Instructor Activity Recognition Using Smartwatch and Smartphone Sensors

Zayed Uddin Chowdhury

INSTRUCTOR ACTIVITY RECOGNITION USING SMARTWATCH AND SMARTPHONE SENSORS

by

ZAYED UDDIN CHOWDHURY

(Under the Direction of Pradipta De)

ABSTRACT

During a classroom session, an instructor performs several activities, such as writing on the board, speaking to the students, gestures to explain a concept. A record of the time spent in each of these activities could be valuable information for the instructors to virtually observe their own style of instruction. It can help in identifying activities that engage the students more, thereby enhancing teaching effectiveness and efficiency. In this work, we present a preliminary study on profiling multiple activities of an instructor in the classroom using smartwatch and smartphone sensor data. We use 2 benchmark datasets to test out the feasibility of classifying the activities. Comparing multiple machine learning techniques, we finally propose a hybrid deep recurrent neural network based approach that performs better than the other techniques.

INDEX WORDS: Instructor activity recognition, Neural network, Smartwatch and smartphone sensors

INSTRUCTOR ACTIVITY RECOGNITION USING SMARTWATCH AND SMARTPHONE SENSORS

by

ZAYED UDDIN CHOWDHURY

B.S., Bangladesh University of Engineering and Technology, Bangladesh, 2015

A Thesis Submitted to the Graduate Faculty of Georgia Southern University in Partial

Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

INSTRUCTOR ACTIVITY RECOGNITION USING SMARTWATCH AND SMARTPHONE SENSORS

by

ZAYED UDDIN CHOWDHURY

Electronic Version Approved:
May 2020

DEDICATION

I dedicate this thesis to my family, my honorable teachers, and all of my friends who inspired me to cross all the hurdles. To my fellow GAs for always covering my back. To my C5 buddies, life is never boring in this apartment.

ACKNOWLEDGMENTS

I would like to sincerely thank Dr. Pradipta De for his guidance throughout these years and for being such a great mentor. I would also like to thank Dr. Andrew Allen, Dr. Mehdi Allahyari for their assistance throughout the project. I can never thank enough Dr. Muralidhar Medidi, Rana and Tasnia for all the supports during difficult times.

TABLE OF CONTENTS

LIST OF TABLES

Page

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

This chapter will first cover the background behind this research and its importance. It will briefly describe the research history on this t opic. Finally, the contributions of this research is explained.

1.1 Motivation

Methodology for evaluating an instructor's performance is an important topic of classroom observational study, as it has a direct effect on students' academic performance. Currently, instructor evaluation is done mainly through student feedback based on a standard survey mechanism known as "Student's Evaluating Teaching (SET)."[1] However, there is a possibility of automating this whole procedure, if we can find a correlation between an instructor's activity and students' attentiveness. There are already researches occurring that examines student attentiveness.[2] We are focusing on tracking the instructor's activities.

Researches on the field of sensor-based HAR is gaining momentum with the increasing use of smart devices like the smartphone and smartwatch. These devices contain multiple sensors like an accelerometer, gyro sensor, microphone, etc. Some other important reasons for the popularity of sensor-based activity recognition[3] are their compact size, low-power requirement, low cost, non-intrusiveness in contrast to the previously popular audio and video data based activity recognition techniques.[4]

1. N. Nida et al., "Bag of Deep Features for Instructor Activity Recognition in Lecture Room," in *International Conference on Multimedia Modeling* (Springer, 2019), 481–492.

2. N. Veliyath et al., "Modeling Students' Attention in the Classroom using Eyetrackers," in *Proceedings of the 2019 ACM Southeast Conference* (ACM, 2019), 2–9.

3. P. Casale, O. Pujol, and P. Radeva, "Human Activity Recognition from Accelerometer Data using a Wearable Device," in *Iberian Conference on Pattern Recognition and Image Analysis* (2011), 289–296.

4. S. Ke et al., "A Review on Video-based Human Activity Recognition," *computers* 2, no. 2 (2013): 88–131.

There are some challenges in automating human activity recognition. Same activity may be performed by different persons in different manners. Also, the same person can perform an activity in different manners at different times based on the environment, physical or mental condition. This is known as interclass variability. On the other hand, there are similarities between different activities. For example, walking and running have similarities between them. Then there is the class imbalance problem. It is a very well known issue in machine learning. A person may rarely do a specific activity compared to other activities. For example, an instructor may sit down in the classroom for very little time than standing up or writing on the board. As a result, it becomes difficult for a machine-learning algorithm to classify rare activities. Activities can be performed simultaneously. For example, an instructor may work on the computer while he is sitting. This type of machine learning problem is known as multi-label classification.[5]

1.2 Contribution

There are two major outcomes to this research. The first one is the comparative analysis (both performance-wise and resource usage wise) of multiple machine learning methods for activity recognition using a smartwatch and smartphone sensor data. We have used 2 datasets, one[6] for single activity detection and the other one[7] for concurrent activity detection. For the first dataset, we used only smartwatch sensors and tried to classify 5 activities: walking, sitting, standing, typing, and writing. For the second dataset, we classified 4 activities: walking, sitting, working on the computer, and standing. Our research shows the

5. G. Tsoumakas and I. Katakis, “Multi-label Classification: An overview,” *International Journal of Data Warehousing and Mining (IJDWM)* 3, no. 3 (2007): 1–13.

6. Gary M Weiss, “WISDM Smartphone and Smartwatch Activity and Biometrics Dataset,” *UCI Machine Learning Repository: WISDM Smartphone and Smartwatch Activity and Biometrics Dataset Data Set*, 2019,

7. Yonatan Vaizman, Katherine Ellis, and Gert Lanckriet, “Recognizing detailed human context in the wild from smartphones and smartwatches,” *IEEE Pervasive Computing* 16, no. 4 (2017): 62–74.

performance of both traditional approaches and deep learning approaches. The models we have used are binary relevance,[8] Random Forest,[9] Decision Tree,[10] Logistic Regression,[11] vanilla neural network,[12] Long Short Term Memory (LSTM)[13] based Recurrent Neural Network (RNN), Convolutional Neural Network (CNN),[14] and a Hybrid Model known as Long-term Recurrent Convolutional Neural Network (LRCN).[15]

The second outcome involves developing a hybrid machine learning model that outperforms previous best results on the 2 benchmark datasets mentioned above. We were able to achieve an average F1 score of 96% for one dataset and an average balanced accuracy of 77% on the other highly imbalanced dataset.

8. K. Trohidis et al., "Multi-label Classification of Music into Emotions," in *ISMIR*, vol. 8 (2008), 325–330.

9. A. Liaw, M. Wiener, et al., "Classification and Regression by RandomForest," *R news* 2, no. 3 (2002): 18–22.

10. S. Safavian and D. Landgrebe, "A Survey of Decision Tree Classifier Methodology," *IEEE transactions on systems, man, and cybernetics* 21, no. 3 (1991): 660–674.

11. D. Kleinbaum et al., *Logistic Regression* (Springer, 2002).

12. S. Wang, "Artificial Neural Network," in *Interdisciplinary computing in java programming* (Springer, 2003), 81–100.

13. Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation* 9, no. 8 (1997): 1735–1780.

14. Yann LeCun et al., "Object recognition with gradient-based learning," in *Shape, contour and grouping in computer vision* (Springer, 1999), 319–345.

15. Jeffrey Donahue et al., "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2015), 2625–2634.

CHAPTER 2
RELATED WORKS

Most of the works related to human activity recognition are based on either wearable sensors, audio, or video. So, we will divide this chapter into two sections. First, we will review the multimedia stream based researches for human activity recognition, and then we will review wearable sensor-based researches.

2.1 Multimedia stream based

Automatic analysis of teachers' instructional strategies was investigated from audio recordings collected in live classrooms in this research.[1] Dataset was collected from classroom recordings of teachers' audio. Supervised machine learning models were used to train five key instructional segments (procedures and directions, supervised seat-work, question and answer, small group work, and lecture). The models were validated independently of the teacher to increase the generalizability. The five instructional segments above were identified with F1 scores ranging from 0.64 to 0.78. The proposed model was able to detect five instructional segments well above chance level. The system used low-level features derived only from teachers' audio.

One of the recent researches[2] uses motion templates of instructors and represents them through Bag-of-Deep features (BoDF). Deep Spatio-temporal features were extracted from motion templates and then utilized to compile a visual vocabulary. After that, the visual vocabularies were quantized to optimize the learning model. The activities given below were recognized with an accuracy of 85.41% - pointing towards the student, pointing towards board or screen, idle, interacting, sitting, walking, using a mobile phone, and using

1. P. Donnelly et al., "Automatic Teacher Modeling from Live Classroom Audio," in *Proceedings of the 2016 conference on user modeling adaptation and personalization* (ACM, 2016), 45–53.
2. Nida et al., "Bag of Deep Features for Instructor Activity Recognition in Lecture Room."

a laptop.

Another research proposed an audio-based activation detection technique in the classroom.[3] They implemented different models, including logistic regression, Deep Neural Network (DNN), RNN based on GRU and LSTM. The activities they classified are *lecture*, *group discussion*, *silent work*. Their method achieved 92.7% f1 score on previously seen instructor and 89.1% f1 score on a previously unseen instructor.

2.2 Wearable sensor based

One research proposed unsupervised learning methods for human activity recognition.[4] They were able to detect activities even when the number of activities was unknown. At first, they collected a series of sensor data from smartphones as the users performed five activities: walking, running, sitting, standing, and lying down. Then a list of feature vectors was generated by aggregating the sensor data over sliding windows. According to their experiment, the k-means algorithm showed a relatively lower accuracy than others. The accuracy never exceeded 0.8 for every k. The mixture of Gaussian (GMM) showed the exact recognition when k was known (k=5). Using Hierarchical Clustering (HIER), the clusters were distinctively recognizable when k was large. HIER showed high accuracy for a large k. DBSCAN does not require the number of clusters. From the experiments in this paper, they found that using DBSCAN the five activities can be detected with 90% accuracy.

One research used CNN based approaches using body-worn inertial sensors for clas-

3. Robin Cosbey, Allison Wusterbarth, and Brian Hutchinson, "Deep Learning for Classroom Activity Detection from Audio," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (IEEE, 2019), 3727–3731.

4. Yongjin Kwon, Kyuchang Kang, and Changseok Bae, "Unsupervised learning for human activity recognition using smartphone sensors," *Expert Systems with Applications* 41, no. 14 (2014): 6067–6074.

sifying different human activities.[5] They proposed a deep Convolutional Neural Network (CNN) model. Using two datasets (Opportunity[6] and Hand Gesture Dataset[7]), they showed that the CNN model performed better than the other models in comparison (SVM, KNN, Means and Variance (MV), Deep Belief Network (DBN)).

Some researches performed wearable sensor-based activity recognition. Three motion sensors (accelerometer, gyroscope, and linear acceleration sensor) were evaluated at both wrist and pocket positions in order to recognize human activities in one research.[8] They found that less-repetitive activities could not be recognized easily at smaller segmentation windows, unlike repetitive activities. Seven window sizes (2–30 s) on thirteen activities were used for the experiments. The effect of window size was also analyzed. It was found that combining the data from the motion sensors from the wrist and pocket positions improved recognition for complex activities. Improvements were seen due to increasing window size for simpler activities when their reference performances were low. Though the sensor combinations improved the recognition of complex activities at smaller window sizes, the paper recommended using a bigger window size for their reliable recognition.

The use of smartphones and smartwatches for human activity recognition was examined in a paper.[9] They compared the results of smartphone-based activity recognition with smartwatch-based activity recognition and ultimately found that the combination of both

5. Jianbo Yang et al., "Deep convolutional neural networks on multichannel time series for human activity recognition," in *Twenty-Fourth International Joint Conference on Artificial Intelligence* (2015).

6. Ricardo Chavarriaga et al., "The Opportunity challenge: A benchmark database for on-body sensor-based activity recognition," *Pattern Recognition Letters* 34, no. 15 (2013): 2033–2042.

7. M. Bachlin et al., "Potentials of Enhanced Context Awareness in Wearable Assistants for Parkinson's Disease Patients with the Freezing of Gait Syndrome," in *2009 International Symposium on Wearable Computers* (2009), 123–130.

8. M. Shoaib et al., "Complex Human Activity Recognition using Smartphone and Wrist-Worn Motion Sensors," *Sensors* 16, no. 4 (2016): 426.

9. Gary M Weiss et al., "Smartwatch-based activity recognition: A machine learning approach," in *2016 IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI)* (IEEE, 2016), 426–429.

devices works best. Their research demonstrated that user-specific supervised machine learning models vastly outperform the impersonal models. They took 18 activities into account and grouped them into three major categories (hand oriented general activities, non-hand oriented general activities, hand oriented eating activities). The algorithms used for classification were random forest, decision tree algorithm, instance-based learning (IB3) algorithm, Naïve Bayes (NB) algorithm, and the multi-layer perceptron (MLP) algorithm. The Random Forest (RF) algorithm performed well for all the configurations. The results from this paper showed that the watch accelerometer provides much better results than the phone accelerometer, with an average accuracy of 91.9% versus 72.6% for personal models and 64.0% versus 25.3% for impersonal models. These differences were largely due to the hand-based activities included in their study. The watch gyroscope performed poorly than the watch accelerometer, with an average accuracy of 72.4% versus 91.9% for the personal models and 53.5% versus 64.0% for the impersonal models.

One research[10] explored deep, convolutional, and recurrent approaches across three datasets. These 3 datasets (Opportunity, PAMAP2 dataset,[11] Daphnet Gait dataset[12]) contain movement data that were recorded using wearable sensors. According to their research, bi-directional LSTMs outperformed the then state-of-the-art models on the opportunity dataset. Recurrent networks outperformed convolutional networks significantly on activities that were short in duration but had a natural ordering, where a recurrent approach benefitted from the ability to contextualize observations across long periods of time. For bi-directional RNNs they found that the number of units per layer had the largest effect

10. Nils Y Hammerla, Shane Halloran, and Thomas Plötz, "Deep, convolutional, and recurrent models for human activity recognition using wearables," *arXiv preprint arXiv:1604.08880*, 2016,

11. Attila Reiss and Didier Stricker, "Introducing a new benchmarked dataset for activity monitoring," in *2012 16th International Symposium on Wearable Computers* (IEEE, 2012), 108–109.

12. Marc Bachlin et al., "Wearable assistant for Parkinson's disease patients with the freezing of gait symptom," *IEEE Transactions on Information Technology in Biomedicine* 14, no. 2 (2009): 436–446.

on performance across all datasets. For prolonged and repetitive activities like walking or running, they recommended using CNNs.

Another research[13] proposed an unsupervised machine learning algorithm (MCODE) for human activity recognition using sensor data from the smartphone accelerometer. They tested their method in 3 datasets: one contained daily living activities, the other two contained sports activities. Users needed to wear the smartphone on the waist or upper arm. Their experimental results showed that their approach was practical for recognizing normal physical activities using smartphone accelerometers. Moreover, they showed that the MCODE-based method is more effective than the K-MEANS method for activity recognition.

