2-16-2023 10:30 AM

Deep Learning for Detection of Upper and Lower Body Movements

Kyle B. Lacroix, *The University of Western Ontario*

Supervisor: Grolinger, Katarina, *The University of Western Ontario*
Co-Supervisor: Trejos, Ana Luisa, *The University of Western Ontario*

A thesis submitted in partial fulfillment of the requirements for the Master of Engineering Science degree in Electrical and Computer Engineering

Abstract

When humans repeat the same motion, the tendons, muscles, and nerves can be damaged, causing repetitive stress injuries (RSI). Symptoms usually begin slowly and become more intense and constant over time. If the motions that lead to RSI are recognized early, these injuries can be prevented. A preventative approach could be implemented in factories to warn workers about possible injuries. By detecting the movements that can cause RSI, the worker can be alerted to stop carrying out those movements. For this purpose, machine learning models can detect human motion with the human activity recognition (HAR) model. HAR models typically require data from each participant before being trained; therefore, they cannot easily be adapted to new participants. This problem arises because humans move differently. To solve this problem, a model can be personalized to a particular individual to help detect their movements more easily. The model training procedure to create a personalized model consists of two phases: create a generic model, and then personalize the generic model with transfer learning. In this thesis, CNN, transformer, and Trans-CNN were selected for the model training procedure. To assess the model training procedure, the WISDM 2019 dataset was selected. Both the generic model and personalized model were evaluated with three different methods: all movement, only upper body movement, and lower body movement. In each of the evaluations, the same following trends were seen: personalization increased all of the performance metrics for all three models; the generic Trans-CNN model significantly outperformed the other two generic models for all four performance metrics; and there were no statistical differences between the personalized model. When evaluating only lower body movement data, the generic model performed substantially higher than when evaluating with only upper body movement data and slightly higher metrics when all movement data are used. A personalized model, however, performed almost identically across all evaluations, no matter the kind of data used (all movement, upper body movement, or lower body movement). This study demonstrates that using HAR models can potentially detect motions that cause RSI, which could result in significant financial benefits for society.

Index terms— human activity recognition, personalized models, convolutional neural network, transformer model, transfer learning, deep learning

Lay Summary

Repeating the same motion can lead to muscle and nerve injuries. Normally, these injuries develop slowly and the pain becomes more constant over time, but if the repetitive movements are recognized early and stopped, injuries can be easily prevented. Most algorithms for detecting human motion require data upfront since people move differently. To solve this problem, an algorithm can be personalized to a particular person to help detect their movements more easily. This thesis proposes a method to detect upper and lower body movements with a personalized algorithm. This method involves creating a general algorithm that generally understands how humans move, then personalizing that algorithm to each person. The algorithms were tested in the following ways: only upper body movement data, lower body movement data, and all movement data. The same trends were seen in all the tests: the personalized algorithm performed better in detecting human motion than the general algorithm. The personalized algorithm performed almost the same with different kinds of data. In contrast, when the same tests were carried out on the general algorithm, the results differed with different data. The personalized algorithm shows promise in accurately detecting repetitive movements to prevent injuries.

Acknowledgements

I want to express my deepest gratitude and appreciation to my supervisors, Dr. Katarina Grolinger and Dr. Ana Luisa Trejos, for their guidance and support throughout the process of completing my thesis. Their expert knowledge and invaluable insights have been crucial to the success of this project. I am truly grateful for the time and effort they have invested in my academic development, and I am confident that the skills and experiences I have gained from working with them will serve me well in my future endeavors.

I am profoundly grateful to have had the support and encouragement from my family, especially my mother, Cynthia Meek, throughout this challenging journey. She has consistently supported my passions and encouraged me to pursue the career path I have chosen, and has been a constant source of inspiration.

I also want to express my sincere appreciation to the Ontario Graduate Scholarship and the Vector Institute Scholarship for providing financial support for my master's studies. Without their generous funding, I would not have been able to pursue my education and achieve my academic goals. I am grateful for their investment in my future and am proud to have completed my masters with their support.

Last but not least, I want to sincerely thank my close friends, Ben, Kiran, and Sam. Your dedication, trust, and loyalty have been invaluable to me and have consistently provided me with the support and encouragement I needed. Your friendship has taught me the importance of seeking true happiness and the value of cherishing those who are dear to us. I am so grateful to have you in my life and for your constant support. Thank you for everything.

Contents

List of Figures

List of Tables

List of Abbreviations

RSI	Repetitive Stress Injuries
ML	Machine Learning
HAR	Human Activity Recognition
CNN	Convolutional Neural Network
NLP	Natural Language Processing
RNN	Recurrent Neural Network
SVM	Support Vector Machine
LSTM	Long Short-Term Memory
TP	True Positives
TN	True Negatives
FP	False Positives
FN	False Negatives

Chapter 1

Introduction

1.1 Motivation

Humans have the tendency to repeat the same motion over and over. Repetitive motion can lead to repetitive stress injuries (RSI) [1]. The damage to the tendons, muscles, and nerves can be temporary or permanent [2]. Many activities such as using a computer mouse, working on an assembly line, and swiping items at a grocery store can lead to RSI. Common injuries include carpal tunnel syndrome, bursitis, tendonitis, trigger finger, and tennis elbow [1]. A range of symptoms can be experienced with RSI, including pain, swelling, stiffness, tingling, numbness, and/or weakness, that may begin gradually and become more severe and persistent over time [3]. According to the Occupational Safety and Health Administration, RSI affects some 1.8 million workers per year [4]. RSI also affects about 15% of Canadians [5]. These injuries reduce the quality of life and are a financial burden to society. In the USA, RSI costs $15 to $20 billion a year in workers' compensation [6]. If a machine learning (ML) model could watch over a person and monitor their activities, the employee could be given advice to adjust their behaviour to prevent injury if they are repeating motions that can lead to RSI. A preventative approach could lead to fewer injuries and substantial financial savings for society. This could be implemented in factories to warn workers about possible injuries.

A production line worker is an employee that works in a factory and completes various jobs. They are required to perform a number of tasks, consisting of assembling and checking products, ensuring all machinery is functioning properly, and assisting with the packaging and shipping of products. These workers can perform the same job every day for years leading to an increased risk of developing RSI. Due to the gradual development of RSI, the warning signs are often ignored by production line workers. As a result, if the symptoms are not treated, they may eventually become constant and can become detrimental to the worker's job performance or even their ability to carry out light duties. In 2001, 5.4 million days (about 15,000 years) were lost in sick leave due to RSI and, every day, six workers leave their jobs because of RSI [7]. When the symptoms are recognized early enough, these injuries can be prevented and treated by altering the way they work. Having a ML model that can warn workers about dangerous movements that can cause RSI could lead to happier workers and therefore, a more efficient factory. Before this problem can be solved, a ML model must be able to detect different types of human movement.

Human activity recognition (HAR) models attempt to detect the activity that a human is carrying out based on raw data from sensors [8]. There are two types of HAR: vision-based and sensor-based [9]. Vision-based HAR uses video or image data, while sensor-based HAR uses time series data collected from sensors, such as accelerometers and/or gyroscopes [10]. In this thesis, only sensor-based HAR is examined due to its advantage over cameras in monitoring activities, since cameras are constrained by a small observation space [11]. Historically, recognizing human activities using HAR models has been successful, but these models rely on having all of the data upfront, which means they cannot adapt to new participants without being retrained [12]. This limitation arises from the fact that people move differently due to differences in body size, gender, age, and other physiological characteristics [13].

Two different people can perform the same movement, and even though it may look the same to the naked eye, on a sensor reading level, it can look very different. Thus, the only accurate and complete way to detect a user's movement is through their own motion data. Since movements have a great deal of variation depending on a person's physiological properties, having one model for everyone will lead to poor results. The typical method to train a HAR model is to split a dataset into two parts: 70% for the training set and 30% for the testing set. For this method to work, data from every participant must be collected before training can begin. Due to this requirement, adding a new participant to the dataset could result in poor performance for the new participant [14]. In general, these models perform well when they are applied to participants on which they have been trained, but their performance drops dramatically when applied to new participants. The personalization of models can solve this problem by tailoring the model to a specific participant.

When a model is personalized for a target participant, the movement of that participant can be detected more easily. In general, the personalization of models is split into two phases: creating the generic model and then personalizing the generic model with a personalization technique. In the case of M participants in the dataset, the generic model is developed by training it with data from all M–1 participants (excluding the target participant). The generic model has insight into the general movement of all participants but lacks precision when applying it to a specific target. In contrast, the personalized model provides more accurate results for a specific target. To personalize the generic model to a specific target, some of the target participant's data will be used along with a personalization technique.

In previous studies, the researchers did not examine the recognition of upper body movements. They have either used data from only the lower body or data from both the upper and lower body. The majority of RSI cases are caused by upper body movements; unfortunately, without the use of a model that can accurately discriminate between upper and lower body movements, the model could not be used to possibly prevent RSI. The goal of this thesis is to determine how accurately an ML model can detect upper and lower body movements.

1.2 Contribution

This work aims to explore the use of deep learning models to detect upper and lower body movements. The main contributions of this thesis can be summarized as follows:

- The design and use of the Trans-CNN model for HAR: Trans-CNN model is a hybrid model combining the CNN model with the transformer model. The Trans-CNN model was designed to capture the advantages of both models.

- Personalized transformer model evaluated with HAR data: in the past year, the transformer model started being used for HAR data; however, personalizing a transformer model for

HAR has not been explored.

- Evaluation of both the generic and personalized models with upper and lower body movement data together and separately: both the generic and personalized models were evaluated with upper and lower body data, as well as with only upper body data and only lower body data.
- Comparing the difference in the performance of the models using upper and lower body movement data together and separately: analyzing the differences in the performance of the models when using upper and lower body movement data together versus using them separately.

1.3 Thesis Outline

The remainder of this thesis is organized as follows: Chapter 2 describes the background, which includes Convolutional Neural Network in Section 2.1, Transformer in Section 2.2, Transfer Learning in Section 2.3, and finally a Chapter Summary in Section 2.4.

Chapter 3 discusses the related work. First, Section 3.1 discusses Sensor-based HAR. Next, Section 3.2 examines Smartphones and Smartwatches HAR, followed by Section 3.3, which reviews Personalized Models HAR, and finally, Section 3.4 is the Chapter Summary.

Chapter 4 presents the methodology for personalized HAR models. First, Data Preparation is described in Section 4.1 and Model Structures in Section 4.2. Next, the Model Training Procedure is presented in Section 4.3 and the Chapter Summary is in Section 4.4.

Chapter 5 describes the evaluation of the methodology: Section 5.1 presents the Dataset and Section 5.2 is the Data Preparation. The Model Structures are defined in Section 5.3 and Model Training Procedure is presented in Section 5.4. Section 5.5 is the Discussion for the evaluations. Lastly, the Chapter Summary is in Section 5.5.

Finally, Chapter 6 concludes the thesis and discusses future work.

Chapter 2

Background

This chapter will detail the two different ML models: Convolutional Neural Network (CNN) and transformers. These models will be used to detect the activities that the human is carrying out. Transfer learning, a ML method that leverages information gained from a related problem to tackle a new problem, will also be discussed.

2.1 Convolutional Neural Network

CNNs are a type of neural network that is optimized for processing data with a grid-like structure, such as images [15]. Because digital images can be represented as a 3D matrix with an RGB grid of pixels, CNNs have been particularly successful in interpreting visual data [16]. CNNs are able to learn features or internal representations of the input data (feature learning) automatically [17]. In addition, CNNs can process one-dimensional (1D) sequence data, such as data from a gyroscope or an accelerometer [18]. When 1D data are the input for a CNN model, the model is referred to as 1D-CNN. For sequence classification tasks, CNNs have the advantage of learning features directly from raw time series data, eliminating the need for manually engineering features [19].

A general CNN model can be seen in Figure 2.1. The model consists of three different types of layers: convolution, pooling, and fully connected. The convolution layer has a matrix of learnable parameters, also known as kernels (filters). The kernels move across the input and perform the dot product with the input matrix to create the activation map. The pooling layer reduces the com-

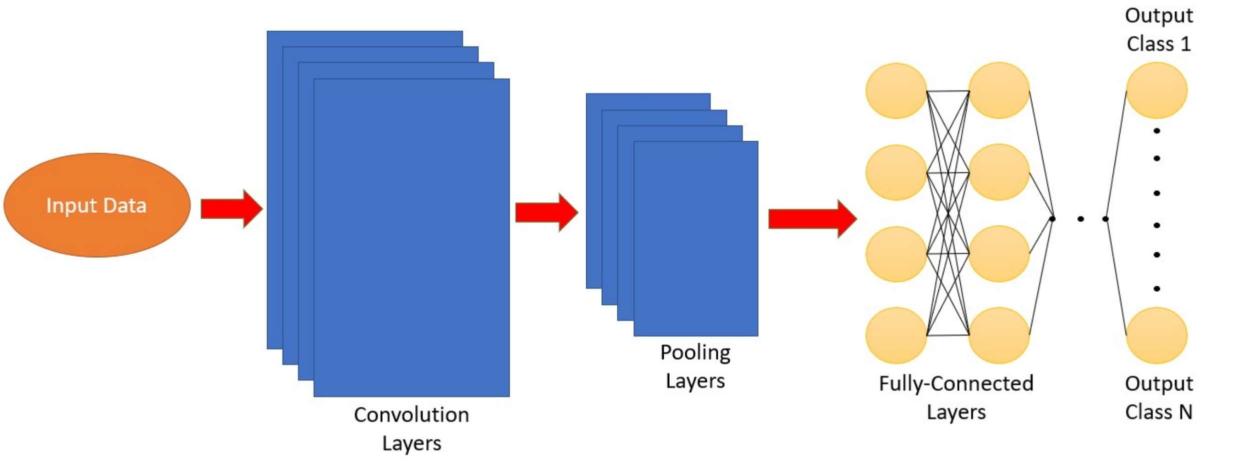


Figure 2.1: CNN Model: The input data go into the convolution layer, where the kernel performs a dot product with the input matrix to create the activation map; the pooling layer uses downsampling to reduce the computational complexity and dimensionality; and the fully connected layers classify the data into various classes.

putational complexity and dimensionality by downsampling. The data can pass through multiple convolution and pooling layer pairs as required before going to the fully connected layer. The final output of the pooling layer is then flattened such that all of the nodes of the fully connected layers are connected to the previous layer. The predictions are produced in the last layer, also known as the classification layer. Lastly, the CNN is updated with backpropagation using gradient descent to update the kernel and weights in the convolution and fully connected layers.

2.2 Transformers

On the other hand, the transformer model, as opposed to CNN, offers an alternative approach for processing sequential data, such as in Natural Language Processing (NLP) tasks. [20]. Transformers as opposed to Recurrent Neural Network (RNN) process the entire input all at once by employing high parallelism, which reduces the training time [21]. The transformer model adopts the mechanism of self-attention, which allows it to learn to focus on different parts of the input sequence in order to understand better the relationships between words or tokens in the sequence [22]. This enables transformer models to learn contextual information about the input, allowing them to perform more complex NLP tasks such as translation and summarization. A transformer consists of an encoder and a decoder [22]. In Figure 2.2, the input enters the embedding and positional encoding unit. The embedding process takes in the input data and transforms the data into a numerical vector. To introduce the order of sequence information into the temporal data, the positional embedding is added to the input embedding. Next, the data go into the encoder, which consists of three types of layers: the multi-head self-attention layer, normalization layer, and feed-forward layer. The multi-head self-attention block consists of multiple parallel attention heads, each of which operates on the input sequence to learn different aspects of the input. This allows the transformer model to learn multiple different aspects of the input simultaneously. Feed-Forward layers generate an abstract representation of the complex input patterns. The decoder has the same layers as the encoder: multi-head self-attention, normalization, and feed-forward. The decoder also has a masked multi-head attention layer, which takes in shifted output embeddings. The masked multi-head attention layer ensures that the prediction for the current position only depends on the outputs of the previous positions. In addition to preserving the auto-regressive features of the transformer network, the masking of the future values allows it to adopt a teacher-forcing learning procedure. Finally, a linear transformation is implemented with a $softmax$ function to produce the probabilities.

In NLP, transformers have been employed to enhance the effectiveness of RNN in various applications [23]. In recent years, models that have been successful in NLP have also shown success in time series data [24]. Therefore, the transformer model has great potential in personalized HAR.

2.3 Transfer Learning

While the CNN and transformer models have achieved success in various tasks, there is still potential for further improvement in performance. One way to address these limitations is through the use of transfer learning, where knowledge learned from one task is transferred to another related task. This technique is widely used in the field of ML as it allows a model that has already been trained on one task to be adapted and applied to a different but related task [25].

Transfer Learning is defined as follows: given a source domain D_s with a source task T_s and a target domain D_T with a target task T_T, transfer learning aims to help improve the learning of

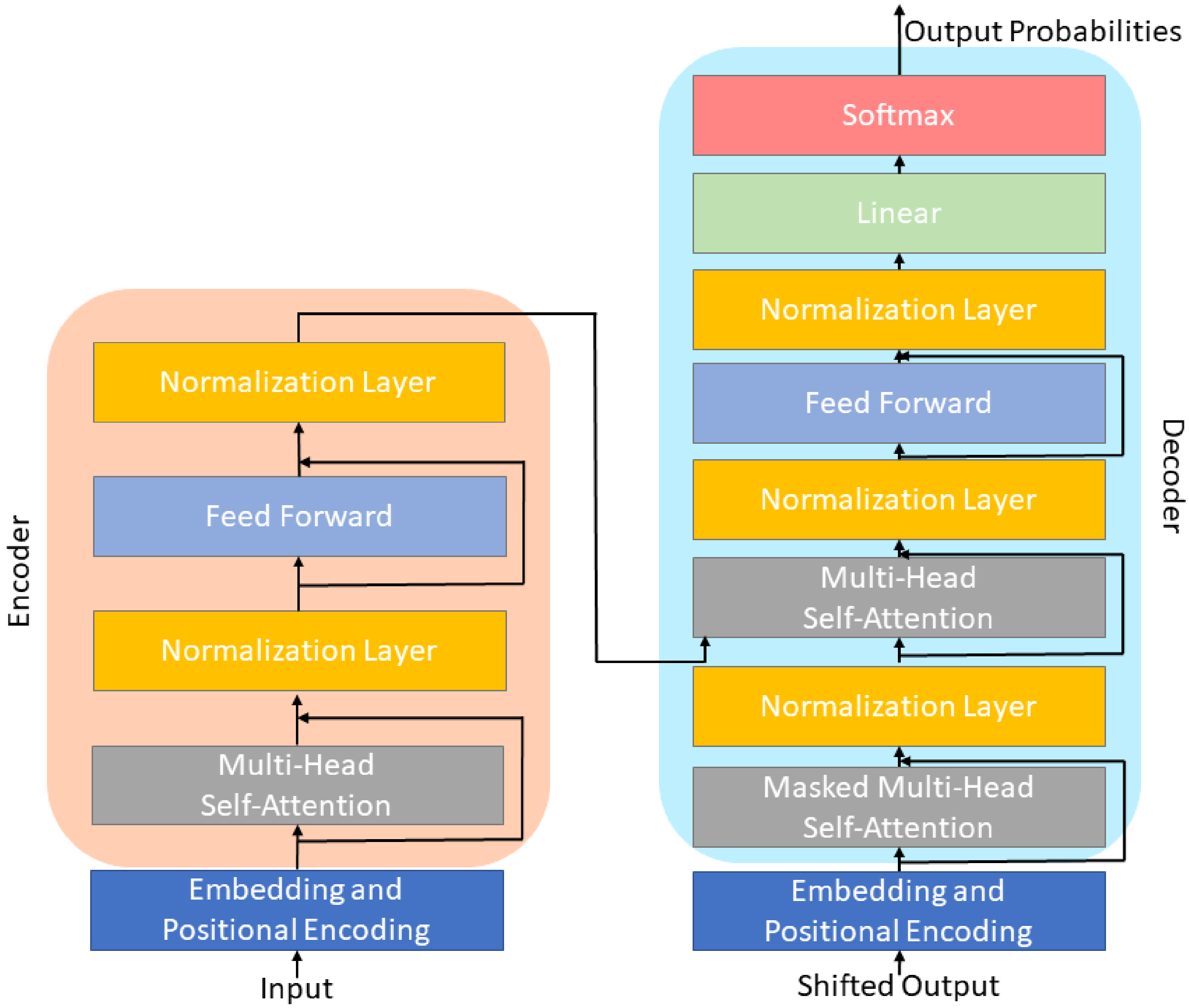


Figure 2.2: Transformer Model: The input enters the Embedding and Positional Encoding block, which converts the input into numerical values; the data then enter the three blocks of the encoder to find association within vector time steps and to generate an abstract representation of the complex input patterns; the data enter the decoder, which has the same three blocks as the encoder plus a Masked Multi-Head Self-Attention. The Masked Multi-Head Self-Attention ensures that the prediction for the current position only depends on the outputs of the previous positions; lastly, a linear transformation and the *softmax* function provide the output as probabilities.

the target predictive function $f_T(\cdot)$ in D_T using the knowledge in D_s and T_s, where $D_s \neq D_T$ or $T_s \neq T_T$ [26].

A visual summary of transfer learning can be seen in Figure 2.3. Transfer learning is particularly useful when there is not enough data available to train a model from scratch on the target task. By using a pre-trained model as a starting point, the model can be quickly fine-tuned to the new task using a smaller amount of training data [27]. This allows the model to utilize the knowledge it has learned from the original task and apply it to the new task, improving the performance of the model on the new task.

2.4 Chapter Summary

In this chapter, two different ML models, CNN and transformer, were described in detail along with a ML technique, transfer learning. CNNs have been historically used for interpreting visual data, such as an image or video, but also have the ability to be used with time series data. Transformers are predominantly used for NLP, but like CNNs, can also be modified for time series data. Transfer learning utilizes the information from one problem to solve another related problem. The next chapter will discuss how other researchers have used these models and other ML models for HAR.

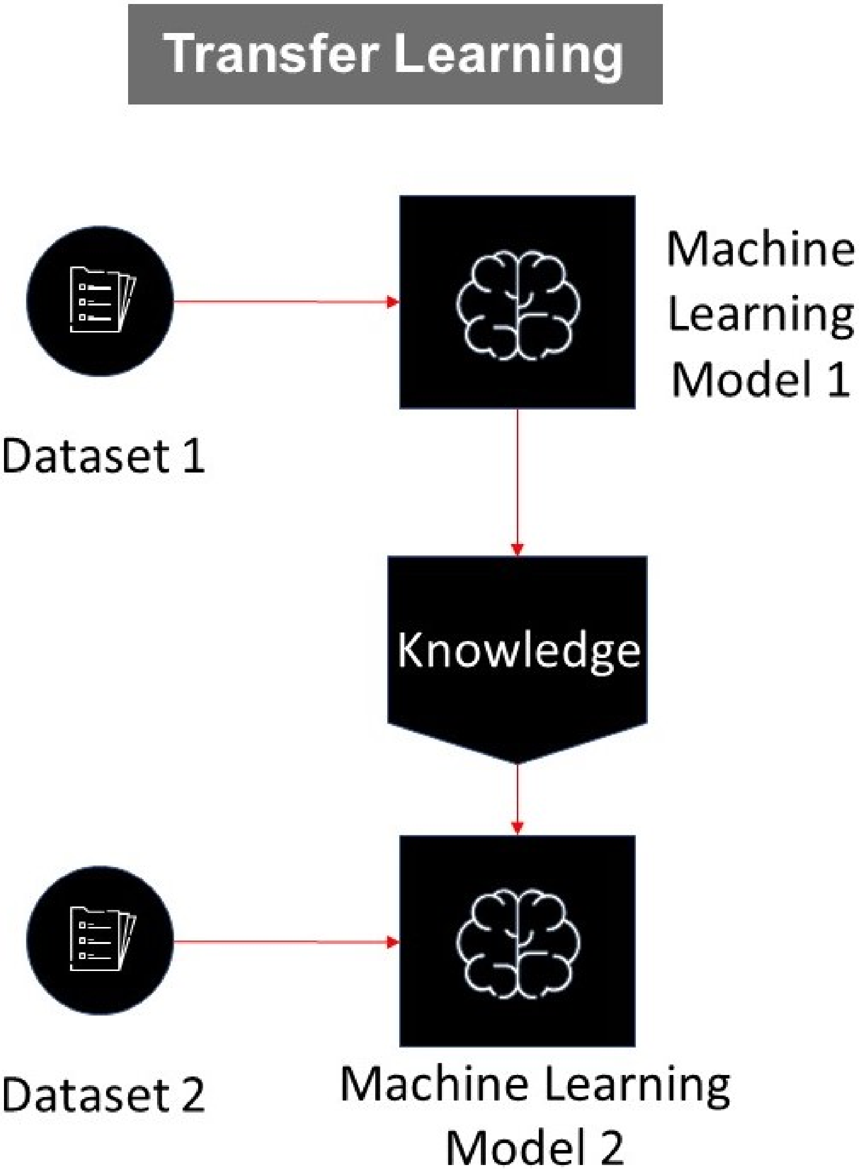


Figure 2.3: Transfer Learning: The knowledge obtained from inputting Dataset 1 into ML Model 1 is applied to ML Model 2 along with Dataset 2.

Chapter 3

Related Work

The previous chapter discussed two ML models, CNN and transformer, as well as transfer learning, a ML technique. This chapter examines how these models and others have been utilized by researchers in the field of HAR. HAR models attempt to detect the activity that a human is carrying out based on raw data from sensors. There are two different types of HAR: vision-based and sensor-based. Vision-based HAR uses video or image data, while sensor-based HAR uses time series data from sensors such as an accelerometer, gyroscope, and/or magnetometer. Only sensor-based HAR will be examined in this thesis. First, sensor-based HAR will be discussed in general followed by an in-depth look at one type of sensor-based HAR: smartphones and smartwatches HAR. Finally, the practicality of personalizing ML models for HAR will be discussed.

3.1 Sensor-based HAR

ML approaches for sensor-based HAR have predominantly been applied to lower body movement data. After surveying the literature to determine how other researchers have approached sensor-based HAR, the most relevant publications related to this topic are discussed.

A study by He et al. [28], used the discrete cosine transform, the principal component analysis and Support Vector Machine (SVM) to recognize human activity from a tri-axial accelerometer sensor. The participants had the tri-axial accelerometer sensor in their pocket and performed four activities for a minute each: walking, jumping, standing still, and running. After the data were collected, the discrete cosine transform extracted features from the data while the principal component analysis reduced the dimension of the features. The SVM model achieved an accuracy of 97.51% for differentiating between the different activities.

