data}} - \left\langle h_j \right\rangle_{\text{model}}, \end{aligned} \tag{6}$$

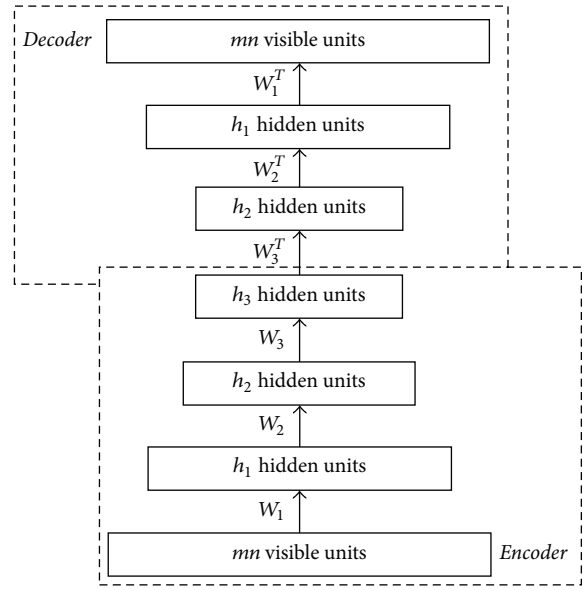


Figure 3: The model of unrolling deep belief network (DBN).

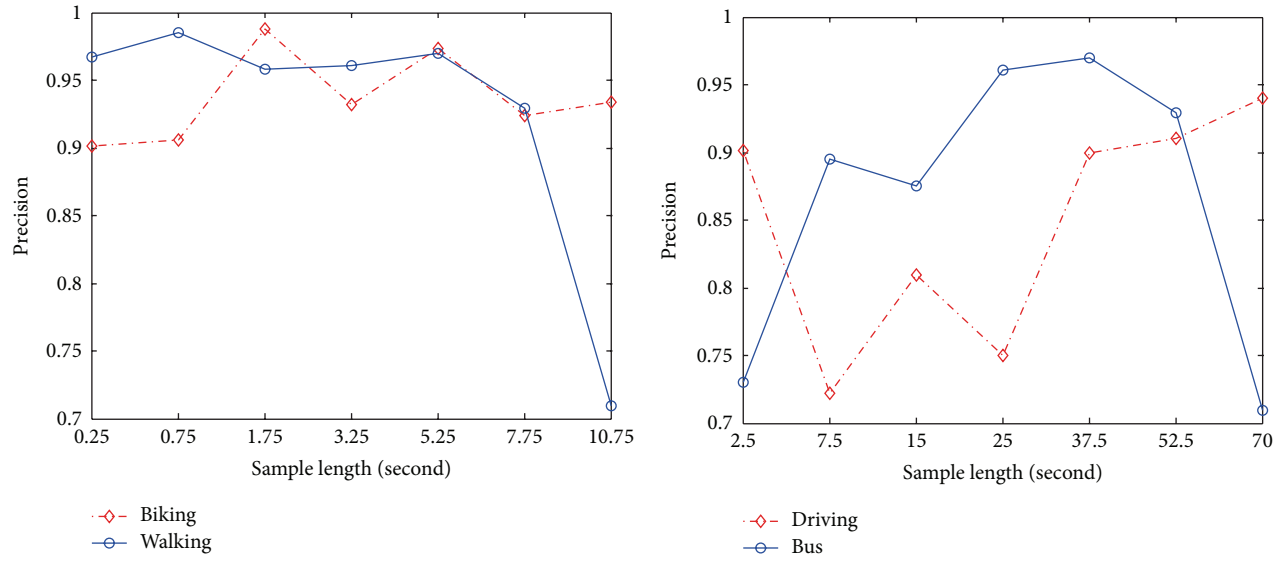
where $\langle \cdot \rangle_{\text{data}}$ is the expectation of the product of the parameters and observed data and $\langle \cdot \rangle_{\text{model}}$ is the expectation for the model observations that is generated according to the model. When the training data and generated data are similar, we obtain the optimal performance. Thus, the parameters can be updated by

