UNIVERSITETET I AGDER

A Deep Learning Approach for Recognizing Daily Movement Patterns through Accelerometer Data

SAHAND JOHANSEN &
TOMMY SANDTORV JOHANNESSEN

SUPERVISORS
Lei Jiao, Vimala Nunavath, Morten Goodwin

University of Agder, 2018
Faculty of Engineering and Science

Abstract

Physical activity is a key factor in the treatment of chronic diseases such as diabetes, cardiovascular disease, and depression. Doctors and personal trainers have limited methods to accurately monitor and classify a patients actual activities based on training diaries and logs that are commonly used today. In this thesis, we apply a tri-axial accelerometer carried by a patient to collect data associated to different activities of daily life (ADL) and utilize deep learning (DL) algorithms for classifying distinct activities based on the data obtained from the accelerometer. Among various DL methods and algorithms, we adopt specifically deep neural networks (DNN) and recurrent neural networks (RNN) to classify movement patterns. In addition, we compare our proposed structures with the state-of-the-art methods via extensive experiments. Numerical results show that our proposed DNN model slightly exceeds, and our RNN model vastly outperforms the state-of-the-art methods in classification of basic movement patterns. The overall solution for data collection and movement classification provides medical doctors and trainers a promising way to precisely track and understand the physical activities of a patient for a better treatment.

Preface

This master's thesis is written for the Faculty of Engineering and Science at the University of Agder, Grimstad. It is part of the study program Engineering and ICT, and was written between August 2018 and January 2019.

We would like to thank our supervisors Associate Professor Lei Jiao, Associate Professor Morten Goodwin and Vimala Nunavath for their guidance and support throughout this project. Their enthusiasm towards our research motivated us to push ourselves to achieve our goals of completing our Master Thesis. We would also like to express our gratitude to Jahn Thomas Fidje, who helped us understand Recurrent Neural Networks better. Finally, we would like to thank our friends and families for encouraging and believing in us during the master thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

It is commonly known that physical inactivity is a risk factor for a variety of chronic diseases such as diabetes, cardiovascular disease, and depression [8, 10]. There are limited methods for doctors and personal trainers to monitor a patient/-customer's actual activities, where training diaries/logs are commonly used. However, questions around the accuracy and credibility of these diaries/logs are raised, as they intentionally, or unintentionally, may be subject to social desirability and recall bias [8].

The use of accelerometers to collect activities and movement patterns are rapidly increasing, where the raw data converted into activity count variables which is further used to classify physical activity intensity and energy expenditure [8]. Furthermore, supervised machine learning algorithms and probability models have been used to classify active traveling methods [8], and activities of daily life (*ADL*) [3] respectively, from the raw accelerometer data. These, state of the art algorithms and models are further described in chapter 2, section 2.3.

Deep learning architectures are commonly used for classifying complex data patterns. Thus, the main motivation of this thesis is to propose a deep learning approach for classifying physical activity movements using accelerometer data.

1.2 Thesis Definition

The main objective for this thesis is to propose a **Deep Learning classifier for motion recognition through wearable sensors**, where the research is divided into three goals and three research questions.

1.2.1 Thesis Goal

Goal 1: Examine state-of-the-art research within the field of recognizing "*Activities of Daily Life*" (ADL), and further improve it by introducing deep learning methods.

Goal 2: Propose and evaluate a *Deep Neural Network* model for time series classification.

Goal 3: Research *Recurrent Neural Network* models, and evaluate the performance of networks with "*memory*" compared to feed forward networks (*goal 2*).

1.2.2 Research Questions

In this section, we discuss and put forward the research question of interest which this thesis makes an effort to answer.

1. Are Deep Neural Network (DNN) classification models suitable for classifying movement from raw accelerometer data?

   To answer this research question, we have formulated subordinate questions designed to thoroughly answer the question. Furthermore, different distributions of the datasets are tested with the developed DNN model to increase the chance of correctly recognizing movement patterns.

   (a) Which combinations of hyper-parameters; optimizers and learning rates, performs better on a already researched dataset[1], wrist-worn dataset[2].

[1] The results of the previous reseach of this dataset is explained in *section 2.3*

[2] An explanation of this dataset is given in *section 3.1*

We are answering this question by testing different combinations of hyper-parameters, which can be found in chapter 4.

(b) Is it possible to correctly classify “unknown” accelerometer data when using a pre-trained deep neural classification model?

This question is answered by trained the models for 2000 training steps, then providing it with unseen data to classify.

(c) Does deep neural network perform better then state-of-the-art algorithms?

To answer this question we consider the result of the research questions above, and compare them to the state of the art results.

2. Are Recurrent Neural Networks (RNN) able to improve the classification of time series compared to simple feed forward DNN?

   We developed a RNN model, which is tested with the hyper-parameter combinations mentioned above, for each of the three sliding windows, to answer this research question.

3. How well does the proposed RNN model perform on a new custom made dataset?

   To answer this question, we collaborated with the health department of the University of Agder, Kristiansand, to collect movement patterns through hip-worn accelerometers. This dataset is dividend into training- and testing-data, which is used to train and test the developed RNN model.

1.3 Contribution

The main contributions of this thesis are:

- A DNN model to recognize movement patterns with comparable accuracies to other implemented recognition model, with minimal data-processing.

- A RNN model designed to recognize time series movement with minimal data-processing compared to previous research.
- Evaluation of the RNN model's performance on a researched dataset and a new dataset.

1.4 Thesis Outline

The structure of the remaining part of the thesis is structured in the following manner:

Chapter 2 provides preliminary information about the deep learning methods proposed in this thesis (*2.1, 2.2*). Furthermore, it provides a summary of the state-of-the-art classification models used for ADL recognition (*2.3*).