A study by Chen et al. [29] investigated HAR using CNN. These researchers placed a single-axis accelerometer onto an individual. The individual then performed seven common activities. The HAR model had an accuracy of 93.8% without any feature extraction methods.

A study by Alsheikh et al. [30] examined the differences between traditional models and deep models for HAR using tri-axial accelerometers. The researchers used three datasets, which involved lower body activities. The researchers proved that deep learning models are more accurate than more traditional models such as K-Nearest Neighbors and Logistic Regression.

Another study by Hendry et al. [31] took a closer look at the relationship between a dancer's training time and the dancer's pain. Dancing involves a lot of training such as jumping and landing. This kind of movement can cause injuries such as foot/ankle, knee and lower back pain if performed incorrectly. The purpose of the study was to develop a ML model with six ActiGraph Link wearable sensors that can accurately detect different movements (jumping and lifting the leg). CNN was applied to the data collected from the wearable sensors and used to develop the ML model. By the end of the study, they developed a ML model that could accurately (97.8%) identify the difference between jumping and leg lifting tasks.

A study by Pienaar et al. [32], looked at using Long Short-Term Memory (LSTM) deep neural architecture to perform HAR. The researchers used a raw sensor dataset called WISDM 2012 [33]. The network was capable of detecting all activities with an accuracy of 94%.

In the studies examined in this section, it can be seen that deep learning models are more accurate for HAR as compared to traditional models, such as SVM and logistic regression. In these studies, the data came from sensors that were placed on the participants. There are some differences between the studies examined and the work presented in this thesis. Overall, the different ML models would not perform well with new participants since all of the studies used a portion of each of the participant's data for training the model. This thesis addresses this problem by personalizing models. Also, since the reviewed studies were limited to lower body movement data, this thesis will look at upper and lower body movement data since RSI results mainly from repeating upper body movements. Finally, this thesis used data from embedded sensors in smart devices instead of data from sensors attached to the participant. As will be discussed in more detail in the following section, the main advantage of using embedded sensors in a smart device is that most people have a smart device; therefore, their motions will be as natural as possible.

3.2 Smartphones and Smartwatches HAR

In the past, HAR required custom hardware to collect sensor data. In today's society, smartphones have the advantage of capturing and processing data over other wearable devices [34]. The participants' movements are more natural when data are collected using smart devices, which are commonly worn by the public and easily collect sensor data.

Staczkiewicz et al. [35] reviewed over 100 articles to determine if smartphone sensors are suitable for human activity recognition and found that smartphones are well-suited for such research in the health sciences.

A study by Ronao et al. [36] looked into using CNN for HAR. The dataset that the researchers used had the participants hold a smartphone in their hand while performing the following activities: standing, walking, going upstairs, going downstairs, and running. For moving activities, CNN was impressively accurate with its predictions. Overall, CNN achieved a performance of 94.79% on the test set of raw sensor data and outperformed other HAR models.

Mekruksavanich et al. [37], examined the benefits of using a hybrid of LSTM and CNN for HAR. The dataset that was used is the 2019 WISDM. The participants had a smartphone and a smartwatch and performed 18 daily activities. All of the models had their hyperparameters tuned with Bayesian optimization. The hybrid deep learning model outperformed other baseline models (only CNN and only LSTM) with an accuracy of 96.2% and an F1 score of 96.3%.

Lastly, Luptáková et al. [38] took a closer look at adapting the transformer model for HAR. They used data from internal sensors (accelerometer and gyroscope) from smartphones. The activities used in the study involved mainly lower body movements. The study concluded that the transformer model could accurately identify the difference between the 18 activities with an accuracy of 99.2%.

In the studies examined in this section, the researchers used deep learning models such as CNN and the data came from embedded sensors in smart devices such as smartphones and smartwatches. The data were both from upper and lower body movements; however, the studies did not focus on upper body movements, which are commonly responsible for RSI. Another difference is that these studies did not employ personalized models. The reviewed studies used data from the same people to train and evaluate models, which is not practical since it would require obtaining upfront all of the needed data from every participant before training can begin.

3.3 Personalized Models HAR

Personalization of the model is necessary since HAR techniques currently rely on a user-independent model that is hard to generalize to new users. In other words, generic or user-independent models only work well on the participants on which they were trained, but their performance decreases greatly with new participants. The first phase in creating a personalized model is to create a generic model. The generic model is trained on many different participants to create a general understanding of the human motions that are being detected. Once the generic model is created, the personalization phase customizes the generic model to a given participant using different techniques.

To create personalized models, Amrani et al. [39] investigated the possibility of using an incremental learning procedure which had three phases: Phase 1 is data preparation; Phase 2 is training the generic model; and Phase 3 is personalizing the model. In Phase 1, one participant was removed from the dataset, and the rest of the dataset was put aside for training. In order to personalize and evaluate the model, the removed participant's data were divided into partitions. In Phase 2, the generic model was created with all of the data except for the removed participant. The generic model was then tested with the portion of the removed participant's data that was not used for personalization. In Phase 3, the model went through the personalization phase. In this phase, the researchers used three approaches to label the data: unsupervised, semi-supervised, and supervised. They found that across all models (Learn++, ResNet, and CNN), the accuracy did increase from the generic model to the personalized model.

A study by Rokni et al [12] looked at using transfer learning to create a personalized CNN model. The researchers initially trained the model on a group of participants to create the generic model. Once the generic model was created, the researchers fixed the weights in all of the layers except the classification layer. The model was then retrained with three labelled instances for each activity. The researchers used two different datasets: SDA and WISDM 2012. Both datasets consist of common lower body human movements. Across both datasets, the personalized model achieved a higher accuracy as compared to models trained using the traditional method (70/30 dataset split).

A study by Gholamiangonabadi and Grolinger [40], examined the use of personalization to select the version of the trained CNN that was best suited for the target participant. The first step in the study was to add the signal decomposition techniques to extract the frequency and time-domain features. They used linear techniques including stationary wavelet transforms variants with mother wavelets db1, db2, db3, and db4 as well as non-linear techniques such as empirical mode decomposition with a linear and cubic spline. Next, to personalize the model, they used a small fragment of the target participant data to select the best suited trained model for the target participant. Personalization did increase the accuracy to a value of 91.2%.

The studies discussed in this section demonstrated different ways of personalizing ML models, as well as the different applications for using personalized models. Most of the studies used deep learning models, such as CNN. This thesis differs from these studies in that it looks at different deep learning models including CNN, transformer, and Trans-CNN in creating models that behave well for new participants. Another important difference is the emphasis on upper body movements. In all of the studies discussed in Chapter 3, none of them examined purely upper body movements. Historically, the focus of HAR has been on analyzing lower body movements. RSI is generally caused by upper body movements; unfortunately, without a model that can accurately detect upper body movements, it is not possible to detect these movements, which if performed repetitively, can cause RSI. This thesis will investigate upper and lower body movements together and separately to see how accurately an ML model can detect different human movements.

3.4 Chapter Summary

In this chapter, sensor-based HAR was discussed in general. In the literature, deep learning models were found to be more accurate for HAR as compared to traditional models such as SVM and logistic regression. Next, an in-depth look at one type of sensor-based HAR: smartphones and smartwatches HAR. These devices produced similar results to sensors placed directly on participants. Lastly, personalized models were shown to be more accurate in detecting the activities performed by new participants than those models trained using traditional methods. The next chapter will detail the entire process to develop a ML model for HAR.

Chapter 4

Design of Deep Learning Techniques for HAR

In order to develop an appropriate model for HAR, the traditional methods for model training were altered by employing the technique of personalization to account for variation in movement from person to person. In this chapter, the entire process to develop a ML model for HAR is discussed. The process involved data preparation, the selection of model structures, and the procedure for model training. This chapter first discusses data preparation which consisted of three components: aligning the data, normalization, and sliding window. Next, the details of the three models selected and their structures are discussed. Lastly, the model training procedure, which is composed of two phases (Generalization Phase and Personalization Phase), is outlined. In order to complete a full analysis, both the generic and personalized models were evaluated in three different ways (All Movement Evaluation, Upper Body Movement Evaluation, and Lower Body Movement Evaluation). Figure 4.1 is a visual summary of the process to develop a ML model for HAR, which involves preparing the data, selecting the model structures, and training the model in two phases.

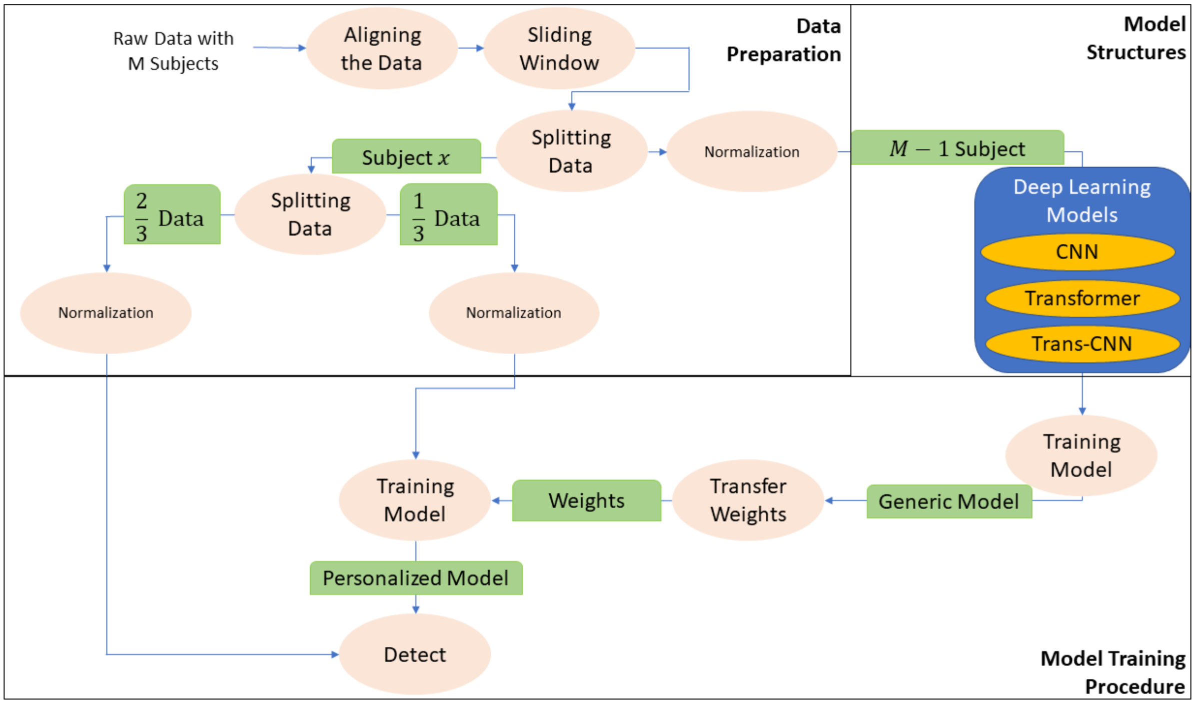


Figure 4.1: The method to develop a ML model for HAR is portrayed as a flow diagram. The method involves the following three sections: data preparation, selection of model structures, and model training procedure. The data preparation consists of three components: aligning the data, sliding window, and normalization. The next step is the selection of the three model structures. As a final step, the model training procedure consists of two phases: the first phase creates the generic model and the second phase creates the personalized model.

4.1 Data Preparation

Data preparation is the process of transforming the raw data into data that can be used for ML algorithms [41]. This process will help the ML model make better predictions. There are three components to this procedure, which are described in the following sections: aligning the data, normalizing it, and compartmentalizing the data (sliding window technique).

4.1.1 Aligning the Data

Aligning the data is a technique to standardize the data to ensure that all of the features have the same length. When using models like CNN, transformer, or similar HAR models, it is necessary to perform this technique because time series data with multiple sensors can result in uneven sample sizes across sensors, even if they are all sampled at the same frequency. This can happen for a number of reasons such as human error, inaccurate sensors, and incorrect clock synchronization between devices. To use the data with ML algorithms, they should be modified to have the same number of readings for each sensor if they have uneven sample sizes. There are a few ways to align the data. One method to align the data is to use the time stamp, which indicates the time at which each sample was collected in time series data.

4.1.2 Sliding Window

For activity recognition, the sliding window technique is widely used to segment accelerometer or gyroscope data [42]. Using this technique, sensor data are partitioned into fixed time slots [43]. In time series data, certain events need to occur before other events. For example, if someone is running then they need to lift their foot up before they move their foot forward. These events can repeat throughout the data, creating a pattern. This pattern can be captured using the sliding window technique. The sliding window technique is applied to help the model capture time dependencies. The sliding window technique transforms time series data into data windows of $w \times f$ size, where w is the number of time steps, and f is the number of features. The first window, the top red box in Figure 4.2, starts at the beginning of the data and has a size of $w \times f$. The window then slides s time steps to create the next window as seen in the bottom red box. This window will have readings from s to $w + s$ time steps. This process will continue to create the rest of the windows.

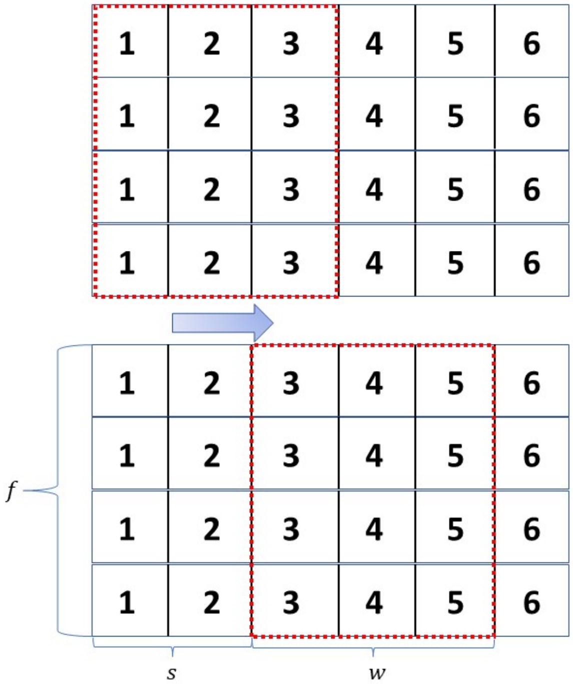


Figure 4.2: Time series data are transformed into data windows of $w \times f$ by using the sliding window technique, where w means the number of time steps and f means the number of features. As shown in the top red box, the first window starts at the beginning of the data and has a size of $w \times f$. As the bottom red box shows, the window slides s time steps to create the next window. This window has readings from s to $w + s$ time steps. The process continues to create the rest of the window as well.

4.1.3 Normalization

Normalization is needed for ML models that are distance based, and for neural networks. Normalization will convert the data to have a similar range throughout the dataset and will prevent features with large values from dominating other features. There are two main normalization techniques, min–max scaling and standardization [44]. Min–max scales the data in the range of [0, 1] or [-1, 1]. In standardization, the data are scaled to have a mean of 0 and a standard deviation of 1. In this thesis, standardization was used to normalize the features because this technique is not sensitive to outliers. The following equation was used to rescale the features to have a mean of zero and a variance of one:

$$z = \frac{x - \alpha}{\sigma}$$

where x is the original feature value, α is the mean of the features, σ is the standard deviation, and z is the normalized value.

4.2 Model Structures

The three models, CNN, transformer, and Trans-CNN, were all examined for their potential use as personalized models for HAR. A comparison was also made between the models that received both upper and lower body movement data as opposed to models that only received lower body movement data or upper body movement data.

CNN models have been used in many different cases for HAR, as seen in Chapter 3. CNNs were designed for images, but are used for time series data as well because time series have a 1D (time) component, which can be extracted [45]. In contrast to other networks, CNNs are able to extract spatial features from the data by using their kernels [46]. This means that manual engineering of input features is not needed. For example, 2D CNN models are commonly used for images [47]. These models can detect edges and other spatial properties in images, which allows them to be robust for image classification [46]. Similar to 2D CNN models, 1D CNN models can also extract spatial features from the data. The main difference between 1D and 2D CNN models is that the kernel only slides in one dimension for 1D CNN, and in two dimensions for 2D CNN. The 1D CNN model is suitable for time series because the kernel only needs to slide in one dimension, which is time [48].

The transformer architecture was modified in this thesis for HAR. Transformers are most commonly used for sequential data such as natural language [22]. Sequential data are any kind of data where the order of the data matters. Since the time series data for HAR are a kind of sequential data, the transformer is an appropriate choice. While the original transformers consist of encoders and decoders, here, only the encoders are used in order to learn latent semantic representations and temporal dependencies.

Moreover, in this thesis, the Trans-CNN model, a hybrid model, was created by combining the CNN and transformer models. The Trans-CNN model was designed for HAR in order to capitalize on the advantages of both models. CNN models have the ability to capture high-level spatial-temporal features while transformers are efficient at capturing latent semantics and global dependencies [49]. By combing the two models, the Trans-CNN structure contains the major components of both models: the encoder block and convolutional block.

4.2.1 CNN Structure

The CNN structure, shown in Figure 4.3, begins at the bottom with the input layer. After the input layer, the convolutional block follows, which contains a convolutional layer, max pooling layer, and dropout layer. For CNN models, convolutional blocks are commonly stacked to ensure that the model can have a hierarchical decomposition of the input [50]. The convolutional layers have weights that need to be trained, while the max pooling layer reduces the dimensions of the feature maps. The dropout layer minimizes overfitting and the generalization error [51]. After the last convolutional block, a flattening layer is added to transform the current output into a one-dimensional vector. Next, three fully connected layers were added to the CNN model to help interpret the features that were learned in convolutional blocks. The output for the last fully connected layer goes to the output layer. The output layer, which is a fully connected layer, outputs the predictions and uses $softmax$ as the activation function. The $softmax$ activation function is represented by the equation below:

$$Softmax(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

The $softmax$ activation function gives each class in a multi-class classification a probability. The probability of all of the classes adds up to 1.0.

4.2.2 Transformer Structure

The modified transformer architecture for HAR can be seen in Figure 4.4. The modified architecture starts with an input layer followed by an encoder. The encoders take in the input and map the input to a higher dimensional space. Next, a global average pooling layer was used. The output from the last encoder goes into the global average pooling layer to reduce the output to a vector of features for each data point in the current batch. The fully connected layer that is used

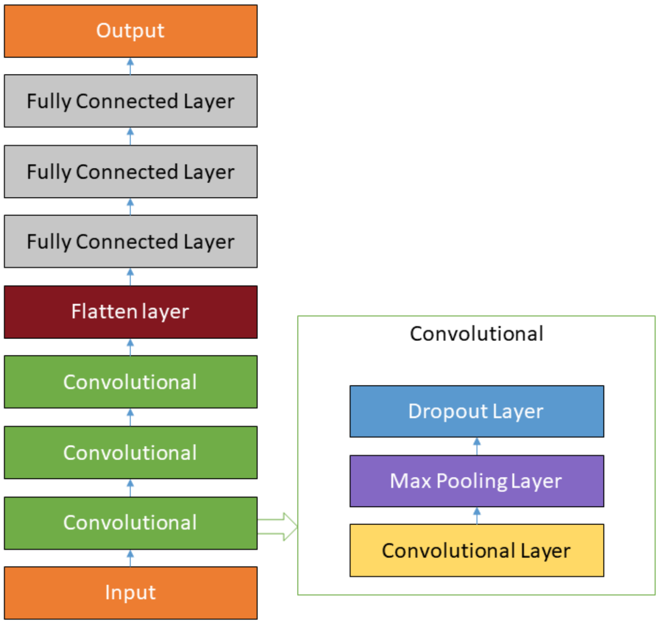


Figure 4.3: The CNN model structure starts with an input layer. A convolutional block consists of the following layers: convolutional layer, max pooling layer, and dropout layer. To convert the output into a one-dimensional vector, a flattening layer is added following the last convolutional block. In order to interpret the features previously learned, three fully connected layers were added to the CNN model structure. The output from the last fully connected layer is taken by the output layer.

as a buffer from the learned features to the predictions is similar to the one in the CNN model. The number of fully connected layers could be varied in this architecture. The output of the fully connected layer is fed into a dropout layer. This layer was added to help reduce overfitting. Lastly, the dropout layer is connected to a fully connected layer, which uses the $softmax$ function as an activation function for predictions.

4.2.3 Trans-CNN Structure

Trans-CNN is a hybrid model that was designed in order to exploit the advantages of both models. The hybrid structure starts with an input layer as seen in Figure 4.5. The structure has encoders after the input similar to the transformer model structure. The output from the last encoder layer is the input to the first convolutional block. Similar to the CNN model structure in Subsection 4.2.1, after each convolutional layer, there is a max pooling layer. Unlike the CNN model structure,

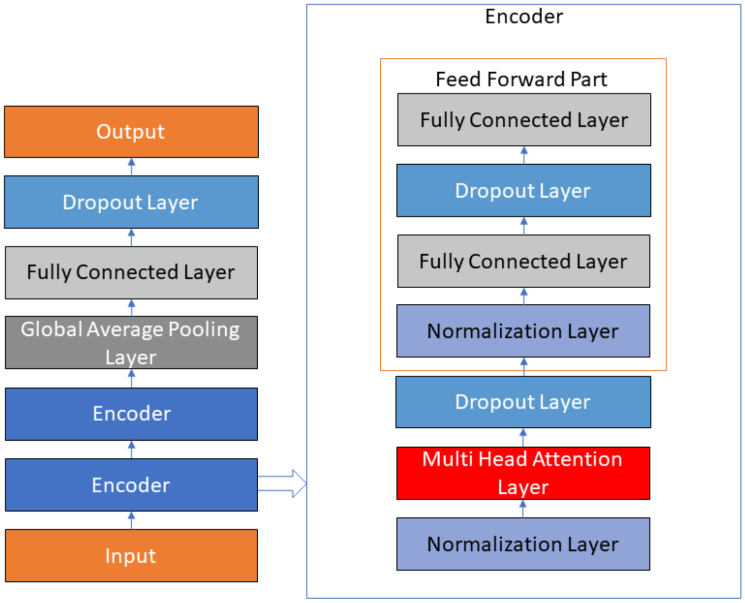


Figure 4.4: Firstly, there is an input layer followed by an encoder in the modified transformer architecture. The output from the last encoder is fed into the global average pooling layer. Next, the global average pooling layer feeds into a fully connected layer. Then, the dropout layer receives the output of the fully connected layer. Lastly, a dropout layer was added after the fully connected layer to help reduce overfitting

there is no dropout layer after the max pooling layer. The dropout layer was omitted since the encoders have two dropout layers inside of them. Next, the output from the last max pooling layer goes to a global average pooling layer. Similar to the transformer structure, only one fully connected layer follows the global average pooling layer. Lastly, a dropout layer was added after the fully connected layer to help reduce overfitting.

4.3 Model Training Procedure

After determining the structure of the three models, the models were trained. In traditional HAR ML model training, some of the data from each participant are in the training dataset. However, when this traditional model is used with new participants, the performance of this model decreases, despite the fact that they work well for participants on which they were previously trained [52]. The decrease in performance with new participants comes from the diversity in humans. People

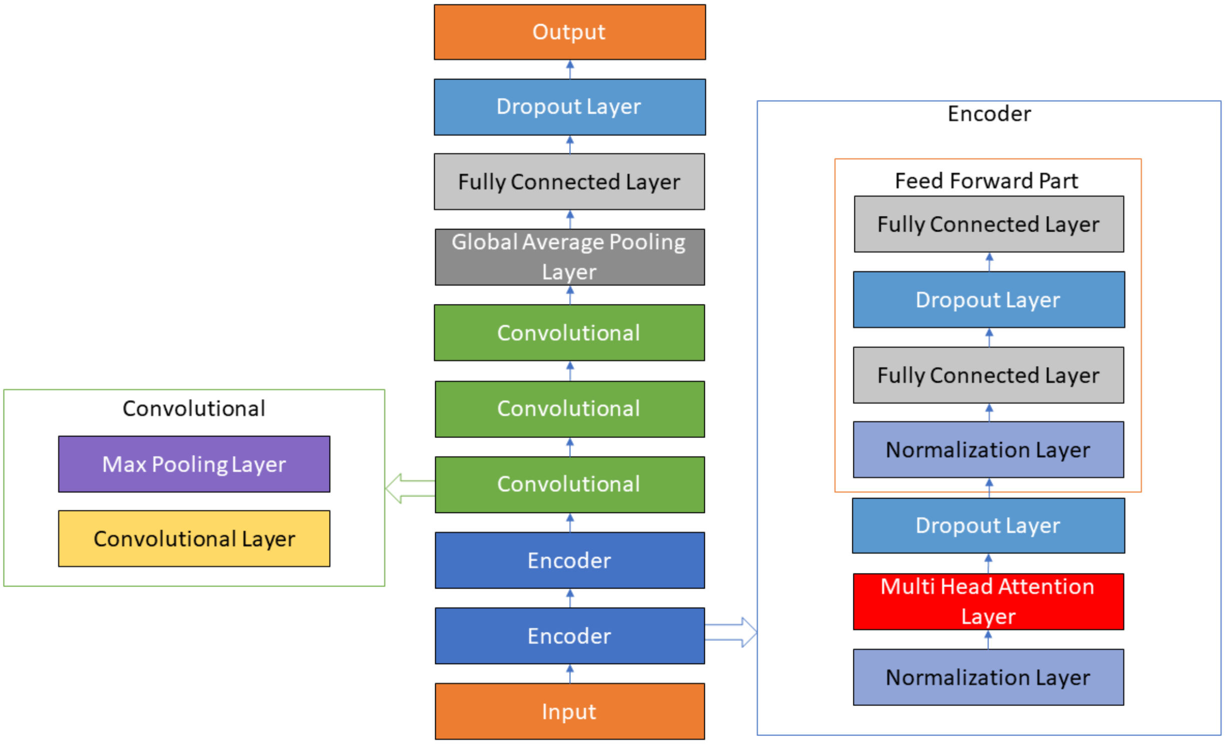


Figure 4.5: The hybrid Trans-CNN model structure starts with an input layer. The input layer is fed into an encoder. The output from the last encoder block is the input to the first convolutional block. The convolutional block for the Trans-CNN model consists of the following layers: the convolutional layer and the max pooling layer. Next, the output from the last max pooling layer is passed to a global average pooling layer and then connected to a fully connected layer. Lastly, a dropout layer was added after the fully connected layer to help reduce overfitting.

can differ in body size, gender, age, and other physiological properties, which leads to the same activity being carried out by two participants in two different ways. As a result, the model cannot be easily adapted to new participants without retraining. In order to solve this problem, ML models can be personalized to a particular participant. Personalization involves learning how a particular participant moves. As discussed in Chapter 3, the personalization of models was split into two parts: creating a general model and then personalizing that model to a certain participant; therefore, the model training procedure consisted of two phases: the Generalization Phase, and the Personalization Phase.

