12-31-2020

Human activity recognition using wearable sensors: a deep learning approach

Jialun Xue
New Jersey Institute of Technology

ABSTRACT

HUMAN ACTIVITY RECOGNITION USING WEARABLE SENSORS: A DEEP LEARNING APPROACH

by
Jialun Xue

In the past decades, Human Activity Recognition (HAR) grabbed considerable research attentions from a wide range of pattern recognition and human–computer interaction researchers due to its prominent applications such as smart home health care. The wealth of information requires efficient classification and analysis methods. Deep learning represents a promising technique for large-scale data analytics. There are various ways of using different sensors for human activity recognition in a smartly controlled environment. Among them, physical human activity recognition through wearable sensors provides valuable information about an individual's degree of functional ability and lifestyle. There is abundant research that works upon real time processing and causes more power consumption of mobile devices. Mobile phones are resource-limited devices. It is a thought-provoking task to implement and evaluate different recognition systems on mobile devices.

This work proposes a Deep Belief Network (DBN) model for successful human activity recognition. Various experiments are performed on a real-world wearable sensor dataset to verify the effectiveness of the deep learning algorithm. The results show that the proposed DBN performs competitively in comparison with other algorithms and achieves satisfactory activity recognition performance. Some open problems and ideas are also presented and should be investigated as future research.

HUMAN ACTIVITY RECOGNITION USING WEARABLE SENSORS: A DEEP LEARNING APPROACH

by
Jialun Xue

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Computer Engineering

Helen and John C. Hartmann Department of
Electrical and Computer Engineering

December 2020

Blank Page

APPROVAL PAGE

HUMAN ACTIVITY RECOGNITION USING WEARABLE SENSORS: A DEEP LEARNING APPROACH

Jialun Xue

Dr. Mengchu Zhou, Thesis Advisor Date
Distinguished Professor of Electrical and Computer Engineering, NJIT

Dr. Edwin Hou, Committee Member Date
Professor of Electrical and Computer Engineering, NJIT

Dr. Xiwang Guo, Committee Member Date
Associate Professor of Computer and Communication Engineering, Liaoning Shihua University

BIOGRAPHICAL SKETCH

Author: Jialun Xue

Degree: Master of Science

Date: December 2020

Undergraduate and Graduate Education:

- Master of Science in Computer Engineering,
  New Jersey Institute of Technology, Newark, NJ, 2020

- Master of Science in Electrical Engineering,
  New Jersey Institute of Technology, Newark, NJ, 2018

- Bachelor of Engineering in Automation,
  Harbin Institute of Technology, Weihai, P. R. China, 2016

Major: Computer Engineering

Publications:

Z. Zhao, S. Liu, M. Zhou, X. Guo and J. Xue, "Iterated greedy algorithm for solving a new single machine scheduling problem", 2019 IEEE 16th International Conference on Networking Sensing and Control (ICNSC), pp. 430-435, 2019.

HUANG Bo, YUAN Shoutong, XUE Jialun, "Spreading Process Simulation and Experimental Investigation for a Carbon Fiber Pneumatic Spreader", Journal of Xi'an Jiaotong University, vol.49 no.6, pp.19-25, March, 2015.

To my parents, family members, and friends.

ACKNOWLEDGMENT

I would like to express my deepest appreciation to Dr. Mengchu Zhou, who not only served as my research supervisor, providing valuable and countless resources, insight, and intuition, but also constantly gave me support and encouragement. Special thanks are given to my parents who always support and encourage me.

Many of the visiting professors and my fellow graduate students in the Discrete Event System Laboratory are deserving of recognition for their support.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Human Activity Recognition

Human Activity Recognition (HAR) aims to identify the actions carried out given a set of observations of a person and his/her surrounding environment. Recognition can be accomplished by exploiting the information retrieved from various sources such as environmental or body-worn sensors [1]. Some approaches [2], [3] have adapted dedicated motion sensors to fit different human body parts such as waist, wrist, chest and thighs. They have achieved great classification performance. However, these sensors usually make a common user not that comfortable and do not provide a long-term solution for activity monitoring, due to such issues, as sensor repositioning after dressing.

HAR has become an attractive research field due to its importance as well as many challenges brought to the research community. Researchers use these HAR systems as a medium to get information about people’s behaviors. The information is commonly collected from the signals of sensors such as ambient and wearable sensors. The data from the signals are then processed through machine learning algorithms and recognize the events. Hence, such HAR systems can be applied in plenty of useful and practical applications in smart environments such as smart home health-care systems. For example, a smart HAR system can continuously observe patients for health diagnosis and medication. Also it can be applied for automated surveillance of public places to predict crimes that may occur in the near future.

1.2 Wearable Sensors

Since the appearance of the first commercial hand-held mobile phones in 1979, it has been observed an accelerated growth in the mobile phone market. Mobile devices have almost become easily accessible to virtually everybody now. Smartphones, which are a new generation of mobile phones, are now offering many other features such as multitasking and the deployment of a variety of sensors, in addition to the basic telephony. Current efforts attempt to incorporate all these features while maintaining similar battery lifespans and device dimensions. The integration of these mobile devices in our daily life is rapidly growing. It is envisioned that such devices can seamlessly keep track of our activities, learn from them, and subsequently help us to make better decisions regarding our future actions.

Smartphones have been bringing up new research opportunities for human-centered applications where the user is a rich source of context information and the phone is the firsthand sensing tool. Latest devices come with embedded built-in sensors such as microphones, dual cameras, accelerometers, gyroscopes, etc. The use of smartphones with inertial sensors is an alternative solution for HAR. These mass-marketed devices provide a flexible, affordable and self-contained solution to automatically and unobtrusively monitor Activities of Daily Living (ADL) while also providing telephony services. Consequently, in the last few years, some works aiming to understand human behavior using smartphones have been proposed. For instance, one of the first approaches has been exploited an Android smartphone for HAR employing its embedded triaxial accelerometers. Improvements are still expected in topics such as in multi-sensor fusion for better HAR classification, standardizing performance evaluation metrics, and providing public data for evaluation.

Currently, smartphones, wearable devices, and internet-of-things (IoT) are becoming more affordable and ubiquitous. Many commercial products, such as the Apple Watch, Fitbit, and Microsoft Band, and smartphone apps including Runkeeper and Strava, are already available for continuous collection of physiological data. These products typically contain sensors that enable them to sense the environment, have modest computing resources for data processing and transfer, and can be placed in a pocket or purse, worn on the body, or installed at home [4]. Accurate and meaningful interpretation of the recorded physiological data from these devices can be applied potentially to HAR. However, most current commercial products only provide relatively simple metrics, such as step count or cadence. The emergence of deep learning methodologies that extract different discriminating features from the data, and increased processing capabilities in wearable technologies. The ability of simultaneous activity classification and the decreasing size of computing platforms give rise to the possibility of performing detailed data analysis in situ and in real time. Today's handheld PCs are often more powerful than desktop computers of the 1990s. Once a rare commodity, computers are now embedded in everything - toys, cars, cell phones, and even bread makers.

In the case of wearable sensors in activity recognition, a smartphone is an alternative to them due to the support of the diversity of sensors in it. Handling sensors such as accelerometers and gyroscopes along with the device with wireless communication capabilities made smartphones a very useful tool for activity monitoring in smart homes. Besides, smartphones are very ubiquitous and require almost no static infrastructure to operate it. This advantage makes it more practically applicable than other ambient multi-modal sensors in smart homes. As recent smart phones consist of inertial sensors (e.g., gyroscopes and accelerometers), they can be appropriate sensing resources to obtain human motion information for HAR.

HAR has been actively explored based on a distinguished kind of ambient and wearable sensors. Some instances of such sensors include motion, proximity, microphone, and video sensors. Most of the ambient sensor-based latest HAR researchers have mainly focused on video cameras as cameras make it easy to retrieve the images of surrounding environment. Video sensors are included with some other prominent sensors in some work related to novel ubiquitous applications. Though video sensors have been very popular for basic activity recognition. They face very many difficulties for ordinary people to accept due to a privacy issue . On the contrary, wearable sensors such as inertial sensors can overcome this kind of privacy issues and hence, deserve more focus for activity recognition in smart homes.

In the past years, many HAR systems used accelerometers to recognize a big range of daily activities such as standing, walking, sitting, running, and lying. For instance, some researchers have already explored the accelerometer data to find out the repeating activities such as grinding, filling, drilling, and sanding [11] [14]. The others, have performed elderly peoples' fall detection and prevention in smart environments [22]. Majority of the afore mentioned systems adopted many accelerometers fixed in different places of a human body. However, this approach apparently not applicable to daily life to observe long-term activities due to attachment of many sensors in the human body and cable connections. Some studies [12] [15] tried to explore the data of single accelerometers at sternum or waist. These studies have reported substantial recognition results of basic daily activities such as running, walking, and lying. However, they could not show good accuracy for some complex activity situations such as transitional activities, e.g., sit to stand, lie to stand, and stand to sit.

1.3 Deep Learning

Deep learning is a paradigm of machine learning that uses multiple processing layers to infer and extract information from a large scale of data. Many studies [5]-[7] have shown that the use of deep learning can achieve better performances in a range of applications than traditional approaches. Traditional approaches use a set of selected features, also known as “shallow” features [8], to represent the data for a specific classification task.

HAR can be accomplished, for example, by exploiting the information retrieved from inertial sensors such as accelerometers. In some smartphones these sensors are embedded by default and we benefit from them to classify a set of physical activities (standing, walking, laying, walking, walking upstairs and walking downstairs) by processing inertial body signals through a supervised Machine Learning (ML) algorithm for hardware with limited resources.

1.3.1 Deep Belief Network

Popular deep learning approaches include deep belief networks (DBN), stacked autoencoders (SAE) and convolutional neural nets (CNN). Among them, deep belief network (DBN) has been used in many complex pattern recognition problems, including speech recognition, image and video processing and classification [23]-[25]. However, it has been addressed in many articles that the DBN shows its superior performance if the configuration of DBN is done appropriately. These studies rarely include details on how it performs best and finds the optimal configuration of parameters. They only shows single score, and it remains unclear how this peak performance is achieved.