Chapter 3 explains the specification of the two proposed deep learning models, DNN (*3.2.1*) and RNN (*3.2.2*) respectively. In addition, an overview of the datasets (*3.1*) are given to provide information about the difficulties of the classification.

Chapter 4 contains experimental results for the algorithms presented in *chapter 3*.

Chapter 5 summarizes the work done in the thesis. In *section 5.1*, the research questions and goals are concluded. While *section 5.2*, outlines a possible roadmap for future research within the domain of movement recognition.

Chapter 2

Background and Related Work

This section starts with an introduction to the algorithms used to answer the research questions of this thesis. Furthermore, a short explanation of the prior research within the field of recognizing movement patterns is presented2.

2.1 Deep Neural Networks

2.1.1 Artificial Neural Networks

Neural networks, or artificial neural networks (ANN), are computational models that are loosely inspired by biological nervous systems such as the brain; A biological neuron fires an electrical signals when its chemical receptors are stimulated by neurotransmitters. A way to emulate this mathematically is to use the perceptron algorithm:

$$f(x) = \begin{cases} 1 & \text{if} \quad \sum_{i=1}^{n}(x_i \cdot w_i) + b > 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

Figure 2.1 illustrates how a perceptron takes a set of inputs $x_1, x_2, x_2, \ldots, x_n$, and a corresponding weight for each input $w_1, w_2, w_2, \ldots, w_n$ and calculates a

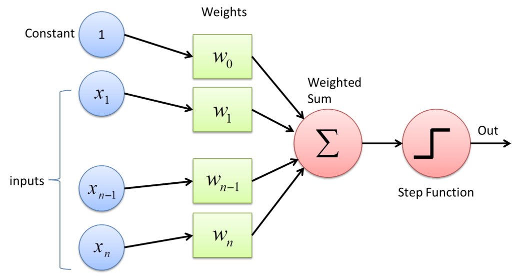


Figure 2.1: Illustration of a perceptron with binary step activation function

weighted sum. This sum is fed into a binary activation function (*Equation 2.1*) which will output zero or one.

A neural *network* is, as the name implies, a network of neurons and is usually structured into separate layers: Starting with the input layer the data is fed into the network and each layer will perform it's calculations and feed it forward to the next layer. This is known as a feed forward neural network, illustrated in *Figure 2.2*.

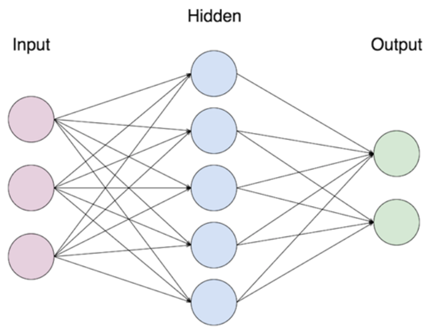


Figure 2.2: Illustration of a feed-forward neural network with a single hidden layer

2.1.2 Activation Function

When a neuron fires is dependant on the activation function. The perceptron illustrated in *Figure 2.1* uses the binary step function as defined in *Equation 2.1*. When the activation function is non-linear, then a two-layer neural network can be proven to be a universal function approximator [1, 4]. Thus, the purpose of the activation function is usually to bring non-linearity into the network, and therefore

Name	Function	Derivative
Identity	$f(x) = x$	$f'(x) = 1$
Sigmoid	$f(x) = \dfrac{1}{1+e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH	$f(x) = \dfrac{2}{1+e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
Binary Step	$f(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & x \neq 0 \\ ? & x = 0 \end{cases}$
ReLU	$f(x) = \begin{cases} 0 & x \leq 0 \\ x & x > 0 \end{cases}$	$f'(x) = \begin{cases} 0 & x \leq 0 \\ 1 & x > 0 \end{cases}$
Leaky ReLU	$f(x) = \begin{cases} \alpha x & x \leq 0 \\ x & x > 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & x \leq 0 \\ 1 & x > 0 \end{cases}$

Table 2.1: Activation functions and their derivatives.

it is natural to use non-linear activation functions in the hidden layers of an ANN [1].

Furthermore, a desirable property of an activation function is for it to be continuously differentiable. The binary step function described earlier is not differentiable at 0, and it differentiates to 0 for all other values. This means that gradient based algorithms can make no progress with it.

Interestingly, it is unknown why an activation function works better for a particular problem and thus trial and error is used to find the best fitting function [9]. The Rectified Linear Unit (ReLU) and the TanH function have worked quite well in ANN's, but there exists several other alternatives. [6, 1, 7] See *Table 2.1* for such examples and their derivatives.

2.1.3 Backpropagation and Optimizers

As mentioned previously, each cell has a bias and each connection between cells has a weight. The weighted sum plus the bias is used with the activation function and determines the output into the next layer. This continues until the last layer, which in case for classification, usually uses a softmax activation function which converts the output into vector whose values are between zero and one. This vector can be interpreted as the probability of some input being in a class equal to the respective component values y_i in the output vector.

The weights matrix is often initialized randomly using a Gaussian normal distribution and the bias vector usually to a small constant value. Naturally this means that the output probabilities are very bad, and for classifying n elements the accuracy will tend to $\frac{1}{n}$. To increase the accuracy the output vector is compared to the ground truth and the error between these two values is calculated. This error is then propagated backwards into the model. Each cell calculates it's gradient and the weights and biases are adjusted proportionally to their contribution to the error. This is known as backpropagation and is performed by the optimization algorithm.