4.3.1 Generalization Phase

The Generalization Phase consisted of training the three models, CNN, transformer, and Trans-CNN, as generic models. For a given dataset with M participants, the generic model was created by training the model using data from all $M-1$ participants (excluding the target participant). This procedure was completed for all M participants. After training, the models became general models for human motion that provide a broad understanding of movement patterns in all participants, but are less precise when it comes to a specific target participant.

4.3.2 Personalization Phase

Personalization provides a more accurate way to achieve results for a particular participant. An individual's movement can be more easily detected when the model is personalized to that individual. As discussed in Chapter 3, other studies have successfully used transfer learning to personalize models. One way of performing transfer learning involves transferring the weights from the generic model to a new model that will be personalized. CNN, transformer, and Trans-CNN will be used to create the generic models. These generic models will provide the initial weights for the personalized model. Next, it is necessary to freeze most of the layers to make sure that all of the knowledge gained from the other participants is preserved. By freezing a layer during training, it prevents its weights from being modified; hence, the knowledge inside the frozen layers is untouched. All of the layers are frozen except for the classification layer (last layer). Since the latter layers are typically learning task-specific features, the classification layer is the only layer that is not frozen. Once the layers are frozen, the data from the target participant were split into two portions, $D1$ and $D2$. The split will be $\frac{1}{3}$ of each class in $D1$ and the remaining $\frac{2}{3}$ in $D2$. $D1$ will be used to train the models after the layers are frozen, and $D2$ will be used to assess each model. During training, only the weights of the classification layer will change to help improve the precision of the model for the target participant.

4.4 Chapter Summary

In this chapter, data preparation was presented, which included the following techniques: aligning the data, sliding window, and normalization. Each of those techniques had its own purpose to help transform the data into useable ML model training data. After the data were ready, the training began on the three models: CNN, transformer, and Trans-CNN. The model training procedure consists of two phases: the Generalization Phase and the Personalization Phase. The Generalization Phase will train the models using all of the participants, M, in the dataset except for the target participant (M-1), which will create the generic model. The Personalization Phase involves personalizing the generic model to the target participant by using transfer learning. Both the generic and personalized models will be evaluated in three different methods: using only lower body movement data, only upper body movement data, and all movement data. The two phases will be performed on all three models for all M participants, and the average of those results will be determined. In the next chapter, the techniques presented in this chapter will be used on an open-source dataset WISDM 2019 to evaluate the appropriateness of these techniques for HAR.

Chapter 5

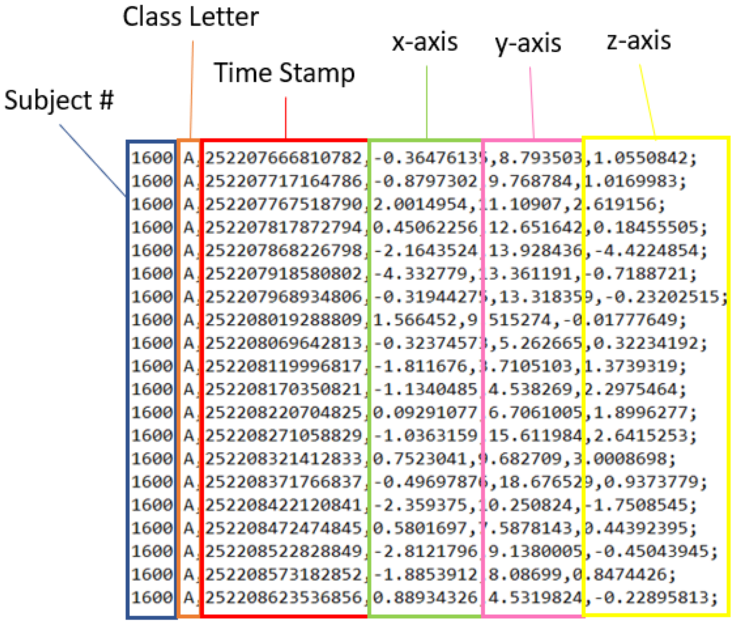
Results and Discussion

In this chapter, the selected dataset is described as well as how the data were prepared for training the model. Next, the details of the three model structures are provided. Lastly, the results of the three evaluations (All Movement Evaluation, Upper Body Movement Evaluation, and Lower Body Movement Evaluation) are analyzed to study whether personalization is beneficial in HAR; whether the transformer model is suitable for personalized HAR; whether there is a difference in the performance of the models depending on whether upper and lower movement data is used.

The algorithms and procedures described were programmed in Python with Keras and TensorFlow deep learning library. The experiments were performed on a computer with Windows 10 OS, Intel(R) Core(TM) i9 CPU, 32 GB RAM, and NVIDIA GeForce RTX 2070 graphics card.

5.1 Dataset

An open-source dataset, WISDM 2019 [53], from the University of California Irvine database was chosen since it contains both upper and lower body movement data. The data in this dataset have been collected using both a smartphone in the participant's pocket and a smartwatch on their dominant hand. A sample of the raw data can be seen in Figure 5.1. The subject number is listed in the first column, followed by the class being performed in the next column. The third column displays the time stamp, while the subsequent three columns record the movements in the x, y, and z directions. Each device has a built-in gyroscope and accelerometer, which were



```
1600 A 252207666810782 -0.36476135,8.793503,1.0550842;
1600 A 252207717164786 -0.8797302 9.768784,1.0169983;
1600 A 252207767518790 2.0014954,11.10907,2.619156;
1600 A 252207817872794 0.45062256 12.651642,0.18455505;
1600 A 252207868226798 -2.1643524 13.928436,-4.4224854;
1600 A 252207918580802 -4.332779,13.361191,-0.7188721;
1600 A 252207968934806 -0.31944275,13.318359,-0.23202515;
1600 A 252208019288809 1.566452,9 515274,-0.01777649;
1600 A 252208069642813 -0.32374573,5.262665,0.32234192;
1600 A 252208119996817 -1.811676,3.7105103,1.3739319;
1600 A 252208170350821 -1.1340485 4.538269,2.2975464;
1600 A 252208220704825 0.09291077 6.7061005,1.8996277;
1600 A 252208271058829 -1.0363159 15.611984,2.6415253;
1600 A 252208321412833 0.7523041,9.682709,3.0008698;
1600 A 252208371766837 -0.49697876,18.676529,0.9373779;
1600 A 252208422120841 -2.359375,10.250824,-1.7508545;
1600 A 252208472474845 0.5801697,7.5878143,0.44392395;
1600 A 252208522828849 -2.8121796 9.1380005,-0.45043945;
1600 A 252208573182852 -1.8853912 8.08699,0.8474426;
1600 A 252208623536856 0.88934326 4.5319824,-0.22895813;
```

Figure 5.1: Raw data collected from a phone accelerometer. The first column displays the subject number, followed by the class of activity being performed in the next column. The third column shows the time stamp, while the following three columns record the movements in the x, y, and z directions.

used to collect data from the participants' movements while carrying out various activities. The data were collected from 51 participants who performed 18 different activities, such as walking, sitting, and eating, for a period of three minutes for each activity. Three readings were collected from each sensor: the phone's gyroscope, the phone's accelerometer, the watch gyroscope, and the watch accelerometer. The three readings collected by each sensor are x, y, and z axis coordinates; therefore, a total of 12 readings were available. The label for each activity was identified by a letter from A–S (no 'N'). Each sensor collected the data at a rate of 20 Hz, but when the data were analyzed, it was found that each sensor had a different number of readings. This discrepancy had to be resolved, as explained in Section 5.2.

5.2 Data Preparation

Following the selection of the dataset, data preparation is necessary in order to transform the raw data into data that can be used for ML models. Data preparation includes aligning and normalizing the data as well as applying the sliding window method.

5.2.1 Aligning the Data

After comparing the number of readings for each of the 12 features in the data, it was found that the participants had a different number of samples for each feature. The feature with the largest number of samples varied from participant to participant. This was unusual since if each sensor was reading at the same rate for the same period of time then all of them should have had the same number of readings. Synchronization issues between the phone and the watch can possibly explain this discrepancy. For all 51 participants, the extra values were truncated by removing the end readings so that all of the features had the same number of samples for that particular participant. The maximum amount of samples that were truncated was 1.9% of the whole amount of samples.

After truncating the extra data points, a check was done to determine whether the dataset was balanced. A balanced dataset is one that has the same number of samples for each class. For a balanced dataset, all 18 activities in this dataset had the same number of samples per class. Some of the participants did not have data for some of the activities and other participants had data for all activities, but insufficient data for some activities. In the end, the data from 12 participants were unbalanced. Those 12 participants were removed from the dataset leaving 39 participants for the analysis presented herein.

5.2.2 Sliding Window

The width of the window was chosen to be 10 seconds since a human can carry out the activities present in the dataset multiple times in that time period. The data were sampled at a rate of 20 Hz; therefore, the width of the window was 200 samples. Since there were 12 features in this dataset, the dimension of the window was 12×200. A 75% overlap was chosen, which indicates that the window will move 50 time steps each time it slides.

5.2.3 Normalization

In addition to using the sliding window technique, it was important to also apply a method of normalization. Both the min–max normalization and standardization methods were considered to normalize the data, but the standardization method was chosen since it is more effective at handling outliers. Each of the 12 participants' features were normalized separately.

5.3 Model Structures

The three ML models discussed in Section 4.2 were examined with respect to their use as personalized models for HAR. As well, the performance of the ML models for detecting upper and lower body movements as opposed to focusing exclusively on lower body movements is discussed. For the three models, the hyperparameters, parameters that are selected before the model is trained, were tuned using grid search with k-fold cross-validation. In grid search, the hyperparameters are divided into discrete grids. Then, each combination of values in this grid is tried and evaluated with k-fold cross-validation to calculate the performance metrics. The value of k was chosen to be 5. The combination with the best performance metric score was chosen. Accuracy was chosen as the performance metric to select the hyperparameters since for a balanced dataset, the scores for precision, recall, and f1-score tend to exhibit similar patterns to accuracy [54].

5.3.1 CNN Structure

The CNN model structure including the different layers can be found in Figure 4.3. The hyperparameters considered in tuning and the values selected by the grid search are shown in Table 5.1. From the tested values, the grid search chose the best among the group.

Table 5.1: The four tuned hyperparameters of the CNN model are shown in the table together with their considered values. Kernel sizes in the vector [i,j,l] correspond to the kernels applied in the three convolutional layers. The selected values are determined through the grid search.

Hyperparameters	**Tested**	**Selected**
Filter Sizes	32, 64, 128	64
Kernel Sizes	[3,3,3], [5,5,5], [11,11,11], [3,5,11]	[3,5,11]
Dropout Rate	0.2, 0.25, 0.3	0.25
Optimizer	Adam, SGD	Adam

Filter sizes 32, 64, and 128 were tested and 64 was chosen. Different combinations of kernel sizes, [3,3,3], [5,5,5], [11,11,11], and [3,5,11], where the first number represents the kernel size for the first convolutional layer, the second number is for the second layer and the last number is for the third layer, were tested and [3,5,11] was chosen. Dropout rates of 0.2, 0.25, and 0.3 were tested and 0.25 was chosen. For the optimizer, Adam and SGD were tested and Adam was chosen.

For the max pooling layer, a size of 2 was chosen since this is commonly used [55]. The number of neurons for the three fully connected layers was chosen to be [128,64,32] to add a buffer between the feature extraction layers and classification layers. Convolutional blocks of 1, 2, 3, and 4 were tested, and the best results were achieved with 3 blocks.

5.3.2 Transformer Structure

The modified transformer model structure can be found in Figure 4.4. Table 5.2 details the hyperparameters and their values tested with grid search. Using grid search, the best value was chosen from all the values tested.

Table 5.2: The four tuned hyperparameters of the transformer model are shown in the table together with their considered values. The selected values are determined through the grid search.

Hyperparameters	**Tested**	**Selected**
Dropout Rate	0.2, 0.25, 0.3	0.25
Number of Heads	1, 2, 4, 8	4
Head Size	16, 32, 64	32
Number of Neurons	512, 1024, 2048	1024

Dropout rates of 0.2, 0.25, and 0.3 were tested and 0.25 was chosen. The number of heads was varied (1, 2, 4, 8) and 4 was chosen. Head sizes of 16, 32, and 64 were tried and a head size of 32 was chosen. The number of neurons for the fully connected layer was varied (512, 1024, and 2048) and 1024 was chosen.

Two encoders were chosen, without testing with grid search, to minimize the complexity of the model and computation time.

5.3.3 Trans-CNN Structure

Figure 4.5 shows the Trans-CNN model structure. For the Trans-CNN model, five hyperparameters were tuned and the other hyperparameters were selected. For each tuned hyperparameter, the considered and chosen values can be found in Table 5.3.

Table 5.3: The five tuned hyperparameters of the Trans-CNN model are shown in the table together with their considered values. Kernel seizes in the vector [i,j,l] correspond to the kernels applied in the three convolutional layers. The selected values are determined through the grid search.

Hyperparameters	**Tested**	**Selected**
Dropout Rate	0.2, 0.25, 0.3	0.2
Filter Sizes	32, 64, 128	128
Kernel Sizes	[3,3,3], [5,5,5], [11,11,11], [3,5,11]	[3,5,11]
Optimizer	Adam, SGD	Adam
Number of Neurons	512, 1024, 2048	2048

Dropout rates of 0.2, 0.25, and 0.3 were tested and 0.2 was chosen. There were three filter sizes tested: 32, 64, and 128 and 128 was chosen. The combination of kernel sizes [3,3,3], [5,5,5], [11,11,11] was tested, and [3,5,11] was chosen. SGD and Adam were tested as optimizers, and Adam was chosen. For the fully connected layer, different numbers of neurons were tested. After testing 512, 1024, and 2048, 2048 was chosen.

The following hyperparameters were chosen without testing with grid search to minimize computation time: 2 encoder blocks, 3 convolutional blocks, 4 for the number of heads, and 32 for the head size.

5.4 Model Training Procedure

The model training procedure was carried out in two phases: the Generalization Phase, and the Personalization Phase. The Generalization Phase involved training with data from all 38 participants (removing the 39th, the target participant) on the following three models: CNN, transformer, and Trans-CNN. For training the CNN model, 150 epochs was used as that was sufficient for the algorithm to converge. For both the transformer model and Trans-CNN model, 100 epochs were sufficient. The Personalization Phase involved personalizing the generic model to the target participant using transfer learning. The target participant's data were split into two portions: $D1$ (33% of the target participant's data) and $D2$ (67% of the target participant's data). Transfer learning was used to personalize the models using $D1$. During personalization, the algorithm converged after 100 epochs for all three types of models. $D2$ was used after each phase to evaluate each model using different data: only lower body movement data, only upper body movement data, and all movement data. For all participants, both phases were conducted on the three models.

The Generalization Phase and the Personalization Phase were the same across all of the evaluations: All Movement Evaluation, Upper Body Movement Evaluation, and Lower Body Movement Evaluation. A single model that can handle both upper and lower body movement data is essential since, in the real world, the type of movement data will not be known beforehand.

Three evaluations were conducted on both the generic and personalized models: All Movement Evaluation, Upper Body Movement Evaluation, and Lower Body Movement Evaluation.

5.4.1 Experimental Methods

When a ML model makes a prediction, the prediction will fall into one of these four categories: true positives, true negatives, false positives, and false negatives. True positives (TP) occur when the actual class value and the predicted class value are both true. In other words, if a participant is walking and the model predicts the participant is walking, the prediction is a true positive. True negatives (TN) are predictions that are false and the value of the actual class is also false. Furthermore, if a participant is not walking and the model predicts the participant is performing an activity other than walking, the prediction is a true negative. False positives (FP) occur when the actual class is false and the predicted class is true. To put it another way, if a participant is not walking and the model predicts the participant is walking, the prediction is a false positive. False negatives (FN) happen when the actual class is true but the predicted class is false. If a participant is walking and the model predicts the participant is performing an activity other than walking, the prediction is a false negative.

TP, TN, FP, and FN are the building blocks of the performance metrics that were used to evaluate the models. The performance metrics quantify how well the model handles data. Four out of the most commonly used performance metrics for classification problems were chosen to evaluate the model: accuracy, precision, recall, and f1-score.

Accuracy is measured by the ratio of correct predictions to total predictions defined as follows:

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

Precision is the measure of true positives divided by all of the positives predictions defined as follows:

$$Precision = \frac{TP}{TP + FP}$$

Recall is the measure of the ratio of the corrected predicted positives over the total number of positives in the sample defined as follows:

$$Recall = \frac{TP}{TP + FN}$$

F1-Score is the weighted average of precision and recall defined as follows:

$$F1\text{-}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} = \frac{2TP}{2TP + FP + FN}$$

These metrics (accuracy, precision, recall, and f1-score) were used in the three evaluations for testing the performance of the generic and personalized models.

A confusion matrix is a tool that is commonly used in the evaluation of the model's performance [56]. It displays the number of correctly and incorrectly classified instances. The confusion matrix also offers insight into which classes are being confused and where the model may be failing. This information can then be used to improve the performance of the model. The confusion matrix is defined as the following matrix:

		Predicted Class		
		p	**n**	**total**
True Class	**p′**	True positive	False negative	P′
	n′	False positive	True negative	N′
	total	P	N	

To determine if there was a significant difference between the models, a statistical analysis was completed. The following methods were used for the statistical analysis to understand the results better:

1. Shapiro–Wilk test

   The Shapiro–Wilk test is used to determine whether the data distributions are normal or not.

2. Paired t-test

   The paired t-test is a statistical test that compares two related parametric groups to determine whether there is a significant difference between them.

3. ANOVA test

   The ANOVA test determines whether there are any statistical differences between the means of three or more parametric groups.

4. Mann-Whitney test

   The Mann-Whitney test is a statistical test that compares two nonparametric groups to determine whether there is a significant difference between them.

5. Kruskal-Wallis test

   The Kruskal-Wallis test determines whether there are any statistical differences between the means of three or more nonparametric groups.

5.4.2 All Movement Evaluation

In the all movement evaluation, all of the activities were included, as seen in Table 5.4. The generic and the personalized models were evaluated with $D2$, the dataset containing $\frac{2}{3}$ of the data that were not used for model training or hyperparameter selection. The generic model is evaluated after phase one is completed and the personalized model is evaluated after phase two is finished.

Table 5.4: The dataset used in this evaluation included 18 activities that involve either upper or lower body movements.

Activities in the Dataset
Walking
Jogging
Stairs
Sitting
Standing
Kicking (Soccer Ball)
Typing
Brushing Teeth
Eating Soup
Eating Chips
Eating Pasta
Drinking from Cup
Eating Sandwich
Playing Catch w/Tennis Ball
Dribbling (Basketball)
Writing
Clapping
Folding Clothes

To determine the best model for HAR, it is important to know whether the data from the results have a normal distribution. To determine whether the data had a normal distribution, the Shapiro–Wilk test was applied to the results of the 39 participants. Table 5.5 outlines which data were normally distributed and which data were non-normally distributed. This information will be used to select the statistical test when examining the significance of the results.

Table 5.5: Shapiro–Wilk test was used to determine if the results were normally distributed or not.

Normal Distributed Data	**Non-Normal Distributed Data**
Generic CNN Accuracy	Generic Trans-CNN Accuracy
Generic CNN Precision	Generic Trans-CNN Precision
Generic CNN Recall	Generic Trans-CNN Recall
Generic CNN F1-Score	Generic Trans-CNN F1-Score
Generic Transformer Accuracy	Personalized CNN Accuracy
Generic Transformer Precision	Personalized CNN Precision
Generic Transformer Recall	Personalized CNN Recall
Generic Transformer F1-Score	Personalized CNN F1-Score
	Personalized Trans-CNN Accuracy
	Personalized Trans-CNN Precision
	Personalized Trans-CNN Recall
	Personalized Trans-CNN F1-Score

Using the results from the 39 participants, the four performance metrics (accuracy, precision, recall, and f1-score) were plotted using a box plot, a graphical summary of a dataset, for both the generic and personalized CNN models, as seen in Figure 5.2. In these box plots, the box represents the interquartile range. The first quartile is the bottom of the box and the third quartile is the top of the box. The maximum value (excluding outliers) in the dataset is at the top of the vertical line and the minimum value (excluding outliers) is at the bottom of the vertical line. The horizontal line that is in the box is the median of the dataset. The points outside of the maximum and minimum values are outliers.

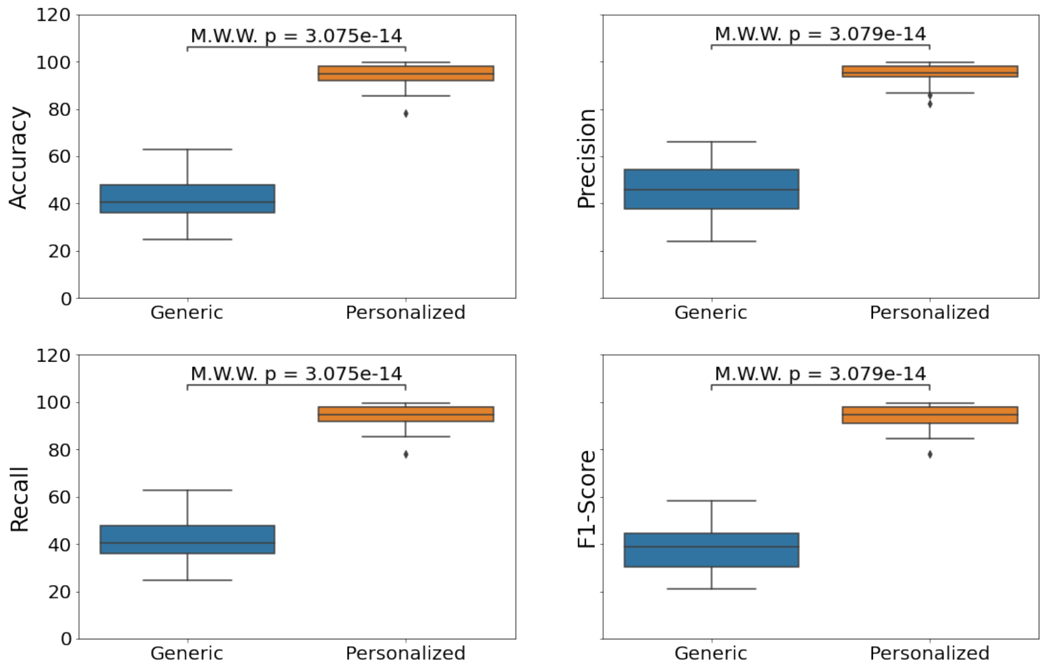


Figure 5.2: For the All Movement Evaluation, these boxplots display the four performance metrics for both the generic and personalized CNN models.

In Figure 5.2, the generic CNN model performance metrics are around 40%. For example, for accuracy, when a new participant is tested on the generic model, the model can detect the correct human motion on average around 40% of the time. On the other hand, the personalized CNN model performance metrics are around 90%. To examine if there is a statistically significant difference between the generic and personalized models, the Mann-Whitney test was performed since the data of the personalized model are nonparametric. The test resulted in a p value of less than 0.05 for all four performance metrics. Hence, there was a statistically significant difference between the generic and personalized models. The p values can be found in Figure 5.2.

Box plots were also created for the transformer and the Trans-CNN models as seen in Figure 5.3 and Figure 5.4. The same trends are seen in both figures as was seen in Figure 5.2. The generic transformer model performance was around 50%, and the personalized transformer model performance was around 90%; whereas the performance for the Trans-CNN generic model was around 60%, while the performance for the personalized Trans-CNN model was around 90%. The Mann-Whitney test was also chosen since both of the data of the personalized models are nonparametric. The p values for all four performance metrics for both the transformer and the Trans-CNN models were less than 0.05; therefore, there was a statistically significant difference between the generic and personalized models for both the transformer and the Trans-CNN models.

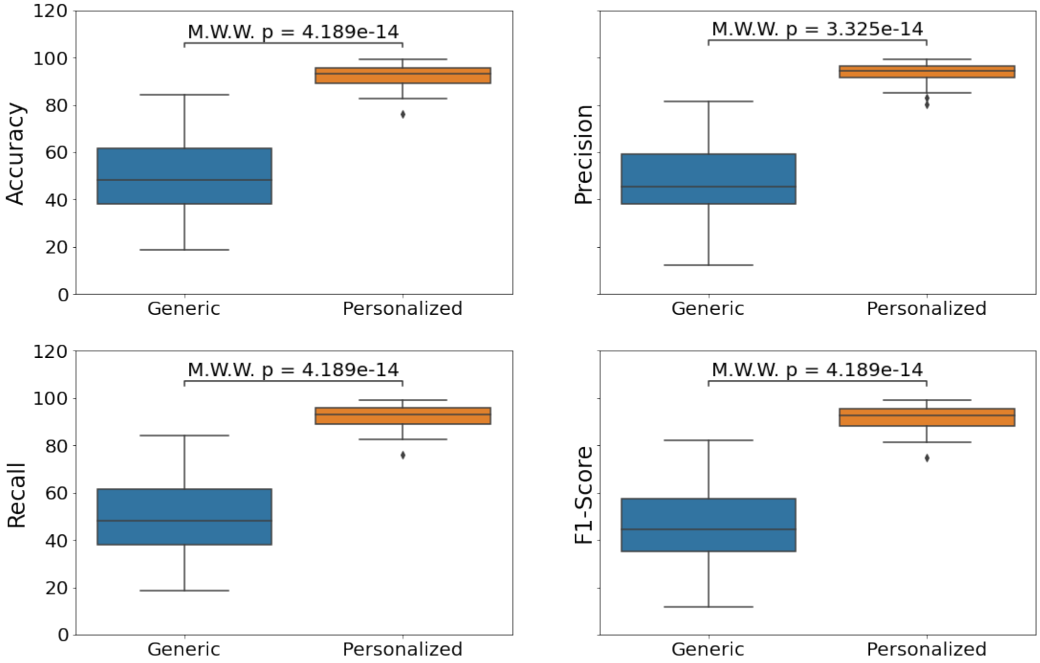


Figure 5.3: For the All Movement Evaluation, these box plots display the four performance metrics for both the generic and personalized the transformer models.