$$\begin{aligned} \Delta w_{ij} &= \epsilon \left(\left\langle v_i h_j \right\rangle_{\text{data}} - \left\langle v_i h_j \right\rangle_{\text{model}} \right) \\ \Delta a_i &= \epsilon \left(\left\langle v_i \right\rangle_{\text{data}} - \left\langle v_i \right\rangle_{\text{model}} \right) \\ \Delta b_j &= \epsilon \left(\left\langle h_j \right\rangle_{\text{data}} - \left\langle h_j \right\rangle_{\text{model}} \right), \end{aligned} \tag{7}$$

where ϵ is a learning rate. Through experimental test, ϵ is set at 0.01. After the first RBM is well trained, the hidden units in this RBM become visible unit for learning the second RBM. The layer-to-layer learning will repeat until the last RBM is well trained. At this time, we obtain coarse grain optimal values of the parameters. To further improve the result, a fine-turning process is implemented in the next phase.

4.2. Fine-Tuning Phase. The phase described above is a bottom to top unsupervised learning process to achieve the network pretraining. After that, the models unfold (as shown in Figure 3) to produce encoder and decoder network. Then, we implement a fine-turning process to optimize the parameters of the deep belief network. In this step, the process is a top to bottom supervised learning.

In order to achieve fine-tuning of the network, in this paper, we use backpropagation (BP) method, which calculates gradient descent of the mean-squared error as a function of the weights [35]. Specifically, backpropagation procedure performs two stages through the unrolling network, forward and backward. During the forward stage, we forward the training data to the input of the network and calculate the difference between the inferred hidden unit and the learned



(a) Cycling versus walking

(b) Driving versus bus

FIGURE 4: Precision statistics of seven different sample lengths n. (a) The precision statistics of walking/cycling with sample length as 0.75 s, 1.75 s, 3.25 s, 5.25 s, 7.75 s, and 10.75 s. (b) The precision statistics of driving/bus with sample lengths 2.5 s, 7.5 s, 15 s, 25 s, 37.5 s, 52.5 s, and 70 s.

hidden unit. In this way, an error can be computed by comparing output with desired output. For the backward stage, we can evaluate the derivatives of the error function with respect to the weights and then use them to adjust weights among all the connections. This process will repeat many times for each of the training data until the network converges. During the whole process, the initial weights are the same weights that are well trained in pretraining phase.