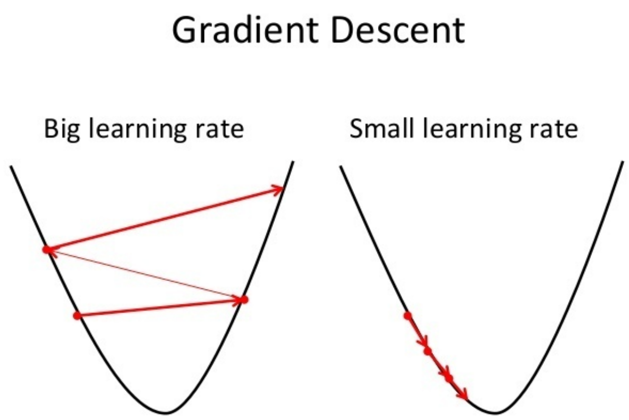


Figure 2.3: Illustration of small and large learning rates.

The optimization algorithm is in charge of propagating the error and updating the weights and biases of the model. Considering that the weights and biases are randomly initialized, it could take a very long time until they converge to a desirable value. A *learning rate* is therefore used to speed up the process of converging to these optimal values. *Figure 2.3* illustrates how a big learning rate cause drastic updates which lead to divergent behaviours. On the other hand, a small learning rate requires many updates before reaching the minimum point and thus extending the time before convergence.

2.1.4 Achieving Depth

The neural network illustrated in *Figure 2.2* has three layers; An input layer, a single hidden layer, and an output layer. As mentioned previously, each layer feeds data forward to the following layer using it's activation function.
A single hidden layer may be sufficient for simple experiments however, stacking these hidden layers gives the network the ability to learn highly complex and abstract information from the initial data [5]. A network that has multiple hidden layers is often regarded as a *Deep* Neural Network (DNN).

2.2 Recurrent Neural Network

In traditional neural networks, there is an underlying assumption that the inputs are all independent from each other. This is not always the case, specifically in sequential information. For instance, to predict the next word in a sentence requires knowledge of the previous words and maybe even the topic of the discussion. Due to the structure of feed-forward networks, the information flows through the network in a single direction. The information never touches a node twice, as such the network has no memory of the input that is received previously and no notion of order in time.

In a recurrent neural network the information cycles through a loop. That is, the network performs the same task for every element of a sequence. The output in such a system is dependent on the previous computations, and every decision takes not only the current input into consideration, but also the output of the previous iteration.

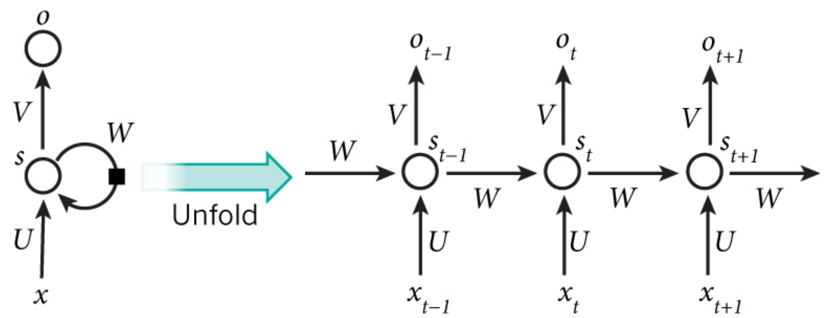


Figure 2.4: Illustration of a recurrent neural network and it's unfolding in time.

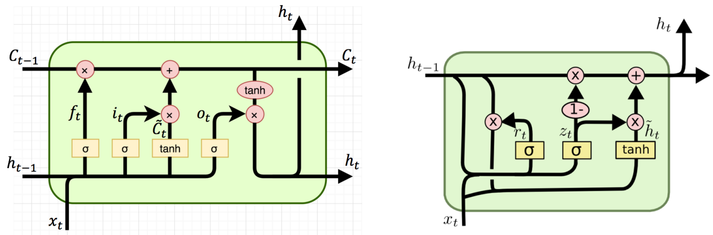
Figure 2.4[1] illustrates a recurrent neural network: The left side is the closed short-hand form of an RNN, while the right hand side shows how the network is unrolled for each time step. Here x_t is the input and o_t is the output at time t .
The hidden state s_t, sometimes noted as h_t, is the state at time step t and can be regarded as the "memory" of the network. It captures the information of the previous time steps, and so basic RNN's use this as short term memory, as it cannot capture information for more than a few iterations. This is limited because of the back-propagation and how the error becomes increasingly smaller for each time step. In order to calculate the first hidden state s_{-1} is typically initialized to all zeroes.

2.2.1 Long Short-Term Memory and Gated Recurrent Units

As mentioned previously, RNN's have the ability to remember information, however not for very long. As the time steps increase, it fails to derive context from time steps which are far behind. By extending the cells to have a specific memory channel one can give them the ability to learn for longer periods of time. Each cell decides to store or delete information from this memory based on the importance of the information. What is important or not is dependant on the weights, thus

[1]This illustration is found at: wildml.com

as the network trains over time it will learn which information is important and which is not. These cells are called Long Short-Term Memory cells (LSTM).



(a) Long Short-Term Memory (b) Gated Recurrent Unit

Figure 2.5: Illustration of a LSTM and GRU cells.

The LSTM cells are computationally expensive as they do many operations to decide what to remember and what to forget. To reduce training time Gated Recurrent Units (GRU) cells can be used. The GRU cells control the flow of information like the LSTM unit, but rather than deciding what to remember and what to forget they expose the full hidden content without any control. This makes them much more efficient and is illustrated in *Figure 2.5*[2].

[2]This illustration is found at: isaacchanghau.github.io

2.3 State-of-the-art

Prior research within the field of recognizing Activities of Daily Life (*ADL*), uses data collected by wearable sensor devices. More accurately a single wrist-mounted tri-axial accelerometer [2]. The previous research divides the categories into 14 low-level activities called motion primitives. These primitives are uniquely related to five major ADL categories shown in table 3.1. Multiple motion primitives can refer to the same ADL, thus it is possible to infer the activity when recognizing the primitive.