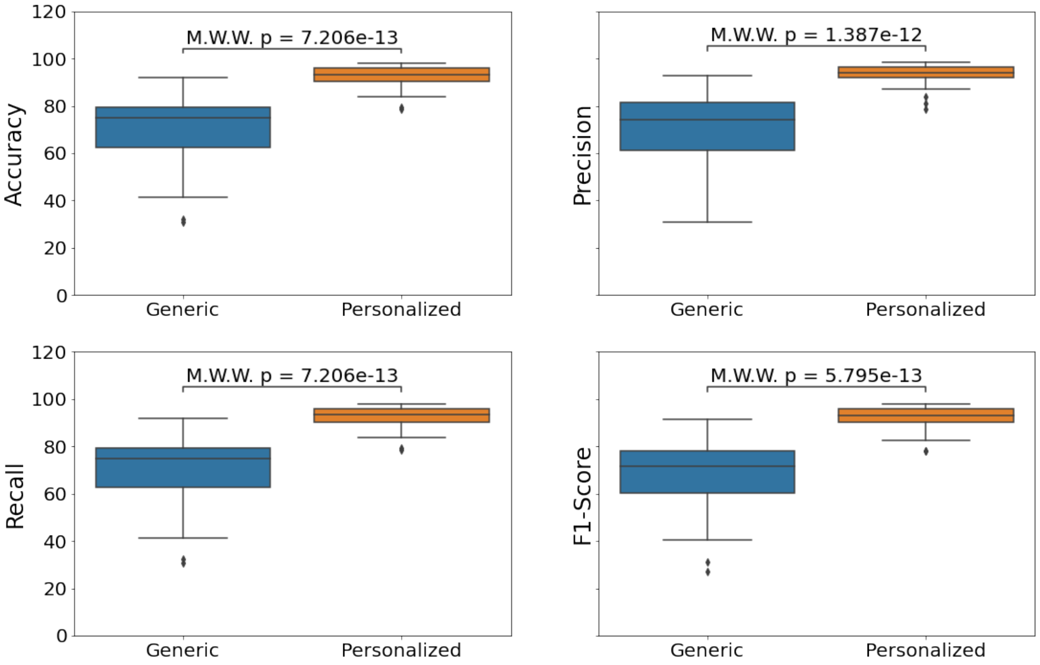


Figure 5.4: For the All Movement Evaluation, these box plots display the four performance metrics for both the generic and personalized Trans-CNN models.

Figures 5.2, 5.3, and 5.4 compared generic and personalized models for each of the three models. To determine which generic model performs the best, Figure 5.5 shows the performance metrics for all three generic models and the p values from a Kruskal-Wallis test. A Kruskal-Wallis test was completed to determine whether there was a significant difference between the three models. The Kruskal-Wallis test was chosen because the data from the generic Trans-CNN model are nonparametric. There was a significant difference in the group of three models due to a p value under 0.05. The p values can be seen in Figure 5.5. To determine where the difference was between the models, a Dunn test with Bonferroni adjusted p value was performed. Table 5.6, 5.7, 5.8, and 5.9 shows the p value from the Dunn test for all four performance metrics.

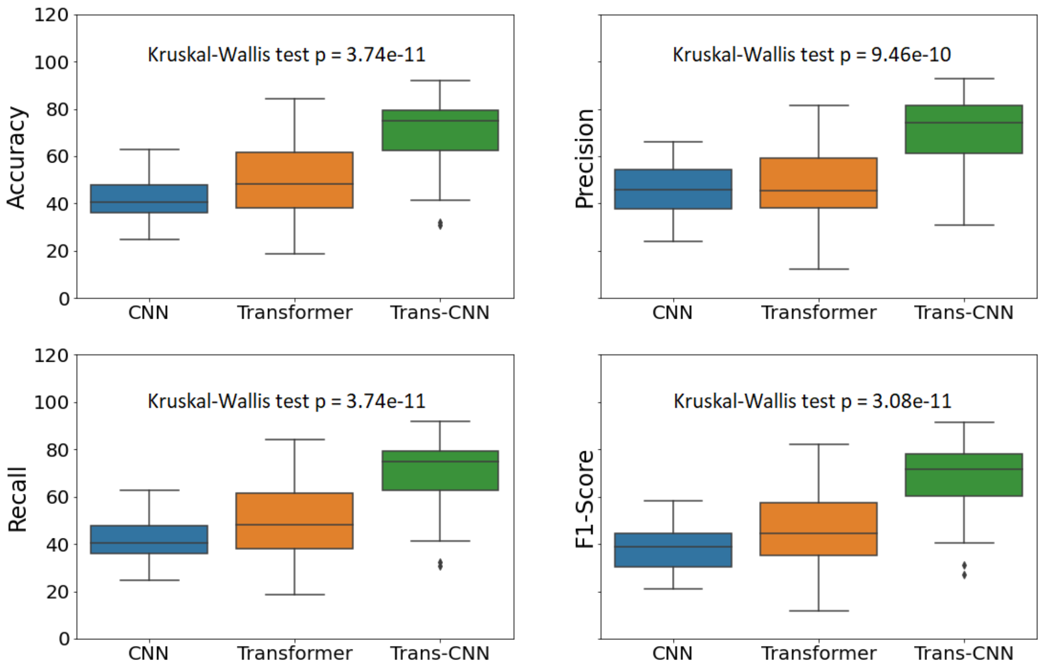


Figure 5.5: For the All Movements Evaluation, box plots were created for each of the four performance metrics for the three generic models.

Table 5.6: Dunn test with Bonferroni adjusted p value was performed for accuracy.

	CNN	**Transformer**	**Trans-CNN**
CNN	1.00	0.122	4.24e-11
Transformer	0.122	1.00	7.42e-06
Trans-CNN	4.24e-11	7.42e-06	1.00

Table 5.7: Dunn test with Bonferroni adjusted p value was performed for precision.

	CNN	**Transformer**	**Trans-CNN**
CNN	1.00	1.00	1.19e-08
Transformer	1.00	1.00	5.34e-07
Trans-CNN	1.19e-08	5.34e-07	1.00

Table 5.8: Dunn test with Bonferroni adjusted p value was performed for recall.

	CNN	**Transformer**	**Trans-CNN**
CNN	1.00	0.122	4.24e-11
Transformer	0.122	1.00	7.42e-06
Trans-CNN	4.24e-11	7.42e-06	1.00

Table 5.9: Dunn test with Bonferroni adjusted p value was performed for f1-score.

	CNN	**Transformer**	**Trans-CNN**
CNN	1.00	0.133	3.78e-11
Transformer	0.133	1.00	5.71e-06
Trans-CNN	3.78e-11	5.71e-06	1.00

The Trans-CNN model outperformed the CNN and transformer models in all four performance metrics, demonstrating a significant difference, as evidenced by a p value of less than 0.05 in the comparison between the Trans-CNN model and the other two models. Comparing the CNN model with the transformer model, a p value that was not under 0.05 for all four performance metrics was obtained; therefore, there was no significant difference between the CNN and transformer models.

Figure 5.6 shows a comparison of three personalized models. Based on the Kruskal-Wallis test, there was no significant difference between the three models, since the p value was greater than 0.05.

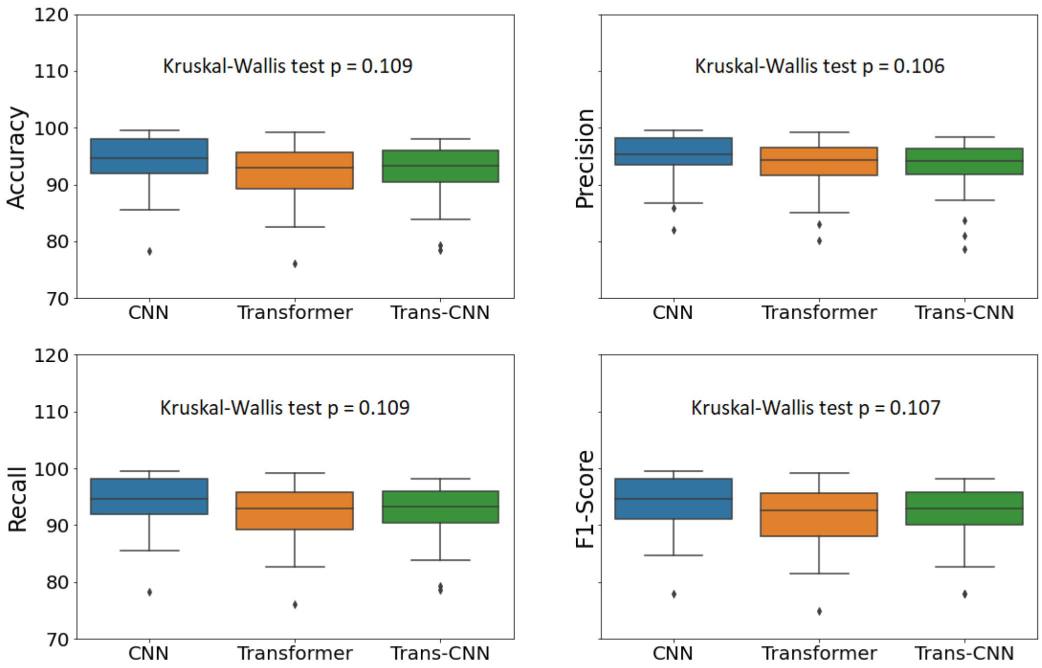


Figure 5.6: For the All Movements Evaluation, boxplots were created for each of the four performance metrics for the three personalized models.

The average of each performance metric using the same data used for the box plots was calculated, as well as the standard deviation, as seen in Table 5.10. The generic Trans-CNN model achieved a value of around 70% for all four performance metrics, which is the highest average value. There was a significant difference compared to the other two models. The other two models archived similar results to each other with no significant difference between them. For the personalized models, each of the three models obtained an average value of around 90% for each of the four metrics with no significant difference between them.

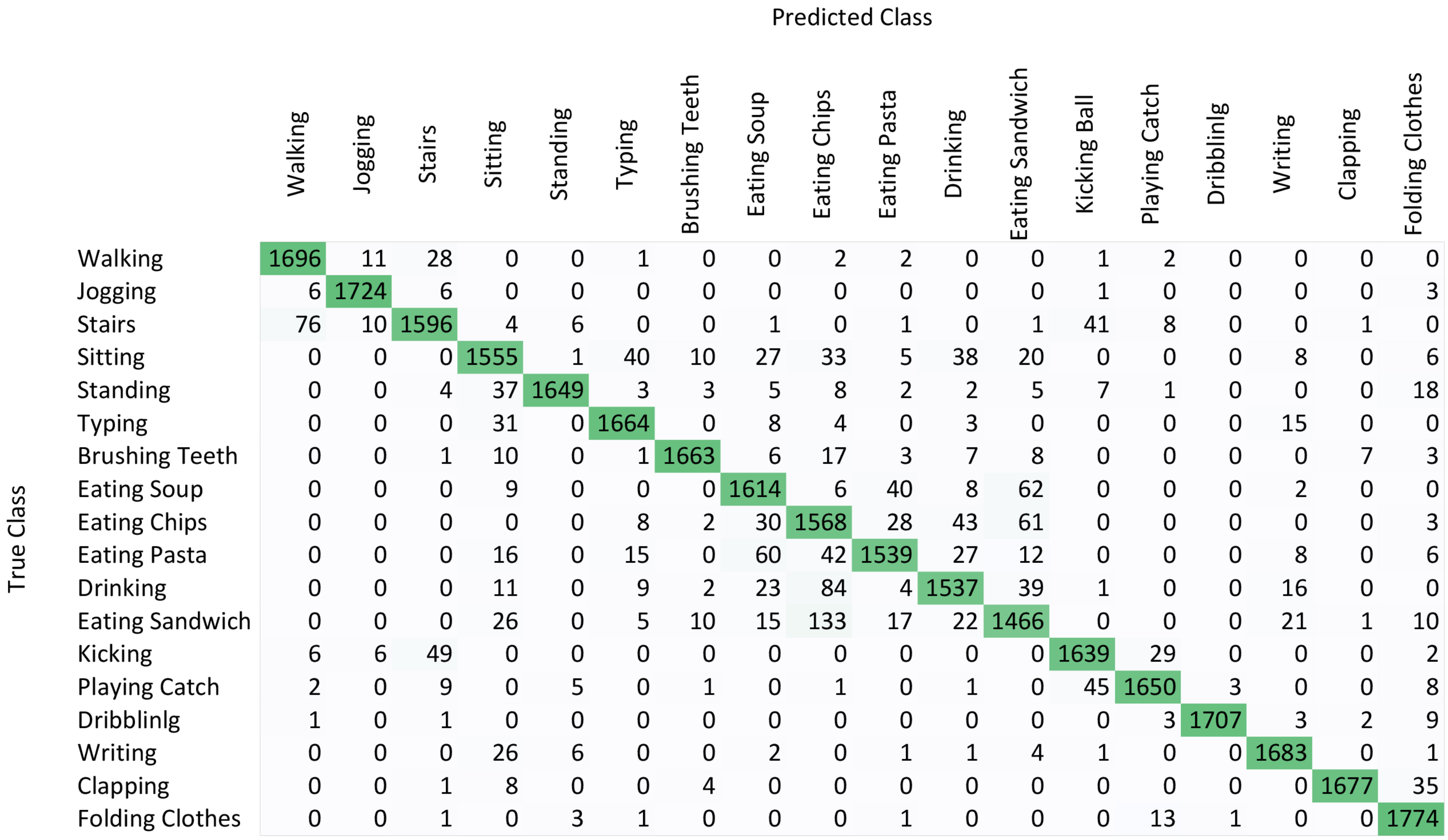
Table 5.10: In the All Movement Evaluation, for all three models, the averages including the standard deviations were calculated for the four performance metrics for both the generic and personalized models using the data from 39 participants.

	CNN	**Transformer**	**Trans-CNN**
Accuracy			
Generic Model	41.2±8.9%	49.2±15.1%	69.6±15.1%
Personalized Model	94.1±4.7%	92.1±5.4%	92.4±4.8%
Precision			
Generic Model	46.3±10.3%	48.4±16.4%	70.5±15.1%
Personalized Model	94.8±4.2%	93.3±4.6%	93.1±4.6%
Recall			
Generic Model	41.2±8.9%	49.2±15.1%	69.6±15.1%
Personalized Model	94.1±4.7%	92.1±5.4%	92.4±4.8%
F1-Score			
Generic Model	37.6±8.9%	45.5±15.7%	67.3±15.7%
Personalized Model	93.8±4.9%	91.6±5.7%	92.0±5.1%

A confusion matrix was created for all three models and for both the generic and personalized models to help determine which activities are confused with other activities. A confusion matrix is a table that has two dimensions (”true class” and ”predicted class”) and is used to define the performance of a classification ML model. The actual class is represented by each row of the matrix, while the predicted class is represented by each column. The confusion matrices that were created in this evaluation can be found in Figures 5.7, 5.8, 5.9, 5.10, 5.11, and 5.12.

True Class	Predicted Class																	
	Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
Walking	1175	11	475	1	1	0	0	0	4	0	0	2	22	26	0	1	1	24
Jogging	61	1293	276	0	0	0	2	0	0	0	0	0	59	9	32	0	0	8
Stairs	269	17	1250	19	4	0	3	0	2	2	9	5	102	34	5	0	0	24
Sitting	1	0	25	830	37	94	3	39	61	51	83	388	1	5	0	4	0	121
Standing	1	0	139	285	182	3	0	61	70	94	103	454	17	12	2	13	0	308
Typing	0	0	26	713	95	169	0	7	67	62	18	321	0	0	0	40	4	203
Brushing Teeth	1	0	153	152	84	10	660	3	24	23	78	185	22	0	0	1	9	321
Eating Soup	2	0	54	163	28	0	0	320	184	240	75	456	3	10	0	0	0	206
Eating Chips	0	0	35	168	44	6	0	30	506	42	137	554	7	11	0	3	0	200
Eating Pasta	0	0	37	236	38	22	0	56	242	252	44	410	1	6	0	3	1	377
Drinking	2	0	25	197	46	2	4	47	165	0	470	621	4	7	0	0	0	136
Eating Sandwich	0	0	41	286	89	0	1	20	232	32	133	614	25	15	0	0	0	238
Kicking	70	1	912	13	9	0	4	0	1	1	0	1	522	114	1	0	0	82
Playing Catch	12	6	121	5	1	0	1	0	0	0	1	0	81	1454	9	1	0	33
Dribblinlg	0	8	104	3	0	0	1	0	2	0	0	7	4	160	1373	2	3	59
Writing	4	0	23	356	47	86	0	77	71	147	104	347	0	3	0	206	0	254
Clapping	2	0	137	107	18	0	71	0	5	8	6	17	25	30	43	2	646	608
Folding Clothes	13	1	304	83	15	0	0	1	5	0	0	21	53	338	3	0	0	957

Figure 5.7: For the All Movement Evaluation, the confusion matrix displays the results from the generic CNN model.



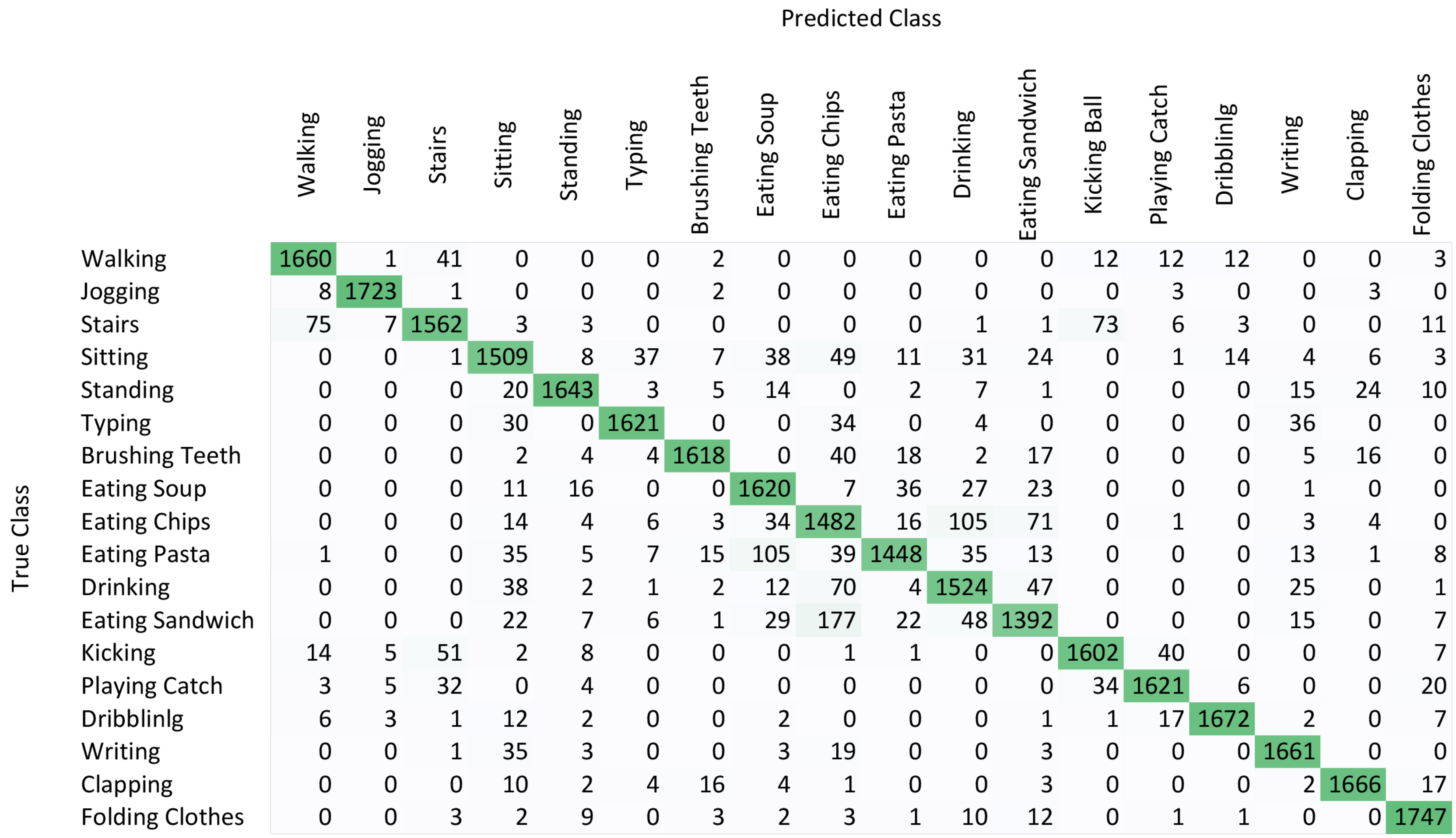
		Predicted Class																
		Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
True Class	Walking	1696	11	28	0	0	1	0	0	2	2	0	0	1	2	0	0	0	0
	Jogging	6	1724	6	0	0	0	0	0	0	0	0	0	1	0	0	0	0	3
	Stairs	76	10	1596	4	6	0	0	1	0	1	0	1	41	8	0	0	1	0
	Sitting	0	0	0	1555	1	40	10	27	33	5	38	20	0	0	0	8	0	6
	Standing	0	0	4	37	1649	3	3	5	8	2	2	5	7	1	0	0	0	18
	Typing	0	0	0	31	0	1664	0	8	4	0	3	0	0	0	0	15	0	0
	Brushing Teeth	0	0	1	10	0	1	1663	6	17	3	7	8	0	0	0	0	7	3
	Eating Soup	0	0	0	9	0	0	0	1614	6	40	8	62	0	0	0	2	0	0
	Eating Chips	0	0	0	0	0	8	2	30	1568	28	43	61	0	0	0	0	0	3
	Eating Pasta	0	0	0	16	0	15	0	60	42	1539	27	12	0	0	0	8	0	6
	Drinking	0	0	0	11	0	9	2	23	84	4	1537	39	1	0	0	16	0	0
	Eating Sandwich	0	0	0	26	0	5	10	15	133	17	22	1466	0	0	0	21	1	10
	Kicking	6	6	49	0	0	0	0	0	0	0	0	0	1639	29	0	0	0	2
	Playing Catch	2	0	9	0	5	0	1	0	1	0	1	0	45	1650	3	0	0	8
	Dribblinlg	1	0	1	0	0	0	0	0	0	0	0	0	0	3	1707	3	2	9
	Writing	0	0	0	26	6	0	0	2	0	1	1	4	1	0	0	1683	0	1
	Clapping	0	0	1	8	0	0	4	0	0	0	0	0	0	0	0	0	1677	35
	Folding Clothes	0	0	1	0	3	1	0	0	0	1	0	0	0	13	1	0	0	1774

Figure 5.8: For the All Movement Evaluation, the confusion matrix displays the results from the personalized CNN model.

Predicted Class

True Class	Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
Walking	1220	69	291	0	1	0	4	0	0	0	0	0	117	12	12	3	7	7
Jogging	30	1669	1	0	0	0	0	1	0	0	0	1	4	6	27	0	0	1
Stairs	189	56	1165	7	2	0	14	0	2	0	0	1	236	26	25	0	6	16
Sitting	5	0	37	753	122	161	38	43	98	153	57	153	0	10	2	52	35	24
Standing	3	0	61	50	883	74	208	82	101	39	93	63	19	9	0	10	42	7
Typing	0	42	46	269	4	520	142	0	221	131	5	71	0	0	0	169	103	2
Brushing Teeth	29	0	37	275	146	51	934	7	83	32	12	20	5	0	0	5	84	6
Eating Soup	0	0	7	118	180	4	103	612	119	325	108	66	0	1	0	65	0	33
Eating Chips	3	0	41	167	85	32	63	105	541	92	177	304	1	0	8	29	1	94
Eating Pasta	2	3	26	307	45	14	66	212	174	411	90	136	13	21	7	78	0	120
Drinking	12	0	10	309	141	19	93	67	217	29	434	332	2	0	0	39	3	19
Eating Sandwich	0	0	21	318	162	3	116	58	281	178	176	349	0	0	0	5	1	58
Kicking	62	49	512	4	20	0	38	1	0	30	6	3	850	109	8	0	1	38
Playing Catch	48	56	163	31	6	0	5	7	2	0	1	0	115	1165	70	2	9	45
Dribblinlg	55	179	78	23	1	3	0	0	0	1	0	2	40	46	1163	4	73	58
Writing	31	1	3	173	119	155	46	118	16	64	58	58	1	0	13	845	11	13
Clapping	113	16	67	21	5	3	180	0	1	0	1	0	0	37	56	41	1097	87
Folding Clothes	19	1	317	82	20	7	18	37	67	167	14	12	81	101	49	16	10	776

Figure 5.9: For the All Movement Evaluation, the confusion matrix displays the results from the generic transformer model.