In one study, a deep convolutional and LSTM based recurrent neural network was proposed for multimodal wearable activity recognition.[14] They tested their system on two public datasets (Opportunity and Skoda). This study shows the effect of sensor fusion and the impact of sequence length. They were able to achieve 91.5% f1 score on the gesture recognition opportunity dataset and 95.8% f1 score on the Skoda dataset. They also demonstrated that the recurrent LSTM cells are fundamental to distinguish gestures of a similar kind (e.g., *Open/Close Door* or *Open/Close Drawer*). The performance of their model improved on average by 15% when fusing accelerometers and gyroscopes. The performance enhanced by 20% when combining accelerometers, gyroscopes, and magnetic sensors.

A proof of concept was designed in a research[15] that used a deep learning framework.

13. Yonggang Lu et al., "Towards unsupervised physical activity recognition using smartphone accelerometers," *Multimedia Tools and Applications* 76, no. 8 (2017): 10701–10719.

14. Francisco Javier Ordóñez and Daniel Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors* 16, no. 1 (2016): 115.

15. M. Panwar et al., "CNN based Approach for Activity Recognition using a Wrist-Worn Accelerometer," in *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* (IEEE, 2017), 2438–2441.

The framework reduced the difficulty of the optimal feature selection problem significantly. A wrist-worn accelerometer was used to identify three basic movements of the human forearm. The validation of the proposed model was done using three possible methods. The results showed that the model achieved an average rating of 99.8%, which was more than K-means clustering, linear discriminant analysis, and support vector machine. A comparative analysis between conventional models and neural network models was done in this paper. As a result, they found that CNNs were very promising in handling the feature engineering process. CNNs also produced high accuracy if the design parameters were defined in an efficient way. Also, the proposed model was able to classify daily living activities in real-time and practical scenarios. The paper suggested that the system could be extended towards increasing the number of subjects and also towards people suffering from neurodegenerative diseases.

Another research[16] proposed a CNN and LSTM based hybrid model for multiple overlapping activity detection. They used the Opportunity dataset to carry out their research. According to their results, the proposed model increased the accuracy relative to the previous researches.

One study[17] proposed an active learning system for activity recognition. They used smartwatch data for training their model. The activities classified were walking, sitting, standing, running, and lying down. They compared results for different traditional machine learning techniques (random forest, naive Bayes, logistic regression, SVM). The results

16. Tsuyoshi Okita and Sozo Inoue, "Recognition of multiple overlapping activities using compositional CNN-LSTM model," in *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers* (2017), 165–168.

17. Farhad Shahmohammadi et al., "Smartwatch based activity recognition using active learning," in *2017 IEEE/ACM International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE)* (IEEE, 2017), 321–329.

of this study revealed that the system could obtain a 93.3% accuracy across 12 participants. They also demonstrated that an interactive learning approach using active learning in smartwatches has significant advantages over smartphones and other devices for activity recognition tasks.

Another study[18] proposed a system for multi-task recognition using wearable sensor data. The system consisted of 2 models, a CNN model for classifying simple activities and an LSTM model for classifying complex activities. They used two benchmark datasets (Opportunity and Ubicomp[19]) to carry out their experiments.

Another research experimented with multiple models using the smartphone and smartwatch sensor data.[20] They compared Random Forest (RF), Hidden Markov Model (HMM), Hybrid of CNN and Multi-Layer Perceptron (MLP), and Hybrid of CNN and LSTM models. They found that for the smartphone recordings, the CNN-LSTM model provided the best results. For smartwatch recordings, HMMs offered slightly better robustness. They used Heterogeneity Human Activity Recognition (HHAR) dataset.[21] The activities they classified were *sitting*, *standing*, *biking*, *walking*, *going upstairs*, and *going downstairs*.

One paper[22] surveyed deep learning methodologies for sensor-based human activity recognition (HAR). According to the paper, several reasons existed for choosing the deep

18. Liangying Peng et al., "Aroma: A deep multi-task learning based simple and complex human activity recognition method using wearable sensors," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, no. 2 (2018): 1–16.

19. Tâm Huynh, Mario Fritz, and Bernt Schiele, "Discovery of activity patterns using topic models," in *Proceedings of the 10th international conference on Ubiquitous computing* (2008), 10–19.

20. Rubén San-Segundo et al., "Robust Human Activity Recognition using smartwatches and smartphones," *Engineering Applications of Artificial Intelligence* 72 (2018): 190–202.

21. Allan Stisen et al., "Smart devices are different: Assessing and mitigatingmobile sensing heterogeneities for activity recognition," in *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems* (2015), 127–140.

22. Jindong Wang et al., "Deep learning for sensor-based activity recognition: A survey," *Pattern Recognition Letters* 119 (2019): 3–11.

learning approaches over traditional Pattern Recognition (PR) techniques. First of all, in traditional PR, the features were extracted using hand-crafted methods that depend on human experience and domain knowledge. Secondly, only shallow features could be learned using human expertise.[23] Those features were mostly statistical. Thirdly, conventional approaches often required a large amount of well-labeled data to train the model. But in real applications, most of the data are unlabeled. Existing deep generative networks[24] are able to exploit the unlabeled samples for model training. According to this survey, six types of deep learning models have been used so far. They are - 1. Deep Neural Network (DNN), 2. Convolutional neural network (CNN), 3. Recurrent neural network (RNN), 4. Deep belief network (DBN) and restricted Boltzmann machine (RBM), 5. Stacked autoencoder (SAE), and 6. The hybrid combination of some deep models. They found that deep hybrid models tend to perform better than single models. They also pointed out six grand challenges for HAR.

Another research[25] used smartwatch sensor data (accelerometer, gyro sensor, step counter, heart rate) to classify eight activities. They used Principal Component Analysis (PCA) first to find out the important features. After that, they carried out tests using random forest, SVM, C4.5, and K-NN algorithms. According to their results, random rorest performed best.

Finally, a recent research work[26] used the same dataset that we have used. It used

23. Yang et al., “Deep convolutional neural networks on multichannel time series for human activity recognition.”

24. Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh, “A fast learning algorithm for deep belief nets,” *Neural computation* 18, no. 7 (2006): 1527–1554.

25. Serkan Balli, Ensar Arif Sağbaş, and Musa Peker, “Human activity recognition from smart watch sensor data using a hybrid of principal component analysis and random forest algorithm,” *Measurement and Control* 52, nos. 1-2 (2019): 37–45.

26. Gary M Weiss, Kenichi Yoneda, and Thaier Hayajneh, “Smartphone and Smartwatch-Based Biometrics Using Activities of Daily Living,” *IEEE Access* 7 (2019): 133190–133202.

the WISDM dataset,[27] which includes the smartphone and smartwatch sensor data (accelerometer and gyro sensor). They classified 18 activities using k-NN, decision tree, and random forest. According to their study, the best biometric performance occurs using the smartphone and smartwatch together.

27. Weiss, “WISDM Smartphone and Smartwatch Activity and Biometrics Dataset.”

CHAPTER 3

DATASET

3.1 DATASET 1

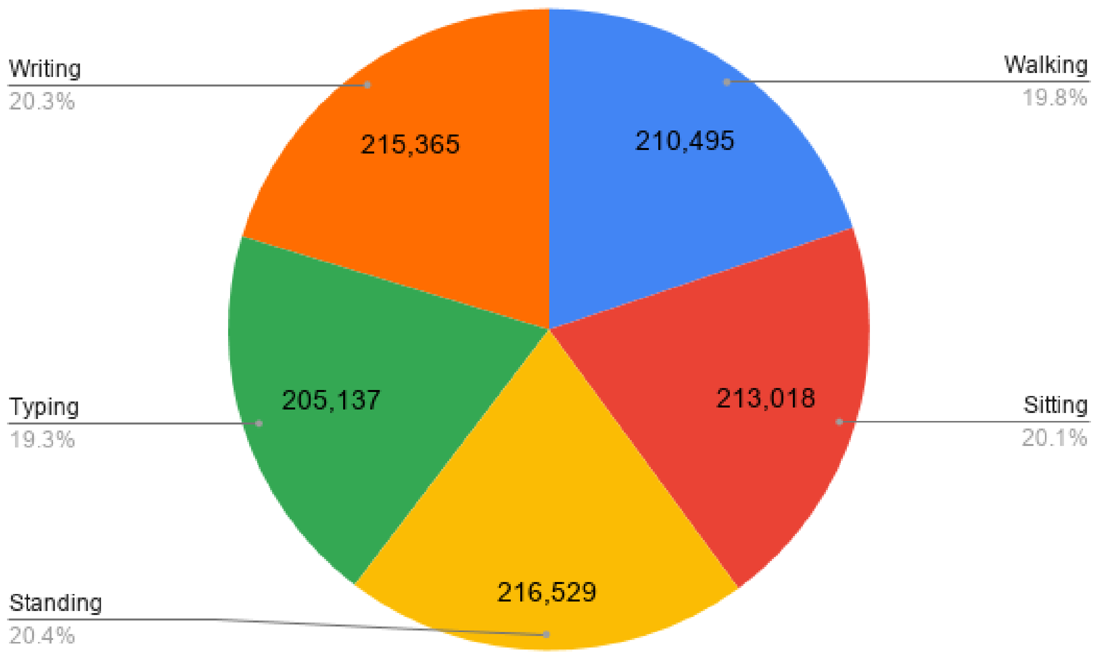
This dataset[1] contains data from the accelerometer and gyroscope sensors of smartphones and smartwatches. These readings were recorded as 51 subjects performed 18 diverse activities of daily living. Each activity was performed for 3 minutes so that each subject contributed 54 minutes of data. Sensor data was collected at a rate of 20Hz. The smartphones used were either Google Nexus 5/5x or Samsung Galaxy S5. The smartwatch used was the LG G watch. However, in our context, we have only used smartwatch sensor data of 5 activities (walking, sitting, standing, typing, writing). The distribution of these data are shown in Figure 3.1a and Figure 3.1b. We can see from the distribution that the dataset is a balanced one. We used this dataset for training models that take both ac-

Overlap	Total	Train	Test
0 overlap	9375	6563	2812
50% overlap	18558	12990	5568
90% overlap	92040	64428	27612

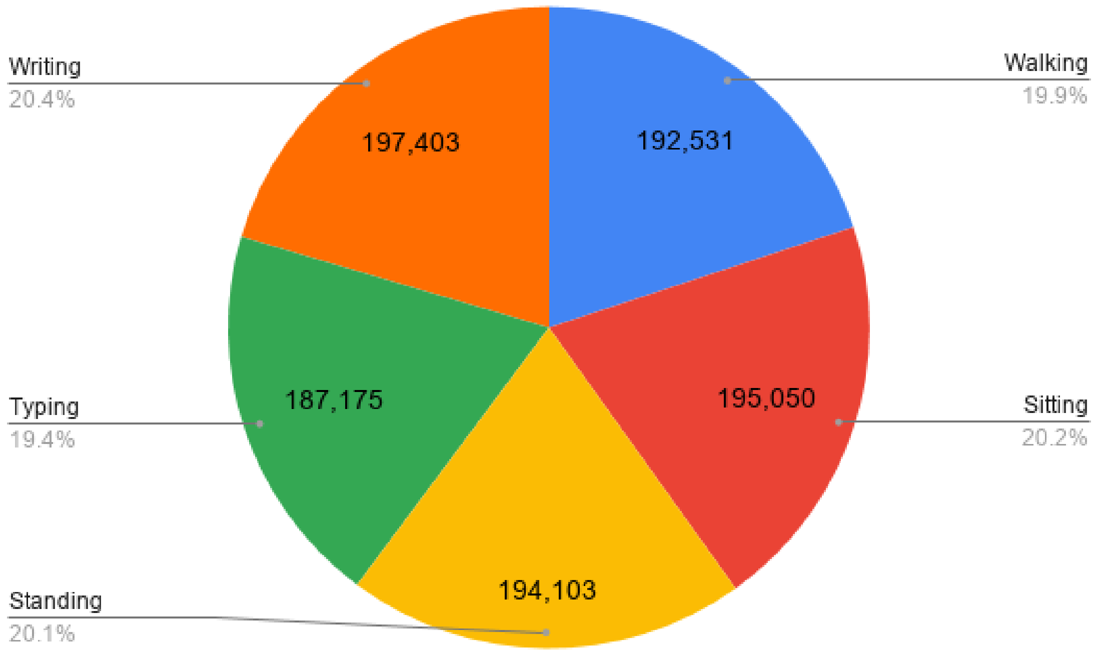
Table 3.1: Sequence Summary

celerometer and gyroscope data as input. For that reason, we merged both data using user id and timestamp. That means each row consisted of smartwatch accelerometer and gyro sensor data for the same user and same time-stamp. As a result, the size of the dataset reduced a bit because both gyroscope and accelerometer value were not always available for all the timestamps and all the users. Using these data we created sequences of length 100. Three amounts of overlapping settings were used for the sliding window technique to

1. Weiss, "WISDM Smartphone and Smartwatch Activity and Biometrics Dataset."



(a) Watch Accelerometer Data



(b) Watch Gyroscope Data

Figure 3.1: Distribution of Dataset 1

test the effect of overlap. The sequence data statistics are shown in Table 3.1.

Number of subjects	51
Number of activities	18
Minutes collected per activity	3
Sensor polling rate	20Hz
Smartphone used	Google Nexus 5/5x or Samsung Galaxy S5
Smartwatch used	LG G Watch
Number raw measurements	15,630,426

Table 3.2: Summary Information for Dataset 1

3.2 Dataset 2

This dataset[2] contains data from 60 users, each identified with a universally unique identifier (UUID). This is a multi-label dataset which means each of the data can represent multiple activities. From every user, it has thousands of examples, typically taken in intervals of one minute. Every example contains measurements from sensors from the user's personal smartphone and smartwatch. Most examples also have context labels self-reported by the user.

There were 34 iPhone users, 26 Android users. 34 of them were female and 26 were male. 56 of the users were right-handed, 2 were left-handed, and 2 defined themselves as using both. The users used a variety of smartphones. IPhones used were from generations 4, 4S, 5, 5S, 5C, 6, and 6S. IPhone operating system versions ranged from iOS-7 to iOS-9. Android devices consisted of Samsung, Nexus, HTC, Moto G, LG, Motorola, One Plus

2. Vaizman, Ellis, and Lanckriet, "Recognizing detailed human context in the wild from smartphones and smartwatches."

One, Sony.