Figure 1.1 shows the workflow of the proposed approach for selecting the optimal structure of DBN for HAR. The proposed system consists of three major parts: sensor data collection, feature extraction with dimensionality reduction, and activity recognition. The sensor data collection system collects various human activities related body sensor data from various sources. In this thesis, we consider the accelerometers and gyroscopes sensor data. The second part of the system extracts robust features and reduces dimensionality of features after removing noise and performing statistical analysis on sensor signals. Finally, the last part of the system trains DBN with these robot features and tries to find the optimal DBN structure for the highest accuracy of HAR.

1.3.2 Other Approaches

Recently, smartphones have attracted many activity recognition researchers as they have fast processing capability, and they are easily deployable. For instance, some researchers [8] use wirelessly connected smartphones to collect a user's data from a chest unit composed of the accelerometer and vital sign sensors. The data is later

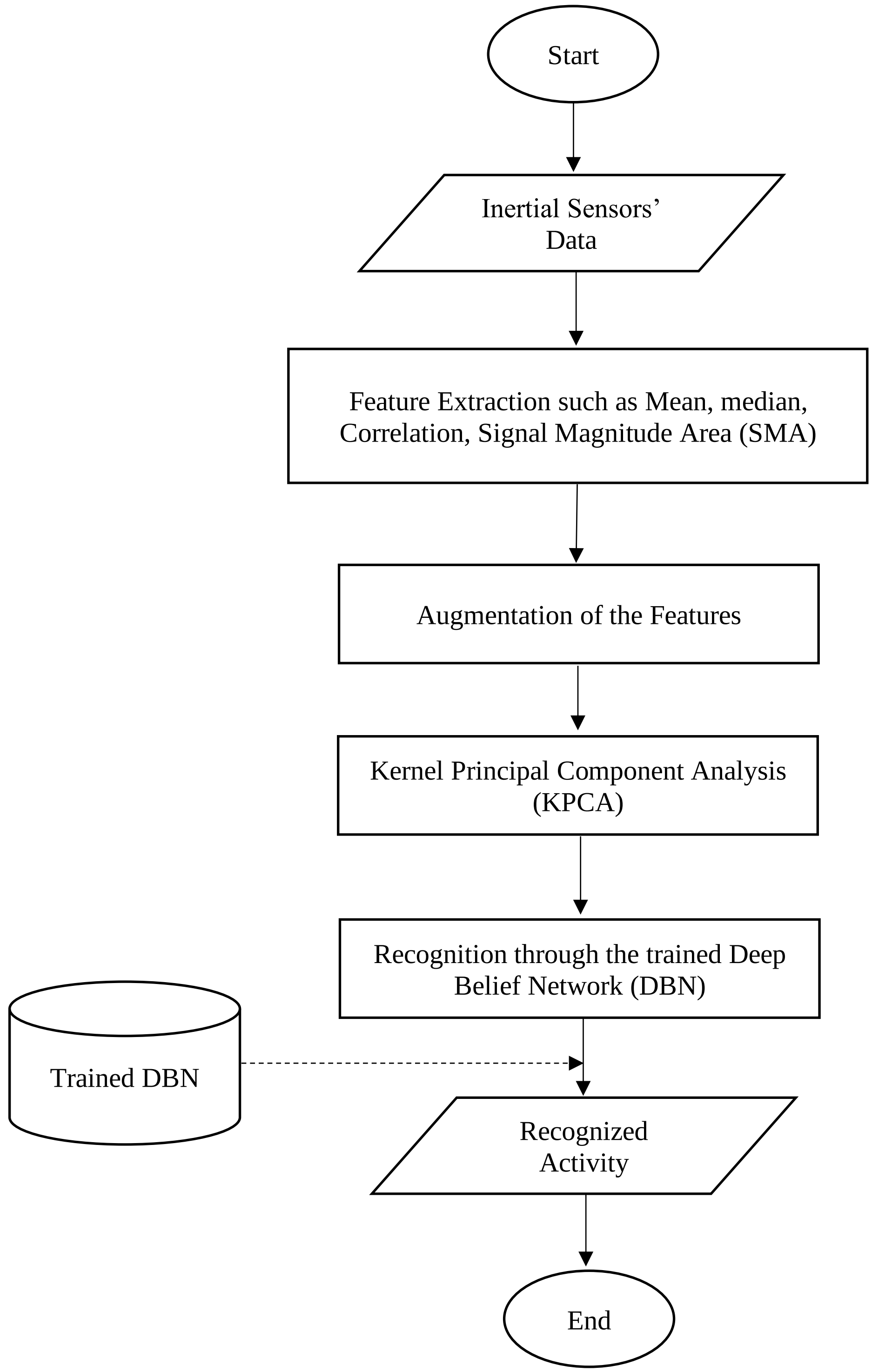

Figure 1.1 A flowchart of the proposed human activity recognition system.

processed and analyzed by using different machine learning algorithms. Some of them [14] develop an HAR system to recognize five different kinds of transportation activities where data from smartphones inertial sensors are used with a mixture-of-expert model for classification. Some researchers [11] proposed an offline HAR system where a smartphone with built-in triaxial accelerometer sensor is used. A phone is kept in the pocket during experiments. Some scientists [18] also used a smartphone mounted in the waist to collect inertial sensors' data for activity recognition. They used Support Vector Machine (SVM) for activity modeling. In some papers [6] [10], a smartphone is used to recognize six different activities in real-time. Moreover, the researchers have proposed a real-time motion recognition system with the help of a smartphone with accelerometer sensors [19]. Some use a smartphone with an embedded accelerometer to recognize four different activities in real time [20].

The development of HAR applications using smartphones has several advantages such as easy device portability without the need for additional fixed equipment, and comfort to a user due to their unobtrusive sensing. This contrasts with other established HAR approaches which use specific-purpose hardware devices such as those in body sensor networks [26]. Although the use of numerous sensors could improve the performance of a recognition algorithm, it is unrealistic to expect that the general public will use them in their daily activities because of the difficulty and the time required to wear them. One drawback of a smartphone-based approach is that energy and services on the mobile phone are shared with other applications and this become critical in devices with limited resources.

ML methods that are previously employed for pattern recognition include Naive

Bayes, and Support Vector Machines (SVMs) [20] [27]. In particular, we make use of SVMs for classification as many other studies [28] [29]. Although it is not fully clear which method performs better for HAR, SVMs have confirmed their successful application in several areas including heterogeneous types of recognition such as intrusion detection, fault detection, handwritten character recognition and speech recognition [30]. In ML, fixed-point arithmetic models [8] were previously studied initially because devices with floating-point units were unavailable or expensive. The possibility of retaking these approaches for HAR systems that require either low cost devices or to allow load reduction in multitasking mobile devices has nowadays become particularly appealing.

CHAPTER 2

RELATED WORK

HAR has been actively explored based on a distinguished kind of ambient and wearable sensors. Some instances of such sensors include motion, proximity, microphone, and video sensors. Most of the ambient sensor-based latest HAR researchers have mainly focused on video cameras as cameras make it easy to retrieve the images of surrounding environment. Video sensors are combined with some other prominent sensors in many applications. They have been very popular for basic activity recognition. However, they pose serious privacy issues. On the other hand, wearable sensors such as inertial sensors do not face this kind of privacy issues. They are thus useful in smart homes.

Many HAR systems apply accelerometers to recognize such daily activities as standing, walking, sitting, running, and lying. This chapter reviews their related problems and structure.

2.1 Problem Statement

In the last few decades, many HAR systems were developed. Researchers have focused on several activities in distinguished application domains. For instance, the activities can include walking, running, cooking, exercising, etc. In terms of their duration and complexity, these activities can be categorized into three key groups: short, simple, and complex activities. The first group consists of activities with very short duration such as transition from *sit to stand*. The second group refers to basic activities like walking and reading. The last group basically include the combinations of progressions of basic activities with the interaction with other objects and individuals. Such kind of activities can be partying or official meeting together.

Some studies have introduced the concept of a Hardware-Friendly SVM (HF-SVM) [9]. It exploits fixed point arithmetic in the feed-forward phase of an SVM classifier, so as to allow the use of this algorithm in hardware-limited devices. The SVM algorithm is originally proposed only for binary classification problems but it has been adapted by using different schemes for multiclass problems such as in [10]. In particular, the One-Vs-All (OVA) method is as its accuracy is comparable to other classification methods as demonstrated [7], and because its learned model uses less memory when compared to an One-Vs-One (OVO) method. This is advantageous when used in resource-limited hardware devices. Utilizing wearable sensors, numerous works [4]-[6] has been done in the literature with various classification algorithms for recognizing human activity. Most of the algorithms include SVM-based classification, neural network-based one and pattern mating based one. For instance, a neural system classifier for line activity recognition is proposed. However, actualizing such a complicated method in a wearable sensor system is restricted by the calculability of the implanted framework. Other more methodical ways to deal with classifying activities based on decision tree classifier are proposed in [19]. However, it has low recognition accuracy rate at 70% [20]. Therefore, to achieve high accuracy with low computation cost is a key challenge of human activity recognition.

2.2 General Structure of HAR Systems

To deal with this challenge, recently, deep learning (DL) based human activity recognition from wearable sensors is becoming popular. The previous approaches [13] [15] in HAR mostly rely on manually designed feature extraction procedures, and various supervised classification methods. The manual feature extracting procedures require prior specific knowledge about the signals for finding important characteristics among different activities and thus lacks the robust physiological basis. In contrast, a deep learning approach can naturally extract representative or optimal features with no earlier learning from the sensor signals and afterward use these features to perform HAR.

Figure 2.1 [10] identifies a generic data acquisition architecture for HAR systems. First, wearable sensors are attached to a person's body to measure attributes of interest such as motion location, temperature, among others. These sensors should communicate with an Integration Device (ID), which can be a cellphone, PDA, laptop, or customized embedded system. The main purpose of the ID is to preprocess the data received from the sensors and, in some cases, send them to an application server for real time monitoring, visualization, and/or analysis. The communication protocol could be UDP/IP or TCP/IP, according to the desired level of reliability.

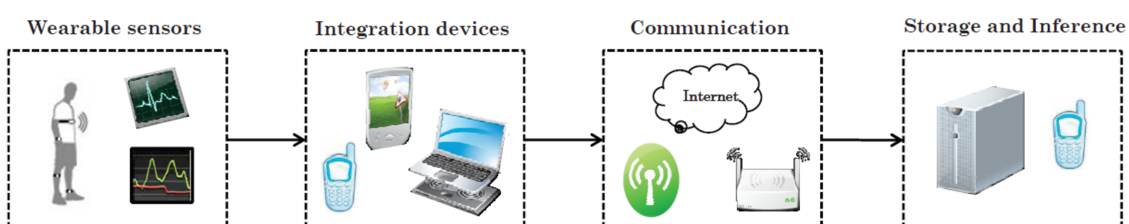

Figure 2.1 Generic data acquisition architecture for HAR.