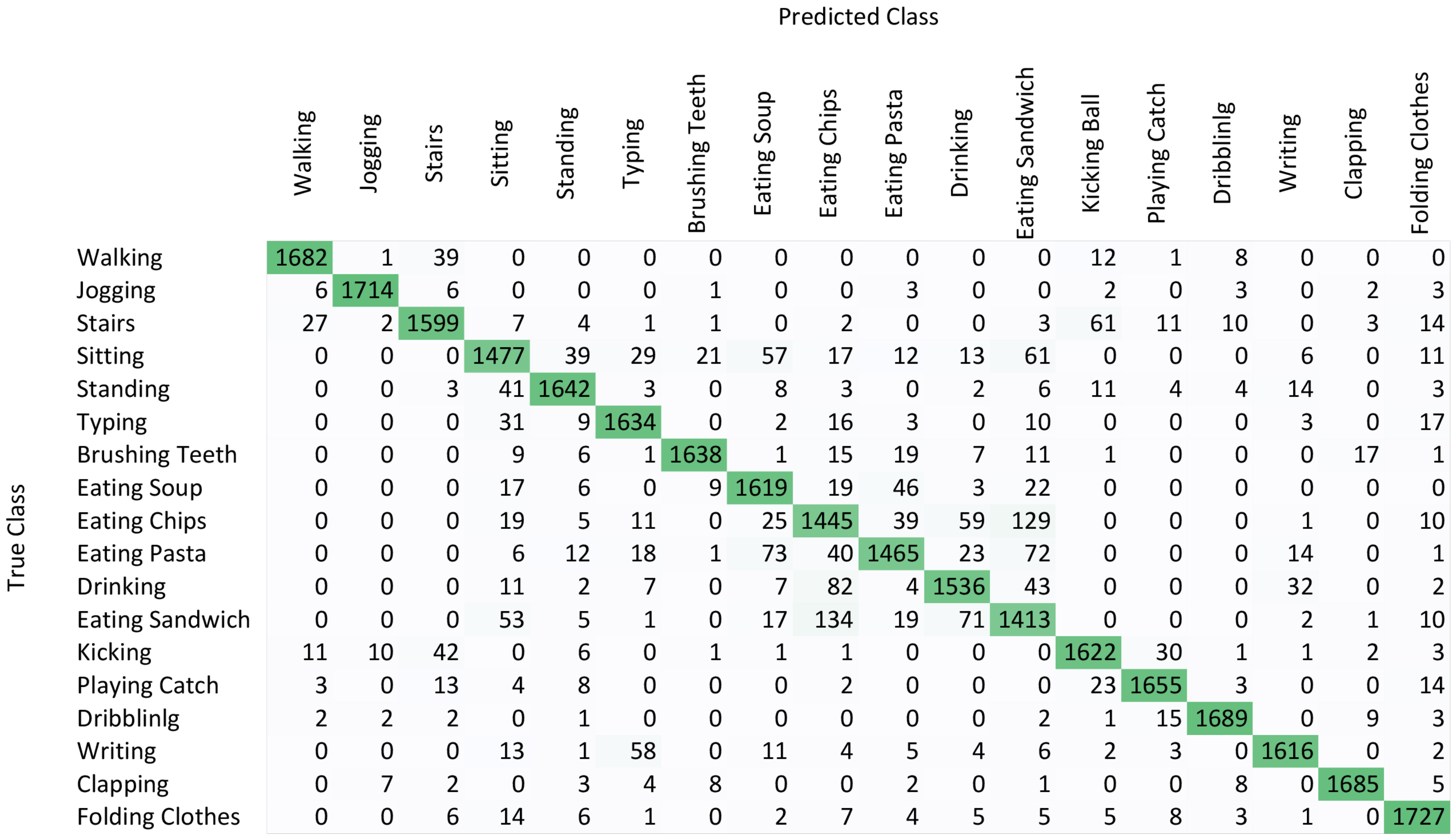
Predicted Class

True Class	Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
Walking	1660	1	41	0	0	0	2	0	0	0	0	0	12	12	12	0	0	3
Jogging	8	1723	1	0	0	0	2	0	0	0	0	0	0	3	0	0	3	0
Stairs	75	7	1562	3	3	0	0	0	0	0	1	1	73	6	3	0	0	11
Sitting	0	0	1	1509	8	37	7	38	49	11	31	24	0	1	14	4	6	3
Standing	0	0	0	20	1643	3	5	14	0	2	7	1	0	0	0	15	24	10
Typing	0	0	0	30	0	1621	0	0	34	0	4	0	0	0	0	36	0	0
Brushing Teeth	0	0	0	2	4	4	1618	0	40	18	2	17	0	0	0	5	16	0
Eating Soup	0	0	0	11	16	0	0	1620	7	36	27	23	0	0	0	1	0	0
Eating Chips	0	0	0	14	4	6	3	34	1482	16	105	71	0	1	0	3	4	0
Eating Pasta	1	0	0	35	5	7	15	105	39	1448	35	13	0	0	0	13	1	8
Drinking	0	0	0	38	2	1	2	12	70	4	1524	47	0	0	0	25	0	1
Eating Sandwich	0	0	0	22	7	6	1	29	177	22	48	1392	0	0	0	15	0	7
Kicking	14	5	51	2	8	0	0	0	1	1	0	0	1602	40	0	0	0	7
Playing Catch	3	5	32	0	4	0	0	0	0	0	0	0	34	1621	6	0	0	20
Dribblinlg	6	3	1	12	2	0	0	2	0	0	0	1	1	17	1672	2	0	7
Writing	0	0	1	35	3	0	0	3	19	0	0	3	0	0	0	1661	0	0
Clapping	0	0	0	10	2	4	16	4	1	0	0	3	0	0	0	2	1666	17
Folding Clothes	0	0	3	2	9	0	3	2	3	1	10	12	0	1	1	0	0	1747

Figure 5.10: For the All Movement Evaluation, the confusion matrix displays the results from the personalized transformer model.

		Predicted Class																	
		Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
True Class	Walking	1479	18	202	0	0	0	0	0	0	0	0	1	19	9	3	0	0	12
	Jogging	3	1728	1	0	0	0	0	1	0	0	0	0	4	0	2	1	0	0
	Stairs	70	24	1399	4	12	0	11	0	1	1	0	1	154	15	41	0	5	7
	Sitting	3	0	20	724	140	157	2	60	48	145	154	239	0	1	2	15	5	28
	Standing	0	0	3	61	1269	40	29	19	20	20	63	148	29	15	14	3	2	9
	Typing	0	0	3	229	47	1139	0	4	72	39	0	15	0	0	0	142	18	17
	Brushing Teeth	2	0	8	86	84	0	1333	12	9	55	13	25	8	1	2	21	32	35
	Eating Soup	1	0	5	50	114	0	0	1095	166	108	27	125	1	20	1	6	0	22
	Eating Chips	1	0	5	136	99	10	2	76	933	85	58	288	4	3	1	15	0	27
	Eating Pasta	0	0	4	98	113	54	5	118	141	945	41	125	1	0	0	21	4	55
	Drinking	0	0	1	130	139	9	0	39	123	43	915	292	1	0	0	28	0	6
	Eating Sandwich	1	0	20	217	133	6	10	109	466	112	186	405	2	9	0	20	0	30
	Kicking	28	22	121	4	11	0	3	2	3	0	0	6	1424	32	9	3	8	55
	Playing Catch	5	5	38	9	8	1	0	2	5	1	1	6	69	1507	42	3	4	19
	Dribblinlg	17	72	21	0	2	0	0	1	1	1	0	5	13	65	1484	4	13	27
	Writing	0	0	1	98	92	163	13	41	17	15	54	34	2	3	2	1181	1	8
	Clapping	19	13	28	40	63	4	88	0	1	4	9	0	0	10	31	12	1340	63
	Folding Clothes	16	2	32	39	17	10	4	5	18	30	17	26	42	60	31	3	1	1441

Figure 5.11: For the All Movement Evaluation, the confusion matrix displays the results from the generic Trans-CNN model.



	Predicted Class																	
True Class	Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
Walking	1682	1	39	0	0	0	0	0	0	0	0	0	12	1	8	0	0	0
Jogging	6	1714	6	0	0	0	1	0	0	3	0	0	2	0	3	0	2	3
Stairs	27	2	1599	7	4	1	1	0	2	0	0	3	61	11	10	0	3	14
Sitting	0	0	0	1477	39	29	21	57	17	12	13	61	0	0	0	6	0	11
Standing	0	0	3	41	1642	3	0	8	3	0	2	6	11	4	4	14	0	3
Typing	0	0	0	31	9	1634	0	2	16	3	0	10	0	0	0	3	0	17
Brushing Teeth	0	0	0	9	6	1	1638	1	15	19	7	11	1	0	0	0	17	1
Eating Soup	0	0	0	17	6	0	9	1619	19	46	3	22	0	0	0	0	0	0
Eating Chips	0	0	0	19	5	11	0	25	1445	39	59	129	0	0	0	1	0	10
Eating Pasta	0	0	0	6	12	18	1	73	40	1465	23	72	0	0	0	14	0	1
Drinking	0	0	0	11	2	7	0	7	82	4	1536	43	0	0	0	32	0	2
Eating Sandwich	0	0	0	53	5	1	0	17	134	19	71	1413	0	0	0	2	1	10
Kicking	11	10	42	0	6	0	1	1	1	0	0	0	1622	30	1	1	2	3
Playing Catch	3	0	13	4	8	0	0	0	2	0	0	0	23	1655	3	0	0	14
Dribblinlg	2	2	2	0	1	0	0	0	0	0	0	2	1	15	1689	0	9	3
Writing	0	0	0	13	1	58	0	11	4	5	4	6	2	3	0	1616	0	2
Clapping	0	7	2	0	3	4	8	0	0	2	0	1	0	0	8	0	1685	5
Folding Clothes	0	0	6	14	6	1	0	2	7	4	5	5	5	8	3	1	0	1727

Figure 5.12: For the All Movement Evaluation, the confusion matrix displays the results from the personalized Trans-CNN model.

In Figure 5.7, the generic CNN predicted eating a sandwich and folding clothes more than other upper body movements. In other words, the model was confusing other upper body movements with eating a sandwich and folding clothes. Upper body movements also got confused with sitting. Lower body movements also got confused with other lower body movements, but there are fewer incorrect labels. After the CNN model was personalized, the model performed much better with fewer incorrect labels as seen in Figure 5.8. The same trends can be seen with the generic and personalized transformer and Trans-CNN models as seen in Figures 5.9, 5.10, 5.11, and 5.12. Therefore, across the three models, the generic model confused upper body movements more than lower body movements.

To examine model variability among participants, the performance data for accuracy from 10 participants using the generic and personalized model for each of the three types of models were plotted, as seen in Figure 5.13. For all three types of models, the results from the personalized model had very little variation (CNN standard deviation: 1.96, Transformer standard deviation: 2.31, and Trans-CNN standard deviation: 3.10) as compared to the generic model (CNN standard deviation: 10.4, Transformer standard deviation: 9.43, and Trans-CNN standard deviation: 12.5). Overall, personalized models, regardless of the deep learning architecture, demonstrated much higher consistency in detecting human activities than the generic models.

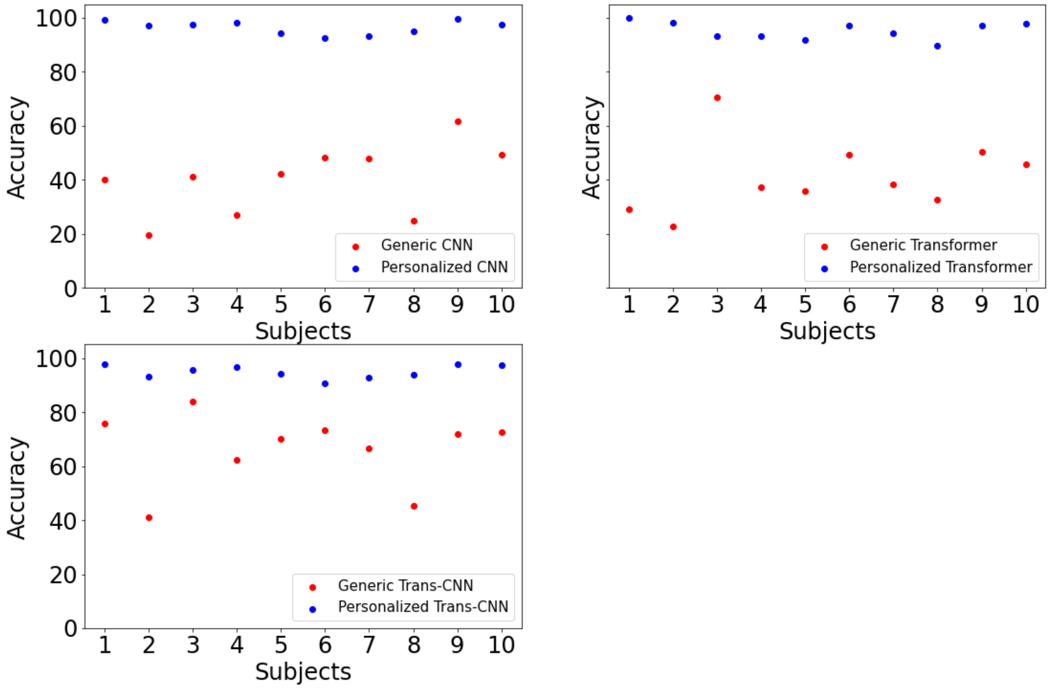


Figure 5.13: For each of the three types of models, the accuracy for 10 participants for both the generic and personalized model was plotted.

5.4.3 Upper Body Movement Evaluation

The upper body movement evaluation examines the performance of the three models on upper body movements, specifically, only the movements listed in Table 5.11. This evaluation used $D2$ but removed the lower body movements. The models were evaluated in the same manner as the All Movement Evaluation.

Table 5.11: The dataset used in this evaluation included 12 activities that involve only upper body movements.

Activities in the Dataset
Typing
Brushing Teeth
Eating Soup
Eating Chips
Eating Pasta
Drinking from Cup
Eating Sandwich
Playing Catch w/Tennis Ball
Dribbling (Basketball)
Writing
Clapping
Folding Clothes

Again, it was important to determine if the data were normally distributed to help select the best model. In order to determine whether the results of the 39 participants were normally distributed, the Shapiro–Wilk test was used. A breakdown of normal versus non-normal distributions can be found in Table 5.19.

Table 5.12: Shapiro–Wilk test was used to determine if the results for the Upper Body Movement Evaluation were normally distributed or not.

Normal Distributed Data	**Non-Normal Distributed Data**
Generic CNN Accuracy	Generic Trans-CNN Accuracy
Generic CNN Precision	Generic Trans-CNN Precision
Generic CNN Recall	Generic Trans-CNN Recall
Generic Transformer Accuracy	Generic Trans-CNN F1-Score
Generic Transformer Precision	Personalized CNN's Accuracy
Generic Transformer Recall	Personalized CNN Precision
Generic Transformer F1-Score	Personalized CNN's Recall
	Personalized CNN F1-Score
	Personalized Trans-CNN Accuracy
	Personalized Trans-CNN Precision
	Personalized Trans-CNN Recall
	Personalized Trans-CNN F1-Score
	Generic CNN F1-Score

Figure 5.14 shows the results from 39 participants plotted using the same type of box plots described in Subsection 5.4.2 for both the generic and personalized CNN models.

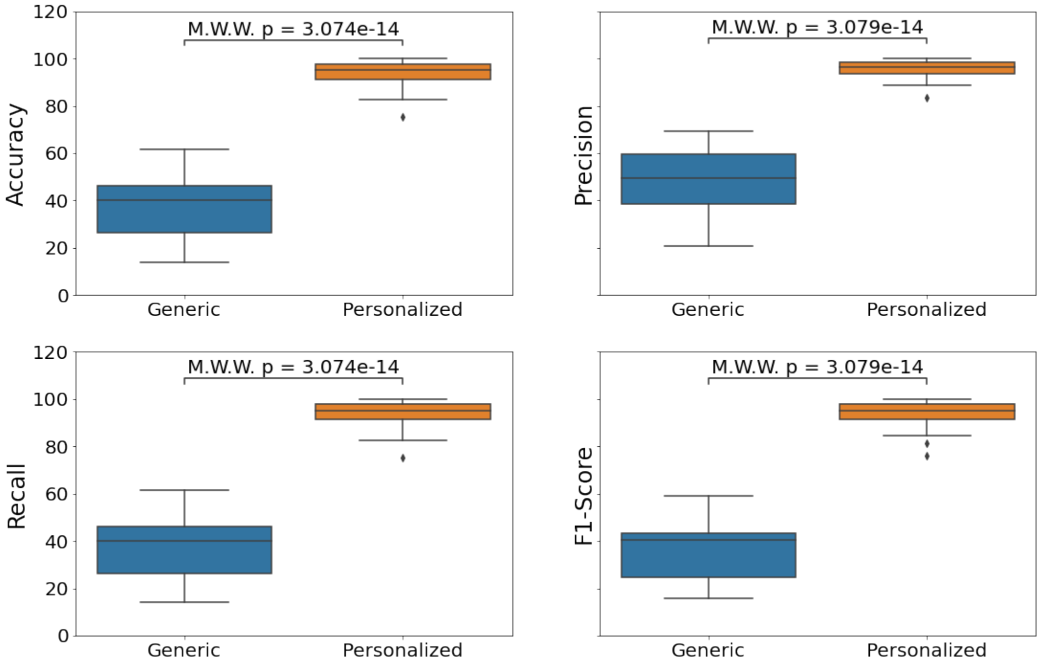


Figure 5.14: For the Upper Movement Evaluation, these boxplots display the four performance metrics for both the generic and personalized CNN models.

Figure 5.14 shows that the Upper Movement Evaluation box plots for CNN have the same trends as the All Movement Evaluation. The performance of the generic model was in the 30s, while the performance of the personalized model was in the 90s. The two models were also compared using the Mann-Whitney test. Based on the p values for each performance metric, the generic model and personalized model had statistically significant differences. Similarly, box plots were created for the transformer and Trans-CNN models seen in Figures 5.15 and 5.16. The same trends can be seen in both figures as seen in Figure 5.14. The generic transformer model performance was approximately 50% and the personalized model had a performance of approximately 90%; in contrast, performance of the generic model was around 60%, while the performance of the personalized model was around 90%. Since the data for both personalized models are nonparametric, the Mann-Whitney test was also chosen. There were statistically significant differences between the generic and personalized models for both the transformer and

Trans-CNN models based on the p values for all four performance metrics.

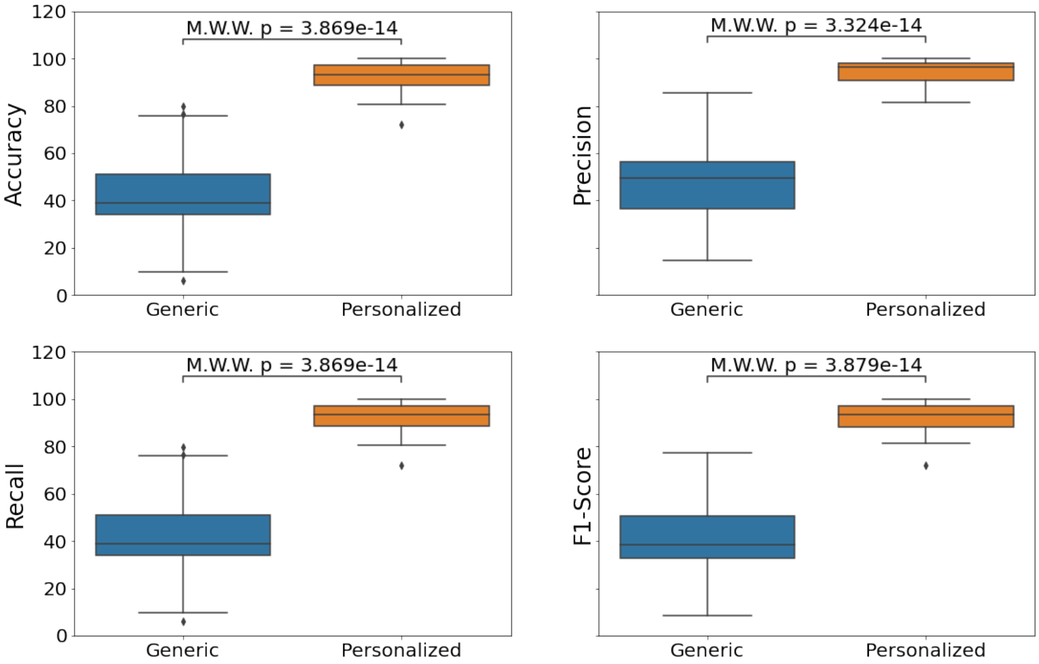


Figure 5.15: For the Upper Movement Evaluation, these boxplots display the four performance metrics for both the generic and personalized transformer models.

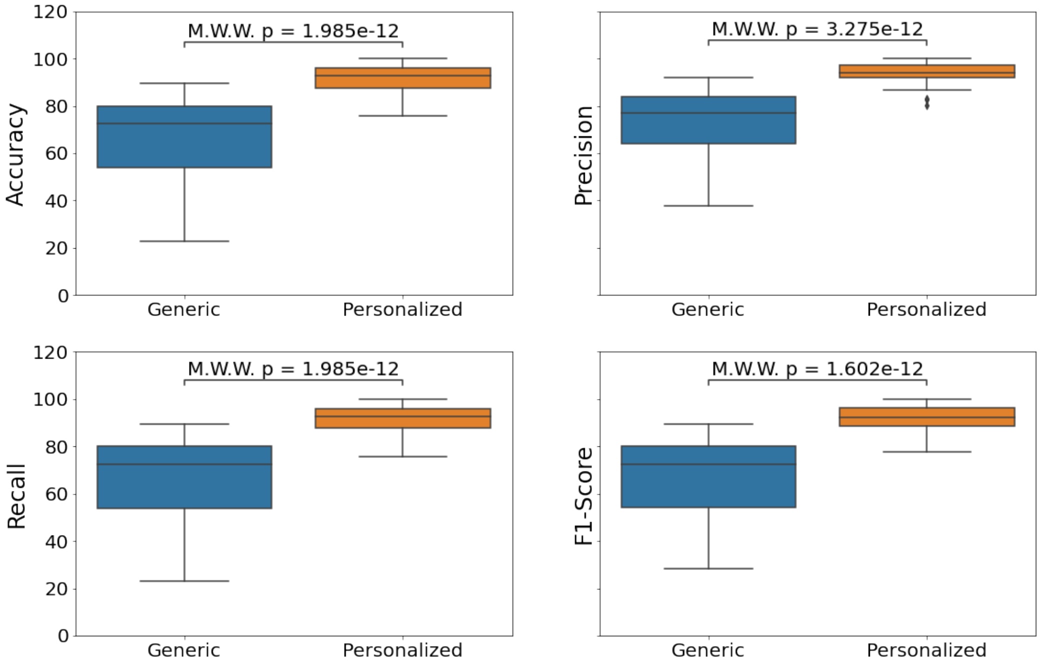


Figure 5.16: For the Upper Movement Evaluation, these boxplots display the four performance metrics for both the generic and personalized Trans-CNN models.

While Figures 5.14, 5.15, and 5.16 compared generic and personalized models for each of the three models, Figure 5.17 compares the three generic models for each of the four performance metrics to determine the best generic model.

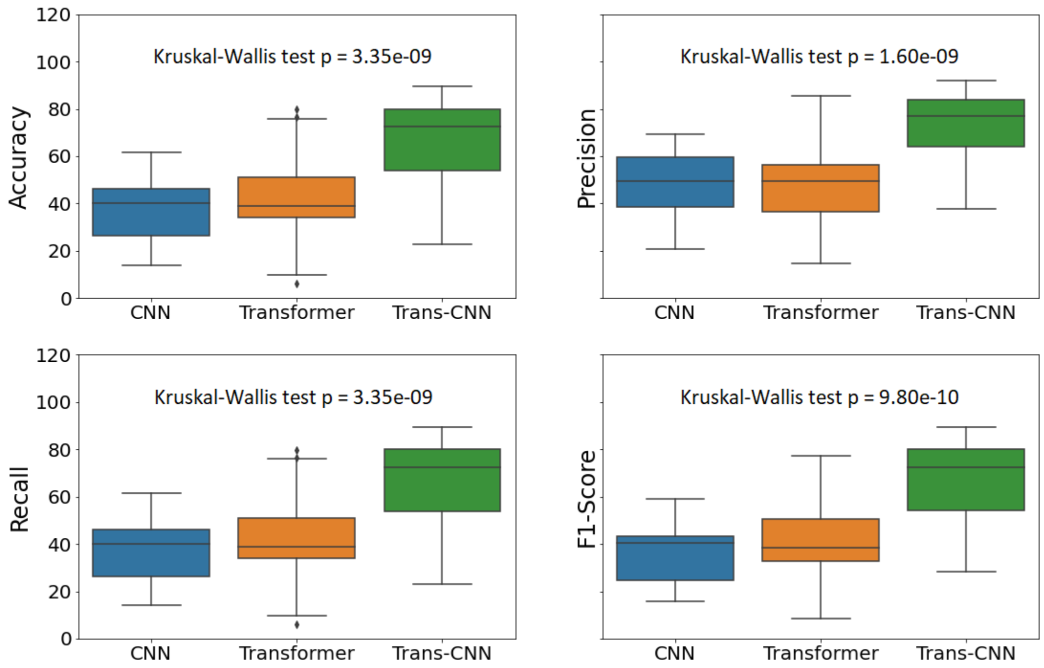


Figure 5.17: For the Upper Body Movements Evaluation, boxplots were created for each of the four performance metrics for the three generic models.

A Kruskal-Wallis test was completed showing that there is a significant difference in the group of three models due to a p value under 0.05. Next, a Dunn test with Bonferroni adjusted p value was performed to find where the difference was in the group. Tables 5.13, 5.14, 5.15, and 5.16 show the p values.

Table 5.13: Dunn test with Bonferroni adjusted p value was performed for accuracy.

	CNN	**Transformer**	**Trans-CNN**
CNN	1.00	0.771	1.18e-08
Transformer	0.771	1.00	5.99e-06
Trans-CNN	1.18e-08	5.99e-06	1.00

Table 5.14: Dunn test with Bonferroni adjusted p value was performed for precision.

	CNN	**Transformer**	**Trans-CNN**
CNN	1.00	1.00	5.68e-08
Transformer	1.00	1.00	2.06e-07
Trans-CNN	5.68e-08	2.06e-07	1.00

Table 5.15: Dunn test with Bonferroni adjusted p value was performed for recall.

	CNN	**Transformer**	**Trans-CNN**
CNN	1.00	0.771	1.18e-08
Transformer	0.771	1.00	5.99e-06
Trans-CNN	1.18e-08	5.99e-06	1.00

Table 5.16: Dunn test with Bonferroni adjusted p value was performed for f1-score.

	CNN	**Transformer**	**Trans-CNN**
CNN	1.00	0.639	3.27e-09
Transformer	0.639	1.00	3.70e-06
Trans-CNN	3.27e-09	3.70e-06	1.00

The Trans-CNN model surpassed the CNN and transformer models in all four performance metrics, indicating a significant difference since all of the p values were less than 0.05 in the comparison between the Trans-CNN model and the other two models. All four performance metrics of the CNN and transformer models were compared. Since the p values were not below 0.05, it can be concluded that there is no significant difference between the two models.

Figure 5.17 was repeated for the personalized models as seen in Figure 5.18. Three ML models are compared in Figure 5.18 as a personalized model. According to the Kruskal-Wallis test, the p value was greater than 0.05 for all three models; therefore, there was no significant difference between the three models.