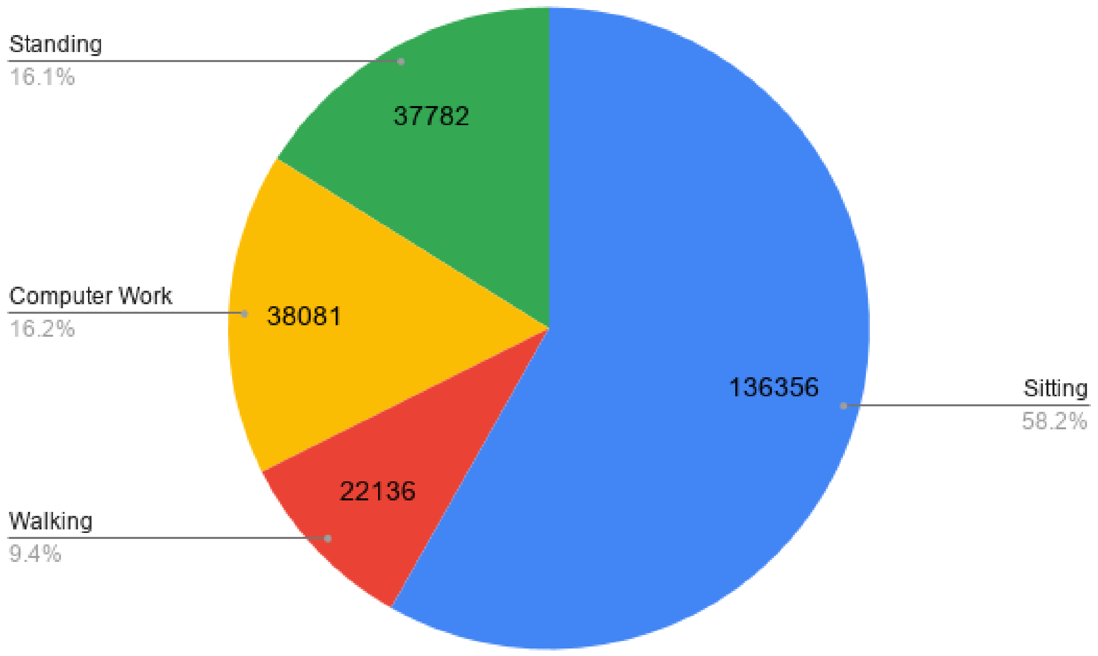


Figure 3.2: Distribution of Dataset 2

The dataset contains different types of sensor readings. But we used only the watch accelerometer and phone gyroscope data. The watch accelerometer data was sampled in 25Hz, and the phone gyroscope data was sampled in 40Hz. The sensor data were recorded in a 20-second window each minute. There are total of 51 cleaned labels or activities in the dataset. However, for our purpose, we have only used data regarding four activities (sitting, walking, computer work, standing). Figure 3.2 shows the distribution of the dataset. We can see that it is a highly imbalanced dataset. The dataset can be divided into two categories - processed data and raw sensor measurements.

3.2.1 Processed Dataset

The processed dataset contains multiple engineered features from the raw sensor data: 26 features for the phone gyroscope data and 46 features for the watch sensor data. It also includes the timestamp and the user id for each data.

Both the phone gyroscope data and the watch accelerometer data consists of 26 standard features each. These are:

- 9 statistics of the magnitude signal: mean, standard deviation, third moment, fourth moment, 25th percentile, 50th percentile, 75th percentile, value entropy, and time-entropy
- 6 spectral features of the magnitude signal: log energies in 5 sub-bands (0–0.5Hz, 0.5–1Hz, 1–3Hz, 3–5Hz, >5Hz), and spectral entropy
- 2 autocorrelation features from the magnitude signal
- 9 statistics of the 3-axis time series: the mean and standard deviation of each axis and the 3 inter-axis correlation coefficients

The watch accelerometer data contains an additional 15 axis-specific features: log energies in the same five sub-bands calculated for each axis' signal separately. Five relative-direction features to account for the changes in watch orientation.

3.2.2 Raw Sensor Measurements

Table 3.3 shows the summary of the sensor data that we have used. The raw sensor data can be divided into two types: one for the gyroscope data and another for the accelerometer data. However, both of them contain three principal values according to the three axes (x, y, z). Each row also consists of the timestamp. The raw dataset was saved in different folders for each user and in different files for each timestamp. We used a sequence

Sensor	Raw Measurements	Examples	Users%
Watch Accelerometer	3-axis (25Hz)	210,716	56
Phone Gyroscope	3-axis (40Hz)	291,883	57

Table 3.3: Sensor Measurements Summary

length of 125 (5 seconds of data) for the watch accelerometer data and sequence length of 200 (5 seconds of data) for the phone gyroscope data.

CHAPTER 4
METHODOLOGY

This chapter describes the methodologies used to detect different types of activities. It is divided into two major categories: methodologies for single activity recognition and methodologies for multiple concurrent activity recognition.

4.1 Single Activity Recognition

For this section we have used **Dataset 3.1**, which contains labeled data for five activities (walking, sitting, standing, writing, typing). Each row of the dataset represents only one activity. The high-level system overview is shown in Figure 4.1.

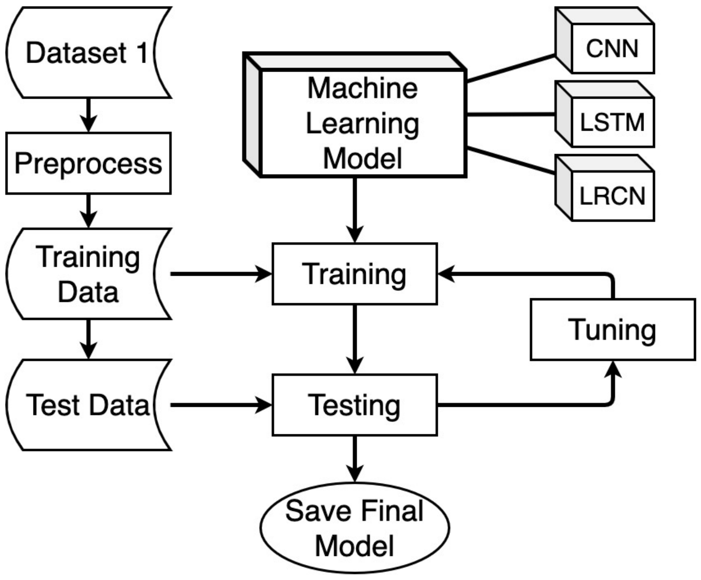


Figure 4.1: High Level Architecture of Single Activity Recognition Module

4.1.1 Data Preprocessing

The first task was to preprocess the data so it could be fed into the machine learning module. As we focused on the smartwatch based activity recognition, we took only the accelerometer and gyroscope reading of the smartwatch. The data were saved into separate files. Each file could be uniquely identified by the user id and the sensor type. Each file was also sorted by timestamps. Each row of a sensor data file (both accelerometer and gyroscope) consisted of five values. They were user id, activity type, x-axis value, y-axis value, z-axis value. First, we took the rows corresponding to the activities in context. Then for each user, we took the accelerometer and the gyroscope values. After that, we merged these values by timestamp. So, each row contained sensor values for the same timestamp. Then we created sequences of length 100 for each activity. As the sampling frequency of each sensor was 20 Hz, each sequence captured 5 seconds of sensor data. Then we used “One Hot Encoding” for transforming the label of each sequence from character to categorical binary representation. So, if there are three activities A, B, and C, one hot encoding represents A by 001, B by 010, and C by 100. After these steps, the dataset became three dimensional (number of sequences, sequence length, and sensor values). Finally, we divided the preprocessed data into training and test set with a ratio of 70:30.

4.1.2 Convolutional Neural Network (CNN)

The first model we have tried is known as Convolutional Neural Network (CNN).[1] CNN is well known for finding out patterns and shapes from images. However, for its pattern-finding capability, researches have also been done for finding patterns from sensor signals,[2] and it has shown excellent results for sensor-based human activity recognition.

The role of CNN is to reduce the input signals into a form that is easier to process without losing features, which are critical for getting accurate predictions. Convolution operations extract high-level features such as edges, from an input image. So, the intuition

1. LeCun et al., “Object recognition with gradient-based learning.”

2. Yang et al., “Deep convolutional neural networks on multichannel time series for human activity recognition.”

was to obtain sensor data shapes or patterns using convolutions similar to images but only in one dimension.

There are some parameters and layers while building a CNN model. The first one is called kernel or filter. A filter is a sliding window that convolves across the input data to find out features. Multiple filters are generally applied to grab various features. Pooling is another widely used concept in CNN. The use of a pooling layer is mainly to avoid overfitting of learned features by taking the maximum or average value of multiple features. Two types of pooling are typically used: max-pooling and average pooling. The dropout layers are used to avoid overfitting by dropping out units in a neural network.

The model we have used is shown in Figure 4.2. First, the input sequence was fed into a one-dimensional convolutional layer with ten filters, and each filter had a size of 25. After that, the output of this layer was fed into a one-dimensional max-pool layer to avoid overfitting of learned features. Another dropout layer was added to reduce overfitting further. The output from this layer was passed to another convolutional layer for higher-level feature learning. Then the outputs were sequentially passed to a max-pool layer and a dropout layer. After that, the outputs were flattened. Then it was passed to a dense layer (a simple fully connected neural network) to interpret the features extracted by the previous layers. Finally, an output layer was used to make predictions.

The different parameters used in our model like kernel size, number of kernels, hidden layers, etc. were chosen by tuning hyperparameters. All the layers used “Relu” activation function except the output layer. The “Softmax” activation function was used in the output layer because we detected one activity for each input sequence. “Categorical cross-entropy” was used as a loss function because it is best suited for single-label classification.

$$L(y,\hat{y}) = -\sum_{j=0}^{M}\sum_{i=0}^{N}(y_{ij} * \log \hat{y}_{ij})$$

where $\hat{y}$ is the predicted value, y is the actual value, M is the number of classes and N is the number of samples. Lastly, ADAM[3] optimizer was used in this model.

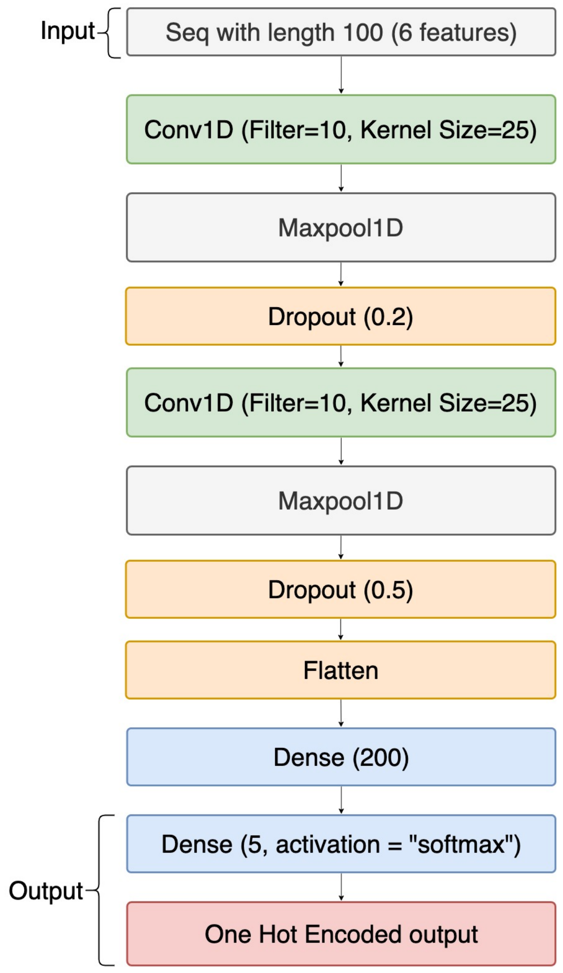

Input
Seq with length 100 (6 features)
Conv1D (Filter=10, Kernel Size=25)
Maxpool1D
Dropout (0.2)
Conv1D (Filter=10, Kernel Size=25)
Maxpool1D
Dropout (0.5)
Flatten
Dense (200)
Dense (5, activation = "softmax")
One Hot Encoded output
Output

Figure 4.2: CNN Model Used for Single Activity Recognition

4.1.3 Recurrent Neural Network (LSTM)

One of the significant shortcomings of traditional neural networks is that they do not learn based on the values they have already seen. In other words, they treat each input independent of previous values, which is not the case in all scenarios. Recurrent Neural

3. Diederik P Kingma and Jimmy Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014,

Network (RNN)[4] solves this problem. RNNs are neural networks with loops in them, allowing information to persist. Figure 4.3 shows the structure of a basic RNN. We can see that a loop enables information to pass from one step of the network to the next, and that is why RNNs perform better for learning sequences. However, there are two significant drawbacks of basic RNN.

- Vanishing gradient: The gradient becomes small, so in backpropagation, there is very little or no change in the earlier steps. That means, if the output in a later stage is dependant on the input in a very early stage, RNN may not grab it.
- Exploding gradient: The gradient becomes big. So, if the output in a later stage is dependant on the input in a very early stage, the gradient will become enormous.

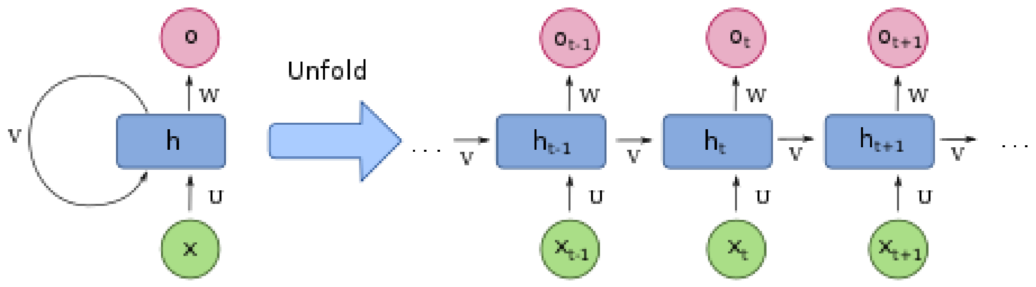


Figure 4.3: Basic Unfolded Architecture of RNN

Long Short Term Memory (LSTM)[5] architecture overcomes these drawbacks of RNN. Figure 4.4 shows the basic architecture of an LSTM cell. An LSTM cell has a sigmoid layer called the "forget gate layer" that decides which previous values to forget. It looks at h_{t-1} and x_t, and outputs a number (F_t) between 0 and 1 for each number in the cell state

4. David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams, "Learning representations by back-propagating errors," *nature* 323, no. 6088 (1986): 533–536.