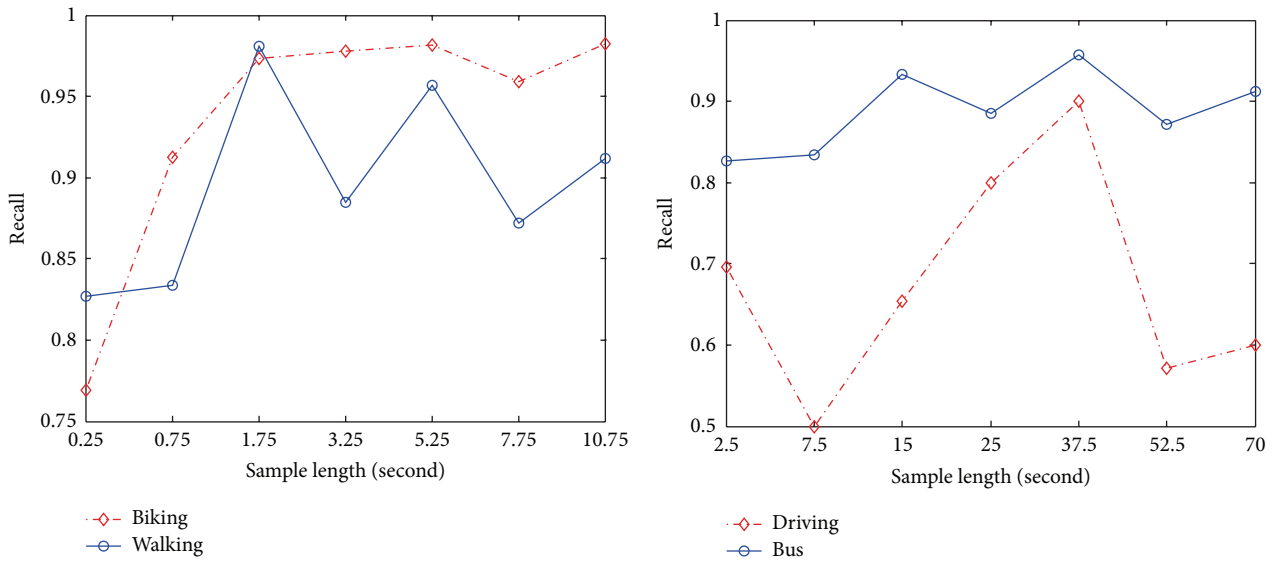
5. Evaluation

5.1. Sample Sets. Analyzing data generated by multiple sensors of smartphone to design and develop a mobile application is not the goal of this paper. The most important thing in this paper is proposing and demonstrating a novel solution for integrating and analyzing multiple time labelled sensory data effectively. As mentioned in Section 3.1, we plan to integrate four kinds of sensors, accelerometer, gyroscope, magnetometer, and compass. We can start our evaluation by explaining how to do preprocessing firstly. The testing sample sets we gathered are corresponding to two groups of mobile activities, *walking/cycling* and *driving private car/taking bus*. We have six volunteers to collect data, including 4 males and 2 females. In two weeks, the volume of sensory data they collected is over 180 hours. They carry six different Android smartphones (different handset makers) equipped with the four sensors we mentioned before. The sampling frequency is 4 Hz. And we do never restrict smartphone orientation during data collecting. It means all volunteers could do sample with the most comfortable gestures as they prefer. Usually, the orientation for woman carrying phone is different from man. But the only thing is that they have to keep one gesture constantly during one sampling period. It means that the gesture with smartphones cannot change during the sampling period. And we do the training and testing with the cross validation method. We grouped the samples into four parts and randomly choose three of them as the training set. The left part is testing set.

5.2. Experimental Results

5.2.1. Running DBN with Different Sample Length n. We do all the tests with two groups of human mobile activities, *walking/cycling* and *driving/bus*. Based on the integrated features, then we use an SVM classifier to distinguish the activities. Precision and recall are the most widely used quality measurements. We observe and compare the precision and recall when tuning sample length n. Figures 4 and 5 compare the precision and recall with different value of n. We firstly test our model on *walking/cycling* testing set with sample lengths 0.25 s (only one sampling point), 0.75 s, 1.75 s, 3.25 s, 5.25 s, 7.75 s, and 10.75 s, respectively. In classifying these two kinds of mobile activities from each other, we firstly define *cycling* as positive class (1) and *walking* as negative class (0). So, the precision of recognizing *cycling* can be calculated. Then, we change *walking* as positive class and get the precision for classifying *walking* as shown in Figure 4(a). From Figure 4(a), we can find that the integrated features achieve an excellent performance with different sample lengths. And a sample length around 1.75 s~5.25 s achieves a stable precision above 92% and with a peak value 98%.

The testing for *driving/bus* works acts in the same way. However, for recognizing *driving* from *bus*, we need to enlarge the sample length because only long enough sample



(a) Cycling versus walking

(b) Driving versus bus

FIGURE 5: Recall statistics of seven different sample lengths n. (a) The recall statistics of walking/cycling with sample length as 0.75 s, 1.75 s, 3.25 s, 5.25 s, 7.75 s, 10.75 s. (b) The recall statistics of driving/bus with sample lengths 2.5 s, 7.5 s, 15 s, 25 s, 37.5 s, 52.5 s, 70 s.