2.3 Evaluation of HAR Systems

In this thesis, we have categorized HAR systems that rely on wearable sensors in two levels. The first one has to do with the learning approach, which can be either supervised or semi-supervised. In the second level, according to the response time, supervised approaches can work either online or offline. The former provides immediate feedback on the performed activities. The latter either needs more time to recognize activities due to high computational demands, or is intended for applications that do not require real-time feedback. This taxonomy has been adopted as the systems within each class have very different purposes and their associated challenges should be evaluated separately. For instance, an effective offline system may not be able to run online due to processing capacity constraints. Finally, although different sets of recognized activities clearly result in different types of HAR systems, incorporating it in the taxonomy would lead to an excessive granularity as most systems define a particular set of activities.

The human activity classifier can be trained online or offline as well as the classification process itself can be done online or offline. Offline classification (non-real-time) is a sufficient solution when a user does not find an urgent need to receive immediate feedback. In the other side, online classification (real-time) assists users in receiving real-time feedback.

2.3.1 Online HAR Systems

Applications of online HAR systems can be easily visualized. In healthcare, continuously monitoring patients with physical or mental pathologies becomes crucial for their protection, safety, and recovery. Likewise, interactive games or simulators may enhance a user's experience by considering activities and gestures. Table 2.1 summarizes the online state-of-the-art activity recognition approaches.

Table 2.1 Summary of Online HAR Systems

Research	**Number of sensors**	**Technique**	**Number of users**	**Accuracy (%)**
Liu et al., 2015	1	SVM	50	88.1
Sazonov et al., 2011	1	SVM	9	80.3
Reiss & Stricker, 2013	3	Boosted Decision Tree	8	90.7
Martin et al,2017	2	K-Nearest Neighbors	5	89.4

2.3.2 Offline HAR Systems

Table 2.2 summarizes state-of-the-art works in offline HAR systems based on wearable sensors. There are cases in which a user does not need to receive immediate feedback. For example, applications that analyze exercise and diet habits in patients

Table 2.2 Summary of Offline HAR Systems

Research	Number of sensors	Technique	Number of users	Accuracy (%)
Feng, Meiling, and Nan ,2011	1	Decision Tree	20	94.1
Czabke, Marsch, and Lueth, 2017	1	SVM	10	83.2
Bayati et al., 2016	--	Artificial Neural Network	30	86.9
Andreu et al., 2011	1	K-Nearest Neighbors	--	87.4
Yuting et al., 2017	3	Naïve Bayes	10	88.6

with heart disease, diabetes, and obesity, as well as applications that estimate the number of calories burned after an exercise routine [22], can work on an offline basis. Another example of an offline HAR system is an application to discover commercial patterns for advertisement. For instance, if an individual performs exercise activities very frequently, they could be advertised on sport wear items. In all these cases, gathered data can be analyzed on a daily or even weekly basis to draw conclusions on the person's behavior.

CHAPTER 3

DESIGN ISSUES

3.1 Selection of Attributes and Sensors

Environmental attributes: These attributes, include temperature, humidity, and audio level, etc., are intended to provide context information describing a person's surroundings. If audio level and light intensity are fairly low, for instance, the subject may be sleeping. Various existing systems have utilized microphones, light sensors, humidity sensors, and thermometers, among others [10], [15]. These sensors alone, though, might not provide sufficient information as individuals can perform each activity under diverse contextual conditions in terms of weather, audio loudness, or illumination. Therefore, environmental sensors are generally accompanied by accelerometers and other sensors associated with a human subject [13].

Acceleration: Triaxial accelerometers are perhaps the most broadly used sensors to recognize ambulation activities (e.g., walking, running, and lying) [14]–[16]. Accelerometers are inexpensive, require relatively low power and are embedded in most of today's cellular phones. Several papers have reported high recognition accuracy 92.25% [15], 90% [17], 91% [18], and up to 93% [19], under different evaluation methodologies. However, other daily activities such as eating, working at a computer, or brushing teeth, are confusing from the acceleration point of view. For instance, eating might be confused with brushing teeth due to arm motion. The impact of the sensor specifications on HAR performance has also been analyzed. In fact, Maurer et al. [11] have studied the behavior of the recognition accuracy as a function of the accelerometer sampling rate (which lies between 10 Hz and 100 Hz). Interestingly, they have found that no significant gain in accuracy is achieved above 20 Hz for ambulation activities. The placement of an accelerometer is another important point of discussion: Marta et al. [19] have found that the best place to wear the accelerometer is inside the trousers pocket. Instead, other studies suggest that the accelerometer should be placed in a bag carried by the user: on the belt, or on the dominant wrist. In the end, the optimal position to place an accelerometer depends on the application and type of activities to be recognized. For instance, an accelerometer on the wrist may not be appropriate to recognize ambulation activities, since accidental arm movements could generate incorrect predictions. On the other hand, in order to recognize an activity such as working at the computer, an accelerometer on the chest would not provide sufficient information.

Location: Global Positioning System (GPS) enables all sort of location based services. Current cellular phones are equipped with GPS devices, making this sensor very convenient for context-aware applications, including the recognition of a user's transportation mode [17]. The place at which a user is can be helpful to infer their activity by using ontological reasoning [12]. As an example, if a person is at a park, they are probably not brushing their teeth but might be doing exercise, e.g., running or walking. Such location information about places can be easily obtained by means of the Google Places Web Service [11], among other tools. However, GPS devices do not work well indoors and they are relatively expensive in terms of energy consumption, especially in real-time tracking applications. For those reasons, this sensor is usually employed along with accelerometers. Finally, location data has privacy issues because users are not always willing to be tracked. Encryption, obfuscation, and anonymization are some of the techniques available to ensure privacy in location data.

Physiological signals: Vital sign data (e.g., heart rate, respiration rate, skin temperature, skin conductivity, and ECG) have also been considered in a few studies [3]. Tapia et al. [20] have proposed an HAR system that combines data from five triaxial accelerometers and a heart rate monitor. They have concluded that the heart rate is not useful in a HAR context because after performing physically demanding activities (e.g., running) the heart rate remains at a high level for a while, even if the individual is lying or sitting. In a previous study, by means of structural feature extraction, vital signs can be exploited to improve recognition accuracy. Now, in order to measure physiological signals, additional sensors would be required, thereby increasing the system cost and introducing obtrusiveness [19]. Also, these sensors generally use wireless communication which entails higher energy expenditures.

3.2 Recognition Performance

The performance of a HAR system depends on several aspects: 1) a concerned activity set, 2) the quality of training data, 3) a feature extraction method, and 4) a machine learning algorithm. Each set of activities brings a totally different pattern recognition problem. For example, discriminating among walking, running, and standing still [7], turns out to be much easier than the cases incorporating more complex activities such as watching TV, eating, ascending, and descending [17]. Secondly, there should be a sufficient amount of training data, which should also be similar to the expected testing data. Finally, a comparative evaluation of several learning methods is desirable as each dataset exhibits distinct characteristics that can be either beneficial or detrimental for a particular method. Such interrelationship among datasets and learning methods can be very hard to analyze theoretically, which accentuates the need of an experimental study. In order to quantitatively understand the recognition performance, some standard metrics are used, e.g., accuracy, recall, precision, F-measure, Kappa statistic, and ROC curves.

3.3 Obtrusiveness

To be successful in practice, HAR systems should not require a user to wear many sensors nor to interact too often with the systems. The more sources of data available, the richer the information that can be extracted from the measured attributes. There are systems which require the users to wear four or more accelerometers [3], [7], [15], or carry a heavy rucksack with recording devices [20]. These configurations may be uncomfortable, invasive, expensive, and hence not suitable for HAR. Other systems are able to work with rather unobtrusive hardware. For instance, a sensing platform that can be worn is presented in [5], which only requires a strap that is placed on the chest and a cellular phone. Finally, the systems introduced in [22], recognize activities with a cellular phone only. Minimizing the number of sensors required to recognize activities is beneficial not only for human subjects' comfort, but also to reduce complexity and energy consumption a smaller amount of data would be processed than the cases with many sensors. Maurer et al. [11] have performed an interesting study with accelerometers and light sensors. They have explored different subsets of features and sensors, as well as different sensor placements. Their conclusion is that all sensors available should be used together in order to achieve the maximum accuracy level. Ravi et al. [14] have carried out another study in the same research issue by placing accelerometers on a person's hip, wrist, arm, ankle, thigh, and combinations of them. Their conclusions suggest that only two accelerometers (i.e., either wrist and thigh or wrist and hip) are sufficient enough to recognize ambulation and other daily activities. Clearly these studies have indicated contractor results.

CHAPTER 4

PROPOSED METHODS AND RESULTS

Figure 4.1 shows the basic structure of a deep learning module. Deep learning is a paradigm of machine learning that uses multiple processing layers to infer and extract information from big data. Research has shown that the use of deep learning can achieve improved performance in a range of applications over traditional approaches. Conventional learning approaches use a set of predesigned features, also known as “shallow” features, to represent the data for a specific classification task. In image processing and machine vision [31]-[33], shallow features such as Spectrogram representation provides a form of time and sampling rate invariance. This enables the classification to be more robust. Frequency selection in the spectrogram domain also allows noise filtering of the data over time.

From each sampled window described above, a vector of features is obtained. Standard measures previously used in HAR literature such as the mean, correlation, signal magnitude area (SMA) and autoregression coefficients are employed for feature mapping.

A set of features is also employed in order to improve the learning performance, including energy of different frequency bands, frequency skewness, and angle between vectors, e.g. mean body acceleration [8]. It contains the list of all the measures applied to the time and frequency domain signals. A total of 561 features are extracted to describe each activity window. In order to ease the performance assessment, the dataset has been also randomly partitioned into two independent sets, where 70% of the data are selected for training and the remaining 30% for testing.

From each window, a vector of features is extracted to 17 features estimated from a set of measurements in the time and frequency domain using previously suggested features. The Fast Fourier Transform (FFT) is used to find the frequency components for each window. Some examples of measurements extracted to obtain a feature vector are depicted in Figure 4.1.