The system architecture of the proposed solution in [3], consist of two distinct modules, a model builder and a classifier. The model builder creates a probabilistic models of the relevant motion primitives extracted from the recordings, while the classifier uses a comparison of new, unseen, acceleration data with the available models. Using Gaussian Mixture Modeling and Gaussian Mixture Regression the previous research achieved promising results for very specific primitives. Table 2.2 from Bruno et al. (2013) [3] illustrates the results.

Model	**TP**	**TN**
Climbing the stairs	20 %	93.34 %
Drinking from a glass	100 %	83.34 %
Getting up from the bed	60 %	66.67 %
Pouring water in a glass	100 %	80 %
Sitting down on a chair	0 %	93.34 %
Standing up from a chair	60 %	83.34 %
Walking	40 %	70 %

Table 2.2: Recognition accuracy focusing on getting up from bed

It is important to note that [3] calculates a model for each movement category and the results shown in 2.2 are incomplete. Furthermore, using True Positives (TP) and True Negatives (TN) as the performance metrics can be misleading. For instance, the model *"Sitting down on a chair"* has zero score in TP, but a score of 93.34 in TN. Any model that always classifies false would achieve similar results.

This thesis will try to achieve greater results than shown in [3] and metrics like Recall, Precision, and F1-score are used in this thesis as they give a clearer picture of performance.

Chapter 3

Dataset and Model Structure

This chapter gives a description of the two datasets which are used for the experiments. Additionally, we are giving an explanation of algorithms which are implemented for recognizing the different movement patterns of the datasets.

3.1 Dataset and Data Structure

This thesis applies the proposed deep learning recognition models on two different movement datasets, where the first one is an existing dataset and the second one is a newly collected dataset.

Dataset 1: A public collection of labelled accelerometer data recordings aquired from UCI Machine learning Repository. It is a dataset for activities-of-dailylife (**ADL**) collected through wrist-worn accelerometers, here by referred to as the wrist-worn dataset.

Dataset 2: A dataset collected by participants associated with the University of Agder, Kristiansand. It is a dataset of labled movement patterns collected through a hip-worn accelerometers, here by referred to as the hip-worn dataset.

3.1.1 Wrist-Worn Dataset

The wrist-worn dataset which is used throughout the experiments consists of 14 different motion primitives performed by 16 volunteers. To test and examine the performance of the proposed algorithms these motion primitives are divided into broader, less complex, ADL categories. The correlation between these ADL categories and the motion primitives is shown in *table 3.1*.

ADL	**Motion Primitives**
Personal hygiene	Brush teeth
	Comb Hair
Mobility	Climb Stairs
	Descend Stairs
	Walk
Feeding	Drink from glass
	Pour water into glass
	Eat with knife and fork
	Eat with spoon
Communication	Use telephone
Functional transfers	Get out of bed
	Lie down in bed
	Stand up from chair
	Sit down on chair

Table 3.1: Correlation between ADL categories and Motion primitives

The distribution of the data-points for these ADL's and motion primitives are shown in *figure 3.1*, where the ADL distribution is on the left and the motion primitives on the right [1] [2].

Looking at the ADL distribution it is clear that it is not evenly distributed as three categories; mobility, feeding and functional transfers, consist of above 100

[1] The indexes of the movement categories in the figure correlates to the categories as shown in *table B.1* and *B.2*, in *appendix A*.

[2] All images in *section 3.1.1* shows the distribution of ADL categories on the left and Motion primitives on the right.

000 data-points each. While personal hygiene and communication only consists of about 50 000 and 20 000 data-points respectively.

The overall distribution of the movement primitives can be considered somewhat even; with the exceptions of walking, eating with spoon, and lying down, as most of them are within the range of 20 000 to 40 000 data-points.

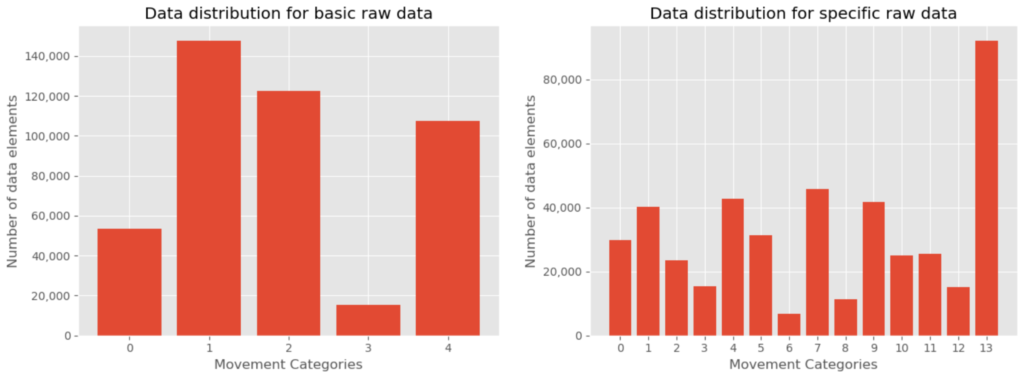


Figure 3.1: Raw data distribution for the wrist-worn dataset

Preparing the raw data for the experiments is done by dividing the data into sliding windows. The accelerometers used when gathering the data, collects 32 data-points each second. Three types of sliding windows are used in the experiments; a three second-, a five second- and a ten second- sliding window.