can capture the characteristic of driving or taking bus. Therefore, in searching an appropriate sample length, we choose the candidate sample length as 2.5 s, 7.5 s, 15 s, 25 s, 37.5 s, 52.5 s, and 70 s, respectively. As shown in Figure 4(b), the sample length n plays a more important role in distinguishing *driving/bus* than *cycling/walking*. An effective range of sample length is 37.5 s~52.5 s. Too small or too large sample length would never achieve a satisfying precision. Actually, not only n but also the number of the hidden layer unit for the two tests is also different. We choose $h_1 = 100$, $h_2 = 60$, and $h_3 = 3$ for the experiment of recognizing *walking/cycling*. For *driving/bus*, we choose $h_1 = 900$, $h_2 = 300$, and $h_3 = 4$. For the visible unit, because the sampling frequency is 4 Hz, $l_1 = 1650(11 * 37.5 * 4)$ if we select 37.5 s as the sample length.

The recall of statistics is shown in Figure 5, which reveals almost the same phenomenon with different sample lengths. However, in the test of classifying driving and bus, it is much easier to distinguish bus than to distinguish driving from testing sample set. As shown in Figures 4(b) and 5(b), both of the two quality measurements, precision and recall, achieve a higher result in distinguishing *bus* from our testing samples. One possible explanation is that bus usually runs more regularly than *driving* a private car or taxi.

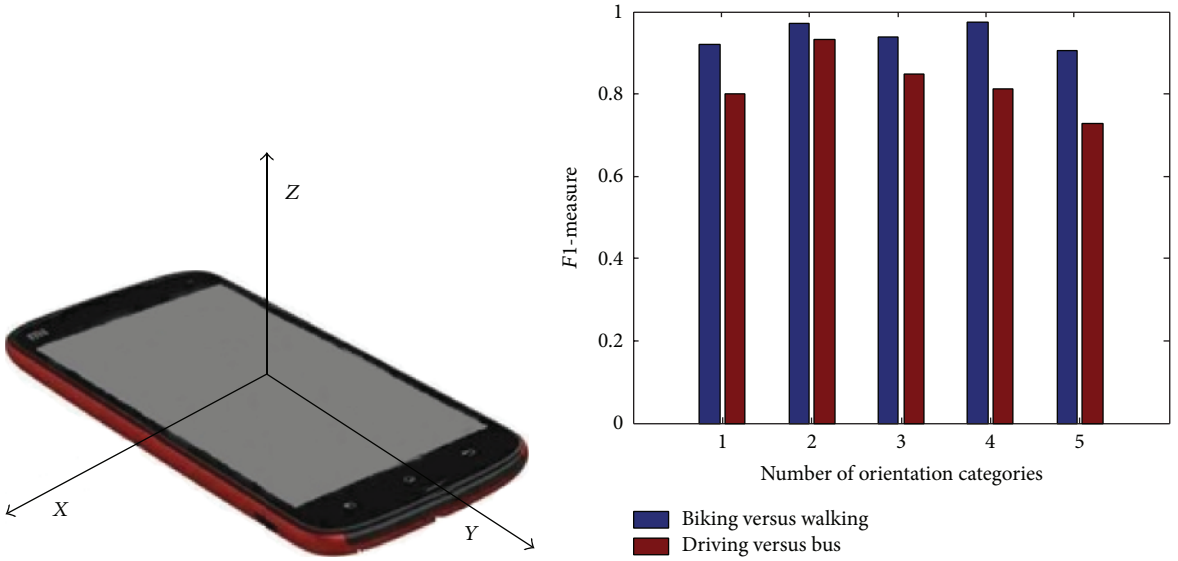
5.2.2. The Overall Performance of DBN. To evaluate the overall performance of our DBN model, we compare DBN with an Activity Recognition System (ARS) with mobile phones [36]. ARS manipulated three kinds of sensors, accelerometer, magnetometer, and gyroscope. There are some differences between DBN and the ARS. Firstly, ARS gets a constant sample length, which is 2 seconds but with a sampling frequency of 50 Hz. Therefore, there are 100 sampling points. Then, ARS calculates mean of temporal differences for the 100 sampling data sets named *Intensity*, where $Intensity := (1/100)\sum_{t=1}^{100}((x'_t - x'_{t-1})/\tau)$. For the second, ARS's network has 9 fully connected neurons in the input layer, which is less than our DBN model.