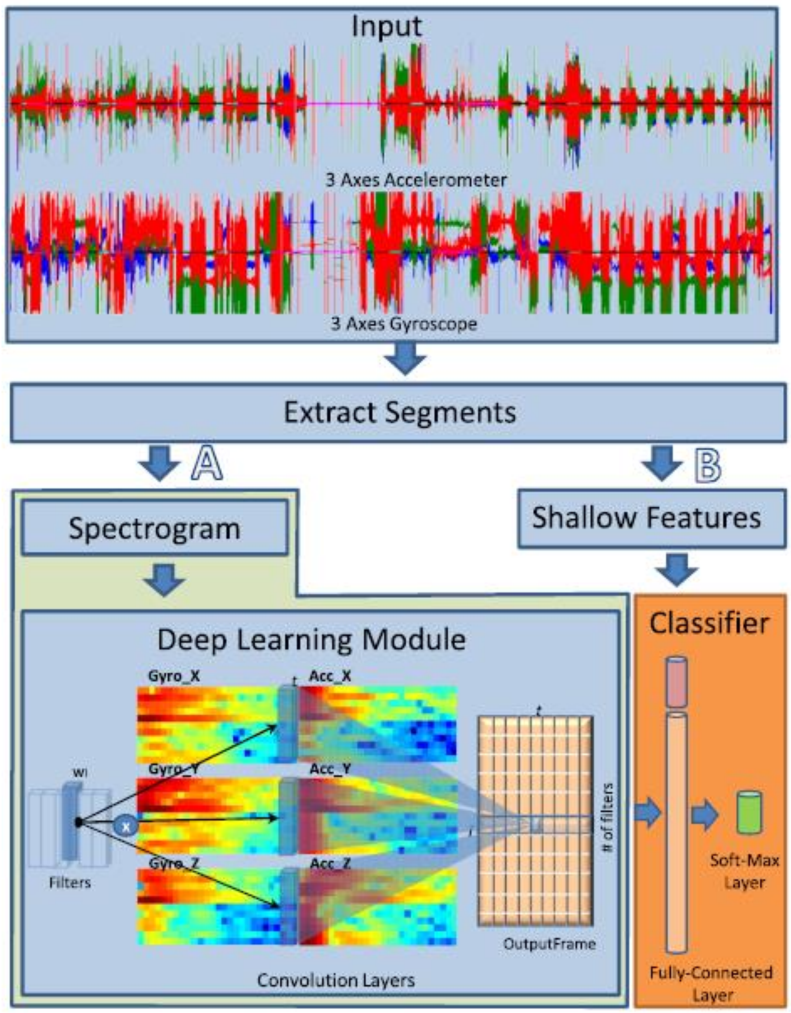

Figure 4.1 Deep learning module.

4.1 Signal Processing

We can collect triaxial linear acceleration and angular velocity signals by using the phone accelerometer and gyroscope at a sampling rate of 50Hz. These signals are preprocessed for noise reduction with a median filter and a 3rd order low-pass Butterworth filter with a 20 Hz cutoff frequency. This rate is sufficient for capturing human body motion since 99% of its energy is contained below 15Hz.

The acceleration signal, which has gravitational and body motion components, is separated by using another Butterworth low-pass filter into body acceleration and gravity. The gravitational force is assumed to have only low frequency components. Therefore, from the experiments, we conclude that 0.3 Hz is an optimal corner frequency for a constant gravity signal. Additional time signals are obtained by calculating the Euclidean magnitude and time derivatives [19] (jerk da/dt and angular acceleration dw/dt) from the triaxial signals. The time signals are then sampled in fixed-width sliding windows of 2.56 sec and 50% overlap between them, since the cadence of an average person walking is within [90,130] steps/min, i.e. a minimum of 1.5 steps/sec.

At least a full walking cycle (two steps) is preferred on each window sample; People with slower cadence such as elderly and disabled should also benefit from this method. It is supposed that a minimum speed is equal to 50% of average human cadence. Signals are also mapped in the frequency domain through a Fast Fourier Transform (FFT), which is optimized for the power of two vectors (2.56 sec×50 Hz = 128 cycles).

4.2 Feature Extraction and Dimension Reduction

Human activities are performed during relatively long periods of time (in the order of seconds or minutes) compared to the sensors' sampling rate (which can be up to 250 Hz). Besides, a single sample on a specific time instant (e.g., the Y-axis acceleration is 2.5g or the heart rate is 130 bpm) does not provide sufficient information to describe the performed activity. Thus, activities need to be recognized in a time window basis rather than in a sample basis.

Now, a question is: how do we compare two given time windows? It would be nearly impossible for the signals to be exactly identical, even if they come from the same subject performing the exactly same activity. This is the main motivation for applying feature extraction (FE) methodologies to each time window: filtering relevant information and obtaining quantitative measures that allow signals to be compared. In general, two approaches have been proposed to extract features from time series data: statistical and structural [30]. The former, such as the Fourier transform and Wavelet transform, use quantitative characteristics of the data to extract features; whereas the latter take into account the interrelationship among data. The criterion to choose either of them is certainly subject to the nature of a given signal. Each instance in the processed dataset corresponds to the feature vector extracted from all the signals within a time window. Most of the existing approaches adhere to this mapping. Next, we cover the most common FE techniques for each of the measured attributes, i.e., acceleration, environmental signals, and vital signs. GPS data are not considered in this section since they are mostly used to compute the speed [19], or include some knowledge about the place where an activity is being performed [1].

1) Acceleration: Acceleration signals are highly fluctuating and oscillatory, which makes it difficult to recognize the underlying patterns using their raw values. Existing HAR systems based on accelerometer data employ statistical feature extraction and, in most of the cases, either time- or frequency-domain features.

Discrete Cosine Transform (DCT) and Principal Component Analysis (PCA) have also been applied with promising results [15], as well as autoregressive model coefficients [12]. All these techniques are conceived to handle the high variability inherent to acceleration signals.

2) Environmental variables: Environmental attributes, along with acceleration signals, have been used to enrich context awareness. For instance, the values from air pressure and light intensity are helpful to determine whether the individual is outdoors or indoors [2]. Also, audio signals are useful to conclude that the user is having a conversation rather than listening to music [19].

3) Vital signs: The very first work that explores vital sign data with the aim of recognizing human activities applies statistical feature extraction. In [13], the authors compute the number of heart beats above the resting heart rate value as the only feature. Instead, Parkka et al. [19] calculate time domain features for heart rate, respiration effort, SaO2, ECG, and skin temperature. Nevertheless, a signal's shape is not described by these features. A heart rate signal S(t) for an individual that was walking is shown with a bold line and the same signal in reverse temporal order, S(t), is displayed with a thin line. Notice that most time domain and frequency domain features (e.g., mean, variance, and energy) are identical for both signals while they may represent different activities. This is he main motivation for applying structural feature

extraction.

4) Selection of window length: Dividing the measured time series in time windows is a convenient way to help solve an HAR problem. A key factor is, therefore, the selection of proper window length because the computational complexity of any FE method depends on the number of samples. Having rather short windows may enhance FE performance, but would entail higher overhead since it would trigger the recognition algorithm more frequently. Besides, short time windows may not provide sufficient information to fully describe a performed activity. Conversely, if window size is too big, there might be more than one activity within a single time window [7]. Different window lengths have been used in the literature. This decision is conditioned to the activities to be recognized and the measured attributes. The heart rate signal, for instance, requires 30s time windows according to [3]. For activities such as swallowing, 1.5s time windows are normally employed.

Time windows can also be either overlapping or disjoint. Overlapping time windows are intended to handle transitions more accurately, although, by using small non-overlapping time windows, misclassifications due to transitions are negligible.

5) Feature selection: Some features in the processed dataset might contain redundant or irrelevant information that can negatively affect the recognition accuracy. Then, implementing techniques for selecting the most appropriate features is a suggested practice to reduce computation and simplify learning models. Bayesian Information Criterion (BIC) and Minimum Description Length (MDL) [9] have been widely used for general machine learning problems. In HAR, a common method is the Minimum Redundancy and Maximum Relevance (MRMR) [5], and has been utilized

in [10]. In [10], the minimum mutual information between features is used as a criterion for minimum redundancy; while the maximal mutual information between the classes and features is used as a criterion for maximum relevance. In contrast, Maurer et al. [11] have applied a Correlation-based Feature Selection (CFS) approach by taking advantage of the fact that this method is built in WEKA [6]. CFS works under the assumption that features should be highly correlated with the given class but uncorrelated with each other. Iterative approaches have also been evaluated to select features.

The next step of the feature extraction is to apply dimension reduction using Kernel PCA (KPCA). In KPCA, a statistical kernel is applied to the input features, followed by typical PCA. Given spatiotemporal robust features F, the covariance matrix of the features can be defined as

$$Y = \frac{1}{q}\sum_{i=1}^{q}(\Theta(F_i) \cdot \Theta(F_i)^T \tag{4.1}$$

$$\Theta F_i = \Theta(F_i) - \bar{\Phi} \tag{4.2}$$

$$\bar{\Phi} = \frac{1}{q}\sum_{i=1}^{q}(\Phi(F_i) \tag{4.3}$$

where q represents the total number of feature segments for training and Φ is a Gaussian kernel. Now, the principal components can be found by solving the following eigenvalue decomposition problem:

$$\lambda E = QE \tag{4.4}$$

$$\bar{\Phi} = \frac{1}{q} \sum_{i=1}^{q} (\Phi(F_i) \tag{4.5}$$

where E represents the principal components and λ the corresponding eigenvalues. The feature vectors using KPCA for a signal segment can be represented as

$$K = FE_m^T \tag{4.6}$$

4.3 Machine Learning

In recent years, the prominent development of sensing devices (e.g., accelerometers, cameras, GPS, etc.) has facilitated the process of collecting attributes related to human beings and their surroundings. However, most applications require much more than simply gathering measurements from variables of interest. In fact, additional challenges for enabling context awareness involve knowledge discovery since the raw data (e.g., acceleration signals or electrocardiogram) provided by the sensors are often useless.