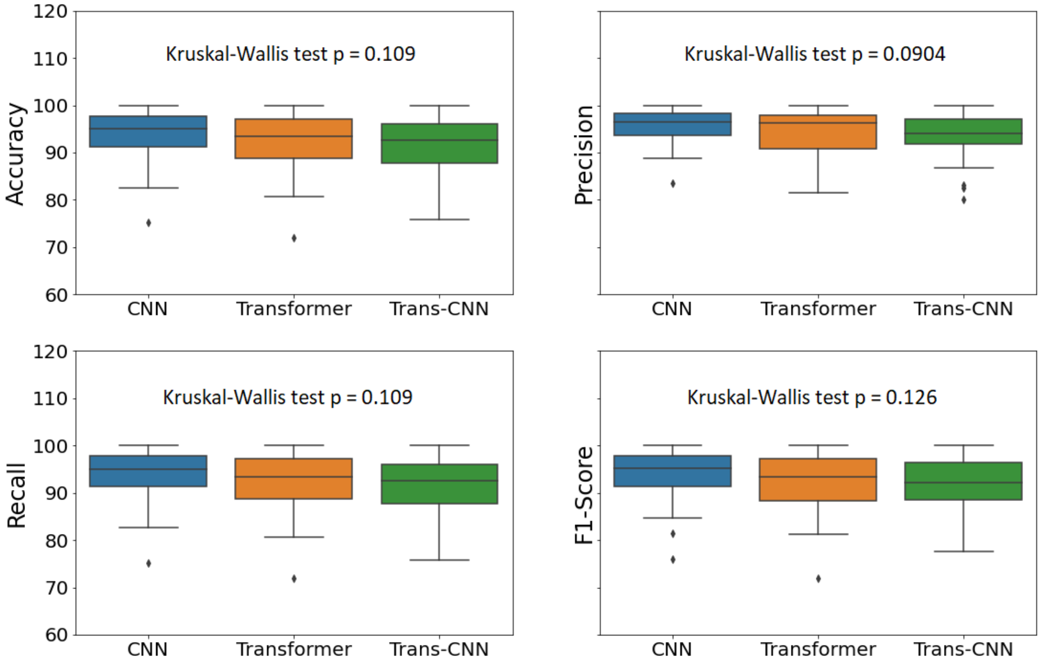


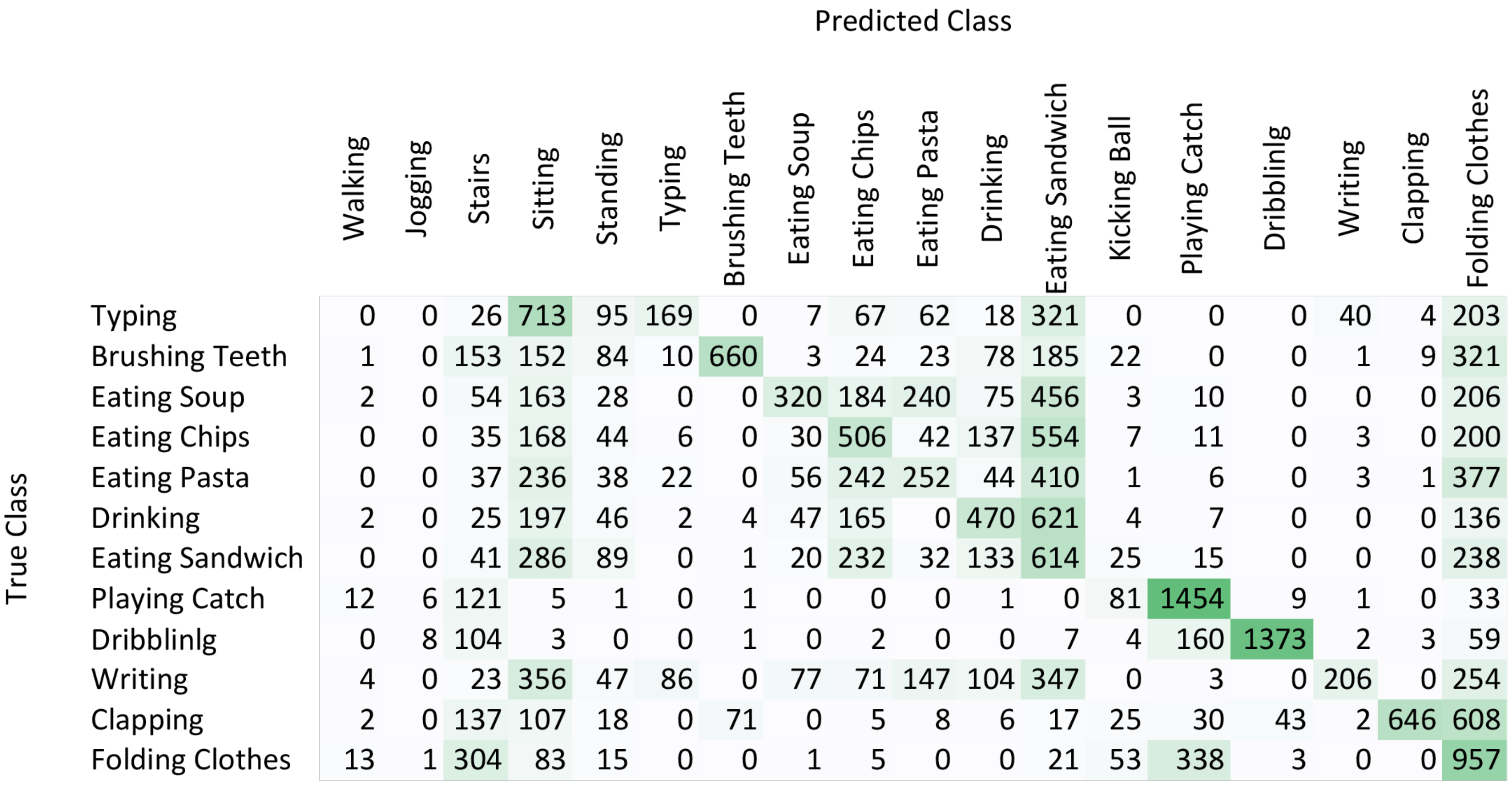
Figure 5.18: For the Upper Body Movements Evaluation, boxplots were created for each of the four performance metrics for the three personalized models.

Based on the same data used for the box plots, Table 5.17 shows the average and standard deviation of each performance metric. Among the three models, the Trans-CNN model as a generic model achieved the highest average value of around 65% for all four performance metrics. There was a significant difference with the Trans-CNN model compared to the other two models. There were no significant differences between the other two models. There was no significant difference between the three personalized models for any of the four performance metrics. They all obtained an average value of around 90%.

Table 5.17: In the Upper Body Movement Evaluation, for all three models, the averages including the standard deviations were calculated for the four performance metrics for both the generic and personalized models using the data from 39 participants.

	CNN	**Transformer**	**Trans-CNN**
Accuracy			
Generic Model	36.6±11.7%	42.5±17.9%	65.9±18.4%
Personalized Model	93.7±5.5%	92.0±6.6%	91.3±6.0%
Precision			
Generic Model	47.5±13.2%	48.7±18.1%	72.4±15.1%
Personalized Model	95.5±3.8%	94.5±4.6%	93.5±4.6%
Recall			
Generic Model	36.6±11.7%	42.5±17.9%	65.9±18.4%
Personalized Model	93.7±5.5%	92.0±6.6%	91.3±6.0%
F1-Score			
Generic Model	35.6±11.5%	41.7±17.3%	65.9±17.7%
Personalized Model	93.7±5.6%	92.2±6.6%	91.4±6.0%

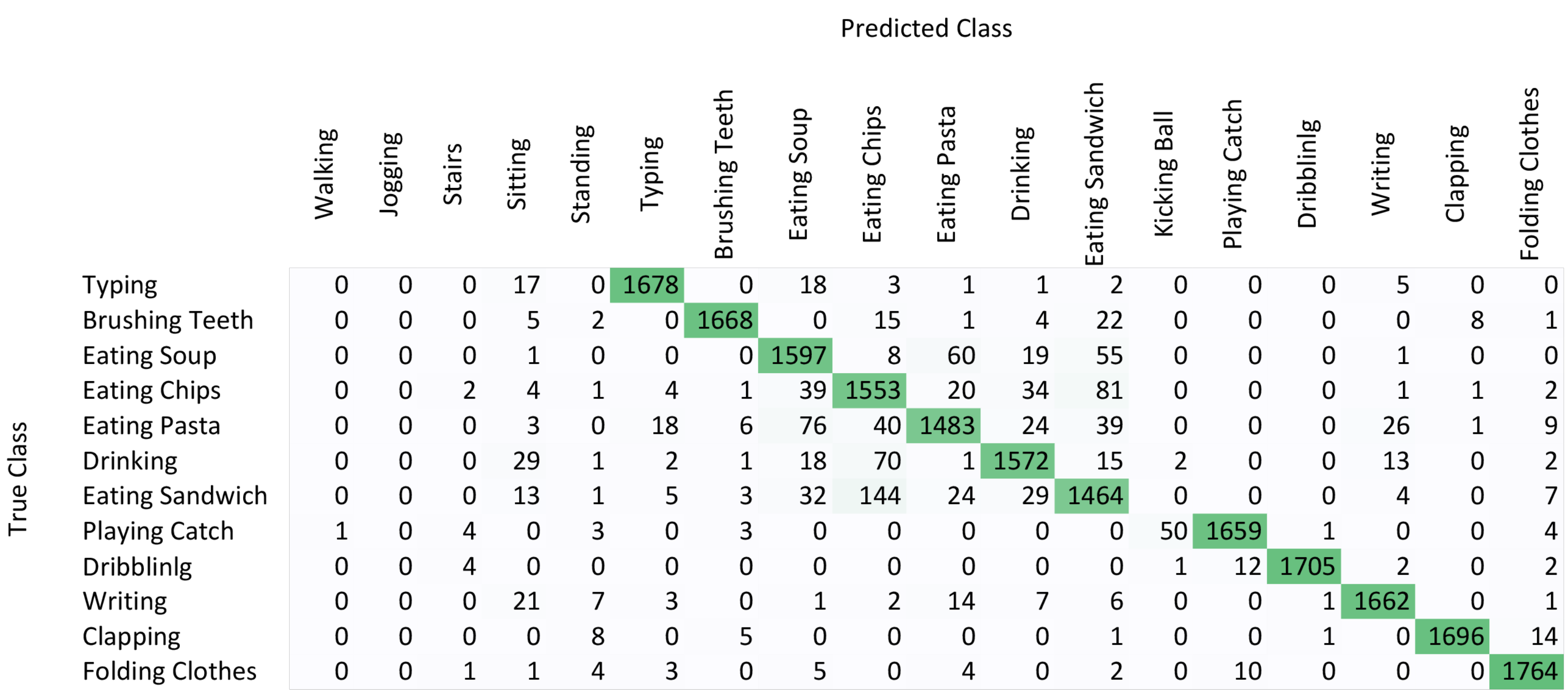
In order to identify which activities were confused with other activities, confusion matrices were created for the generic and personalized models of all three models. Each row of the matrix represents the actual class, and each column represents the predicted class. The confusion matrices in this evaluation can be found in Figures 5.19, 5.20, 5.21, 5.22, 5.23, and 5.24.



Predicted Class

True Class	Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
Typing	0	0	26	713	95	169	0	7	67	62	18	321	0	0	0	40	4	203
Brushing Teeth	1	0	153	152	84	10	660	3	24	23	78	185	22	0	0	1	9	321
Eating Soup	2	0	54	163	28	0	0	320	184	240	75	456	3	10	0	0	0	206
Eating Chips	0	0	35	168	44	6	0	30	506	42	137	554	7	11	0	3	0	200
Eating Pasta	0	0	37	236	38	22	0	56	242	252	44	410	1	6	0	3	1	377
Drinking	2	0	25	197	46	2	4	47	165	0	470	621	4	7	0	0	0	136
Eating Sandwich	0	0	41	286	89	0	1	20	232	32	133	614	25	15	0	0	0	238
Playing Catch	12	6	121	5	1	0	1	0	0	0	1	0	81	1454	9	1	0	33
Dribblinlg	0	8	104	3	0	0	1	0	2	0	0	7	4	160	1373	2	3	59
Writing	4	0	23	356	47	86	0	77	71	147	104	347	0	3	0	206	0	254
Clapping	2	0	137	107	18	0	71	0	5	8	6	17	25	30	43	2	646	608
Folding Clothes	13	1	304	83	15	0	0	1	5	0	0	21	53	338	3	0	0	957

Figure 5.19: For the Upper Body Movement Evaluation, the confusion matrix displays the results from the generic CNN model.



Predicted Class

True Class

	Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
Typing	0	0	0	17	0	1678	0	18	3	1	1	2	0	0	0	5	0	0
Brushing Teeth	0	0	0	5	2	0	1668	0	15	1	4	22	0	0	0	0	8	1
Eating Soup	0	0	0	1	0	0	0	1597	8	60	19	55	0	0	0	1	0	0
Eating Chips	0	0	2	4	1	4	1	39	1553	20	34	81	0	0	0	1	1	2
Eating Pasta	0	0	0	3	0	18	6	76	40	1483	24	39	0	0	0	26	1	9
Drinking	0	0	0	29	1	2	1	18	70	1	1572	15	2	0	0	13	0	2
Eating Sandwich	0	0	0	13	1	5	3	32	144	24	29	1464	0	0	0	4	0	7
Playing Catch	1	0	4	0	3	0	3	0	0	0	0	0	50	1659	1	0	0	4
Dribblinlg	0	0	4	0	0	0	0	0	0	0	0	0	1	12	1705	2	0	2
Writing	0	0	0	21	7	3	0	1	2	14	7	6	0	0	1	1662	0	1
Clapping	0	0	0	0	8	0	5	0	0	0	0	1	0	0	1	0	1696	14
Folding Clothes	0	0	1	1	4	3	0	5	0	4	0	2	0	10	0	0	0	1764

Figure 5.20: For the Upper Body Movement Evaluation, the confusion matrix displays the results from the personalized CNN model.

Predicted Class

True Class	Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
Typing	0	42	46	269	4	520	142	0	221	131	5	71	0	0	0	169	103	2
Brushing Teeth	29	0	37	275	146	51	934	7	83	32	12	20	5	0	0	5	84	6
Eating Soup	0	0	7	118	180	4	103	612	119	325	108	66	0	1	0	65	0	33
Eating Chips	3	0	41	167	85	32	63	105	541	92	177	304	1	0	8	29	1	94
Eating Pasta	2	3	26	307	45	14	66	212	174	411	90	136	13	21	7	78	0	120
Drinking	12	0	10	309	141	19	93	67	217	29	434	332	2	0	0	39	3	19
Eating Sandwich	0	0	21	318	162	3	116	58	281	178	176	349	0	0	0	5	1	58
Playing Catch	48	56	163	31	6	0	5	7	2	0	1	0	115	1165	70	2	9	45
Dribblinlg	55	179	78	23	1	3	0	0	0	1	0	2	40	46	1163	4	73	58
Writing	31	1	3	173	119	155	46	118	16	64	58	58	1	0	13	845	11	13
Clapping	113	16	67	21	5	3	180	0	1	0	1	0	0	37	56	41	1097	87
Folding Clothes	19	1	317	82	20	7	18	37	67	167	14	12	81	101	49	16	10	776

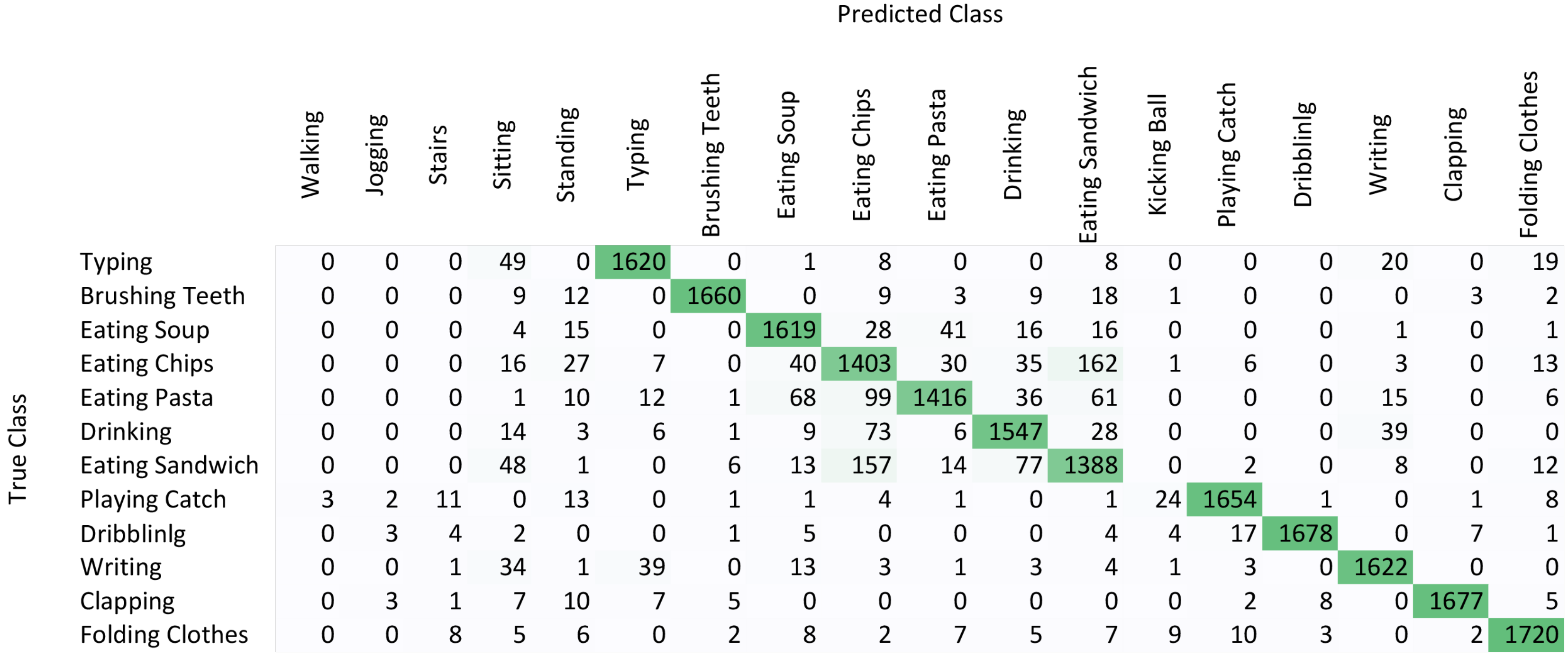
Figure 5.21: For the Upper Body Movement Evaluation, the confusion matrix displays the results from the generic transformer model.

True Class	Predicted Class																	
	Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
Typing	0	0	0	40	0	1671	3	0	0	0	4	0	0	0	0	7	0	0
Brushing Teeth	0	0	0	1	4	1	1632	0	11	4	2	43	0	3	0	0	25	0
Eating Soup	0	0	0	0	13	0	0	1604	7	58	30	29	0	0	0	0	0	0
Eating Chips	0	0	2	14	2	7	0	25	1534	20	84	53	0	0	0	2	0	0
Eating Pasta	2	0	2	15	36	14	24	46	53	1457	25	32	0	0	0	3	10	6
Drinking	0	0	0	28	1	2	0	11	118	6	1486	42	0	0	0	32	0	0
Eating Sandwich	0	0	0	20	0	17	6	9	176	30	28	1420	0	0	0	16	0	4
Playing Catch	5	22	6	0	9	0	0	0	0	0	0	0	42	1611	1	0	0	29
Dribblinlg	6	4	2	6	0	0	0	0	0	0	0	5	1	22	1672	0	0	8
Writing	0	0	4	29	0	0	0	2	0	1	0	0	3	0	0	1686	0	0
Clapping	1	0	0	8	4	10	43	2	6	0	0	3	0	0	0	2	1629	17
Folding Clothes	0	0	4	0	6	0	0	3	2	2	0	13	0	13	0	0	3	1748

Figure 5.22: For the Upper Body Movement Evaluation, the confusion matrix displays the results from the personalized transformer model.

		Predicted Class																	
		Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
True Class	Typing	0	0	3	229	47	1139	0	4	72	39	0	15	0	0	0	142	18	17
	Brushing Teeth	2	0	8	86	84	0	1333	12	9	55	13	25	8	1	2	21	32	35
	Eating Soup	1	0	5	50	114	0	0	1095	166	108	27	125	1	20	1	6	0	22
	Eating Chips	1	0	5	136	99	10	2	76	933	85	58	288	4	3	1	15	0	27
	Eating Pasta	0	0	4	98	113	54	5	118	141	945	41	125	1	0	0	21	4	55
	Drinking	0	0	1	130	139	9	0	39	123	43	915	292	1	0	0	28	0	6
	Eating Sandwich	1	0	20	217	133	6	10	109	466	112	186	405	2	9	0	20	0	30
	Playing Catch	5	5	38	9	8	1	0	2	5	1	1	6	69	1507	42	3	4	19
	Dribblinlg	17	72	21	0	2	0	0	1	1	1	0	5	13	65	1484	4	13	27
	Writing	0	0	1	98	92	163	13	41	17	15	54	34	2	3	2	1181	1	8
	Clapping	19	13	28	40	63	4	88	0	1	4	9	0	0	10	31	12	1340	63
	Folding Clothes	16	2	32	39	17	10	4	5	18	30	17	26	42	60	31	3	1	1441

Figure 5.23: For the Upper Body Movement Evaluation, the confusion matrix displays the results from the generic Trans-CNN model.



		Predicted Class																	
		Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
True Class	Typing	0	0	0	49	0	1620	0	1	8	0	0	8	0	0	0	20	0	19
	Brushing Teeth	0	0	0	9	12	0	1660	0	9	3	9	18	1	0	0	0	3	2
	Eating Soup	0	0	0	4	15	0	0	1619	28	41	16	16	0	0	0	1	0	1
	Eating Chips	0	0	0	16	27	7	0	40	1403	30	35	162	1	6	0	3	0	13
	Eating Pasta	0	0	0	1	10	12	1	68	99	1416	36	61	0	0	0	15	0	6
	Drinking	0	0	0	14	3	6	1	9	73	6	1547	28	0	0	0	39	0	0
	Eating Sandwich	0	0	0	48	1	0	6	13	157	14	77	1388	0	2	0	8	0	12
	Playing Catch	3	2	11	0	13	0	1	1	4	1	0	1	24	1654	1	0	1	8
	Dribblinlg	0	3	4	2	0	0	1	5	0	0	0	4	4	17	1678	0	7	1
	Writing	0	0	1	34	1	39	0	13	3	1	3	4	1	3	0	1622	0	0
	Clapping	0	3	1	7	10	7	5	0	0	0	0	0	0	2	8	0	1677	5
	Folding Clothes	0	0	8	5	6	0	2	8	2	7	5	7	9	10	3	0	2	1720

Figure 5.24: For the Upper Body Movement Evaluation, the confusion matrix displays the results from the personalized Trans-CNN model.

In Figure 5.19, the same trends can be seen as in Figure 5.7. The generic CNN model confused different upper body movements with eating a sandwich, folding clothes, and sitting. As shown in Figure 5.20, the CNN model performed much better after it was personalized. The other two generic models showed the same trends as the generic CNN model, seen in Figures 5.21 and 5.23. The generic model confused upper body movements with other movements whereas the personalized models, seen in Figures 5.22 and 5.24, confused upper body movements a lot less.

5.4.4 Lower Body Movement Evaluation

The lower body movement evaluation investigates the performance of the three models on lower body movements using the lower body activities shown in Table 5.18. The upper body movements were removed from $D2$ for this evaluation. The models in this evaluation were evaluated in a similar way as the models in All Movement Evaluation and Upper Body Movement Evaluation.

Table 5.18: The dataset used in this evaluation included 6 activities that involve only lower body movements.

Activities in the Dataset
Walking
Jogging
Stairs
Sitting
Standing
Kicking (Soccer Ball)

Using the Shapiro-Wilk test, 39 participants were analyzed to determine whether their results were normally distributed. Table 5.19 summarizes the normal and non-normal distributions.

Table 5.19: Shapiro–Wilk test was used to determine if the results for the Upper Body Movement Evaluation were normally distributed or not.

Normal Distributed Data	**Non-Normal Distributed Data**
Generic CNN Accuracy	Generic Transformer Precision
Generic CNN Precision	Generic Trans-CNN Precision
Generic CNN Recall	Generic Trans-CNN Recall
Generic Transformer Accuracy	Generic CNN F1-Score
Generic Transformer Recall	Personalized CNN Accuracy
Generic Transformer F1-Score	Personalized CNN Precision
Generic Trans-CNN Accuracy	Personalized CNN Recall
Generic Trans-CNN F1-Score	Personalized CNN F1-Score
	Personalized Trans-CNN Accuracy
	Personalized Trans-CNN Precision
	Personalized Trans-CNN Recall
	Personalized Trans-CNN F1-Score

Both the generic CNN model and the personalized CNN model results are shown in Figure 5.25 using the same type of box plot described in Subsection 5.4.2. In Figure 5.25, the box plots for the Lower Movement Evaluation of CNN demonstrate the same trend as the two other evaluations. Generic models performed around 50%, while personalized models performed around 90% for all four performance metrics. Next, the Mann-Whitney test was used to compare the two models. There was a significant difference between the generic model and the personalized model based on the p values being under 0.05 for each performance metric.

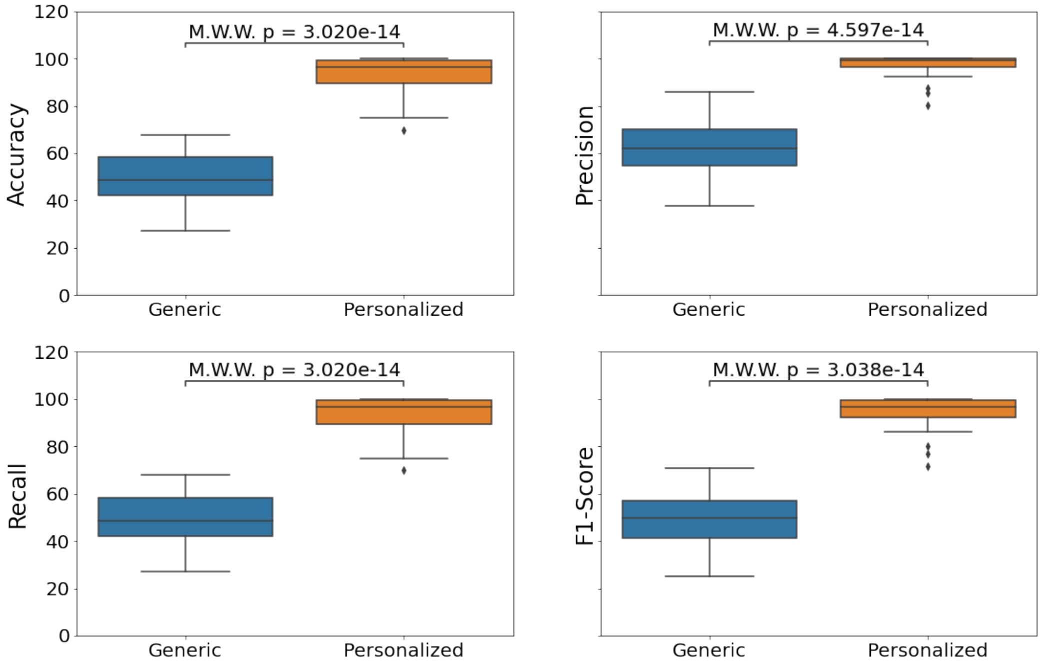


Figure 5.25: For the Lower Body Movement Evaluation, these boxplots display the four performance metrics for both the generic and personalized CNN models.

Figures 5.26 and 5.27 are box plots that were also created for the transformer and Trans-CNN models. The trends in both figures are similar to those in Figure 5.25. The generic transformer model performance was approximately 60% and the personalized transformer model had a performance of approximately 90%; in contrast, the performance of the generic Trans-CNN model was around 80%, while the personalized Trans-CNN model performed around 90%. The Mann-Whitney test was also chosen since the data for both personalized models are nonparametric. There were statistically significant differences between the generic and personalized models for both the transformer and Trans-CNN models based on the p values for all four performance metrics.