5. Hochreiter and Schmidhuber, "Long short-term memory."

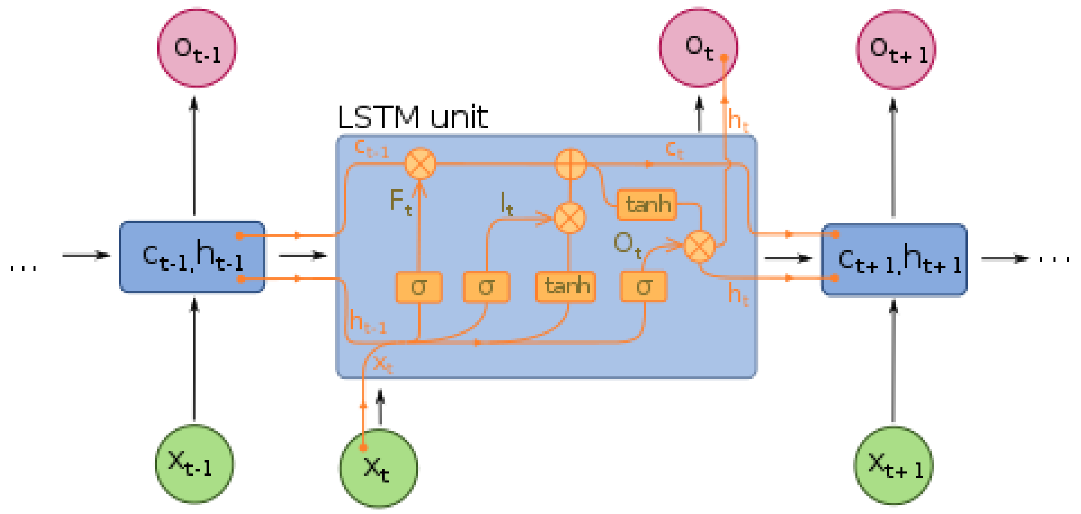


Figure 4.4: Basic Architecture of LSTM

(C_{t-1}). A 1 represents "completely keep the value," while a 0 represents "completely get rid of this." The next layer consists of two parts. First, a "sigmoid" layer called the "input gate layer" (I_t) decides which values to update. Next, a *tanh* layer creates a vector of new candidate values that can be added to the state. Then these two are multiplied and added to the old state to update each state value. Finally, for the output (O_t), a "sigmoid" layer decides what parts of the cell state are going to the output. Then, the cell state goes through *tanh* and is multiplied by the output of the sigmoid gate. As a result, it outputs only the selected portions.

Figure 4.5 shows the LSTM model that we have used. At first, the 3-dimensional input goes into a recurrent layer consisting of 200 LSTM cells, which learn the input sequences. Then we pass it through a dropout layer to avoid overfitting. The outputs from this layer are then passed to a fully connected dense layer with the "Relu" activation function to interpret the results from the previous layer. Finally, the output layer, which is also a dense layer, outputs the classification for the input sequences. This layer uses the "Softmax" activation function. The reason behind using these activations are described in Section 4.1.2.

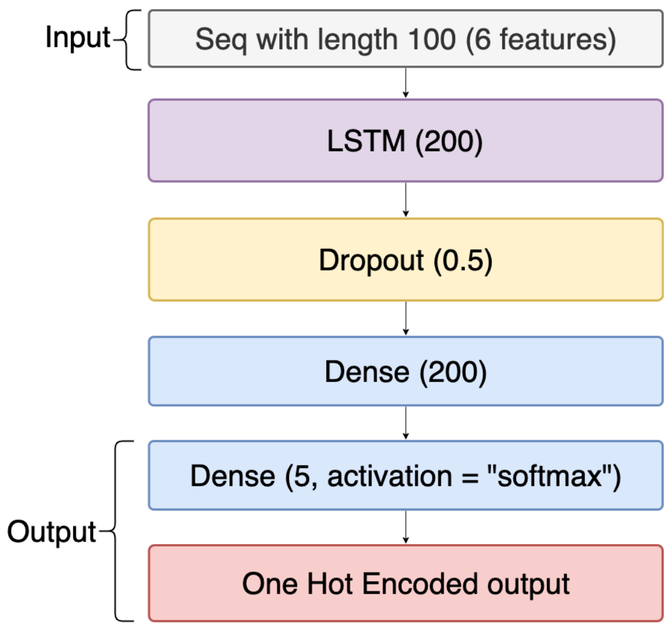


Figure 4.5: LSTM Model Used for Single Activity Recognition

4.1.4 Hybrid Neural Network

This model is a hybrid of multiple neural networks. We have seen that CNN models can find out patterns or spatial features from time-series data. On the other hand, LSTM is useful for learning the temporal dynamics of sensor data. The idea is to merge these two so that CNN first learns features from the sensor data, and the outputs of CNN are passed to the LSTM layer to learn the temporal dynamics. As a result, CNN makes the complex signal easier for the LSTM model to understand. One research first used this idea mainly to learn patterns from image sequences.[6] Another study used a similar approach for human activity recognition.[7]

The hybrid architecture that we have used is described in Figure 4.6. Multiple sequences go into the model as input and get classified by the model. For our case, the

6. Donahue et al., "Long-term recurrent convolutional networks for visual recognition and description."

7. Ordóñez and Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition."

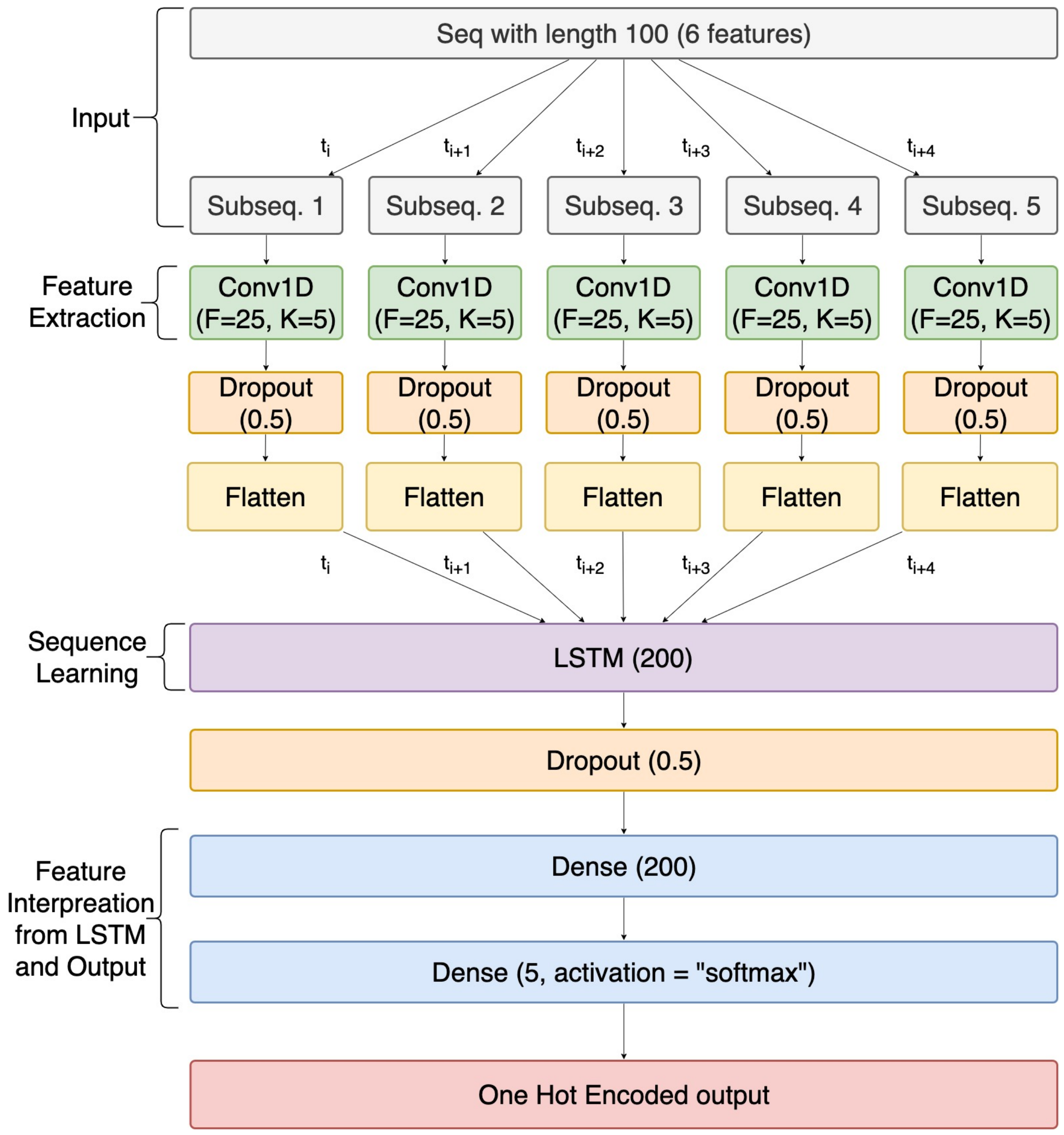


Figure 4.6: Hybrid Model Used for Single Activity Recognition

length of each sequence was 100, and each sequence contained six values corresponding to the accelerometer and gyroscope axes. For extracting features from each sequence, we used time distributed CNN. The time-distributed layer applies the same layer to several inputs. It produces one output per input to get the result in time. We used this to be able to get several subsequences as input, making the same filtering for each subsequence, and finally, to be able to continue the model with the LSTM layer. First, we broke down the 100 length sequence to multiple subsequences (five in our case, so each subsequence had one second of data). We passed it through a time-distributed one dimensional convolutional layer with five filters and a kernel size of five. After that, to avoid overfitting, we used a time-distributed dropout layer. Then we flattened the input using a time-distributed flattened layer so that it could be passed to the next LSTM layer. The LSTM layer contained 200 cells. It learned the sequences from the features extracted by the previous layers. Then we passed it through a dropout layer to avoid further overfitting. A fully connected dense layer with 200 units was then applied to interpret the results from the LSTM layer. Finally, a fully connected dense layer with five units and the "softmax" activation function worked as the output layer. The model was compiled with ADAM[8] optimizer. The loss function used was "categorical cross-entropy." The reason behind using these activation functions and loss functions are described in 4.1.2.

4.2 Multiple Concurrent Activity Recognition

In this section, we describe methodologies for detecting multiple concurrent activities. For example, if an instructor is working on a computer while sitting on a chair, there are two activities that he is doing concurrently. Our target is to detect both the activities simultaneously. We have used Dataset 3.2 in this section as it contains multi-label data. The activities in context are walking, sitting, standing, and working on the computer.

8. Kingma and Ba, "Adam: A method for stochastic optimization."

4.2.1 METHODOLOGY FOR USING DATA WITH HAND-CRAFTED FEATURES

The dataset in the current context contains two types of data. One of them is data with hand-crafted features. For this type of data, we have tried two methods: *one vs. rest classifier* and *feed-forward neural network*. But first, some pre-processing steps were needed before feeding the data into the models. Figure 4.7 shows the whole architecture.

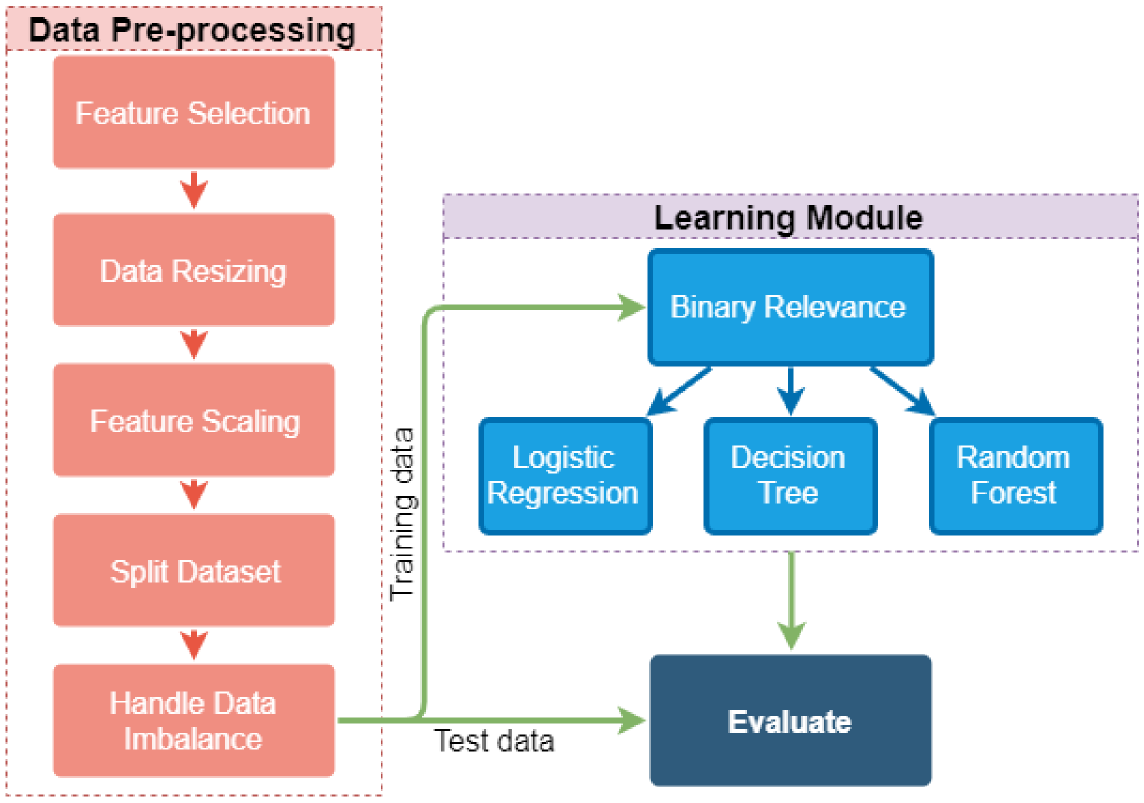


Figure 4.7: Architecture for Classifying Activities With Hand-Crafted Features

Data Resizing

There are many features and labels in the dataset. Still, as our context was to detect activities of the instructor in a classroom using smartwatch sensors, we selected only the relevant ones. In this analysis, we worked exclusively with the accelerometer data. For the reasons above, the first thing we had to do was to get rid of the rows in the dataset that were not relevant to the selected activities. The second step was to remove rows that did not have any accelerometer data.

Feature Scaling

Feature scaling or normalization is a crucial step in data preprocessing. It normalizes the values of features to a common scale. It is essential when features have different ranges. Otherwise, the model may skew towards specific features only because of its range of values. We have used the min-max scaling method.

$$x' = \frac{x - min(x)}{max(x) - min(x)}$$

where x' is the normalized value and x is the actual value.

Dataset Splitting

We primarily split the dataset into training and test datasets with a ratio of 70:30. We used the training dataset for training the models. The models were tested using the test dataset. For tuning hyperparameters of the neural network, we took 30% of the training dataset to validate the data.

Handling Data Imbalance

When the instances of one class outnumber the instances of another class, it is called an imbalanced dataset.[9] The dataset used in our work is highly imbalanced. Implementing machine learning models using an imbalanced dataset is always challenging.[10] There are

9. S. Elrahman and A. Abraham, "A Review of Class Imbalance Problem," *Journal of Network and Innovative Computing* 1, no. 2013 (2013): 332–340.

10. B. Krawczyk, "Learning from Imbalanced Data: Open Challenges and Future Directions," *Progress in Artificial Intelligence* 5, no. 4 (2016): 221–232.

multiple methods for overcoming the effect of data imbalance.[11] We have used an algorithmic level approach known as cost-sensitive learning, which is to define fixed and unequal misclassification costs between classes.[12] We adjusted the weights in such a way that it is inversely proportional to class frequencies in the input data

$$w_i = \frac{n}{k \times n_i}$$

where w_i is the weight to class i, n is the number of observations, n_i is the number of observations in class i and k is the total number of classes.