For this purpose, HAR systems make use of machine learning tools, which are helpful to build patterns to describe, analyze, and predict data. In a machine learning context, patterns are to be discovered from a set of given examples or observations denominated instances. Such input set is called a training set. In our specific case, each instance is a feature vector extracted from signals within a time window. The examples in the training set may or may not be labeled, i.e., associated to a known class, e.g., walking, and running. In some cases, labeling a vast amount of data is not feasible because it may require an expert to manually examine the examples and assign a label based upon their experience. This process is usually tedious, expensive, and extremely time-consuming in many data mining applications. There exist two learning approaches, namely supervised and unsupervised learning, which deal with labeled and unlabeled data, respectively. Since an HAR system should return a label such as walking, sitting, and running, most HAR systems work in a supervised learning fashion. Indeed, it might be very hard to discriminate activities in a completely unsupervised context. Some other systems [17] work in a semi-supervised fashion allowing part of the data to be unlabeled.

4.3.1 Supervised Learning

Labeling sensed data from individuals performing different activities is a technically easy task. Some systems store sensor data in a non-volatile medium while a person from the research team supervises the collection process and manually registers activity labels and time stamps. Other systems feature a mobile application that allows the user to select the activity to be performed from a list. In this way, each sample is matched to an activity label, and then stored in the server. Supervised learning (referred to as classification for discrete-class problems) has been a very productive field, in which a great number of algorithms have been proposed.

Decision trees can be used to build a hierarchical model in which attributes are mapped to nodes and edges represent the possible attribute values. Each branch from the root to a leaf node is a classification rule. C4.5 is perhaps the most widely used decision tree classifier and is based on the concept of information gain to select the attributes that should be placed in the top nodes.

Bayesian methods calculate posterior probabilities for each class using estimated conditional probabilities from a training set. The Bayesian Network (BN) classifier and Naive Bayes (NB) (which is a specific case of BN) are the principal players of this family of classifiers. A key issue in Bayesian Networks is the topology construction, as it is necessary to make assumptions on the independence among features. For instance, the NB classifier assumes that all features are conditionally independent given a class value. Yet such assumption does not hold in many cases. As a matter of fact, acceleration signals are highly correlated, as well as physiological signals such as heart rate, respiration rate, and ECG amplitude.

Instance based learning (IBL) methods classify an instance based upon the most similar instance(s) in the training set. For that purpose, they define a distance function to measure similarity between each pair of instances. This makes IBL classifiers quite expensive in their evaluation phase as each new instance to be classified needs to be compared to the entire training set. Such high cost in terms of computation and storage, makes IBL models inconvenient to be implemented in a mobile device.

Support Vector Machines (SVM) and Artificial Neural Networks (ANN) have also been broadly used in HAR although they do not provide a set of rules understandable by human beings. Instead, knowledge is hidden within the model, which may hinder the analysis and incorporation of additional reasoning. SVMs rely on kernel functions that project all instances to a higher dimensional space with the aim of finding a linear decision boundary (i.e., a hyperplane) to partition the data. Neural networks [34]-[36] replicate the behavior of biological neurons in human brain, propagating activation signals and encoding knowledge in the network links. Besides, ANNs have been shown to be universal function approximators. The high computational cost and the need for a large amount of training data are two common drawbacks of neural network-based approaches.

Ensembles of classifiers [37] [38] combine the output of several classifiers to improve classification accuracy. Some examples are bagging, boosting, and stacking. Classifier ensembles are clearly more expensive, computationally speaking, as they require several models to be trained and evaluated.

4.3.2 Semi-Supervised Learning

Relatively few approaches [10] have implemented activity recognition in a semi-supervised fashion, thus, having part of the data without labels. In practice, annotating data might be difficult in some scenarios, particularly when the granularity of activities is very high or a user is not willing to cooperate with a data collection process. Since semi-supervised learning is a minority in HAR, there are no standard algorithms or methods, but each system implements its own approach.

4.3.3 Evaluation Metrics

In general, the selection of a classification algorithm for HAR has been merely supported by empirical evidence. The vast majority of studies use cross validation with statistical tests to compare classifiers' performance for a particular dataset. The classification results for a particular method can be organized in an $n \times n$ confusion matrix M for a classification problem with n classes. This is a matrix such that its element Mij is the number of instances from class i that are actually classified as class j.

The following values can be obtained from the confusion matrix in a binary classification problem:

1) True Positives (TP): The number of positive instances that are classified as positive;

2) True Negatives (TN): The number of negative instances that are classified as negative;

3) False Positives (FP): The number of negative instances that are classified as positive;

4) False Negatives (FN): The number of positive instances that are classified as negative.

The accuracy is the most standard metric to summarize the overall classification performance for all classes and it is defined as follows:

$$\mathrm{Accuracy} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{4.7}$$

The precision, often referred to as positive predictive value, is the ratio of correctly classified positive instances to the total number of instances classified as positive:

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{4.8}$$

The recall, also called true positive rate, is the ratio of correctly classified positive instances to the total number of positive instances:

$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{4.9}$$

The F-measure combines precision and recall in a single value:

$$F - \text{measure} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precisio} + \text{Recall}} \quad (4.10)$$

Although defined for binary classification, these metrics can be generalized for a multi-classification problem with n classes. In such case, an instance could be positive or negative according to a particular class, e.g., positives might be all instances of running while negatives would be all instances other than running.

4.3.4 Machine Learning Tools

The Waikato Environment for Knowledge Analysis (WEKA) is among the best known tools in the machine learning research community. It contains implementations of a number of learning algorithms and allows researchers to easily evaluate them for a particular dataset using cross validation and random split, among others. WEKA also offers a Java API that facilitates the incorporation of new learning algorithms and evaluation methodologies on top of the pre-existing framework. One of the limitations of current Machine Learning APIs such as WEKA and the Java Data Mining (JDM) platform [8] is that they are not fully functional in current mobile platforms. In that direction, the work [20] has proposed MECLA, a mobile platform for the evaluation of classification algorithms based on the Android operating system.

4.4 Results and Analysis

In order to evaluate the state-of-the-art HAR systems, it is required to first define a taxonomy that allows us to compare and analyze them within groups that share common characteristics. To the best of our knowledge, no comprehensive taxonomy has been proposed in the literature to encompass all sorts of HAR systems.

For tests, an openly accessible database has to be gathered. The database [1] consists of twelve exercises: Standing, Sitting, Walking, Lying Down, Stand-to-Sit, Walking-downstairs, Walking-upstairs, Sit-to-Lie, Sit-to-Stand, Lie-to-Sit, Lie-to-Stand, and Stand-to-Lie. An aggregate of 7767 and 3162 occasions can be utilized for preparing and testing exercises separately. Every occasion has 561 fundamental highlights. It is to be noticed that in the database utilized as a part of this work, the number of tests for preparing and testing distinctive action is not uniformly disseminated. A few exercises contain an extensive number of tests though some of them have few experiments.

We started a network structure with 10 hidden units for layer-1 and layer-2, then increase the number of hid-den units up to 860.

The rest of the structures has a different number of hidden units for layer-1 and layer-2. The total number of epochs is 1000. Momentum = 0.7, learning rate = 2, and batch size = 881. The reconstruction error of Restricted Boltzmann Machine (RBM) layer for DBN structure-16 is plotted in Figure 4.2. It is seen that reconstruction error rate is decreased sharply as the number of epochs increases. The weight matrix of a trained DBN is used as the initial weight of an artificial neural network (ANN) where the out-puts of ANN is kept same as the number of activity types. ANN is trained by using a backpropagation algorithm and an activation function of optimal tan hyperbolic [39]. Scaling factor for the learning rate in each epoch for ANN is 1, learning rate is 2, and momentum is 0.5. The training error rate of ANN for a DBN structure is presented in Figure 4.3. It is observed that error rate decreases as the number of epoch increases.

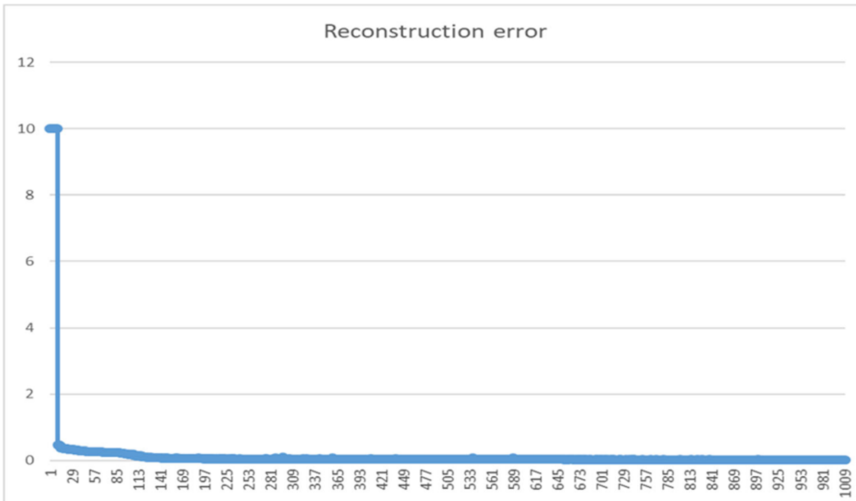

Figure 4.2 Reconstruction error.

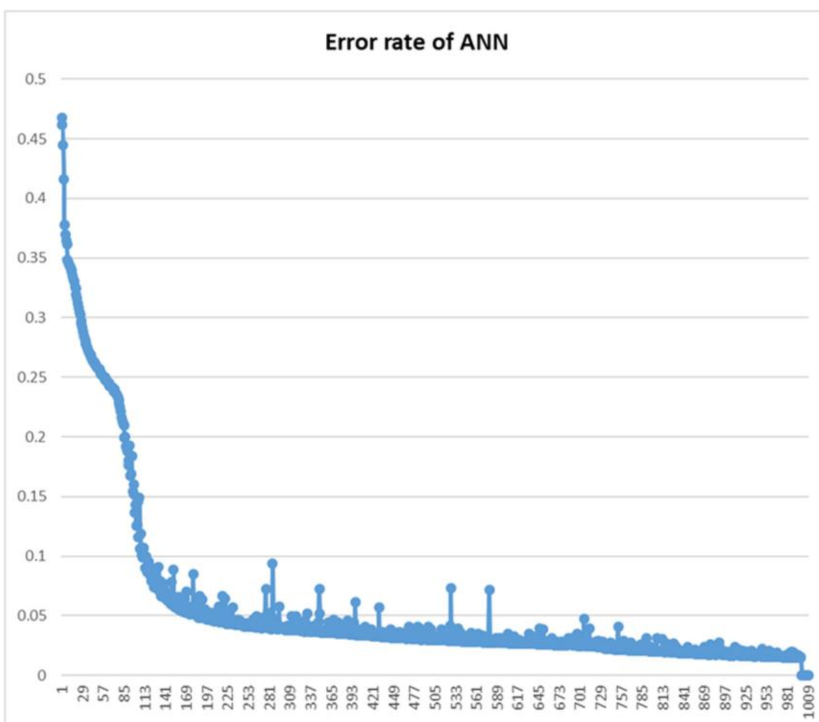

Figure 4.3 Error rate of ANN.