TABLE 1: Experiment performance compare ($F1$-measurement).

Activity	ARS	DBN
Walking	92.35%	96.36%
Cycling	75.96%	97.77%
Bus	70.10%	93.75%
Driving	58.33%	90.10%
Average	74.19%	94.50%

We conduct experiments on classifying the four mobile activities of walking, cycling, driving private car, and taking bus using DBN and ARS. For each activity, ARS uses the same sampling length, whereas DBN selects different sampling lengths for different activities. As tested above, DBN uses 2 s, 5 s, 40 s, and 35 s sampling length with 4 Hz sampling frequency, respectively, for classifying walking, cycling, driving private car, and taking bus. In order to evaluate performance overall, we introduce $F1$-score as a new quality measurement [37]. It is a measure of a test's accuracy. $F1$-score is defined as $2 * precision * recall/(precision + recall)$. The $F1$-score results of ARS and DBN are shown in Table 1.

As shown in Table 1, we can see that the integrated features extracted from our DBN perform better than ARS in recognizing all of the four mobile activities, especially for the last three activities. The reason is that the long enough temporal sequential data can capture more features of some mobile activities. Besides, DBN also considers time labels of the sensory data. Although ARS has higher sampling frequency, in a short time, there will be no significant change of the characteristic of sensory data. And it will increase the computational load.



(a) Phone orientation

(b) $F1$-measure of different number of orientations

FIGURE 6: The robustness testing for smartphone orientation invariance.

5.2.3. Evaluation of Orientation Invariance. Some of the sensors with which smartphone is equipped are orientation related as shown in Figure 6. For different orientations, data will present differently in the same context. For example, during walking, different gestures of carrying smartphone will generate different data records of accelerometer. Traditionally, in dealing with those different representations of data, we have to determine the orientation by some rules firstly [5]. However, in our model, we do not need to do this kind of job of adjusting orientations. It can learn context's different data representations for the same context effectively. In other words, it is orientation invariant.

In evaluating the orientation invariance of our method, we test it with a number of orientations and observe the corresponding performances. As discussed before, we never restrict smartphone orientation during sampling. And all volunteers could do sample with the most comfortable gestures as they prefer. Actually, there are totally five gestures volunteers used, putting their phones in coat pockets, trouser pockets, backpacks, lady's handbags, and their hands. We firstly do test on the testing data collected from only one gesture of phone carrying. Then, we increase the category of gestures to renew our testing. After that we use the extracted features to observe the varieties of performance. We also do the same experiments on cycling/walking and driving/bus. As shown in Figure 6(b), our method achieves stable results on both of the two classifying experiments. Despite data with multiple orientations, it could also be recognized with good performance.

6. Conclusion

In this paper, we propose and demonstrate a novel model to analyze multiple time labelled sensory data using deep learning in an overall view. Our method tries to integrate feature of each sensor into a combined feature (context fingerprint) and then set it as input for the DBN model. Besides, it captures and learns not only the sensory data itself but also tagged time information of data and utilizes both of them to do context inferring. When analyzing data extracted using our method, we even need not care about the orientation of smartphone during sampling. We demonstrate our model with capturing a reliable fingerprint of four sensory data sets in inferring two categories of mobile activities, *walking/biking* and *driving/bus*.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

This work was supported in part by the Application Foundation and Advanced Technology Research Project of Tianjin (no. 15JCQNJC01400) and the Scientific Special Commissioner Project of Tianjin (no. 15JCTPJC58300).

References

[1] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39, 2011.