One vs Rest Classifier

We have used a problem transformation method for the multi-label binary classification known as binary relevance or one vs. rest[13] strategy. This method mainly considers the prediction of each class as an independent classification problem. For example, if we are detecting the activity *walk*, we treat the samples with this label as positive and all the others as negative. In one vs. rest approach, this strategy is done for all the classes. After transforming the problem, we tried three algorithms for classification and compared them. These three estimators were logistic regression, decision tree, and random forest. We have used *Scikit-learn*[14] for implementing the whole architecture.

Logistic regression: Logistic regression is a supervised machine learning algorithm for binary classification. It uses the logistic function (sigmoid function), $\sigma(x) = \frac{1}{1+e^{-x}}$ that can take any real-valued number and map it into a value between 0 and 1.

11. S. Kotsiantis, D. Kanellopoulos, P. Pintelas, et al., "Handling Imbalanced Datasets: A review," *GESTS International Transactions on Computer Science and Engineering* 30, no. 1 (2006): 25–36.

12. P. Domingos, "Metacost: A General Method for Making Classifiers Cost-sensitive," in *KDD*, vol. 99 (1999), 155–164.

13. Christopher M Bishop, *Pattern recognition and machine learning* (springer, 2006).

14. Fabian Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of machine learning research* 12, no. Oct (2011): 2825–2830.

Decision Tree: Linear regression and logistic regression models fail in situations where the relationship between features and outcome is nonlinear or where features interact with each other. The decision tree works better in such cases. Tree-based models split the data multiple times according to specific cutoff values in the features. Through splitting, different subsets of the dataset are created, with each instance belonging to one subset. The final subsets are called terminal or leaf nodes, and the intermediate subsets are called internal nodes or split nodes. To predict the outcome in each leaf node, the average outcome of the training data in this node is used.

Random Forest: Random forest[15] is an ensemble learning (uses multiple learning algorithms to obtain better predictive performance) method for classification or regression. It operates by constructing a multitude of decision trees at training time. Then it outputs the class that is the mode of the classes (classification) or the mean prediction (regression) of the individual trees. Random decision forests correct decision trees' habit of overfitting to their training set.

Feed Forward Neural Network

A feed-forward neural network consists of some simple neuron-like processing units, organized in layers. Every unit in a layer is connected with all the units in the previous layer. Each connection may have a different strength or weight. Data enters at the inputs and passes through the network, layer by layer until it arrives at the outputs. During normal operation, there is no feedback between layers. It uses a method named backpropagation for learning. In backpropagation learning, every time an input vector of a training sample is presented, the output vector is compared to the desired value. The comparison is made by a defined error function. The goal of backpropagation is to minimize the sum of errors

15. Andy Liaw and Matthew Wiener, "Classification and Regression by randomForest," *R News* 2, no. 3 (2002): 18–22, `https://CRAN.R-project.org/doc/Rnews/`.

for all the training samples.

Our model consisted of one input layer, one hidden layer, and one output layer. The number of neurons in each of the input and hidden layers was equal to the number of features. The output layer had four outputs corresponding to each of the activities. We also used dropout layers in between the input and the output layer to avoid overfitting. The output layer used the “sigmoid” activation function as each of the outputs could either be 0 or 1.

4.2.2 Methodology for using Data Without Feature Engineering

In this method, we tried to train a Recurrent Neural Network (RNN) to learn the sequence of sensor data.

Problem Formulation

Let, $D = (S_i, Y_i), 1 <= i <= N_d$ represent the training dataset. Here, N_d = number of training samples, S_i = i^{th} sequence of training data, Y_i = labels of i^{th} sequence. Each sequence S_i is a $N_s \times N_f$ dimensional vector, where N_s = sequence length and N_f = number of features. We define the labels for each sequence as, $y = \{y_1, y_2, y_3, ..., y_{N_l}\}$, which is a set of binary values and where N_l = number of classes (activities).

For an unseen sequence instance $x = \{x_1, x_2, x_3, ..., x_{N_s}\}$, our target was to build a classifier $h(.)$ which predicts $y = \{y_1, y_2, y_3, ..., y_{N_l}\}$ as a vector of labels for x.

Sequence Creation and Labeling

For learning the sequences, we first needed to pre-process the raw data to create sequences. Raw accelerometer data recorded at 25Hz frequency were used to create sequences with a size of 125. For the gyroscope data, we used a sequence length of 200, as the gyroscope data was recorded at 40Hz frequency. So, each sequence (both the watch

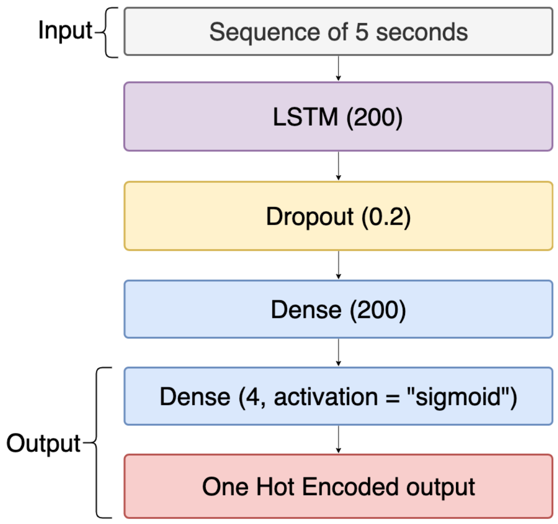


Figure 4.8: Structure of LSTM Model

accelerometer and the phone gyroscope data) represented 5 seconds of data. We first found out the labels for each user's raw data from the processed dataset's timestamp. Then saved the sequences and labels in a compressed format for minimal memory consumption. The compression task was important, as the actual raw accelerometer data size was around 10GB.

Dataset splitting

We split the dataset into training and test set in a 70:30 ratio. The training set was used to train the model. 30% of the training data were used for validating the dataset and tune the hyper-parameters. Finally, the test dataset was used to evaluate the model.

Recurrent Neural Network (LSTM)

We used LSTM (Long Short Term Memory) based Recurrent Neural Network (RNN) for the sequence learning approach. Figure 4.8 shows the high level architecture. The raw sensor data are multivariate time-series data. So, we can describe an activity by a sequence of raw sensor data.

A single LSTM layer with 200 cells learns the sequences. Then a dropout layer avoids overfitting. After that, a fully connected neural network with 200 neurons interprets the results of previous layers. The output layer is a fully connected neural network with four neurons, each of which outputs one if the sample sequence falls into the corresponding class. Otherwise, it outputs 0.

Activation Function: As the output is 0 or 1 for each class, sigmoid activation function was used.

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Loss Function: The models were compiled using binary cross-entropy loss function because this was a multi-label classification, and we had to treat each output label independently.

$$L(y,\hat{y}) = -\frac{1}{N}\sum_{i=0}^{N}\{y\log\hat{y} + (1-y)(\log(1-\hat{y})\}$$

where $\hat{y}$ is the predicted value, y is the actual value and N is the number of samples. Binary cross-entropy measures how far away from the true value (which is either 0 or 1) the prediction is for each of the classes and then averages these class-wise errors to obtain the final loss. However, we also used another version of the same formula which takes class imbalance into account.

Convolutional Neural Network (CNN)

Figure 4.9 describes the high-level overview of the CNN model. For both accelerometer and gyroscope data, we used the same architecture. The first layer was a convolutional layer with twenty kernels, each with a size of five. After that, we used six more convolutional layers, each containing ten kernels with a kernel size of five. These convolutional layers learned the shape of the sequences. After that, a max-pool and a dropout layer were used to get rid of the overfitting problem. Then the outputs of these layers were flattened so that it could be passed to a fully connected dense layer with 200 neurons. And finally,

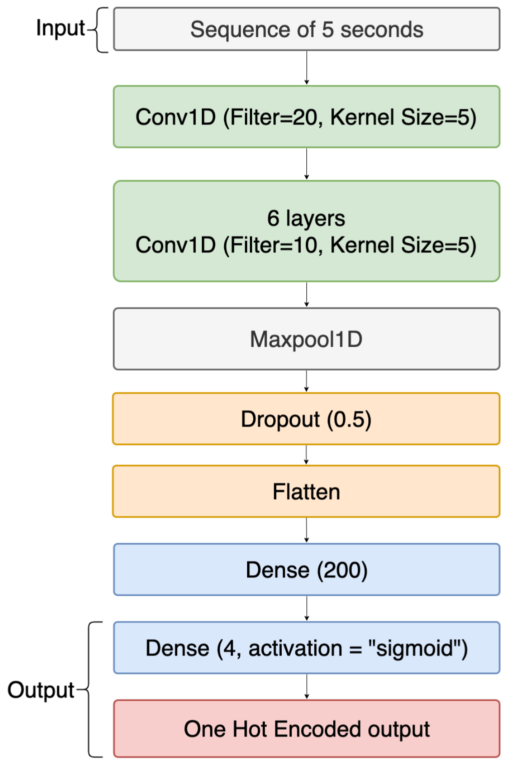


Figure 4.9: Structure of CNN model

a dense output layer was used. The loss function used was “binary cross-entropy,” and the activation function of the output layer was “sigmoid.” The reason for choosing these is described in 4.2.2.

Long Recurrent Convolutional Neural Network (LRCN)

The hybrid architecture used here is described in Figure 4.10. We used this architecture for building both the smartwatch accelerometer-based model and smartphone gyroscope-based model. For extracting features from each sequence, we used time-distributed CNN. The time-distributed layer applies the same layer to several inputs. It changes the outputs to continue the model with the LSTM layer. First, we passed the inputs through 5 layers of the time-distributed convolutional layer. Each of the convolutional layers contained 25 filters, each with a size of 5. Then we used a time-distributed max-pool layer that takes only the max values from each filter. After that, a time-distributed dropout layer reduced overfitting. Then we flattened the input using a time-distributed flattened layer to pass it to the next LSTM layer. The LSTM layer contained 200 cells. It learned the sequence from the features extracted by the previous layers. Then we passed it through a dropout layer to avoid further overfitting. A fully connected dense layer with 200 units interpreted the results from the LSTM layer. Finally, a fully connected dense layer with four units and the “sigmoid” activation function worked as the output layer. The model was compiled with ADAM[16] optimizer. The loss function used was “binary cross-entropy.” The reasons

behind using these activation functions and loss functions are described in 4.2.2.

Sensor Fusion

Figure 4.11 shows the high-level overview of sensor fusion architecture. For this architecture, we used six previously trained models: 2 CNN models, 2 LSTM models, and

16. Kingma and Ba, “Adam: A method for stochastic optimization.”

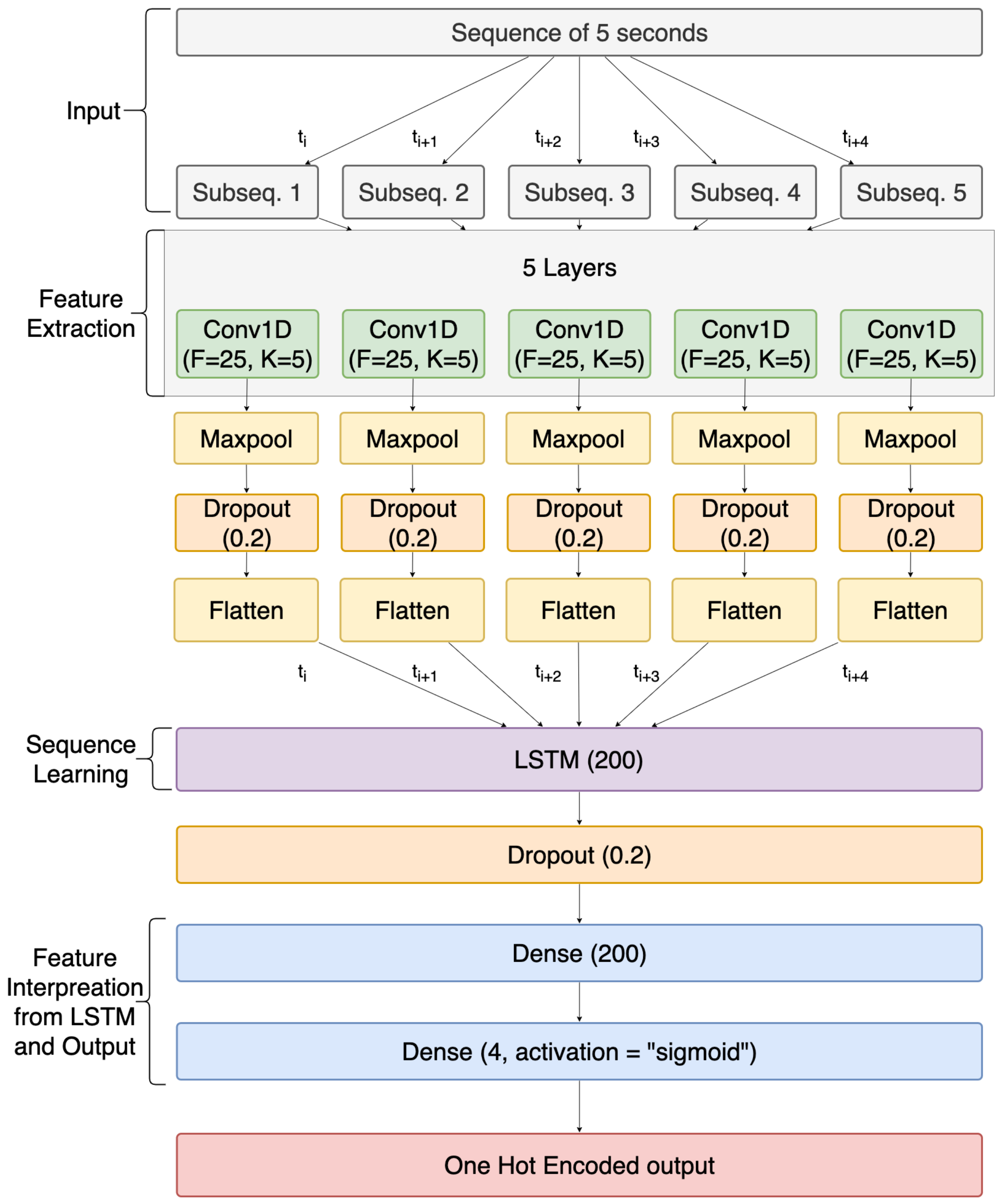


Figure 4.10: Structure of LRCN Model

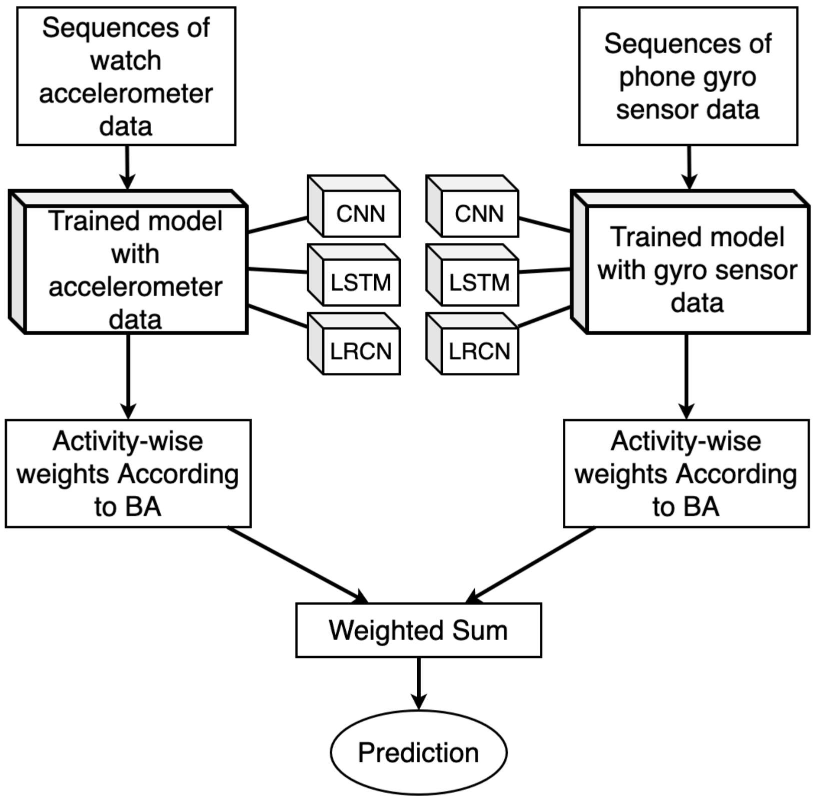

Sequences of watch accelerometer data
Sequences of phone gyro sensor data
Trained model with accelerometer data
CNN
LSTM
LRCN
CNN
LSTM
LRCN
Trained model with gyro sensor data
Activity-wise weights According to BA
Activity-wise weights According to BA
Weighted Sum
Prediction