Table 4.1 Different Types of Activities in the Training Data and Error Rate

Activity	Recognition Rate (%)	Mean
Standing	98.60	87.65
Sitting	95.72	
Lying Down	95.15	
Walking	92.34	
Walking Upstairs	94.65	
Walking Downstairs	95.36	
Stand-to-sit	79.37	
Sit-to Stand	86.18	
Sit-to-Lie	80.39	
Lie-to-Sit	73.10	
Stand-to-Lie	81.21	
Lie-to-Stand	79.72	

Table 4.1 shows our experimental results about different types of activities through DBN. As we can see, all the six basic activities achieve, i.e., Standing, Sitting, Lying Down, Walking, Walking-downstairs and Walking-upstairs, have achieved the accuracies over 90%. The result of distinguishing short-term activities (Stand-to-Sit, Sit-to-Lie, Sit-to-Stand, Lie-to-Sit, Lie-to-Stand, and Stand-to-Lie) with high statistical similarities cannot achieve as what we expect, which should be our future work. Our proposed approach achieved an average accuracy of 87.65% which is pretty good for this data set. It is computed as the sum of all recognition rates divided by 12

(activities). Therefore, the excellent performance of the proposed approach is experimentally verified.

Table 4.2 depicts the performance of different features ranked based on information gain with different classifier learning approaches, and time taken to build the model. We examine the effects of using the top 2,8,16, 32, 64, 128, 256 and 561 (all features) for HAR.

Table 4.2 Recognition Accuracies

Number of Features	**NB**	**DT**	**DBN**
2	49.45	**56.30**	53.18
8	48.26	**61.39**	60.18
16	48.57	**69.02**	67.84
32	52.34	70.24	**71.74**
64	56.10	77.30	**77.51**
128	55.31	81.46	**88.97**
256	53.86	88.81	**90.55**
561	79.00	91.00	**91.89**

Tables 4.2 shows the performance for our approach. compared with others' methods, in terms of classification accuracy. As can be seen, Decision Tree (DT) performs well with the recognition accuracy (91%). The Naïve Bayes Classifier performs moderately well for such a large dataset, with 79% accuracy. The best performing classifier is the proposed DBN model as its accuracy is 91.89%. In detail, it performs with an accuracy of 90.55% with 256 features and 91.89% with 561 features. It also performs well with an accuracy of 88.97% by using only 128 features, which has its advantage over the other two approaches when only 128 features are used. A good trade-off between accuracy and model building time is necessary for a

smartphone-based activity recognition system, as real time activity monitoring needs an accurate model to be built dynamically from the captured data.

Meanwhile, there is much room to extend the presented work. An observation from Table 4.2 is that Decision Tree has the best performance through 2, 8, and 16 features. Therefore, which approach can use the fewest features to achieve the same satisfying accuracy is an interesting topic as our future work.

CHAPTER 5

CONCLUSION

The idea of this thesis is to build a deep learning model to solve the Human Activity Recognition (HAR) problems. This thesis surveys the state-of-the-art work in human activity recognition based on wearable sensors. HAR systems are introduced according to their response time and learning scheme. Meanwhile, several systems are also qualitatively compared in terms of response time, learning approach, obtrusiveness, flexibility, recognition accuracy, and other important design issues. The fundamentals of feature extraction and machine learning are also included, as they are important components of every HAR system.

In this thesis, we explore the performance of a Deep Belief Network (DBN) for HAR by using wearable body sensors. We describe how to extract the robust features from the sensor signals and use them to train DBN. We also show how to find the optimal DBN architecture by varying the hyper-parameters of a DBN structure. Our experimental results using the proposed DBN method on a public human activity recognition dataset shows its superiority as compared to traditional approaches. The overall accuracy obtained by the proposed method is over 91%.

As future work, we plan to try with some different kinds of functions and methods. We can also attempt to perceive body exercises from the point of view of inadequate portrayal and arbitrary projections in real-time environments. Various ideas are also proposed for future research to extend this field to more realistic and pervasive scenarios. Multiple applications such as intelligent homes and smart healthcare may also be realized using the model presented in the thesis.