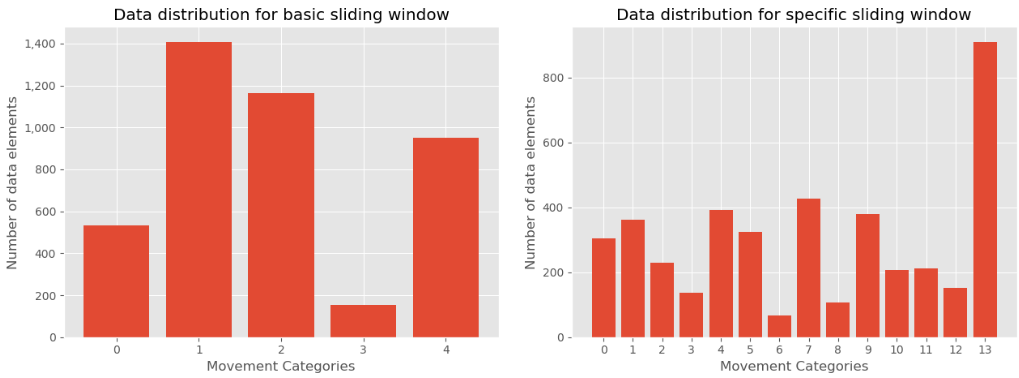


Figure 3.2: Three second sliding window distribution for the wrist-worn dataset

The three second sliding window differs from the other two sliding windows, as the shift for each window also is set to three seconds. It is created by collecting three seconds of data - which is a total of 96 data-points - then sliding three seconds to create the next sliding window. Thus, there is no overlapping of data-points within this type of sliding window. Dividing the dataset into these three second sliding windows, yields a dataset consisting of approximately 4000 sliding windows distributed as shown in *figure 3.2* [1].

When generating the five second sliding windows, the shift for each window is set to one second. Thus, each sliding window contains an overlap of four seconds to the previous window. This yields a dataset of approximately 10 000 sliding windows, with the distribution shown in *figure 3.3* [1].

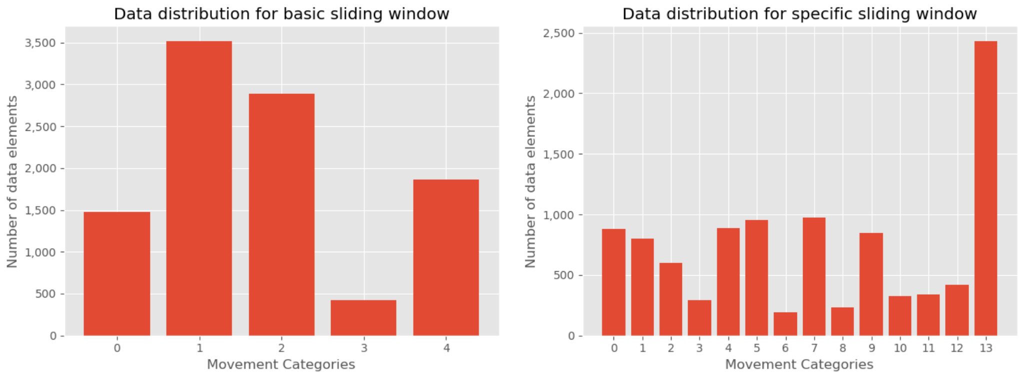


Figure 3.3: Five second sliding window distribution for the wrist-worn dataset

The final type of sliding window is ten seconds of data, also sliding by one second for each window. This creates a dataset consisting of approximately 6000 sliding windows, and its distribution is shown in *figure 3.4* [1].

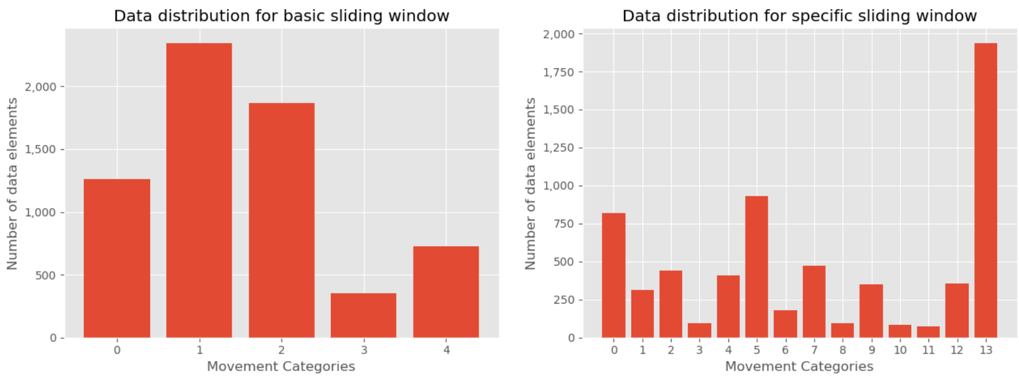


Figure 3.4: Ten second sliding window distribution for the wrist-worn dataset

3.1.2 Hip-Worn Dataset

The hip-worn dataset is consists of ten movement patterns performed by eight volunteers, where some of the movement patterns are different variations of a movement, such as different speeds of walking. The collected movement patterns are cycling, jogging, laying still, sitting, sitting in a vehicle, sitting relaxed, walking stairs, standing, walking fast, and walking normal. This dataset consists of 1.6 million data-points distributed among the categories as shown in *figure 3.5* [3].

Similar to the wrist-worn dataset, the hip-worn dataset is divided into three types of sliding windows; three-, five-, and ten second windows respectively. The hip-worn dataset in unevenly distributed, as most of the data-points are in the relaxing categories such as lying down, and sitting. This pattern in the distribution is carried over to the sliding window datasets.