Figure 4.11: Architecture of Sensor Fusion Model

2 LRCN models. Each model was trained using the smartwatch accelerometer data and smartphone gyroscope data. For each of the models, we calculated activity-wise balanced accuracy. Then for prediction, we first passed accelerometer and gyroscope data to corresponding models to get a primary prediction probability. After that, we calculated the weighted average according to the previously described weights.

$$WeightedMean = \frac{\sum_{i=1}^{n}(w_i x_i)}{\sum_{i=1}^{n}(w_i)}$$

Here, w_i = weight of i^{th} activity and x_i = predicted probability of i^{th} activity

CHAPTER 5
RESULTS

This chapter will first describe the experimental setup used for all the tests. Then it will explain the metrics used for evaluating different models. After that, it will do performance analysis for single activity recognition models. Finally, it will describe the results of multiple concurrent activity recognition models.

5.1 Experimental Setup

The development environment that we used for our experimental setup has the following configurations:
Operating system: Ubuntu 18.04
Processor: Intel(R) Xeon(R) W-2123 CPU @ 3.60GHz
System memory: 64 GB DDR4
GPU: Two GeForce GTX 1080

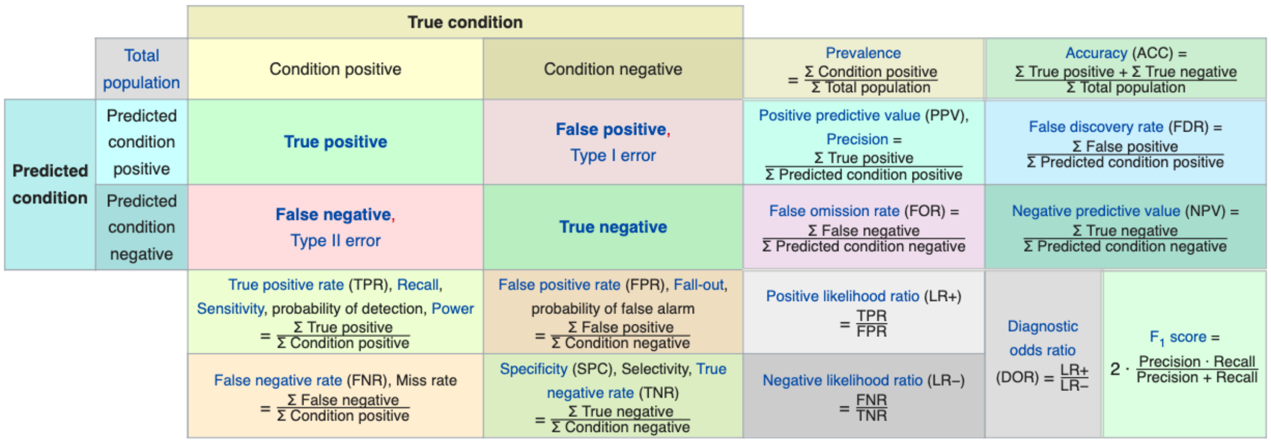
5.2 Metrics

For evaluating the performances of the models, we calculated multiple metrics. To understand these, we will first describe the confusion matrix which is shown in 5.1.

Terminologies: TP = True positive, TN = True negative, FP = False positive, FN = False negative, $Sensitivity(TPR) = \frac{TP}{TP+FN}$, $Specificity(TNR) = \frac{TN}{TN+FP}$, $Precision(PPV) = \frac{TP}{TP+FP}$.

- *Accuracy:*

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



		True condition				
	Total population	Condition positive	Condition negative	Prevalence $= \frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) $= \frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error	Positive predictive value (PPV), Precision $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$	
	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$	
		True positive rate (TPR), Recall, Sensitivity, probability of detection, Power $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$	F_1 score $= 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
		False negative rate (FNR), Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) $= \frac{\text{FNR}}{\text{TNR}}$		

Figure 5.1: Confusion Matrix and Metrics

Accuracy is the most common metric. But, it can be a misleading metric for imbalanced data sets. Consider a sample with 95 negative and five positive values. Classifying all the values as negative examples, in this case, gives an accuracy of 0.95. So, accuracy is a bad measure in case the dataset is imbalanced (contains examples of one class more than another).

- *F1 Score:*

$$F1 = \frac{2 * Sensitivity * Precision}{Sensitivity + Precision}$$

Calculating the harmonic mean (F1) is a commonly used metric. However, precision and F1 are less-fitting for a highly imbalanced dataset, since they are susceptible to how rare labels are. When averaging precision or F1 over many labels, certain labels will unfairly dominate the score. Additionally, if the data is noisy, Precision and F1 will be too sensitive to noises.

- *Area Under Curve (AUC):* The receiver operating characteristic (ROC) curve is a performance measurement for a classification problem at various threshold settings. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. AUC represents the degree or mea-

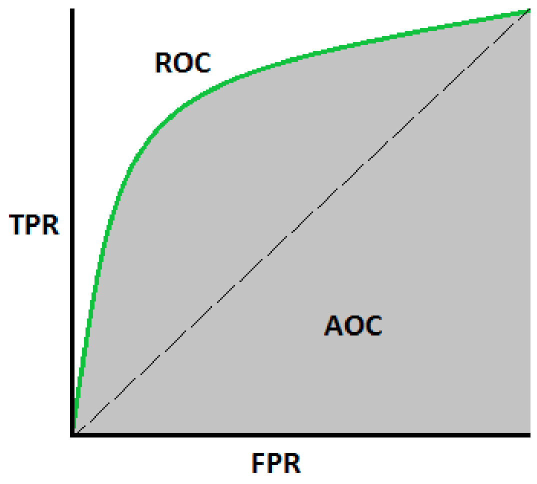


Figure 5.2: ROC Curve

sure of separability. It tells how much a model is capable of distinguishing between classes. The higher the AUC, the better the model is at predicting.

- *Balanced Accuracy:*

$$BA = \frac{Sensitivity + Specificity}{2}$$

Balanced Accuracy normalizes true positive and true negative predictions by the number of positive and negative samples, respectively, and divides their sum by two. It does not suffer from issues like F1 score and Accuracy.

5.3 Single Activity Recognition Results

We tested the performance of the models with different metrics. Dataset 3.1 is a balanced dataset. So, all the different metrics accuracy, F1, BA, AUC are more or less proper for this dataset. We also tested all three models using three types of sequence overlapping.

- No overlap
- 50% overlap
- 90% overlap

5.3.1 Results with Non Overlapping Sequence

Label	LSTM	LRCN	CNN
Walk	0.84	0.97	0.83
Sit	0.67	0.85	0.47
Stand	0.81	0.90	0.63
Type	0.79	0.88	0.59
Write	0.84	0.88	0.72

Table 5.1: Activity Wise Average F1 Score for Non Overlapping Sequence

Table 5.1 shows the activity-wise F1 score for all the models using non-overlapping sequence. From the table, we can see that the best-classified activity is “walk,” and the worst classified activity is “sit,” according to all the models. We cannot rank the activities “stand” and “write,” according to all the models. However, “type” is the second-worst classified activity according to all the models.

Figure 5.3 shows the average results for all activities for each of the models. The hybrid model (LRCN) outperformed both LSTM and CNN according to all the metrics (accuracy, BA, AUC, and F1). On the other hand, CNN performed the worst according to all the metrics.

Figure 5.4 shows the learning curve for all of the three models. We can see that the training stopped when the training-validation curve started to flatten. This way, we can say that the models did not overfit. We used a method named early-stopping, which stops the training when validation loss begins to increase and thus avoid overfitting.

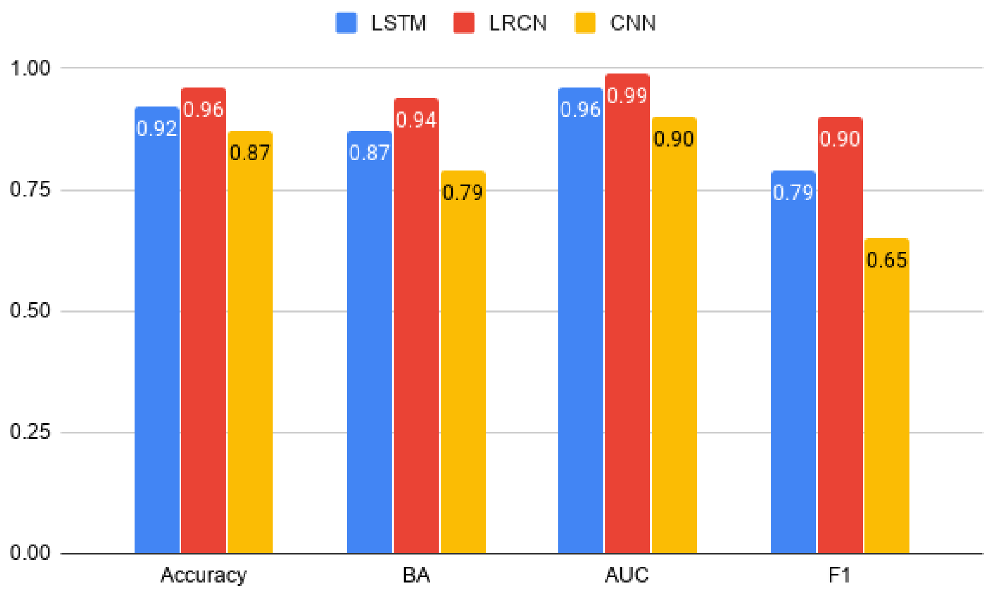
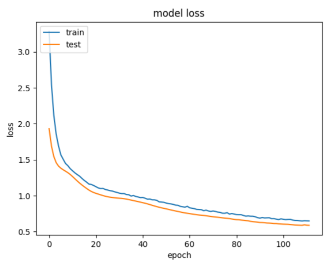
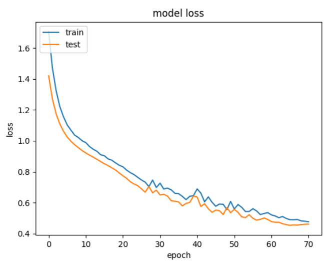


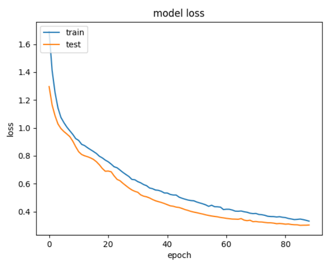
Figure 5.3: Average Results for All Models (No Overlap)



(a) CNN



(b) LSTM



(c) LRCN

Figure 5.4: Learning Curve (No Overlap)

Label	LSTM	LRCN	CNN
Walk	0.92	0.97	0.95
Sit	0.70	0.84	0.66
Stand	0.86	0.91	0.82
Type	0.81	0.88	0.75
Write	0.86	0.91	0.78

Table 5.2: Activity Wise Average F1 Score (50% Overlapping)

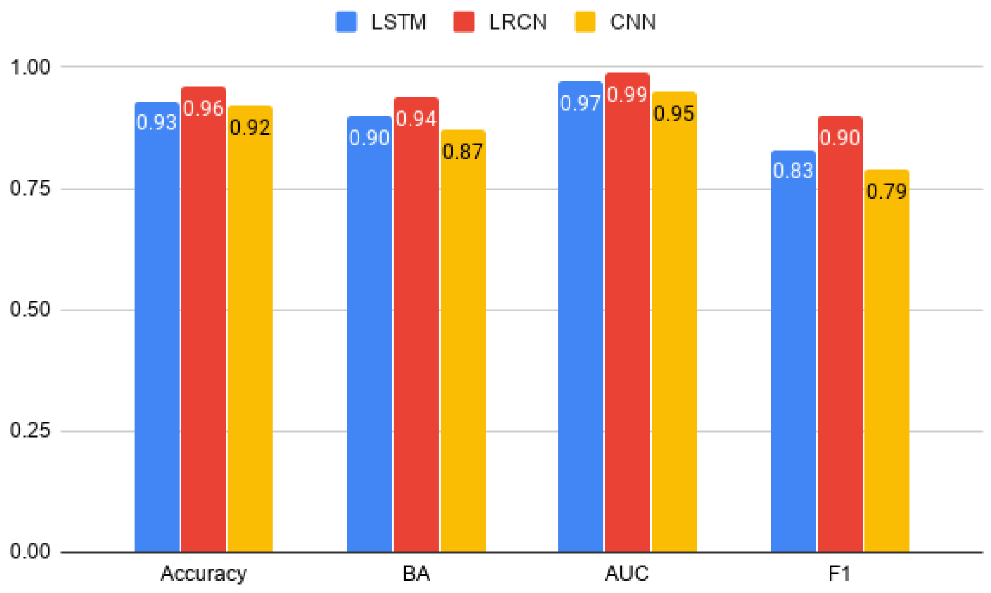
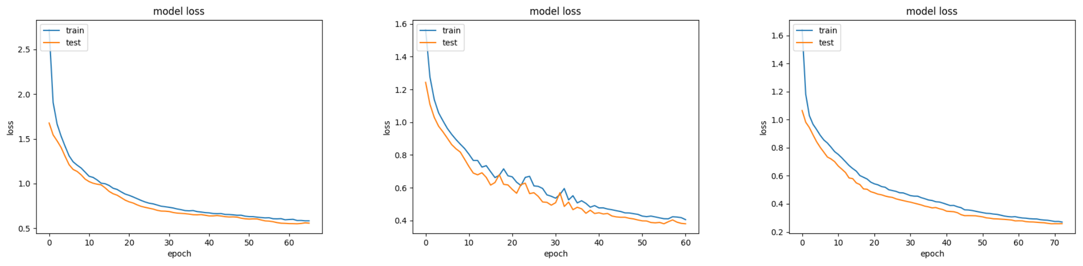


Figure 5.5: Average Results for All Models (50% Overlap)

5.3.2 Results with 50% Overlapping Sequence

Table 5.2 shows the activity-wise F1 score for all the models using sequence with 50% overlapping. Similar to the non-overlapping sequence models, "walk" is the best-classified activity, and "sit" is the worst classified activity according to all the models. "Stand" and "write" both are ranked two according to the LSTM and LRCN models. But according to CNN, "stand" ranks 2nd, and "write" ranks 3rd. "Type" is ranked 4th according to all the models, which is similar to the results of the non-overlapping sequence.