APPENDIX

SOURCE CODE

```
Function test_example_DBN.
clear all; close all; clc;
load mnist_uint8;
addpath('E:\00.lab\Jialun\000DeepLearning\DeepLearnToolbox9faf641\DeepLearnT
oolbox-9faf641');
train_x = double(train_x) / 255;
test_x  = double(test_x)  / 255;
train_y = double(train_y);
test_y  = double(test_y);
%%  ex1 train a 100 hidden unit RBM and visualize its weights
rand('state',0)
dbn.sizes = [100];
opts.numepochs =   1;
opts.batchsize = 100;
opts.momentum  =   0;
opts.alpha     =   1;
dbn = dbnsetup(dbn, train_x, opts);
dbn = dbntrain(dbn, train_x, opts);
figure; visualize(dbn.rbm{1}.W');   %  Visualize the RBM weights
%train dbn
```

```
dbn.sizes = [64  70];

opts.numepochs =   1;

opts.batchsize = 100;

opts.momentum  =   0;

opts.alpha     =   1;

dbn = dbnsetup(dbn, train_x, opts);

dbn = dbntrain(dbn, train_x, opts);

nn = dbnunfoldtonn(dbn, 10);

nn.activation_function = 'sigm';

%train nn

opts.numepochs =  1;

opts.batchsize = 100;

nn = nntrain(nn, train_x, train_y, opts);

labels = nnpredict(nn, test_x);

[dummy, expected] = max(test_y,[],2);

err = find(labels~=expected);

assert(er < 0.10, 'Too big error');

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

function dbn = dbnsetup(dbn, x, opts)

    n = size(x, 2);

    dbn.sizes = [n, dbn.sizes];

    for u = 1 : numel(dbn.sizes) - 1

        dbn.rbm{u}.alpha    = opts.alpha;
```

```
        dbn.rbm{u}.momentum = opts.momentum;
        dbn.rbm{u}.W  = zeros(dbn.sizes(u + 1), dbn.sizes(u));
        dbn.rbm{u}.vW = zeros(dbn.sizes(u + 1), dbn.sizes(u));
        dbn.rbm{u}.b  = zeros(dbn.sizes(u), 1);
        dbn.rbm{u}.vb = zeros(dbn.sizes(u), 1);
        dbn.rbm{u}.c  = zeros(dbn.sizes(u + 1), 1);
        dbn.rbm{u}.vc = zeros(dbn.sizes(u + 1), 1);
    end
end
//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
function dbn = dbntrain(dbn, x, opts)
    n = numel(dbn.rbm);
    dbn.rbm{1} = rbmtrain(dbn.rbm{1}, x, opts);
    for i = 2 : n
        x = rbmup(dbn.rbm{i - 1}, x);
        dbn.rbm{i} = rbmtrain(dbn.rbm{i}, x, opts);
    end
end
//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
function nn = dbnunfoldtonn(dbn, outputsize)
%   layer of size outputsize added.
    if(exist('outputsize','var'))
        size = [dbn.sizes outputsize];
```

```
    else
        size = [dbn.sizes];
    end
    nn = nnsetup(size);
    for i = 1 : numel(dbn.rbm)
        nn.W{i} = [dbn.rbm{i}.c dbn.rbm{i}.W];
    end
end
/////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
function nn = nnapplygrads(nn)
% weights and biases
    for i = 1 : (nn.n - 1)
        if(nn.weightPenaltyL2>0)
            dW=nn.dW{i}+nn.weightPenaltyL2*[zeros(size(nn.W{i},1),1)
nn.W{i}(:,2:end)];
        else
            dW = nn.dW{i};
        end
        dW = nn.learningRate * dW;
        if(nn.momentum>0)
            nn.vW{i} = nn.momentum*nn.vW{i} + dW;
            dW = nn.vW{i};
        end
```

```
        nn.W{i} = nn.W{i} - dW;
    end
end
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
function nn = nnbp(nn)
%NNBP performs backpropagation
    n = nn.n;
    sparsityError = 0;
    switch nn.output
        case 'sigm'
            d{n} = - nn.e .* (nn.a{n} .* (1 - nn.a{n}));
        case {'softmax','linear'}
            d{n} = - nn.e;
    end
    for i = (n - 1) : -1 : 2
        % Derivative of the activation function
        switch nn.activation_function
            case 'sigm'
                d_act = nn.a{i} .* (1 - nn.a{i});
            case 'tanh_opt'
                d_act = 1.7159 * 2/3 * (1 - 1/(1.7159)^2 * nn.a{i}.^2);
        end
        if(nn.nonSparsityPenalty>0)
```

```
            pi = repmat(nn.p{i}, size(nn.a{i}, 1), 1);
            sparsityError = [zeros(size(nn.a{i},1),1) nn.nonSparsityPenalty * (-nn.sparsityTarget ./ pi + (1 - nn.sparsityTarget) ./ (1 - pi))];
        end
        % Backpropagate first derivatives
        if i+1==n % in this case in d{n} there is not the bias term to be removed
            d{i} = (d{i + 1} * nn.W{i} + sparsityError) .* d_act; % Bishop (5.56)
        else % in this case in d{i} the bias term has to be removed
            d{i} = (d{i + 1}(:,2:end) * nn.W{i} + sparsityError) .* d_act;
        end
        if(nn.dropoutFraction>0)
            d{i} = d{i} .* [ones(size(d{i},1),1) nn.dropOutMask{i}];
        end
    end
    for i = 1 : (n - 1)
        if i+1==n
            nn.dW{i} = (d{i + 1}' * nn.a{i}) / size(d{i + 1}, 1);
        else
            nn.dW{i} = (d{i + 1}(:,2:end)' * nn.a{i}) / size(d{i + 1}, 1);
        end
    end
end
function nnupdatefigures(nn,fhandle,L,opts,i)
```

```
if i > 1 %dont plot first point, its only a point
    x_ax = 1:i;
    % create legend
    if opts.validation == 1
        M = {'Training','Validation'};
    else
        M = {'Training'};
    end
    %create data for plots
    if strcmp(nn.output,'softmax')
        plot_x       = x_ax';
        plot_ye      = L.train.e';
        plot_yfrac   = L.train.e_frac';
    else
        plot_x       = x_ax';
        plot_ye      = L.train.e';
    end
    if opts.validation == 1
        plot_x       = [plot_x, x_ax'];
        plot_ye      = [plot_ye,L.val.e'];
    end
    if opts.validation == 1 && strcmp(nn.output,'softmax')
        plot_yfrac   = [plot_yfrac, L.val.e_frac'];
```

```
    end
%    plotting
    figure(fhandle);
    if strcmp(nn.output,'softmax')  %also plot classification error
        p1 = subplot(1,2,1);
        plot(plot_x,plot_ye);
        xlabel('Number of epochs'); ylabel('Error');title('Error');
        title('Error')
        legend(p1, M,'Location','NorthEast');
        set(p1, 'Xlim',[0,opts.numepochs + 1])
        p2 = subplot(1,2,2);
        plot(plot_x,plot_yfrac);
        xlabel('Number of epochs'); ylabel('Misclassification rate');
        title('Misclassification rate')
        legend(p2, M,'Location','NorthEast');
        set(p2, 'Xlim',[0,opts.numepochs + 1])
    else
        p = plot(plot_x,plot_ye);
        xlabel('Number of epochs'); ylabel('Error');title('Error');
        legend(p, M,'Location','NorthEast');
        set(gca, 'Xlim',[0,opts.numepochs + 1])
    end
    drawnow;
```

```
end
end
///////////////////////////////////////////////////////////////////////////////////////////////////////////////////
function [nn, L, K]  = nntrain(nn, train_x, train_y, opts, val_x, val_y)
assert(isfloat(train_x), 'train_x must be a float');
assert(nargin == 4 || nargin == 6,'number ofinput arguments must be 4 or 6')
loss.train.e               = [];
loss.train.e_frac          = [];
loss.val.e                 = [];
loss.val.e_frac            = [];
opts.validation = 0;
if nargin == 6
    opts.validation = 1;
end
fhandle = [];
if isfield(opts,'plot') && opts.plot == 1
    fhandle = figure();
end
m = size(train_x, 1);
batchsize = opts.batchsize;
numepochs = opts.numepochs;

numbatches = m / batchsize;
```

```
assert(rem(numbatches, 1) == 0, 'numbatches must be a integer');
L = zeros(numepochs*numbatches,1);
K=zeros(numepochs,1);
n = 1;
for i = 1 : numepochs
    tic;
    kk = randperm(m);
    for l = 1 : numbatches
        batch_x = train_x(kk((l - 1) * batchsize + 1 : l * batchsize), :);
        %Add noise to input (for use in denoising autoencoder)
        if(nn.inputZeroMaskedFraction ~= 0)
            batch_x = batch_x.*(rand(size(batch_x))>nn.inputZeroMaskedFraction);
        end
        batch_y = train_y(kk((l - 1) * batchsize + 1 : l * batchsize), :);
        nn = nnff(nn, batch_x, batch_y);
        nn = nnbp(nn);
        nn = nnapplygrads(nn);
        L(n) = nn.L;
        n = n + 1;
    end
    t = toc;

    if opts.validation == 1
```

```
        loss = nneval(nn, loss, train_x, train_y, val_x, val_y);
        str_perf = sprintf('; Full-batch train mse = %f, val mse = %f', loss.train.e(end),
loss.val.e(end));
    else
        loss = nneval(nn, loss, train_x, train_y);
        str_perf = sprintf('; Full-batch train err = %f', loss.train.e(end));
    end
    if ishandle(fhandle)
        nnupdatefigures(nn, fhandle, loss, opts, i);
    end
    disp(['epoch ' num2str(i) '/' num2str(opts.numepochs) '. Took ' num2str(t) ' seconds'
'. Mini-batch mean squared error on training set is ' num2str(mean(L((n-
numbatches):(n-1)))) str_perf]);
    nn.learningRate = nn.learningRate * nn.scaling_learningRate;
    K(i)=loss.train.e(end);
end
end
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
function nn = nnff(nn, x, y)
% performs a feedforward pass
    n = nn.n;
    m = size(x, 1);
    x = [ones(m,1) x];
```

```
nn.a{1} = x;

%feedforward pass

for i = 2 : n-1

    switch nn.activation_function

        case 'sigm'

            % Calculate the unit's outputs (including the bias term)

            nn.a{i} = sigm(nn.a{i - 1} * nn.W{i - 1}');

        case 'tanh_opt'

            nn.a{i} = tanh_opt(nn.a{i - 1} * nn.W{i - 1}');

    end

    %dropout

    if(nn.dropoutFraction > 0)

        if(nn.testing)

            nn.a{i} = nn.a{i}.*(1 - nn.dropoutFraction);

        else

            nn.dropOutMask{i} = (rand(size(nn.a{i}))>nn.dropoutFraction);

            nn.a{i} = nn.a{i}.*nn.dropOutMask{i};

        end

    end

    %calculate running exponential activations for use with sparsity

    if(nn.nonSparsityPenalty>0)

        nn.p{i} = 0.99 * nn.p{i} + 0.01 * mean(nn.a{i}, 1);

    end
```

```
        %Add the bias term
        nn.a{i} = [ones(m,1) nn.a{i}];
    end
    switch nn.output
        case 'sigm'
            nn.a{n} = sigm(nn.a{n - 1} * nn.W{n - 1}');
        case 'linear'
            nn.a{n} = nn.a{n - 1} * nn.W{n - 1}';
        case 'softmax'
            nn.a{n} = nn.a{n - 1} * nn.W{n - 1}';
            nn.a{n} = exp(bsxfun(@minus, nn.a{n}, max(nn.a{n},[],2)));
            nn.a{n} = bsxfun(@rdivide, nn.a{n}, sum(nn.a{n}, 2));
    end
    %error and loss
    nn.e = y - nn.a{n};
    switch nn.output
        case {'sigm', 'linear'}
            nn.L = 1/2 * sum(sum(nn.e .^ 2)) / m;
        case 'softmax'
            nn.L = -sum(sum(y .* log(nn.a{n}))) / m;
    end
end
function nn = nnsetup(architecture)
```

```
%NNSETUP creates a Feedforward Backpropagate Neural Network
% nn = nnsetup(architecture) returns an neural network structure with
n=numel(architecture)
% layers, architecture being a n x 1 vector of layer sizes e.g. [784 100 10]

nn.size   = architecture;
nn.n      = numel(nn.size);

nn.activation_function              = 'tanh_opt';
layers: 'sigm' (sigmoid) or 'tanh_opt' (optimal tanh).
nn.learningRate                     = 2;
nn.momentum                         = 0.5;
    nn.scaling_learningRate            = 1;
nn.weightPenaltyL2                  = 0;
    nn.nonSparsityPenalty              = 0;
    nn.sparsityTarget                  = 0.05;
    nn.inputZeroMaskedFraction         = 0;
    nn.dropoutFraction=0;
    nn.testing                         = 0;
    nn.output                          = 'sigm';
    for i = 2 : nn.n
        nn.W{i - 1} = (rand(nn.size(i), nn.size(i - 1)+1) - 0.5) * 2 * 4 * sqrt(6 / (nn.size(i)
+ nn.size(i - 1)));
```

```
        nn.vW{i - 1} = zeros(size(nn.W{i - 1}));
        nn.p{i}     = zeros(1, nn.size(i));
    end
end
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
function [loss] = nneval(nn, loss, train_x, train_y, val_x, val_y)
% Evaluates performance of neural network
% Returns a updated loss struct
assert(nargin == 4 || nargin == 6, 'Wrong number of arguments');
nn.testing = 1;
% training performance
nn                    = nnff(nn, train_x, train_y);
loss.train.e(end + 1) = nn.L;
% validation performance
if nargin == 6
    nn                    = nnff(nn, val_x, val_y);
    loss.val.e(end + 1)   = nn.L;
end
nn.testing = 0;
%calc misclassification rate if softmax
if strcmp(nn.output,'softmax')
    [er_train, dummy]               = nntest(nn, train_x, train_y);
    loss.train.e_frac(end+1)    = er_train;
```

```
    if nargin == 6
        [er_val, dummy]             = nntest(nn, val_x, val_y);
        loss.val.e_frac(end+1)  = er_val;
    end
end
end
```

//

```
function nn = nnff(nn, x, y)
%Performs a feedforward pass
    n = nn.n;
    m = size(x, 1);
    x = [ones(m,1) x];
    nn.a{1} = x;
    %feedforward pass
    for i = 2 : n-1
        switch nn.activation_function
            case 'sigm'
                % Calculate the unit's outputs
                nn.a{i} = sigm(nn.a{i - 1} * nn.W{i - 1}');
            case 'tanh_opt'
                nn.a{i} = tanh_opt(nn.a{i - 1} * nn.W{i - 1}');
        end
        %dropout
```

```
    if(nn.dropoutFraction > 0)
        if(nn.testing)
            nn.a{i} = nn.a{i}.*(1 - nn.dropoutFraction);
        else
            nn.dropOutMask{i} = (rand(size(nn.a{i}))>nn.dropoutFraction);
            nn.a{i} = nn.a{i}.*nn.dropOutMask{i};
        end
    end
    %calculate running exponential activations for use with sparsity
    if(nn.nonSparsityPenalty>0)
        nn.p{i} = 0.99 * nn.p{i} + 0.01 * mean(nn.a{i}, 1);
    end
    %Add the bias term
    nn.a{i} = [ones(m,1) nn.a{i}];
end
switch nn.output
    case 'sigm'
        nn.a{n} = sigm(nn.a{n - 1} * nn.W{n - 1}');
    case 'linear'
        nn.a{n} = nn.a{n - 1} * nn.W{n - 1}';
    case 'softmax'
        nn.a{n} = nn.a{n - 1} * nn.W{n - 1}';
        nn.a{n} = exp(bsxfun(@minus, nn.a{n}, max(nn.a{n},[],2)));
```

```
            nn.a{n} = bsxfun(@rdivide, nn.a{n}, sum(nn.a{n}, 2));
    end
    %error and loss
    nn.e = y - nn.a{n};
    switch nn.output
        case {'sigm', 'linear'}
            nn.L = 1/2 * sum(sum(nn.e .^ 2)) / m;
        case 'softmax'
            nn.L = -sum(sum(y .* log(nn.a{n}))) / m;
    end
end
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
clc;
clear all;
addpath(genpath('.\'));
load Actvity_Dataset_Normalized_0_1_Zia.mat;
rand('state',0)
dbn.sizes = [60 20];
opts.numepochs =  10;
opts.batchsize = 881;
opts.momentum  = 0;
opts.alpha     =   0.0000000000001;
dbn = dbnsetup(dbn, train_x, opts);
```

```
dbn = dbntrain(dbn, train_x, opts);

nn = dbnunfoldtonn(dbn, 12);

nn.activation_function = 'sigm';

opts.numepochs =  1000;

opts.batchsize = 881;

[nn, L]  = nntrain(nn, train_x, train_y, opts);

labels = nnpredict(nn, test_x);

[dummy, expected] = max(test_y,[],2);

good = find(labels == expected);

Accuracy=size(good,1)/3162*100;

fprintf(1,"The accuracy is = %.2f%%.\n", Accuracy);
```

REFERENCES

Lichman,M., UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School ofInformation and Computer Science, 2013.