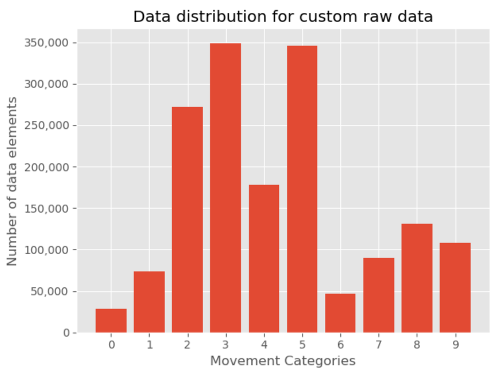


Figure 3.5: Raw data distribution for the hip-worn dataset

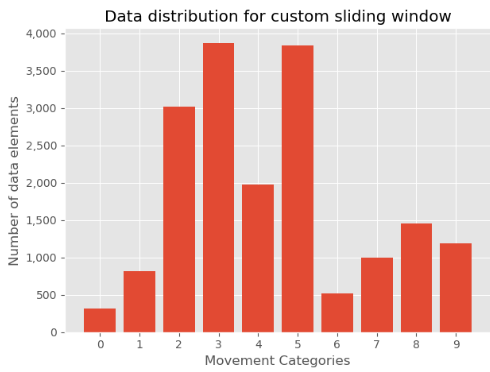


Figure 3.6: Three second sliding window distribution for the hip-worn dataset

Generating the dataset for three second sliding windows is done the same as for the wrist-worn, three seconds of data shifting by three seconds. However, the accelerometers used on this dataset collects 30 data-points each second, giving the three second sliding window dataset around 15 000 sliding windows with the distribution shown in *figure 3.6* [3].

The five- and ten- second sliding window datasets are created with a shift of one second, the same as for the wirst-worn dataset. Both types of sliding window

[3]. The indexes of the movement categories in the figure correlates to the categories as shown in *table A.1*, in *appendix A*

creates a dataset with approximately 54 000 sliding windows, where the distribution for the five seconds are shown in figure 3.7 [3], and for the ten seconds are shown in figure 3.8 [3].

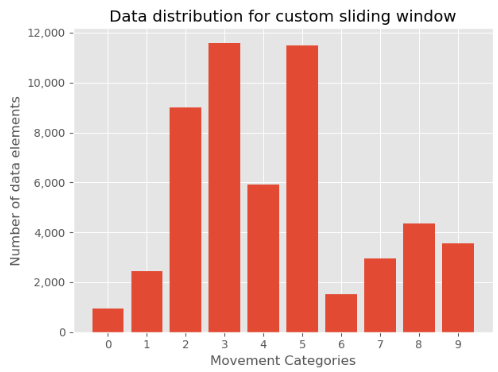


Figure 3.7: Five second sliding window distribution for the hip-worn dataset

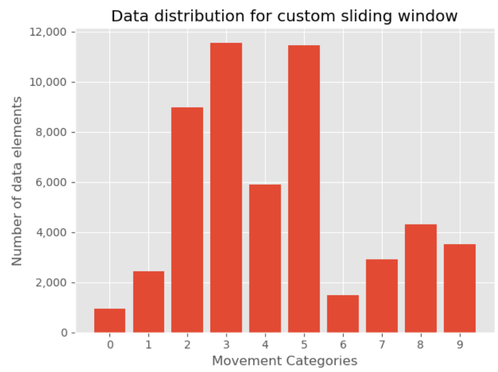


Figure 3.8: Ten second sliding window distribution for the hip-worn dataset

3.2 Model Structure

In this section we describe the experimental setup for the two implemented algorithms for recognizing the movement types describe in the previous section[4].

3.2.1 Deep Neural Network Model

The implemented model for the deep neural network consists of five layers; An input and output layer, and three hidden layers. The three hidden layers have a constant number of cells, more specifically, the first hidden layer has 512 cells, the second hidden layer has 258 cells, and the last hidden layer has 128 cells. While the input and the output layer adapt to the dataset; The input layer will always fit the length of the samples, while the output layer will always have n cells, where n is the number of classes that need to be classified.

To evaluate performance of the model, the output of the network is compared with the true label and the different performance metrics; accuracy, recall, precision,

[4]Code available at GitHub: https://github.com/SahandJ/MovementRecognition

and f1 score, are calculated. A simplified illustration of the model is shown in *figure 3.9*

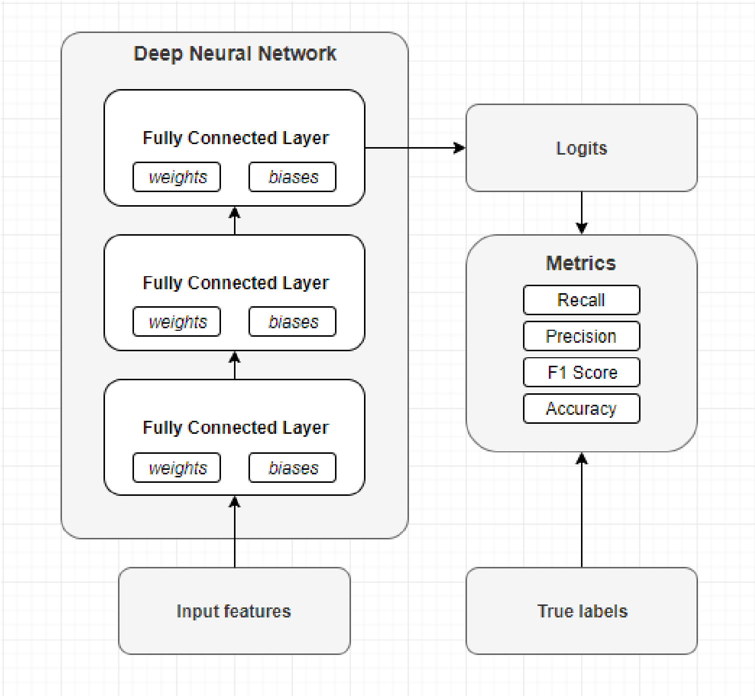


Figure 3.9: The structure of the Deep Neural Network model

During the training phase, the input data is split up into smaller batches of 100 elements. The model will predict a label for each element and the weights and biases are adjusted from the sum of the errors between the predicted labels and true labels. The testing dataset is not split up into smaller batches. Thus, the entire testing dataset is used for calculating the performance metrics.