[2] A. M. Khan, Y.-K. Lee, S. Y. Lee, and T.-S. Kim, "Human activity recognition via an accelerometer-enabled-smartphone using Kernel discriminant analysis," in *Proceedings of the 5th International Conference on Future Information Technology (FutureTech '10)*, pp. 1–6, Busan, South Korea, May 2010.

[3] D. A. Johnson and M. M. Trivedi, "Driving style recognition using a smartphone as a sensor platform," in *Proceedings of the 14th IEEE International Intelligent Transportation Systems*

Conference (ITSC '11), pp. 1609–1615, Washington, DC, USA, October 2011.

[4] C.-W. You, N. D. Lane, F. Chen et al., "CarSafe app: alerting drowsy and distracted drivers using dual cameras on smartphones," in *Proceedings of the 11th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '13)*, pp. 13–26, ACM, Taipei, Taiwan, June 2013.

[5] P. Mohan, V. N. Padmanabhan, and R. Ramjee, "Nericell: rich monitoring of road and traffic conditions using mobile smartphones," in *Proceedings of the 6th ACM Conference on Embedded Networked Sensor Systems (SenSys '08)*, pp. 323–336, November 2008.

[6] Y. Wang, J. Yang, H. Liu, Y. Chen, M. Gruteser, and R. P. Martin, "Sensing vehicle dynamics for determining driver phone use," in *Proceedings of the 11th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '13)*, pp. 41–54, Taipei, Taiwan, June 2013.

[7] R. K. Ganti, S. Srinivasan, and A. Gacic, "Multisensor fusion in smartphones for lifestyle monitoring," in *Proceedings of the International Conference on Body Sensor Networks (BSN '10)*, pp. 36–43, Singapore, June 2010.

[8] H. Kuehne, J. Gall, and T. Serre, "An end-to-end generative framework for video segmentation and recognition," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV '16)*, pp. 1–8, Lake Placid, NY, USA, March 2016.

[9] H. Xu, Y. Lee, and C. Lee, "Activity recognition using Eigenjoints based on HMM," in *Proceedings of the 12th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI '15)*, pp. 300–305, IEEE, Goyang, Reoublic of Korea, October 2015.

[10] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan, "The Pothole Patrol: Using a mobile sensor network for road surface monitoring," in *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services*, pp. 29–39, Breckenridge, Colo, USA, June 2008.

[11] S. K. Vashist, E. M. Schneider, and J. H. Luong, "Commercial smartphone-based devices and smart applications for personalized healthcare monitoring and management," *Diagnostics*, vol. 4, no. 3, pp. 104–128, 2014.

[12] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan, "The pothole patrol: using a mobile sensor network for road surface monitoring," in *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services*, pp. 29–39, ACM, Breckenridge, Colo, USA, June 2008.

[13] J. Cherian, J. Luo, H. Guo, S.-S. Ho, and R. Wisbrun, "Poster: Parkgauge: gauging the congestion level of parking garages with crowdsensed parking characteristics," in *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, pp. 395–396, ACM, Seoul, Republic of Korea, November 2015.

[14] S. Madansingh, T. A. Thrasher, C. S. Layne, and B. Lee, "Smartphone based fall detection system," in *Proceedings of the 15th International Conference on Control, Automation and Systems (ICCAS '15)*, pp. 370–374, Busan, South Korea, October 2015.

[15] J.-H. Hong, B. Margines, and A. K. Dey, "A smartphone-based sensing platform to model aggressive driving behaviors," in *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems (CHI '14)*, pp. 4047–4056, Toronto, Canada, April 2014.

[16] Q. T. Huynh, U. D. Nguyen, L. B. Irazabal, N. Ghassemian, and B. Q. Tran, "Optimization of an accelerometer and gyroscope-based fall detection algorithm," *Journal of Sensors*, vol. 2015, Article ID 452078, 8 pages, 2015.

[17] Sensorfusion, http://en.wikipedia.org/wiki/Principal_component_analysis.