Clarkson, B.P., Life patterns: structure from wearable sensors (Ph.D. thesis), Massachusetts Institute of Technology. 2002.

Avci, A., Bosch S., Marin-PerianuM.,Marin-Perianu R., Havinga P.Activity recognition using inertial sensing for healthcare, wellbeingand sports applications: a survey. In: International Conference on Architecture of Computing Systems, pp. 1–10. ARCS, Berlin, 2010.

Nham, B., Siang liulue, K., Yeung, S., Predicting mode of transport from iphone accelerometer data. Technical Report, Stanford University, 2008.

Minnen, D., Starner, T., Ward, J., Lukowicz, P., and Troester, G., Recognizing and discovering human actions from on-body sensor data. In Proc. IEEE Int. Conf. Multimedia Expo. 1545–1548, 2005. Bao, L., Intille, S., Activity recognition from user-annotated acceleration data. In: International Conference on Pervasive Computing, pp. 1–17. Springer, Berlin, Heidelberg, 2004.

Aggarwal, J., and Ryoo, M. S., Human activity analysis: a review.ACM Comput. Surv. 43(3):1–16, 2011.

Tasoulis, S. K., Doukas, N., Plagianakos, V. P., and Maglogiannis, I., Statistical data mining of streaming motion data for activity and fall recognition in assistive environments. Neurocomputing 107: 87–96, 2013.

Ebied, H. M., Feature extraction using PCA and Kernel-PCA for face recognition. 8th International Conference on Informatics and Systems (INFOS), 72–77, 2017.

D. Townsend, F. Knoefel, R. Goubran, Privacy versus autonomy: a tradeoff model for smart home monitoring technologies. In: 2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 4749–4752. EMBC, 2011.

Khan, A. M., Lee, Y. K., Lee, S. Y., and Kim, T. S., A triaxial accelerometer-based physical-activity recognition via augmented signal features and a hierarchical recognizer. IEEE Trans. Inf. Technol. Biomed. 14(5):1166–1172, 2010.

Maurer, U., Smailagic, A., Siewiorek, D., and Deisher, M., Activity recognition and monitoring using multiple sensors on different body positions. In: Proc. Int. Workshop Wearable Implantable Body Sens. Netw. pp. 113–116, 2006.

Kern, N., Schiele, B., Junker, H., Lukowicz, P., and Troster, G.,Wearable sensing toannotate meeting recordings. Pers. Ubiquit. Comput. 7:263–274, 2003.

Lin, W., Sun, M.-T., Poovandran, R, Zhang, Z., Human activity recognition for video surveillance. In: IEEE International Symposium on Circuits and Systems, pp. 2737–2740. IEEE, Seattle, 2008.

Ravi, D., Wong, C., Lo, B., and Yang, G. Z., A deep learning approach to on-node sensor data analytics for mobile or wearable devices. IEEE Journal of Biomedical and Health Informatics. 21(1):56–64, 2017.

Hinton, G. E., Osindero, S., and Teh, Y.-W., A fast learning algorithm for deep belief nets. Neural Comput. 18(7):1527–1554, 2006.

Uddin,M. Z., Hassan,M.M., Almogren, A., Zuair, M., Fortino, G.and Torresen, J., A facial expression recognition system using robust face features from depth videos and deep learning. Comput.Electr. Eng., 2017.

Bulling, A., Blanke, U., and Schiele, B., A tutorial on human activity recognition using body-worn inertial sensors. ACM Computing Surveys (CSUR) 46(3):33, 2014.

Behera, A., Hogg, D., Cohn, A., Egocentric activity monitoring and recovery. In: Asian Conference on Computer Vision, pp. 519–532. Springer, Berlin, Heidelberg, 2012.

Cornacchia, M., Ozcan, K., Zheng, Y., and Velipasalar, S., A survey on activity detection and classification using wearable sensors. IEEE Sensors 17(2):386–403, 2017.

Zebin, T., Scully, P. J., and Ozanyan, K. B., Human activity recognition with inertial sensors using a deep learning approach. InSENSORS, 2016 I.E. (pp. 1–3). IEEE, 2016.

Cheng, L., Guan, Y., Zhu, K., and Li, Y., Recognition of human activities usingmachine learning methods with wearable sensors. In Computing and Communication Workshop and Conference (CCWC), 2017 I.E. 7th Annual (pp. 1–7). IEEE, 2017.

M. M. Hassan, A. Gumaei, G. Aloi, G. Fortino and M. Zhou, "A Smartphone-Enabled Fall Detection Framework for Elderly People in Connected Home Healthcare," IEEE Network, vol. 33, no. 6, pp. 58-63, Nov.-Dec. 2019

H. Zhang, Y. Li, Z. Lv, A. K. Sangaiah and T. Huang, "A real-time and ubiquitous network attack detection based on deep belief network and support vector machine," in IEEE/CAA Journal of Automatica Sinica, vol. 7, no. 3, pp. 790-799, May 2020

G. Wang, J. Qiao, J. Bi, Q. -S. Jia and M. Zhou, "An Adaptive Deep Belief Network With Sparse Restricted Boltzmann Machines," IEEE Transactions on Neural Networks and Learning Systems, vol. 31, no. 10, pp. 4217-4228, Oct. 2020

G. Wang, J. Qiao, J. Bi, W. Li and M. Zhou, “TL-GDBN: Growing Deep Belief Network with Transfer Learning,” IEEE Transactions on Automation Science and Engineering, 16(2), pp. 874 – 885, April 2019

S. Qiu et al., "Body Sensor Network-Based Robust Gait Analysis: Toward Clinical and at Home Use," in IEEE Sensors Journal, vol. 19, no. 19, pp. 8393-8401, 1 Oct.1, 2019

Q. Kang, L. Shi, M. Zhou, X. Wang, Q. Wu, and Z. Wei, “A Distance-based Weighted Undersampling Scheme for Support Vector Machines and its Application to Imbalanced Classification,” IEEE Transactions on Neural Networks and Learning Systems, 29(9), pp. 4152 – 4165, Sept. 2018

P. Y. Zhang, S. Shu, and M. C. Zhou, “An online fault detection model and strategies based on SVM-grid in clouds,” IEEE/CAA J. Autom. Sinica, vol. 5, no. 2, pp. 445-456, Mar. 2018

S. H. Teng, N. Q. Wu, H. B. Zhu, L. Y. Teng, and W. Zhang, “SVM-DT-based adaptive and collaborative intrusion detection,” IEEE/CAA J. Autom. Sinica, vol. 5, no. 1, pp. 108-118, Jan. 2018

E. F. Ohata et al., "Automatic detection of COVID-19 infection using chest X-ray images through transfer learning," in IEEE/CAA Journal of Automatica Sinica, vol. 8, no. 1, pp. 239-248, January 2021

H. Han, M. Zhou, and Y. Zhang, “Can Virtual Samples Solve Small Sample Size Problem of KISSME in Pedestrian Re-identification of Smart Transportation?” IEEE Trans. on Intelligent Transportation Systems, 21(9), pp. 3766-3776, Sept. 2020.

H. Han, M. Zhou, Y. Zhang, and W. Cao, “KISS+ for Rapid and Accurate Pedestrian Re-identification," IEEE Transactions on Intelligent Transportation Systems, DOI: 10.1109/TITS.2019.2958741, Online 2020.

Y. Wen, L. Zhang, L. He, and M. Zhou “Incorporation of Structural Tensor and Driving Force into Log-Demons for Large-Deformation Image Registration", IEEE Trans. on Image Processing, 28(12), pp. 6091-6102, Dec. 2019.

S. Li, M. Zhou, and X. Luo, “Modified Primal-Dual Neural Networks for Motion Control of Redundant Manipulators with Dynamic Rejection of Harmonic Noises,” IEEE Transactions on Neural Networks and Learning Systems, 29(10), pp. 4791 - 4801, Oct. 2018.

S. Gao, M. Zhou, Y. Wang, J. Cheng, H. Yachi, and J. Wang, "Dendritic neuron model with effective learning algorithms for classification, approximation and prediction," IEEE Transactions on Neural Networks and Learning Systems, 30(2), pp. 601 - 614, Feb. 2019.

Z. Huang, X. Xu, H. Zhu, and M. Zhou, “An Efficient Group Recommendation Model With Multiattention-based Neural Networks,” IEEE Transactions on Neural Networks and Learning Systems, vol. 31, no. 11, pp. 4461-4474, Nov. 2020.

Q. Kang, X. Chen, S. Li and M. Zhou, "A Noise-Filtered Under-Sampling Scheme for Imbalanced Classification," in IEEE Transactions on Cybernetics, vol. 47, no. 12, pp. 4263-4274, Dec. 2017.

Q. Kang, S. Liu, M. Zhou, and S. Li, “A weight-incorporated similarity-based clustering ensemble method based on swarm intelligence,” Knowledge-based Systems, 104, pp.156–164, 2016.

F. M. Shakiba and M. Zhou, "Novel Analog Implementation of a Hyperbolic Tangent Neuron in Artificial Neural Networks," IEEE Transactions on Industrial Electronics, doi: 10.1109/TIE.2020.3034856, 2020

Davila, J.; Cretu, A.-M.; Zaremba, M. Iterative Learning for Human Activity Recognition from Wearable Sensor Data. In Proceedings of the 3rd International Electronic Conference on Sensors and Applications, Barcelona, Spain, 15–30 November 2016.