3.2.2 Recurrent Neural Network Model

The second implemented model is reccurent neural network which consists of two components; the recurrent element of the network and a fully connected network.

The first component, the recurrent element, consists of cells which in our case are Gated Recurrent Unit's (**GRU**). Each of these cells are connected to each other and they all have a fixed number of states. The number of cells, and the number of states each cell has is provided as a parameter to the network. The recurrent element of the network returns two values; The current state of the [recurrent] network S_t, and the output of each state O_t. The output of each state will be an array of length t where t is the number of items in the timeseries that is sent into the network.

The last element of the outputs is used and sent forwarded to the fully connected network. The fully connected network is much smaller than the previous model and only consists of two layers. The input layer, which takes the input from the recurrent network and an output layer that is used to predict the class. A simplified illustration of the recurrent neural network model is shown in *figure 3.10*.

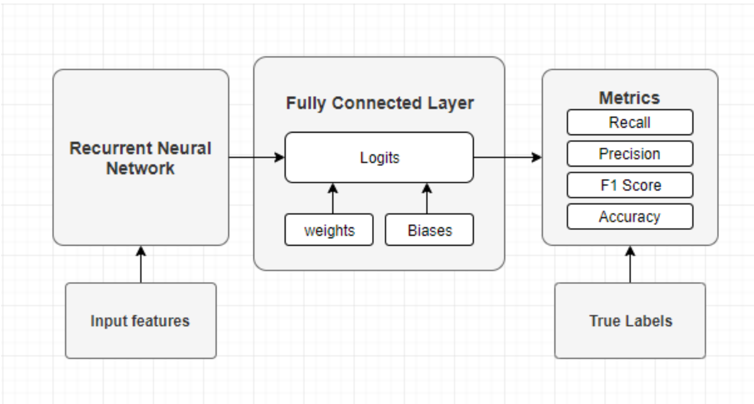


Figure 3.10: The structure of the Recurrent Neural Network model

Chapter 4

Experiments and Results

The dataset provided by *UCI Machine Learning Reporistory*, which we refer to as the wrist-worn dataset, is thoroughly tested for the proposed deep learning models, the *Deep Neural Network* (**DNN**) model and a *Reccurent Neural Network* (**RNN**) model respectively.

Four different learning rates are used; *0.1*, *0.01*, *0.001* and *0.0001*. These are tested in combination with three well known optimizers; *Adagrad*, *Adam* and *Stochastic Gradient Descent* (**SGD**). Furthermore, these combinations of optimizers and learning rates are tested for all three types of sliding windows; three-, five- and ten- seconds, both for the basic- and specific- movement categories. Considering the number of data-points available in the datasets, all experiments are stopped after 2000 training steps. Thus, reducing the chance of overfitting through repeatedly training on the same data.

Furthermore, the hip-worn dataset, provided by UiA Kristiansand, is tested using only the proposed *RNN* model, with the different hyper-parameter combinations discussed above. The decision to not run this dataset through the proposed *DNN* model is the fact that the results of the wrist-worn dataset shows that the *RNN* model consistently outperforms the *DNN* for this type of time series movement patterns.

4.1 Deep Neural Network

Throughout this section a detailed explanation of the best performing results, with regards to the overall F1 score [1], is given. For each type of sliding window - three-, five- and ten- seconds - the best performing combination of optimizer and learning rate will be given for both the basic- and specific- movement categories.

As mentioned, multiple combinations of hyper-parameters are tested for the UCI dataset. The overall results of all the tested combinations are show in the appendices, *appendix B.1* for the basic movement categories, and *appendix B.2* for the specific movement types.

Looking at these result overviews, low learning rates are shown to be more suited for movement recognition, for both the Adagrad- and the Adam- optimizer. When using a high learning rate of 0.1, both the Adagrad and Adam optimizer get a low F1 score. The recall of these results shows that the algorithms guesses “randomly”. The probable cause of this might be the high learning rate. Thus, the algorithm diverges from the global minimum, which prevents it from learning the patterns of the movements.

Furthermore, the result of the SGD optimizer is to some extent unexpected. However, the result can be explained through the proposed structure of the DNN. The model uses ReLU as its activation function. Using ReLU with SGD often leads to vanishing gradients, which stagnates the learning of the algorithm.

4.1.1 Wrist-Worn Dataset: Basic Movement

Three second sliding window:

Testing the different hyper-parameter combinations on the three second sliding window, shows the highest achieved overall accuracy and F1 score are 80.94% and 78.46% respectively. This result is reached when using a low learning rate, 0.0001, with the Adam optimizer.