[18] M. Azizyan, I. Constandache, and R. Roy Choudhury, "SurroundSense: mobile phone localization via ambience fingerprinting," in *Proceedings of the 15th Annual ACM International Conference on Mobile Computing and Networking (MobiCom '09)*, pp. 261–272, ACM, Beijing, China, September 2009.

[19] G. Filios, S. Nikoletseas, C. Pavlopoulou, M. Rapti, and S. Ziegler, "Hierarchical algorithm for daily activity recognition via smartphone sensors," in *Proceedings of the 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT '15)*, pp. 381–386, IEEE, Milan, Italy, December 2015.

[20] M. Zeng, X. Wang, L. T. Nguyen, P. Wu, O. J. Mengshoel, and J. Zhang, "Adaptive activity recognition with dynamic heterogeneous sensor fusion," in *Proceedings of the 2014 6th International Conference on Mobile Computing, Applications and Services (MobiCASE '14)*, pp. 189–196, IEEE, November 2014.

[21] C. Song, J. Wu, M. Liu, H. Gong, and B. Gou, "RESen: sensing and evaluating the riding experience based on crowdsourcing by smart phones," in *Proceedings of the 8th International Conference on Mobile Ad Hoc and Sensor Networks (MSN '12)*, pp. 147–152, Chengdu, China, December 2012.

[22] T. V. Duong, H. H. Bui, D. Q. Phung, and S. Venkatesh, "Activity recognition and abnormality detection with the switching hidden semi-Markov model," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, vol. 1, pp. 838–845, IEEE, San Diego, Calif, USA, June 2005.

[23] L. Xie, S.-F. Chang, A. Divakaran, and H. Sun, "Unsupervised discovery of multilevel statistical video structures using hierarchical hidden Markov models," in *Proceedings of the International Conference on Multimedia and Expo (ICME '03)*, vol. 3, pp. III-29–III-32, Baltimore, Md, USA, July 2003.

[24] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[25] I. Arel, D. C. Rose, and T. P. Karnowski, "Deep machine learning-a new frontier in artificial intelligence research," *IEEE Computational Intelligence Magazine*, vol. 5, no. 4, pp. 13–18, 2010.

[26] M. Lenzerini, "Data integration: a theoretical perspective," in *Proceedings of the 21st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS '02)*, pp. 233–246, ACM, Madison, Wis, USA, June 2002.

[27] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[28] R. Socher, A. Perelygin, J. Y. Wu et al., "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP '13)*, pp. 1631–1642, Citeseer, October 2013.

[29] R. Socher, C. C.-Y. Lin, C. D. Manning, and A. Y. Ng, "Parsing natural scenes and natural language with recursive neural networks," in *Proceedings of the 28th International Conference on Machine Learning (ICML '11)*, pp. 129–136, Bellevue, Wash, USA, June 2011.

[30] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[31] Q. Zhu, Z. Chen, and Y. C. Soh, "Smartphone-based human activity recognition in buildings using locality-constrained linear coding," in *Proceedings of the IEEE 10th Conference on Industrial Electronics and Applications (ICIEA '15)*, pp. 214–219, IEEE, Auckland, New Zealand, June 2015.

[32] S. Hemminki, P. Nurmi, and S. Tarkoma, "Accelerometer-based transportation mode detection on smartphones," in *Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems*, p. 13, ACM, 2013.

[33] P. Zhou, Y. Zheng, and M. Li, "How long to wait? Predicting bus arrival time with mobile phone based participatory sensing," in *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services (MobiSys '12)*, pp. 379–392, ACM, Lake District, UK, June 2012.

[34] Z. Zhang and S. Poslad, "A new post correction algorithm (PoCoA) for improved transportation mode recognition," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC '13)*, pp. 1512–1518, IEEE, Manchester, UK, October 2013.

[35] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," DTIC Document, 1985.

[36] N. Győrbíró, Á. Fábián, and G. Hományi, "An activity recognition system for mobile phones," *Mobile Networks and Applications*, vol. 14, no. 1, pp. 82–91, 2009.

[37] "F1 score," https://en.wikipedia.org/wiki/F1_score.