Figure 5.5 shows the average results for all activities for each of the models. The hybrid model (LRCN) outperformed both LSTM and CNN with an accuracy of 0.96, BA of 0.94, AUC of 0.99, and F1 of 0.90. On the other hand, CNN performed the worst according to all the metrics.



(a) CNN (b) LSTM (c) LRCN

Figure 5.6: Learning Curve (50% Overlap)

Figure 5.6 shows the learning curve for all of the three models. We can see that the training stopped when the training-validation curve began to flatten. This way, we can say that the models did not overfit. Again, we used "early stopping" to avoid overfitting.

5.3.3 Results with 90% Overlapping Sequence

Table 5.3 shows the activity-wise F1 score for all the models using sequences with 90% overlapping. Similar to previous results, "walk" is the best-classified activity, and

Label	LSTM	LRCN	CNN
Walk	0.99	1.00	0.97
Sit	0.83	0.93	0.77
Stand	0.94	0.96	0.85
Type	0.87	0.96	0.86
Write	0.91	0.97	0.85

Table 5.3: Activity Wise Average F1 Score (90% Overlapping)

"sit" is the worst classified activity according to all the models. The rankings of the other three activities vary based on different models.

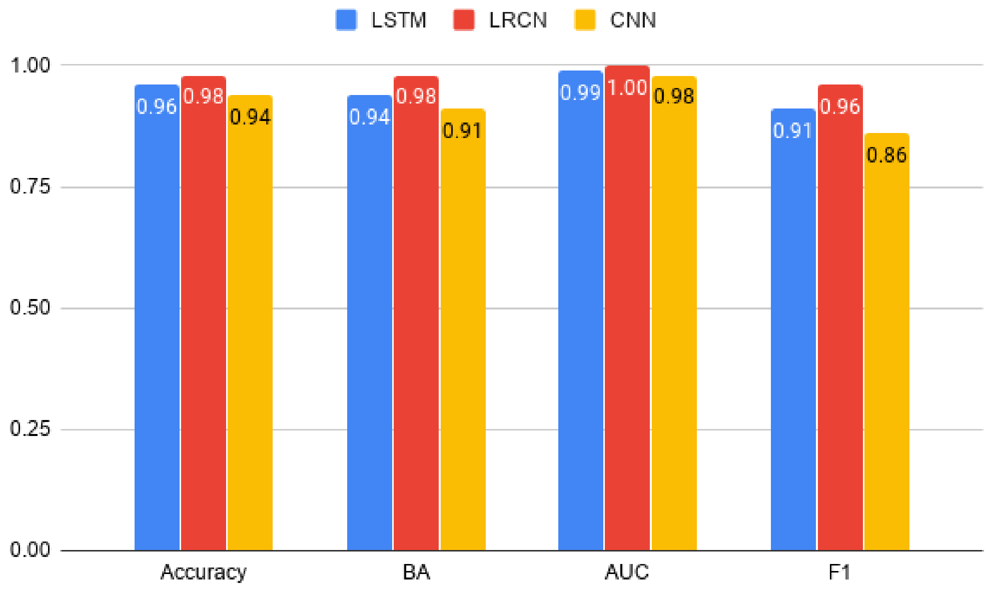
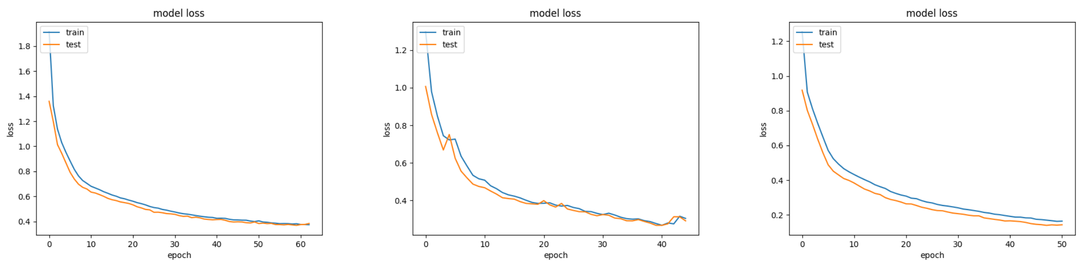


Figure 5.7: Average Results for All Models (90% Overlap)

Figure 5.7 shows the average results across all the activities for each of the models. Again, the hybrid model (LRCN) outperformed both LSTM and CNN with an accuracy of 0.99, BA of 0.98, AUC of 1, and F1 of 0.98. On the other hand, CNN performed the worst according to all the metrics.



(a) CNN (b) LSTM (c) LRCN

Figure 5.8: Learning Curve (90% Overlap)

Figure 5.8 shows the learning curve for all of the three models. We can see that training stopped when the training-validation curve began to flatten. This way we can say that the models did not overfit.

5.3.4 Resource Usage

Runtime Analysis

Figure 5.9 shows the runtime analysis of all the models. We tested all the models with no overlap, 50% overlap, and 90% overlap of sequences. We can see that for each of the setups, LSTM took the most amount of time, and CNN took the least amount of time. Runtime wise the hybrid model performed better than the LSTM model but worse than the CNN model. The LSTM model took five times more runtime than the CNN model.

Memory Analysis

Figure 5.10 shows the memory consumption of all the models. Like runtime, the LSTM model consumed the most amount of memory among the three models. The CNN model consumed the least. The LRCN model consumed more memory than the CNN model, but it was still a lot less than the LSTM model. The LRCN model consumed around

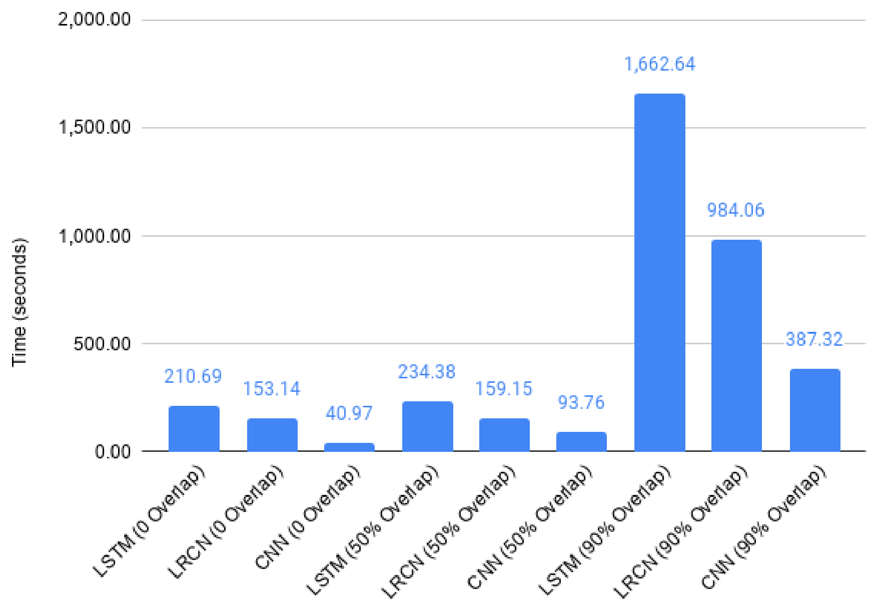


Figure 5.9: Runtime Analysis of Different Models using Different Overlaps

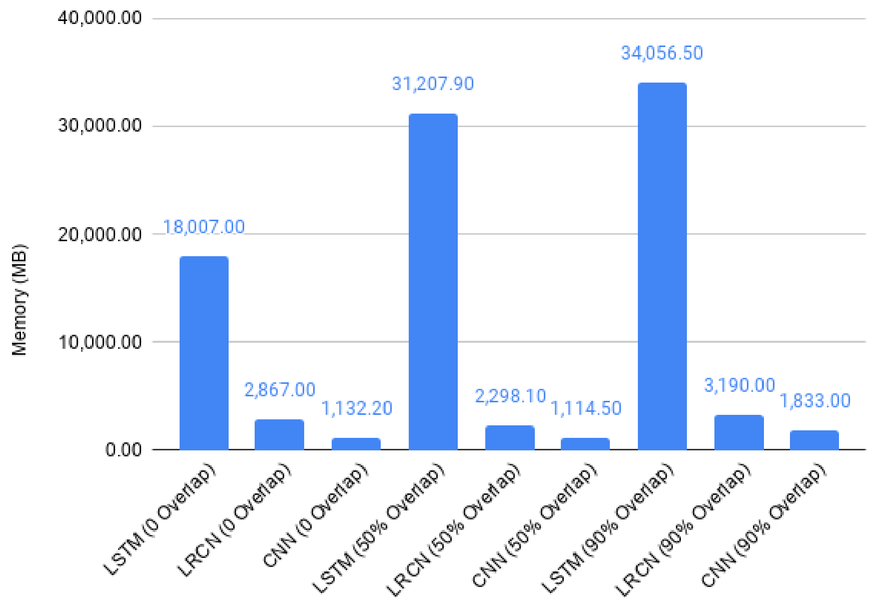


Figure 5.10: Memory Analysis of Different Models and Overlaps

two times more memory than the CNN model. But the LSTM model consumed around 18 times more memory than the CNN model in all settings. We can also observe that, as the overlapping of sequences increase, memory consumption also increases. Especially for LSTM, it increases drastically.

5.4 Multiple Concurrent Activity Recognition Results

The models described in this section use Dataset 3.2. We tested the performance of the models using different metrics. However, the dataset is highly imbalanced and contains noises. That is why balanced accuracy (BA) is the fittest for evaluating the models. We have also measured the F1 score and AUC.

5.4.1 Results Using Data with Hand-crafted Features

Table 5.4 shows F1 score, BA, and AUC as performance metrics for the processed dataset with weight adjustment. We can see that according to average BA, logistic regression did the best to classify the activities. According to the average AUC, the random forest did the best. According to the average F1 score, the decision tree performed better. If we take the average of all the scores, we can see that random forest outperformed the others.

5.4.2 Results Without Feature Engineering

Results Using Watch Accelerometer Data

Table 5.5 shows balanced accuracy for all the models using raw watch accelerometer data. We can see that “computer work” is the worst classified activity according to all the models. According to the LSTM and CNN models, the best-classified activity is “walk” and “sit” is the 2nd best. According to the LRCN model, the best-classified activity is “sit” and “walk” is the 2nd best. Overall, LRCN performed best with an average balanced

Label	BA	AUC	F1
Sit	0.67	0.72	0.62
Walk	0.76	0.84	0.30
CW	0.64	0.69	0.28
Stand	0.66	0.71	0.30
AVG	**0.66**	0.71	0.30

(a) Logistic regression

Label	BA	AUC	F1
Sit	0.69	0.69	0.63
Walk	0.63	0.63	0.31
CW	0.62	0.62	0.33
Stand	0.60	0.60	0.28
AVG	0.64	0.64	**0.39**

(b) Decision Tree

Label	BA	AUC	F1
Sit	0.77	0.86	0.72
Walk	0.61	0.87	0.34
CW	0.57	0.85	0.24
Stand	0.54	0.82	0.16
AVG	0.62	**0.85**	0.37

(c) Random Forest

Label	BA	AUC	F1
Sit	0.73	0.80	0.68
Walk	0.61	0.86	0.34
CW	0.51	0.79	0.04
Stand	0.51	0.76	0.04
AVG	0.59	0.80	0.28

(d) Neural Network

Table 5.4: Results of Feature Engineered Watch Accelerometer Dataset (Weighted)

Label	LSTM	LRCN	CNN
Sit	0.75	0.76	0.73
Walk	0.76	0.75	0.79
CW	0.68	0.70	0.68
Stand	0.70	0.72	0.69
AVG	0.72	**0.73**	0.72

Table 5.5: Activity Wise BA Using Watch Accelerometer Data (No Feature Engineering)

accuracy of 0.73.

Results Using Phone Gyro Sensor Data

Label	LSTM	LRCN	CNN
Sit	0.58	0.63	0.61
Walk	0.79	0.80	0.79
CW	0.57	0.61	0.58
Stand	0.63	0.65	0.64
AVG	0.64	**0.67**	0.65

Table 5.6: Activity Wise BA Using Phone Gyroscope Data (No Feature Engineering)

Table 5.6 shows balanced accuracy for all the models using raw phone gyroscope data. According to all the models, "walk" is the best-classified activity, while "computer work" is the worst classified. "Sit" is 2nd best classified for all the models. Again, the LRCN model performed best with an average balanced accuracy of 0.67.

Results after Sensor Fusion

We did sensor fusion on similar models. That means, two LSTM based models (one uses watch accelerometer data, and the other uses phone gyroscope data) were used for the LSTM based fusion. The CNN and LRCN based fused models are also similar. Table 5.7 shows the results. We can see that both the LSTM and LRCN based fused models performed better than the CNN model according to balanced accuracy. "Walk" is the best-classified activity according to all the models, and "computer work" is the worst classified activity. "Sit" is the 2nd best according to all the models. However, according to average AUC, LRCN based fused model performed best.

Label	LSTM	LRCN	CNN
Sit	0.78	0.77	0.74
Walk	0.83	0.84	0.83
CW	0.71	0.73	0.70
Stand	0.75	0.74	0.73
AVG	**0.77**	**0.77**	0.75

(a) Activity Wise BA Using Sensor Fusion

Label	LSTM	LRCN	CNN
Sit	0.86	0.85	0.81
Walk	0.91	0.93	0.91
CW	0.79	0.80	0.77
Stand	0.82	0.82	0.79
AVG	0.84	**0.85**	0.82

(b) Activity Wise AUC Using Sensor Fusion

Table 5.7: Activity Wise Results Using Sensor Fusion

5.5 Summary

We tested several deep learning models for single activity recognition. The dataset[1] we used was well balanced. According to the results, we found that the Hybrid Model (LRCN) outperforms both the LSTM and CNN based models. However, if we consider the usage of resources, LRCN was heavier than CNN but considerably lighter than LSTM. We also found that if sequences are created with more overlap, the accuracy of the model increases. However, it also increases the training time.