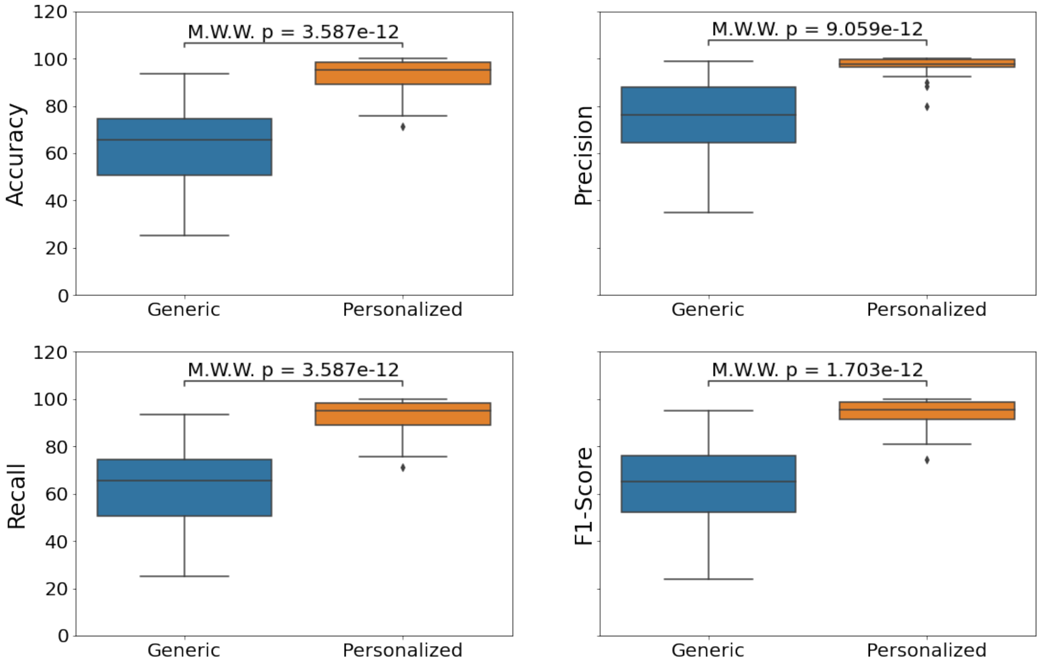


Figure 5.26: For the Lower Movement Evaluation, these boxplots display the four performance metrics for both the generic and personalized transformer models.

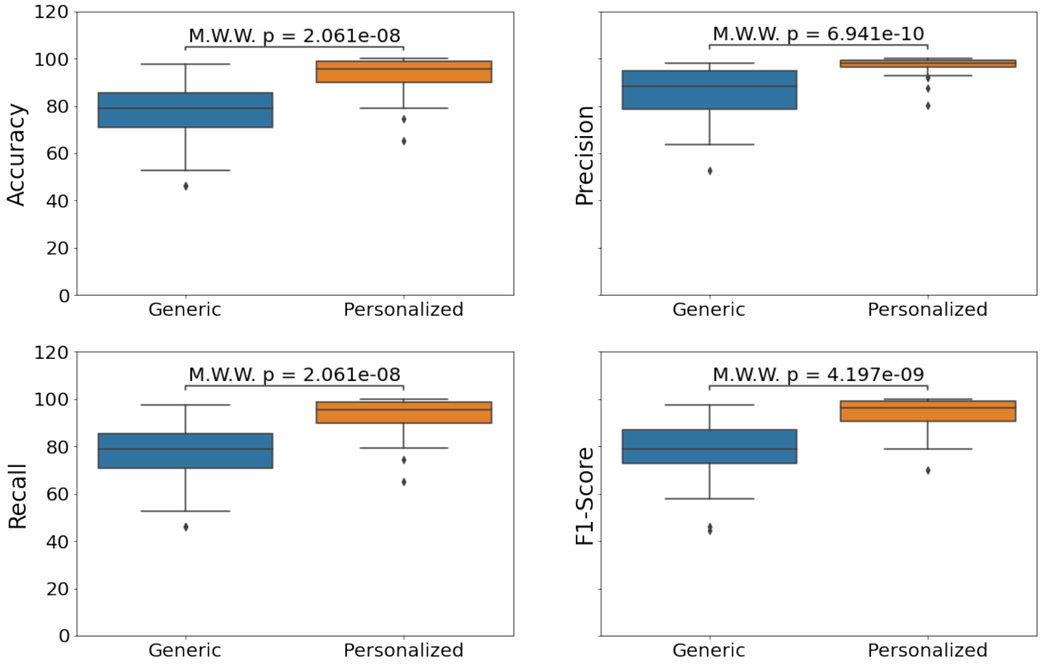


Figure 5.27: For the Lower Movement Evaluation, these boxplots display the four performance metrics for both the generic and personalized Trans-CNN models.

A comparison of the generic and personalized models was presented in Figures 5.25, 5.26, and 5.27. To determine the best generic model, Figure 5.28 compares the three generic models.

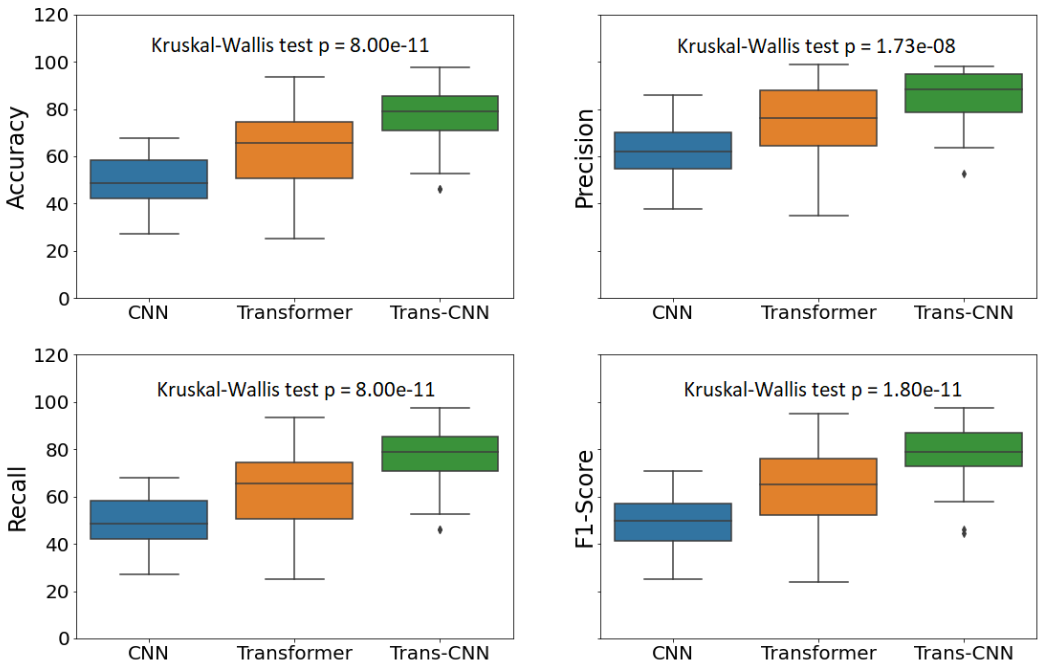


Figure 5.28: For the Lower Body Movements Evaluation, boxplots were created for each of the four performance metrics for the three generic models.

There was a significant difference in the three groups based on a Kruskal-Wallis test with a p value below 0.05. To determine where the difference lies, a Dunn test was performed with a Bonferroni adjusted p value. The results are shown in Tables 5.20, 5.21, 5.22, and 5.23.

Table 5.20: Dunn test with Bonferroni adjusted p value was performed for accuracy.

	CNN	**Transformer**	**Trans-CNN**
CNN	1.00	0.00241	2.76e-11
Transformer	0.00241	1.00	1.58e-03
Trans-CNN	2.76e-11	1.58e-03	1.00

Table 5.21: Dunn test with Bonferroni adjusted p value was performed for precision.

	CNN	**Transformer**	**Trans-CNN**
CNN	1.00	9.40e-03	6.45e-09
Transformer	9.40e-03	1.00	7.47e-03
Trans-CNN	6.45e-09	7.47e-03	1.00

Table 5.22: Dunn test with Bonferroni adjusted p value was performed for recall.

	CNN	**Transformer**	**Trans-CNN**
CNN	1.00	0.00241	2.76e-11
Transformer	0.00241	1.00	1.58e-03
Trans-CNN	2.76e-11	1.58e-03	1.00

Table 5.23: Dunn test with Bonferroni adjusted p value was performed for f1-score.

	CNN	**Transformer**	**Trans-CNN**
CNN	1.00	1.02e-03	6.04e-12
Transformer	1.02e-03	1.00	1.67e-03
Trans-CNN	6.04e-12	1.67e-03	1.00

The Trans-CNN model outperformed both the CNN and transformer models in all four metrics, with a significant statistical difference shown by a p value of less than 0.05 when comparing the Trans-CNN model to the other two models. There was also a significant difference between CNN and transformer models for the four performance metrics analyzed since the p value was under 0.05; therefore, the transformer model outperformed the CNN model.

This process was repeated for the personalized models as seen in Figure 5.29. As a personalized model, Figure 5.29 compares the three ML models. Based on the Kruskal-Wallis test, there was no significant difference between the three models since the p value was over 0.05.

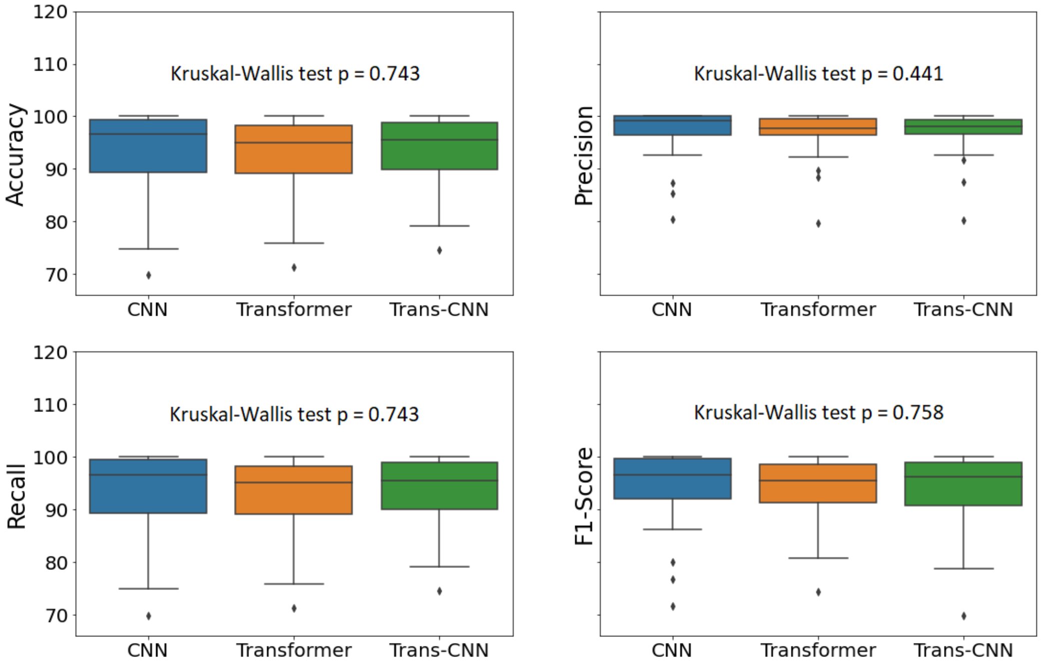


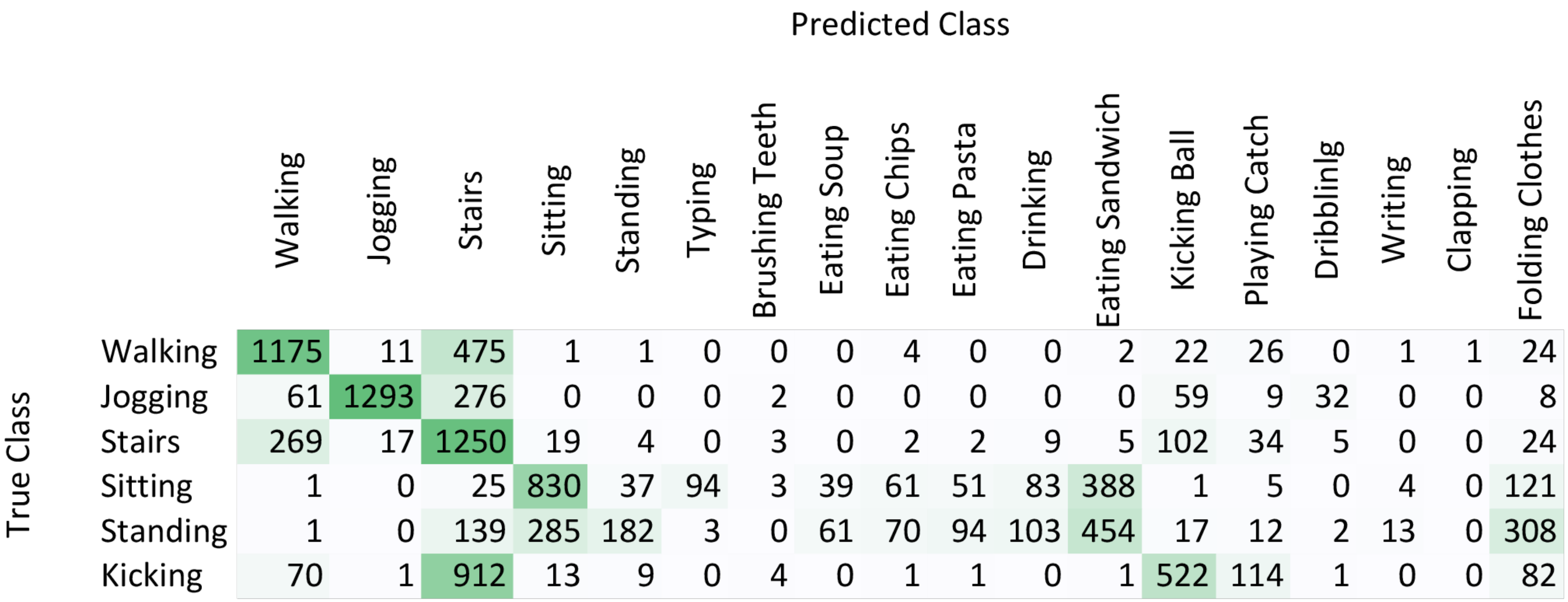
Figure 5.29: For the Lower Body Movements Evaluation, boxplots were created for each of the four performance metrics for the three personalized models.

Table 5.24 shows the averages and standard deviations for each performance metric based on the same data that were used for the box plots. For all four performance metrics, the Trans-CNN model as a generic model performed the best and achieved the highest average value of around 75%, which was significantly different from the other two. The transformer averaged a value of around 60% and CNN averaged a value of around 50%. There was also a significant difference between the generic CNN model and the generic transformer model. None of the four performance metrics for the personalized models showed a significant difference between the three models, which all scored around 90% on average.

Table 5.24: In the Lower Body Movement Evaluation, for all three models, the averages including the standard deviations were calculated for the four performance metrics for both the generic and personalized models using the data from 39 participants.

	CNN	**Transformer**	**Trans-CNN**
Accuracy			
Generic Model	50.4±10.9%	62.8±17.6%	76.9±13.0%
Personalized Model	93.5±7.6%	92.7±7.3%	92.9±8.0%
Precision			
Generic Model	62.9±12.4%	73.3±17.5%	84.7±11.7%
Personalized Model	97.2±4.4%	97.0±4.0%	97.1±3.9%
Recall			
Generic Model	50.4±10.9%	62.8.2±17.6%	76.9±13.0%
Personalized Model	93.5±7.6%	92.7±7.3%	92.9±8.0%
F1-Score			
Generic Model	49.4±11.0%	63.5±18.1%	77.8±13.0%
Personalized Model	94.3±7.0%	93.9±6.2%	94.1±6.8%

A confusion matrix was created for the generic and personalized models of all three structures to identify which activities were confused with each other. The confusion matrix consists of rows representing actual classes and columns representing predicted classes. Figures 5.30, 5.31, 5.32, 5.33, 5.34, and 5.35 contains the confusion matrices for this evaluation.



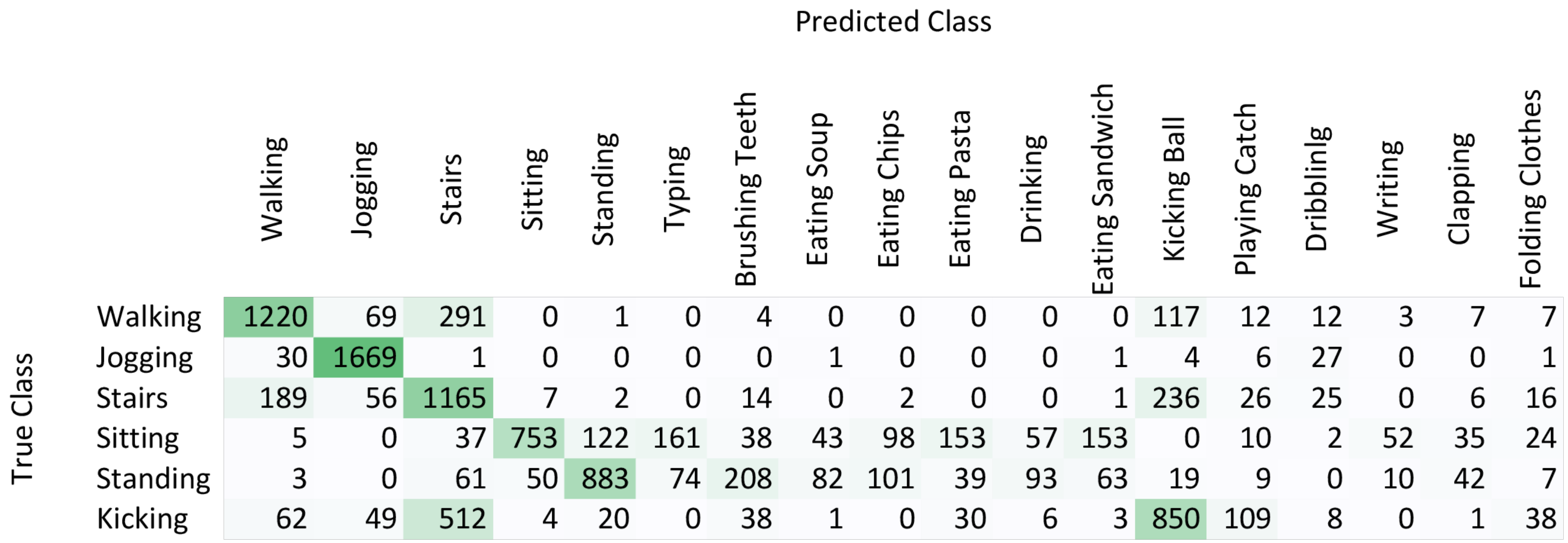
Predicted Class

True Class	Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
Walking	1175	11	475	1	1	0	0	0	4	0	0	2	22	26	0	1	1	24
Jogging	61	1293	276	0	0	0	2	0	0	0	0	0	59	9	32	0	0	8
Stairs	269	17	1250	19	4	0	3	0	2	2	9	5	102	34	5	0	0	24
Sitting	1	0	25	830	37	94	3	39	61	51	83	388	1	5	0	4	0	121
Standing	1	0	139	285	182	3	0	61	70	94	103	454	17	12	2	13	0	308
Kicking	70	1	912	13	9	0	4	0	1	1	0	1	522	114	1	0	0	82

Figure 5.30: For the Lower Body Movement Evaluation, the confusion matrix displays the results from the generic CNN model.

		Predicted Class																	
		Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
True Class	Walking	1667	2	51	0	0	1	0	0	2	0	0	0	5	2	10	0	0	3
	Jogging	10	1713	7	1	0	0	0	0	0	0	0	0	6	0	0	0	0	3
	Stairs	68	10	1586	2	3	0	0	0	0	0	0	0	63	13	0	0	0	0
	Sitting	0	0	0	1510	4	55	6	16	56	0	28	52	0	1	1	5	0	9
	Standing	0	0	1	30	1653	4	2	4	13	0	12	2	7	0	0	0	0	16
	Kicking	8	8	53	0	1	0	2	0	0	0	0	0	1621	38	0	0	0	0

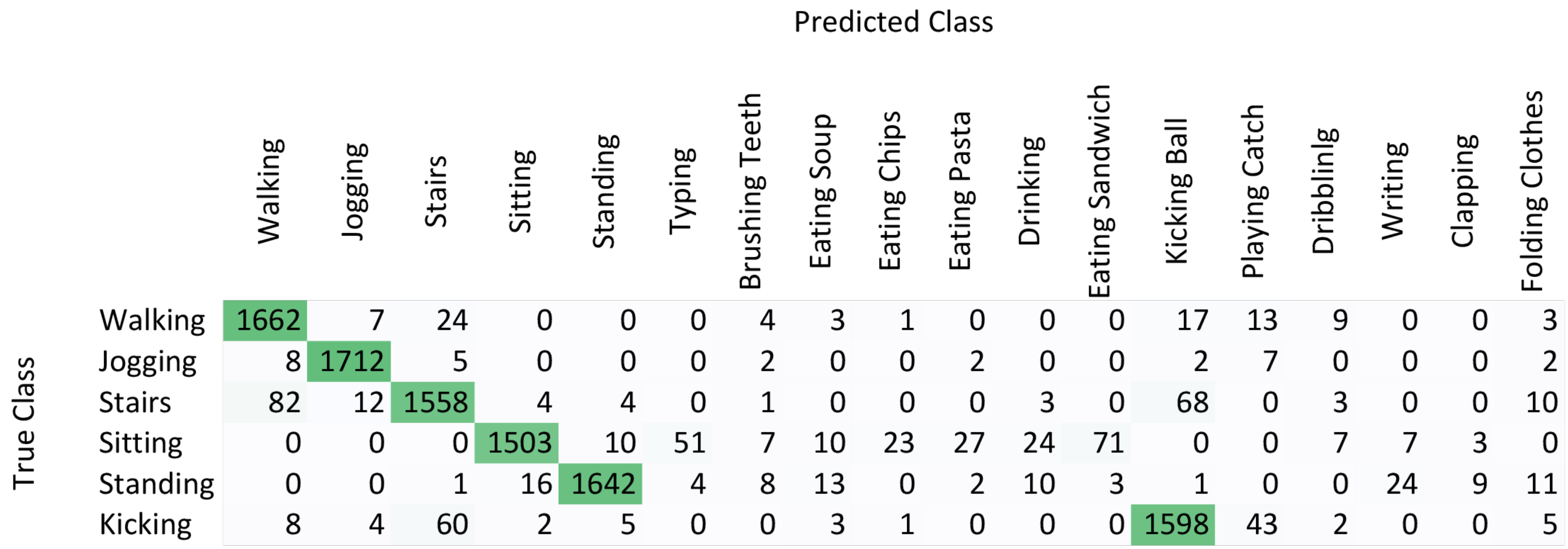
Figure 5.31: For the Lower Body Movement Evaluation, the confusion matrix displays the results from the personalized CNN model.



Predicted Class

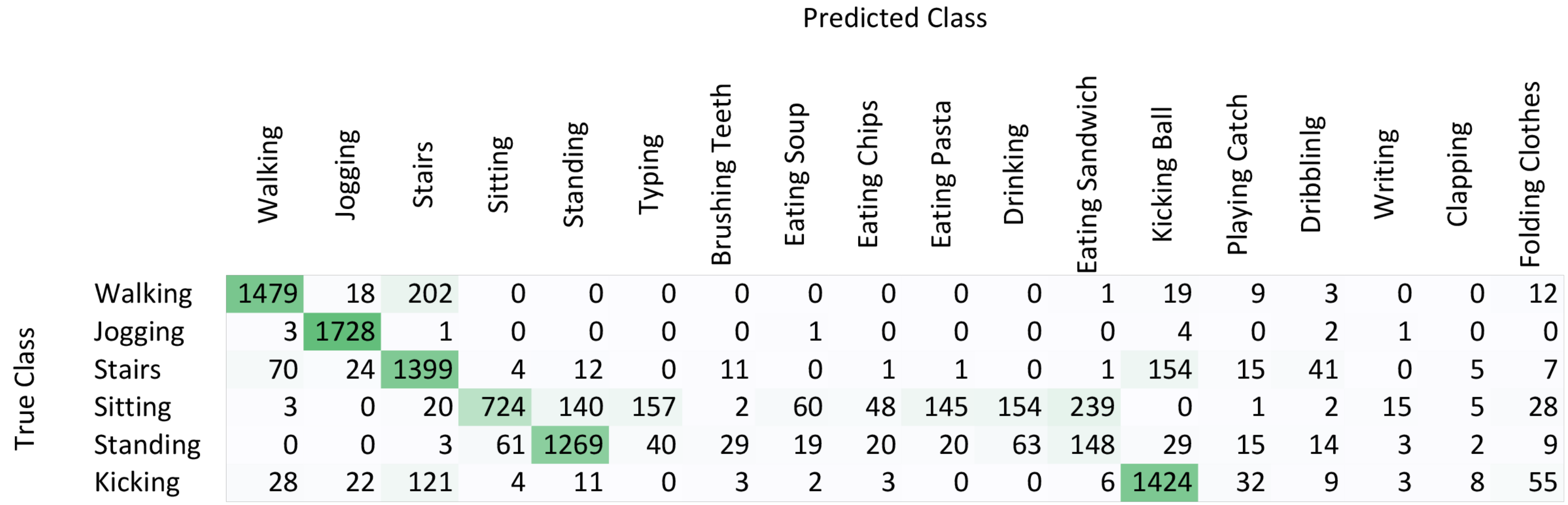
True Class		Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
	Walking	1220	69	291	0	1	0	4	0	0	0	0	0	117	12	12	3	7	7
	Jogging	30	1669	1	0	0	0	0	1	0	0	0	1	4	6	27	0	0	1
	Stairs	189	56	1165	7	2	0	14	0	2	0	0	1	236	26	25	0	6	16
	Sitting	5	0	37	753	122	161	38	43	98	153	57	153	0	10	2	52	35	24
	Standing	3	0	61	50	883	74	208	82	101	39	93	63	19	9	0	10	42	7
	Kicking	62	49	512	4	20	0	38	1	0	30	6	3	850	109	8	0	1	38

Figure 5.32: For the Lower Body Movement Evaluation, the confusion matrix displays the results from the generic transformer model.



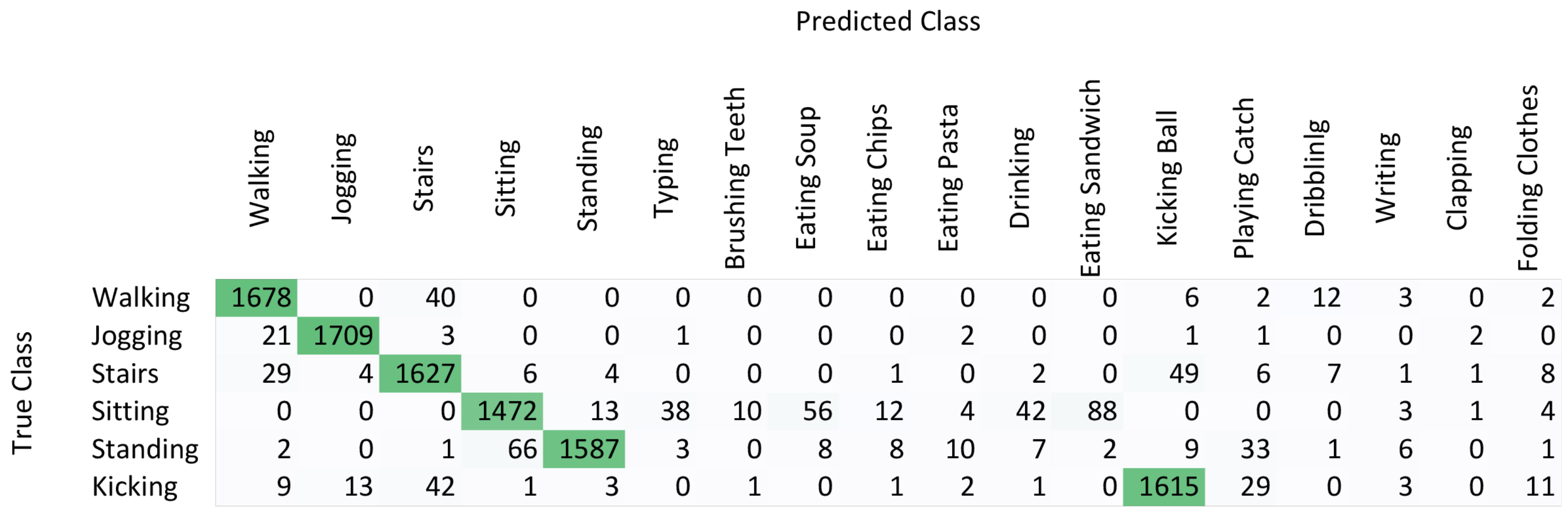
		Predicted Class																	
		Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
True Class	Walking	1662	7	24	0	0	0	4	3	1	0	0	0	17	13	9	0	0	3
	Jogging	8	1712	5	0	0	0	2	0	0	2	0	0	2	7	0	0	0	2
	Stairs	82	12	1558	4	4	0	1	0	0	0	3	0	68	0	3	0	0	10
	Sitting	0	0	0	1503	10	51	7	10	23	27	24	71	0	0	7	7	3	0
	Standing	0	0	1	16	1642	4	8	13	0	2	10	3	1	0	0	24	9	11
	Kicking	8	4	60	2	5	0	0	3	1	0	0	0	1598	43	2	0	0	5

Figure 5.33: For the Lower Body Movement Evaluation, the confusion matrix displays the results from the personalized transformer model.



True Class \ Predicted Class	Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
Walking	1479	18	202	0	0	0	0	0	0	0	0	1	19	9	3	0	0	12
Jogging	3	1728	1	0	0	0	0	1	0	0	0	0	4	0	2	1	0	0
Stairs	70	24	1399	4	12	0	11	0	1	1	0	1	154	15	41	0	5	7
Sitting	3	0	20	724	140	157	2	60	48	145	154	239	0	1	2	15	5	28
Standing	0	0	3	61	1269	40	29	19	20	20	63	148	29	15	14	3	2	9
Kicking	28	22	121	4	11	0	3	2	3	0	0	6	1424	32	9	3	8	55

Figure 5.34: For the Lower Body Movement Evaluation, the confusion matrix displays the results from the generic Trans-CNN model.



Predicted Class

True Class

	Walking	Jogging	Stairs	Sitting	Standing	Typing	Brushing Teeth	Eating Soup	Eating Chips	Eating Pasta	Drinking	Eating Sandwich	Kicking Ball	Playing Catch	Dribblinlg	Writing	Clapping	Folding Clothes
Walking	1678	0	40	0	0	0	0	0	0	0	0	0	6	2	12	3	0	2
Jogging	21	1709	3	0	0	1	0	0	0	2	0	0	1	1	0	0	2	0
Stairs	29	4	1627	6	4	0	0	0	1	0	2	0	49	6	7	1	1	8
Sitting	0	0	0	1472	13	38	10	56	12	4	42	88	0	0	0	3	1	4
Standing	2	0	1	66	1587	3	0	8	8	10	7	2	9	33	1	6	0	1
Kicking	9	13	42	1	3	0	1	0	1	2	1	0	1615	29	0	3	0	11

Figure 5.35: For the Lower Body Movement Evaluation, the confusion matrix displays the results from the personalized Trans-CNN model.

In Figure 5.30, the CNN model confused most of the lower body movements with stairs. This is due to the fact that going upstairs is a similar motion to kicking a ball. Both activities require the participant to lift their leg. The other two generic models showed a similar trend, as seen in Figures 5.32 and 5.34. The personalized models across all three model structures performed well with a lot fewer incorrect labels as seen in Figures 5.31, 5.33, and 5.35.

5.5 Discussion

In this thesis, combined data (both upper and lower movements) were used to train the model since the label of data in a real-world application is unknown beforehand. In the All Movement Evaluation, the model was evaluated using combined data. Figure 5.2 demonstrated the value of personalization since the performance metrics increased from a value in the 40s to a value in the 90s for the CNN model. The same trend can be seen with the other two models as seen in Figures 5.3 and 5.4. The performance metrics increased for all three models when personalization was applied.

For the generic models, the Trans-CNN model outperformed the other two models for all four performance metrics as seen in Figure 5.5. There was also a significant difference between the generic Trans-CNN and the CNN model as well as the generic Trans-CNN and the transformer model. From reviewing Table 5.10, many trends were seen. The Trans-CNN average accuracy was 69.6% whereas the other two models were under 50%. These results suggests that the transformer model can perform as well as the CNN model, the dominant model for HAR in recent years.

Figure 5.6 showed that after personalization the three models across the four metrics achieved similar results with values close to 90%. From Table 5.10, the personalized models performed well with a score in the low 90s across all four metrics, with no statistical difference between the three models. Based on this finding, any of the three models can be used for personalizing the models. Although the performance of the three models was in the low 90s, the CNN model benefited the most from personalizing the models. The CNN model went from 41.2% to 94.1% for accuracy, whereas the transformer model went from 49.2% to 92.1% and the Trans-CNN went from 69.6% to 92.4%.

From Figure 5.13, accuracy was plotted for 10 participants using both the generic (red points) and personalized (blue points) models. When the generic model was used, a fair bit of variation from participant to participant was seen. However, when the personalized models were used, the variation was minimal. These results suggest that there was a variation in participants that could be reduced with personalization.

In the Upper Body Movement Evaluation, only upper body data were used to evaluate both the generic and personalized model. The results obtained were similar to the previous evaluation except overall the performance metrics were lower in value. Figures 5.14, 5.15, and 5.16 showed the same trend with each performance metric increasing from a value in the range of 30-70% to a value in the 90s as seen in the All Movement Evaluation. For the generic model, the Trans-CNN performed significantly better than either CNN or transformer as seen in Figure 5.17. For the personalized model, the three models performed similarly in the low 90s as seen in Figure 5.18. Even though the performance of the generic model metrics were lower in this evaluation, the performance metrics for the personalized model were comparable in value to the All Movement Evaluation, as seen in Tables 5.10 and 5.17. The model did not perform as well with only upper body movement data; however, as long as the model was personalized, the model performed equally well with any type of data.

In the Lower Body Movement Evaluation, only lower body data were used to evaluate both the generic and personalized models. Figures 5.25, 5.26, and 5.27 showed the same trend with each performance metric increasing from a value in the range of 50-80% to a value in the 90s as seen in the All Movement and Upper Body Movement Evaluation. For the generic model, the same trend was observed as seen in the other two evaluations: the Trans-CNN performed significantly better than either CNN or transformer (see Figure 5.28). For the personalized model, the same trend was seen with the models performing in the low 90s (see Figure 5.29). As seen in Tables 5.10, 5.17, 5.24, the performance metrics for the generic model were substantially higher in the Lower Body Movement Evaluation as compared to the Upper Body Movement Evaluation and slightly higher than the All Movement Evaluation. However, for the personalized model, the performance metrics were almost identical to the other two evaluations. Therefore, the type of data is irrelevant to the performance of the model as long as the model is personalized. For all three evaluations, the boxplots for the models demonstrated the amount of variation in the results for each of the four performance metrics. Looking at the boxplots, a lot of variation was seen with the generic models as compared to the personalized models. As seen in the boxplots, personalization eliminates participant to participant variation.

5.6 Chapter Summary

In this chapter, the WISDM 2019 dataset was presented along with how the data were prepared for training. Grid search was used to finalize the details of the three model structures. In the three evaluations, a lot of variation from participant to participant was found; however, this variation can be reduced by personalizing the model. As a generic model, the Trans-CNN model performed better than the other two models with respect to all four performance metrics. Trans-CNN accuracy metric achieved 69.6% despite having never seen the target participant's data before. Although transformers have not been used traditionally with HAR data, these evaluations demonstrated that it performs equally to the CNN model with HAR data. In all three evaluations, the value of personalization was seen since the four performance metrics (accuracy, precision, recall, and f1-score) improved substantially to values in the 90s for all three models. Therefore, data type (combined, upper, or lower) does not matter if the model is personalized. This finding is significant since in the real world the type of data is not known beforehand.

Chapter 6

Conclusion and Future Work

This chapter provides a high-level overview of the proposed process for detecting upper and lower body movements followed by an examination of the findings. A discussion of future work takes place in Section 6.2.

6.1 Conclusion

In this thesis, a personalized approach to HAR was presented that leverages individual characteristics and past individual's activity data to improve the performance. As a result of the model training procedure, it was demonstrated that the personalized HAR approach presented in this thesis could adapt to new users with all of the performance metrics above 90%. Traditional HAR models have difficulty adapting to new participants because humans have a number of physiological properties that influence their movement. In order to accurately detect an individual's movement, a model should be personalized to that individual. Chapter 4 presented a personalization approach that involved creating a generic model and then using transfer learning to personalize it to a particular participant.

The three model structures in this thesis were selected for a particular reason. The CNN model has been successfully used in a variety of different studies as seen in Chapter 3. Next, selected was the modified transformer model structure for HAR. The transformer model is successful with sequential data such as natural language; therefore, since time series data for HAR are a kind of sequential data, the transformer model has the potential to be successful for personalized HAR. Lastly, the Trans-CNN model structure is a hybrid model that was created by combining the major components from both the CNN and transformer models in order to exploit the advantages of the models.

The model training procedure that was presented was assessed with the WISDM 2019 dataset because the dataset includes both upper and lower body movements. After the generic and personalized models were trained, the models were evaluated with three different methods: All Movement Evaluation, Upper Body Movement Evaluation, and Lower Body Movement Evaluation.

Using both upper and lower body data for the All Movement Evaluation, the generic and personalized models were evaluated and then compared. Figure 5.2 illustrated how valuable personalization is. The performance metrics for the CNN model increased from a value in the 40s to a value in the 90s. The other two models showed the same trend; therefore, personalization increased all of the performance metrics for all three models. Personalization of models can also reduce the variation in the results from participant to participant as seen in Figure 5.13. When comparing the three generic models, it was determined that the Trans-CNN model outperformed the other two models for all four performance metrics with a significant difference. Trans-CNN accuracy metric was 69.6% despite having never seen the target participant's data before. On the other hand, when comparing the personalized models, it was determined that there was no statistical difference between the models.

In the Upper Body Movement Evaluation, the results were similar to those obtained in the previous evaluation except for the results of the generic models, which were overall lower in value. When comparing the generic model to the personalized model, the performance metrics increased after personalization occurred. Similar to the last evaluation, the generic Trans-CNN model performed better with a significant difference from the other two generic models. The personalized models, however, did not differ statistically when compared.

As a result of using only lower body data in the last evaluation, similar results were obtained as in the other two evaluations, except that the generic model gave higher results. After personalization happened, the performance metrics increased in value similar to the other two evaluations. In the same way as the previous two evaluations, Trans-CNN performed significantly better than the other two models. When comparing the personalized models, there was not a significant difference.

Comparing the three evaluations, the generic model had significantly higher performance metrics in Lower Body Movement Evaluation as compared to Upper Body Movement Evaluation and slightly higher performance metrics in All Movement Evaluation. However, the performance metrics for the personalized model were almost identical to the other two evaluations. In other words, it does not matter whether combined, upper, or lower body data are used in a personalized model.

These results suggest that the transformer model can perform as well as the CNN model, the dominant model for HAR in recent years.

6.2 Future Work

Future work will explore the potential to improve the models, the model training procedure, and the procedure in different scenarios. Thus, future work will include the following:

- Evaluating the model training procedure with repetitive motions that cause RSI: In Chapter 5, it was seen that the procedure performed fairly well with the WISDM 2019 dataset. To see how well this procedure will perform in a real-world application, the data need to be motions known to cause RSI.

- Exploring the use of incremental learning (also known as online learning) instead of transfer learning for personalizing the generic model: As seen in Chapter 4, transfer learning involves transferring the initial weights from the generic model to a new model and then retrain the model with the frozen layers to create the personalized model. On the other hand, with incremental learning, models are continuously learning and extending their knowledge, as they adjust what they have already learned in response to new data. Further research is needed to understand the potential of using incremental learning better.

- Investigating what is the minimal amount of data needed from the target participant for personalization: This work used one third of the target participant's data to personalize the model. It is unclear how much data are required for personalization, as different amounts could potentially be used. The topic requires further research.

- Examining the model training procedure scalability: The model training procedure was evaluated with 39 participants; however, it is unknown how well the model would perform if the data had more participants. Further research is needed.

- Investigating the model training procedure with data from wearable sensors: In the WISDM dataset, people are in a controlled environment instead of people completing activities naturally. The dataset came from internal sensors such as gyroscopes and accelerometers which only capture limited data. On the other hand, wearable sensors can measure full biomechanics and muscle activity, not just motion.

- Exploring the number of participants needed to create the generic model: In this thesis, the generic model was trained with 38 participants; however, it is unknown if 38 participants creates the best performing generic model. For example, maybe 70 participants create the best model or even 150 participants. In order to answer this question, further research is needed.

- Investigating the use of other HAR models instead of the three models used in this thesis: even though CNN and transformer have been successful with HAR, other deep learning models might prove to be more successful.

- Examining the use of a dataset acquired at a higher frequency: A dataset with data collected at a higher frequency may reveal patterns or trends that are not evident in a dataset with data collected at a lower frequency, which could lead to new insights or discoveries.

- Exploring the application of the personalization technique for identifying differences in individuals during rehabilitation: The use of the personalization technique could enable the monitoring of an individual's progress as they recover.

The model training procedure presented performed well and showed potential in personalizing HAR models. However, there is still room for improvement, as discussed in this section.

References

[1] B. A. O'Neil, M. E. Forsythe, and W. D. Stanish, "Chronic occupational repetitive strain injury." *Canadian Family Physician*, vol. 47, no. 2, pp. 311–316, 2001.

[2] B. McConnell, "Repetitive strain injuries," *International Review of Law, Computers & Technology*, vol. 7, no. 1, pp. 231–236, 1993.

[3] A. Yassi, "Repetitive strain injuries," *The Lancet*, vol. 349, no. 9056, pp. 943–947, 1997.

[4] P. Bierma, "Repetitive stress injury," Aug 2022. [Online]. Available: https://consumer.healthday.com/encyclopedia/pain-management-30/pain-health-news-520/repetitive-stress-injury-rsi-646236.html#:~:text=According%20to%20the%20Occupational%20Safety,and%20%2420%20billion%20a%20year.

[5] Canadian Centre for Occupational Health and Safety, "Painful disorders focus of international repetitive strain injury awareness day," Oct 2021. [Online]. Available: https://www.ccohs.ca/newsroom/news_releases/RSIDay2017_22Feb2017.html#:~:text=Painful%20Disorders%20Focus%20of%20International%20Repetitive%20Strain%20Injury%20(RSI)%20Awareness%20Day,-For%20Immediate%20Release&amp;text=From%20carpal%20tunnel%20syndrome%2C%20to,)%2C%20according%20to%20Statistics%20Canada.

[6] Microsoft, "Reducing the incidence and cost of work-related musculoskeletal." [Online]. Available: https://webobjects.cdw.com/webobjects/media/pdf/CDWCA/Ergo_Whitepaper_June-2017.pdf

[7] RSI UK, "Rsi awareness." [Online]. Available: http://www.rsi.org.uk/pdf/ULDs_Overview.pdf

[8] M. M. Hassan, M. Z. Uddin, *et al.*, "A robust human activity recognition system using smartphone sensors and deep learning," *Future Generation Computer Systems*, vol. 81, pp. 307–313, 2018.

[9] L. M. Dang, K. Min, *et al.*, "Sensor-based and vision-based human activity recognition: A comprehensive survey," *Pattern Recognition*, vol. 108, p. 107561, 2020.

[10] J. Wang, Y. Chen, *et al.*, "Deep learning for sensor-based activity recognition: A survey," *Pattern Recognition Letters*, vol. 119, pp. 3–11, 2019.

[11] A. O. Ige and M. H. M. Noor, "A survey on unsupervised learning for wearable sensor-based activity recognition," *Applied Soft Computing*, p. 109363, 2022.

[12] S. A. Rokni, M. Nourollahi, and H. Ghasemzadeh, "Personalized human activity recognition using convolutional neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[13] A. Ferrari, D. Micucci, *et al.*, "Personalization in human activity recognition," *ArXiv Preprint ArXiv:2009.00268*, 2020.

[14] D. M. Burns and C. M. Whyne, "Personalized activity recognition with deep triplet embeddings," *ArXiv Preprint ArXiv:2001.05517*, 2020.

[15] A. Khan, A. Sohail, *et al.*, "A survey of the recent architectures of deep convolutional neural networks," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5455–5516, 2020.

[16] Y. Wei, W. Xia, *et al.*, "Hcp: A flexible cnn framework for multi-label image classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 9, pp. 1901–1907, 2015.

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

[18] S.-M. Lee, S. M. Yoon, and H. Cho, "Human activity recognition from accelerometer data using convolutional neural network," in *2017 IEEE International Conference on Big Data and Smart Computing (Bigcomp).* IEEE, 2017, pp. 131–134.

[19] Y. Xu and T. T. Qiu, "Human activity recognition and embedded application based on convolutional neural network," *Journal of Artificial Intelligence and Technology*, vol. 1, no. 1, pp. 51–60, 2021.

[20] T. Wolf, L. Debut, *et al.*, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020, pp. 38–45.

[21] L. E. Lopez, D. K. Cruz, *et al.*, "Transformer-based end-to-end question generation," *arXiv preprint arXiv:2005.01107*, vol. 4, 2020.

[22] A. Vaswani, N. Shazeer, *et al.*, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[23] A. Gillioz, J. Casas, *et al.*, "Overview of the transformer-based models for nlp tasks," in *2020 15th Conference on Computer Science and Information Systems (FedCSIS).* IEEE, 2020, pp. 179–183.

[24] Q. Wen, L. Sun, *et al.*, "Time series data augmentation for deep learning: A survey," *arXiv preprint arXiv:2002.12478*, 2020.

[25] L. Torrey and J. Shavlik, "Transfer learning," in *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques.* IGI global, 2010, pp. 242–264.

[26] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big data*, vol. 3, no. 1, pp. 1–40, 2016.

[27] R. Ribani and M. Marengoni, “A survey of transfer learning for convolutional neural networks,” in *2019 32nd SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*. IEEE, 2019, pp. 47–57.

[28] Z.-Y. He and L.-W. Jin, “Activity recognition from acceleration data using ar model representation and svm,” in *2008 International Conference on Machine Learning and Cybernetics*, vol. 4, 2008, pp. 2245–2250.

[29] Y. Chen and Y. Xue, “A deep learning approach to human activity recognition based on single accelerometer,” in *2015 IEEE International Conference on Systems, Man, and Cybernetics*, 2015, pp. 1488–1492.

[30] M. A. Alsheikh, A. Selim, *et al.*, “Deep activity recognition models with triaxial accelerometers,” in *Workshops at the Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[31] D. Hendry, K. Chai, *et al.*, “Development of a human activity recognition system for ballet tasks,” *Sports Medicine-open*, vol. 6, no. 1, pp. 1–10, 2020.

[32] S. W. Pienaar and R. Malekian, “Human activity recognition using lstm-rnn deep neural network architecture,” in *2019 IEEE 2nd Wireless Africa Conference (WAC)*, 2019, pp. 1–5.

[33] WISDM-2012, “Wisdm 2012 dataset.” [Online]. Available: https://www.cis.fordham.edu/wisdm/dataset.php

[34] W. Sousa Lima, E. Souto, *et al.*, “Human activity recognition using inertial sensors in a smartphone: An overview,” *Sensors*, vol. 19, no. 14, p. 3213, 2019.

[35] M. Straczkiewicz, P. James, and J.-P. Onnela, “A systematic review of smartphone-based human activity recognition methods for health research,” *NPJ Digital Medicine*, vol. 4, no. 1, pp. 1–15, 2021.

[36] C. A. Ronao and S.-B. Cho, “Human activity recognition with smartphone sensors using deep learning neural networks,” *Expert Systems with Applications*, vol. 59, pp. 235–244, 2016.

[37] S. Mekruksavanich and A. Jitpattanakul, “Smartwatch-based human activity recognition using hybrid lstm network,” in *2020 IEEE SENSORS*, 2020, pp. 1–4.

[38] I. Dirgová Luptáková, M. Kubovčík, and J. Pospíchal, “Wearable sensor-based human activity recognition with transformer model,” *Sensors*, vol. 22, no. 5, p. 1911, 2022.

[39] H. Amrani, D. Micucci, and P. Napoletano, “Personalized models in human activity recognition using deep learning,” in *2020 25th International Conference on Pattern Recognition (ICPR)*, 2021, pp. 9682–9688.

[40] D. Gholamiangonabadi and K. Grolinger, “Personalized models for human activity recognition with wearable sensors: Deep neural networks and signal processing,” *Applied Intelligence*, pp. 1–21, 2022.

[41] J. Brownlee, “Data preparation for machine learning,” 2022.

[42] A. Dehghani, O. Sarbishei, *et al.*, "A quantitative comparison of overlapping and non-overlapping sliding windows for human activity recognition using inertial sensors," *Sensors*, vol. 19, no. 22, p. 5026, 2019.

[43] A. Ignatov, "Real-time human activity recognition from accelerometer data using convolutional neural networks," *Applied Soft Computing*, vol. 62, pp. 915–922, 2018.

[44] S. Geller, "Normalization vs standardization. the two most important feature..." Apr 2019. [Online]. Available: https://towardsdatascience.com/normalization-vs-standardization-cb8fe15082eb

[45] M. Kravchik and A. Shabtai, "Detecting cyber attacks in industrial control systems using convolutional neural networks," in *Proceedings of the 2018 Workshop on Cyber-physical Systems Security and Privacy*, 2018, pp. 72–83.

[46] S. Verma, "Understanding 1d and 3d convolution neural network: Keras," Apr 2022. [Online]. Available: https://towardsdatascience.com/understanding-1d-and-3d-convolution-neural-network-keras-9d8f76e29610

[47] N. Lang, "Using convolutional neural network for image classification," Dec 2021. [Online]. Available: https://towardsdatascience.com/using-convolutional-neural-network-for-image-classification-5997bfd0ede4

[48] H. Cho and S. M. Yoon, "Divide and conquer-based 1d cnn human activity recognition using test data sharpening," *Sensors*, vol. 18, no. 4, p. 1055, 2018.

[49] M. C. Leong, H. Zhang, *et al.*, "Combined cnn transformer encoder for enhanced fine-grained human action recognition," *arXiv preprint arXiv:2208.01897*, 2022.

[50] J. Brownlee, "How do convolutional layers work in deep learning neural networks?" Apr 2020. [Online]. Available: https://machinelearningmastery.com/convolutional-layers-for-deep-learning-neural-networks/

[51] ——, "A gentle introduction to dropout for regularizing deep neural networks," Aug 2019. [Online]. Available: https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/

[52] D. Gholamiangonabadi, N. Kiselov, and K. Grolinger, "Deep neural networks for human activity recognition with wearable sensors: Leave-one-subject-out cross-validation for model selection," *IEEE Access*, vol. 8, pp. 133 982–133 994, 2020.

[53] WISDM-2019. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/WISDM+Smartphone+and+Smartwatch+Activity+and+Biometrics+Dataset+

[54] M. Olugbenga, "Balanced accuracy: When should you use it?" Jul 2022. [Online]. Available: https://neptune.ai/blog/balanced-accuracy#:~:text=Balanced%20Accuracy%20Multiclass%20Classification,be%20the%20same%20as%20Accuracy.

[55] J. Brownlee, "A gentle introduction to pooling layers for convolutional neural networks," Jul 2019. [Online]. Available: https://machinelearningmastery.com/pooling-layers-for-convolutional-neural-networks/

[56] E. Beauxis-Aussalet and L. Hardman, “Visualization of confusion matrix for non-expert users,” in *IEEE Conference on Visual Analytics Science and Technology (VAST)-Poster Proceedings*, 2014.

Curriculum Vitae

Name:	Kyle B. Lacroix
Post-secondary Education and Degrees:	Western University London, Ontario, Canada 2017–2021 B.E.Sc., Mechatronic Systems Engineering
	Western University London, Ontario, Canada 2021-2023 M.E.Sc., Electrical and Computer Engineering (Artificial Intelligence)
Honours and Awards:	Institute of Electrical and Electronic Engineers Inc. I.E.E.E. Award Dr. E. V. Buchanan Prize Ontario Graduate Scholarship Vector Scholarship in Artificial Intelligence
Related Work Experience:	Teaching Assistant Western University 2022

- S. Vecile, K. Lacroix, K. Grolinger, and J. Samarabandu. ”Malicious and benign URL dataset generation using character-level LSTM models.” In 2022 IEEE Conference on Dependable and Secure Computing (DSC), pp. 1-8. IEEE, Edinburgh, United Kingdom, 22-24 June 2022.