Few other pieces of research used this dataset to test out different methods. In one research[2] k-NN, decision tree, and random forest algorithms were used in the same dataset achieving an accuracy of 89.1%, 84%, 89.7%, 92.9%, 91.2% for walking, sitting, standing, typing, and writing activities. In another recent work,[3] they have used this dataset and were able to achieve an f1-score of 0.89, 0.75, 0.8, 0.82, and 0.84 for walking, sitting, standing,

1. Weiss, "WISDM Smartphone and Smartwatch Activity and Biometrics Dataset."
2. Weiss, Yoneda, and Hayajneh, "Smartphone and Smartwatch-Based Biometrics Using Activities of Daily Living."
3. Susana Benavidez and Derek McCreight, "A Deep Learning Approach for Human Activity Recognition Project Category: Other (Time-Series Classification)."

typing, and writing activities. Our proposed model outperformed these results.

For multiple concurrent activity prediction, we generated the activity prediction models based on two classes of techniques. One requires extensive feature selection and engineering, such as the random forest. The second approach uses RNN, where the features are automatically discovered from the raw data. Although random forest performs well, we would recommend using the RNN for the ease of use for this specific problem. The disadvantage of using RNNs is that the raw dataset is an order of magnitude larger than the processed feature set provided as input to generate the random forest. Therefore, training an RNN is significantly costlier in terms of resources and time. Note that the dataset is imbalanced concerning sample count per label type. We observed that assigning class weights has a positive impact on the performance of a machine learning model.

In one research,[4] 5-fold performance evaluation (BA) was done in this same dataset. In their paper, the results using only watch accelerometer data for classifying sitting, walking, computer work, and standing are respectively 0.68, 0.75, 0.62, and 0.67. They used logistic regression as the classification technique. In another research,[5] Multi-Layer Perceptron (MLP) was used with multiple layers on the same dataset. They got an accuracy of 0.75, 0.8, 0.72, 0.63 for the same activities. In our research, using LRCN, we were able to achieve a BA of 0.76, 0.75, 0.70, 0.72 for those activities using only watch accelerometer, which is better than the previous results. With sensor fusion, we achieved even better results.

4. Vaizman, Ellis, and Lanckriet, "Recognizing detailed human context in the wild from smartphones and smartwatches."

5. Y. Vaizman, N. Weibel, and G. Lanckriet, "Context Recognition In-the-Wild: Unified Model for Multi-Modal Sensors and Multi-Label Classification," *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* (New York, NY, USA) 1, no. 4 (January 2018), doi:10.1145/3161192, https://doi.org/10.1145/3161192.

CHAPTER 6
CONCLUSION AND FUTURE WORK

In this research, we have presented both traditional and neural network-based models to classify different activities. In single activity recognition settings, we classified five activities (walk, stand, sit, type, write). In concurrent multiple activity recognition settings, we classified four activities (sit, walk, computer work, stand) with class imbalance problem. Our proposed system shows promising results on activity recognition using the smartwatch and smartphone sensors data. As a part of our contributions, we tried to solve three open problems,[1] which are time-series modeling, multi-task modeling, and feature learning. We were able to achieve better results than previous experiments done on the WISDM dataset.[2]

However, there are some limitations that we want to overcome in the future. We have used data that were not collected in a classroom environment. We plan to build a dataset collecting smartwatch sensor data from instructors while they teach in the classroom. Labeling the data is also challenging. So, we plan to build a video-based crowd-sourcing system for labeling the sensor data. In short, classroom video will be recorded, focusing on the instructor. People will label short video clips uploaded on a website to describe the activities the instructor was performing at that time. From those timestamps and labels, we will label the sensor data by majority voting. The number of activities we have considered so far are few. In the future, we plan to include more activities like talking, writing on board, pointing towards the board, etc. We also plan to use more sensors available in smartwatches like microphones. We have used a fixed sequence size for the RNN models. In the future, we want to experiment with the effects of different sequence lengths. Auto-encoders are also popular to extract features automatically. In the future, we can also test the performance of auto-encoders as a method for feature learning. Finally, we plan to

1. Vaizman, Ellis, and Lanckriet, "Recognizing detailed human context in the wild from smartphones and smartwatches."
2. Weiss, "WISDM Smartphone and Smartwatch Activity and Biometrics Dataset."

build a fully working online activity detection system, including a smartwatch app. It will connect with a student attentiveness measuring system (which is also an active research field). A dashboard will show activity-wise attention analytics.

REFERENCES

Bachlin, M., D. Roggen, G. Troster, M. Plotnik, N. Inbar, I. Meidan, T. Herman, et al. "Potentials of Enhanced Context Awareness in Wearable Assistants for Parkinson's Disease Patients with the Freezing of Gait Syndrome." In *2009 International Symposium on Wearable Computers*, 123–130. 2009.

Bachlin, Marc, Meir Plotnik, Daniel Roggen, Inbal Maidan, Jeffrey M Hausdorff, Nir Giladi, and Gerhard Troster. "Wearable assistant for Parkinson's disease patients with the freezing of gait symptom." *IEEE Transactions on Information Technology in Biomedicine* 14, no. 2 (2009): 436–446.

Balli, Serkan, Ensar Arif Sağbaş, and Musa Peker. "Human activity recognition from smart watch sensor data using a hybrid of principal component analysis and random forest algorithm." *Measurement and Control* 52, nos. 1-2 (2019): 37–45.

Benavidez, Susana, and Derek McCreight. "A Deep Learning Approach for Human Activity Recognition Project Category: Other (Time-Series Classification)."

Bishop, Christopher M. *Pattern recognition and machine learning.* springer, 2006.

Casale, P., O. Pujol, and P. Radeva. "Human Activity Recognition from Accelerometer Data using a Wearable Device." In *Iberian Conference on Pattern Recognition and Image Analysis*, 289–296. 2011.

Chavarriaga, Ricardo, Hesam Sagha, Alberto Calatroni, Sundara Tejaswi Digumarti, Gerhard Tröster, José del R Millán, and Daniel Roggen. "The Opportunity challenge: A benchmark database for on-body sensor-based activity recognition." *Pattern Recognition Letters* 34, no. 15 (2013): 2033–2042.

Cosbey, Robin, Allison Wusterbarth, and Brian Hutchinson. “Deep Learning for Classroom Activity Detection from Audio.” In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3727–3731. IEEE, 2019.

Domingos, P. “Metacost: A General Method for Making Classifiers Cost-sensitive.” In *KDD*, 99:155–164. 1999.

Donahue, Jeffrey, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. “Long-term recurrent convolutional networks for visual recognition and description.” In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2625–2634. 2015.

Donnelly, P., N. Blanchard, B. Samei, A. Olney, X. Sun, B. Ward, S. Kelly, M. Nystran, and S. D’Mello. “Automatic Teacher Modeling from Live Classroom Audio.” In *Proceedings of the 2016 conference on user modeling adaptation and personalization*, 45–53. ACM, 2016.

Elrahman, S., and A. Abraham. “A Review of Class Imbalance Problem.” *Journal of Network and Innovative Computing* 1, no. 2013 (2013): 332–340.

Hammerla, Nils Y, Shane Halloran, and Thomas Plötz. “Deep, convolutional, and recurrent models for human activity recognition using wearables.” *arXiv preprint arXiv:1604.08880*, 2016.

Hinton, Geoffrey E, Simon Osindero, and Yee-Whye Teh. “A fast learning algorithm for deep belief nets.” *Neural computation* 18, no. 7 (2006): 1527–1554.

Hochreiter, Sepp, and Jürgen Schmidhuber. “Long short-term memory.” *Neural computation* 9, no. 8 (1997): 1735–1780.

Huynh, Tâm, Mario Fritz, and Bernt Schiele. "Discovery of activity patterns using topic models." In *Proceedings of the 10th international conference on Ubiquitous computing*, 10–19. 2008.

Ke, S., H. Thuc, Y. Lee, J. Hwang, J.Yoo, and K. Choi. "A Review on Video-based Human Activity Recognition." *computers* 2, no. 2 (2013): 88–131.

Kingma, Diederik P, and Jimmy Ba. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980*, 2014.

Kleinbaum, D., K. Dietz, M. Gail, and M. Klein. *Logistic Regression.* Springer, 2002.

Kotsiantis, S., D. Kanellopoulos, P. Pintelas, et al. "Handling Imbalanced Datasets: A review." *GESTS International Transactions on Computer Science and Engineering* 30, no. 1 (2006): 25–36.

Krawczyk, B. "Learning from Imbalanced Data: Open Challenges and Future Directions." *Progress in Artificial Intelligence* 5, no. 4 (2016): 221–232.

Kwon, Yongjin, Kyuchang Kang, and Changseok Bae. "Unsupervised learning for human activity recognition using smartphone sensors." *Expert Systems with Applications* 41, no. 14 (2014): 6067–6074.

LeCun, Yann, Patrick Haffner, Léon Bottou, and Yoshua Bengio. "Object recognition with gradient-based learning." In *Shape, contour and grouping in computer vision*, 319–345. Springer, 1999.

Liaw, A., M. Wiener, et al. "Classification and Regression by RandomForest." *R news* 2, no. 3 (2002): 18–22.

Liaw, Andy, and Matthew Wiener. "Classification and Regression by randomForest." *R News* 2, no. 3 (2002): 18–22. `https://CRAN.R-project.org/doc/Rnews/`.

Lu, Yonggang, Ye Wei, Li Liu, Jun Zhong, Letian Sun, and Ye Liu. "Towards unsupervised physical activity recognition using smartphone accelerometers." *Multimedia Tools and Applications* 76, no. 8 (2017): 10701–10719.

Nida, N., M. Yousaf, A. Irtaza, and S. Velastin. "Bag of Deep Features for Instructor Activity Recognition in Lecture Room." In *International Conference on Multimedia Modeling*, 481–492. Springer, 2019.

Okita, Tsuyoshi, and Sozo Inoue. "Recognition of multiple overlapping activities using compositional CNN-LSTM model." In *Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM International Symposium on Wearable Computers*, 165–168. 2017.

Ordóñez, Francisco Javier, and Daniel Roggen. "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition." *Sensors* 16, no. 1 (2016): 115.

Panwar, M., S. Dyuthi, K. Prakash, D. Biswas, A. Acharyya, K. Maharatna, A. Gautam, and G. Naik. "CNN based Approach for Activity Recognition using a Wrist-Worn Accelerometer." In *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 2438–2441. IEEE, 2017.

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. "Scikit-learn: Machine learning in Python." *Journal of machine learning research* 12, no. Oct (2011): 2825–2830.

Peng, Liangying, Ling Chen, Zhenan Ye, and Yi Zhang. "Aroma: A deep multi-task learning based simple and complex human activity recognition method using wearable sensors." *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, no. 2 (2018): 1–16.

Reiss, Attila, and Didier Stricker. "Introducing a new benchmarked dataset for activity monitoring." In *2012 16th International Symposium on Wearable Computers*, 108–109. IEEE, 2012.

Rumelhart, David E, Geoffrey E Hinton, and Ronald J Williams. "Learning representations by back-propagating errors." *nature* 323, no. 6088 (1986): 533–536.

Safavian, S., and D. Landgrebe. "A Survey of Decision Tree Classifier Methodology." *IEEE transactions on systems, man, and cybernetics* 21, no. 3 (1991): 660–674.

San-Segundo, Rubén, Henrik Blunck, José Moreno-Pimentel, Allan Stisen, and Manuel Gil-Martın. "Robust Human Activity Recognition using smartwatches and smartphones." *Engineering Applications of Artificial Intelligence* 72 (2018): 190–202.

Shahmohammadi, Farhad, Anahita Hosseini, Christine E King, and Majid Sarrafzadeh. "Smartwatch based activity recognition using active learning." In *2017 IEEE/ACM International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE)*, 321–329. IEEE, 2017.

Shoaib, M., S. Bosch, O. Incel, H. Scholten, and P. Havinga. "Complex Human Activity Recognition using Smartphone and Wrist-Worn Motion Sensors." *Sensors* 16, no. 4 (2016): 426.

Stisen, Allan, Henrik Blunck, Sourav Bhattacharya, Thor Siiger Prentow, Mikkel Baun Kjærgaard, Anind Dey, Tobias Sonne, and Mads Møller Jensen. "Smart devices are different: Assessing and mitigatingmobile sensing heterogeneities for activity recognition." In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, 127–140. 2015.

Trohidis, K., G. Tsoumakas, G. Kalliris, and I. Vlahavas. “Multi-label Classification of Music into Emotions.” In *ISMIR*, 8:325–330. 2008.

Tsoumakas, G., and I. Katakis. “Multi-label Classification: An overview.” *International Journal of Data Warehousing and Mining (IJDWM)* 3, no. 3 (2007): 1–13.

Vaizman, Y., N. Weibel, and G. Lanckriet. “Context Recognition In-the-Wild: Unified Model for Multi-Modal Sensors and Multi-Label Classification.” *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* (New York, NY, USA) 1, no. 4 (January 2018). doi:10.1145/3161192. https://doi.org/10.1145/3161192.

Vaizman, Yonatan, Katherine Ellis, and Gert Lanckriet. “Recognizing detailed human context in the wild from smartphones and smartwatches.” *IEEE Pervasive Computing* 16, no. 4 (2017): 62–74.

Veliyath, N., P. De, A. Allen, C. Hodges, and A. Mitra. “Modeling Students’ Attention in the Classroom using Eyetrackers.” In *Proceedings of the 2019 ACM Southeast Conference*, 2–9. ACM, 2019.

Wang, Jindong, Yiqiang Chen, Shuji Hao, Xiaohui Peng, and Lisha Hu. “Deep learning for sensor-based activity recognition: A survey.” *Pattern Recognition Letters* 119 (2019): 3–11.

Wang, S. “Artificial Neural Network.” In *Interdisciplinary computing in java programming*, 81–100. Springer, 2003.

Weiss, Gary M. “WISDM Smartphone and Smartwatch Activity and Biometrics Dataset.” *UCI Machine Learning Repository: WISDM Smartphone and Smartwatch Activity and Biometrics Dataset Data Set*, 2019.

Weiss, Gary M, Jessica L Timko, Catherine M Gallagher, Kenichi Yoneda, and Andrew J Schreiber. “Smartwatch-based activity recognition: A machine learning approach.”

In *2016 IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI)*, 426–429. IEEE, 2016.

Weiss, Gary M, Kenichi Yoneda, and Thaier Hayajneh. "Smartphone and Smartwatch-Based Biometrics Using Activities of Daily Living." *IEEE Access* 7 (2019): 133190–133202.

Yang, Jianbo, Minh Nhut Nguyen, Phyo Phyo San, Xiao Li Li, and Shonali Krishnaswamy. "Deep convolutional neural networks on multichannel time series for human activity recognition." In *Twenty-Fourth International Joint Conference on Artificial Intelligence*. 2015.