[1]F1 score is defined as: 2 * ((Precision * Recall) / (Precision + Recall))

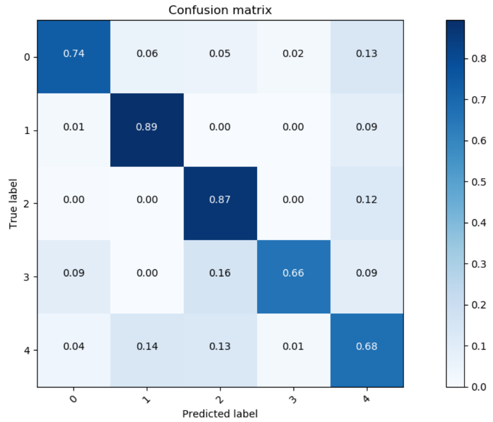


Figure 4.1: Confusion matrix of the best result for basic movement three second window using *DNN*

	Category	**Recall**	**Precision**	**F1 Score**
0	Hygiene	73.7%	83.61%	78.35%
1	Mobility	89.33%	88.7%	89.01%
2	Feeding	87.13%	84.44%	85.76%
3	Communication	66.23%	79.69%	72.34%
4	F-Transfer	67.57%	64.6%	66.05%

Table 4.1: Best result of basic movement three second window using *DNN*

Figure 4.1 shows a confusion matrix for the results of each movement type. This matrix is created to evaluate the algorithm, and interpret why the algorithm achieved its results and where the incorrect classifications occurs. *Table 4.1* adds additional information; the recall-, precision- and F1- percentages, about the classification of the movement types.

The classification inaccuracies for three seconds sliding windows are understandable considering the data distribution shown in *figure 3.2*. The most distinct pattern in the matrix is that each category is to some extent inaccurately recognized as a functional transfer, even without it being the category with the largest amount of data-points.

However, the probable cause of this pattern is the fact that the data is collected through a wrist-worn accelerometer; Each of the 14 participants might have different personal traits when moving, especially hand movements. This could affect the classification. Thus, giving functional transfers a lower precision score [2], shown in *table 4.1*.

The ADL category which is inaccurately classified the most is the communication category, which contains the least amount of data-points. Furthermore, when it is incorrectly classified it is often interpreted as similar movement patterns, either hygiene or feeding. Functional transfers is the category with the second highest

[2]Precision score is defined as: True Positives / (True Positives + False Positives).

percentage of inaccurate classification. Again, some of these classification might be affected by traits of the participants, thus classifying it as feeding. On the other hand, functional transfers are often classified as mobility, which is a more comparable category with regards to the movement types in those categories.

Five second sliding window:

Using the Adam optimizer combined with a low learning rate of 0.0001, on the five second sliding windows, the proposed *DNN* achieves an overall accuracy of 86.01% and an overall F1 score of 82.37%.

There are clear patterns in the inaccurate classifications of the ADL categories shown in the confusion matrix in *figure 4.2*.

The *DNN* model incorrectly classifies the communication data as feeding (*20%* of the time), similar to the three second sliding window. Considering the resemblance in the movement patterns when using a telephone and eating, this confusion is understandable, especially when seeing the difference in the data distribution, shown in *figure 3.3*.

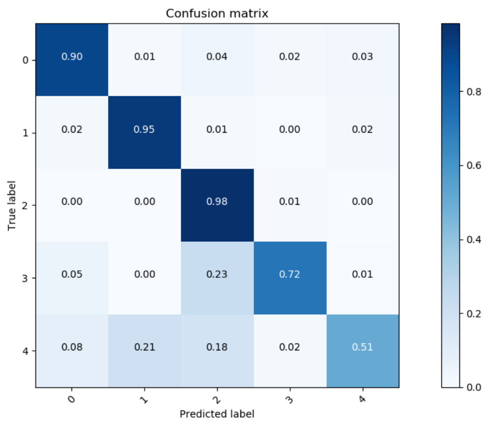


Figure 4.2: Confusion matrix of the best result for basic movement five second window using *DNN*

	Category	Recall	Precision	F1 Score
0	Hygiene	89.69%	84.33%	86.93%
1	Mobility	94.53%	89.38%	91.88%
2	Feeding	98.37%	83.69%	90.43%
3	Communication	71.72%	72.82%	72.26%
4	F-Transfer	50.54%	88.89%	64.44%

Table 4.2: Best result of basic movement five second window using *DNN*

Explaining the miss-classification of the functional transfer category is partially based on assumptions. The two most commonly classified classes are mobility and feeding. As mentioned for the three second slidining window, the specific movement types in the mobility class and the functional transfer class (see table 3.1) are comparable. It is understandable that getting up from a chair or the bed is interpreted as one of the mobility movements. However, looking at possible reasons for the *DNN* to interpret functional transfers as feeding, the most reasonable explanation would, as mentioned, be the placement of the accelerometer.

Ten second sliding window:

The Adam optimizer and a low learning rate of 0.0001 is again the best performing combination when testing it on ten second sliding windows. Achieving an impressive overall accuracy of 92.4% and an overall F1 score of 89.43%.

Looking at the confusion matrix in *figure 4.3*, it is clear to see that a ten second sliding window allows the *DNN* model to observe and distinguish the differences between these basic movement patterns. The probable cause of the lower F1 score of communication is its low number of data-points; shown in the data distribution in *figure 3.4*, which influences its precision score. Thus, predicting 5% of the hygiene data-points as communication heavily affects the precision score.

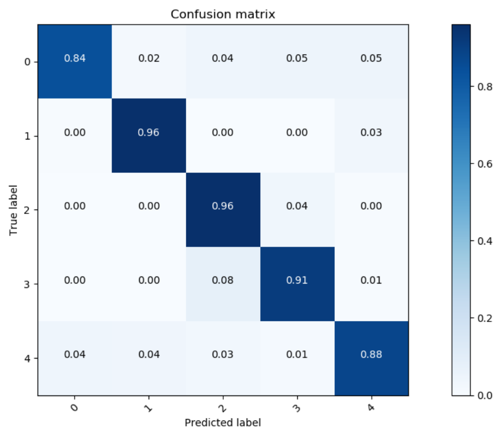


Figure 4.3: Confusion matrix of the best result for basic movement ten second window using *DNN*

	Category	Recall	Precision	F1 Score
0	Hygiene	84.1%	96.49%	89.87%
1	Mobility	95.95%	97.51%	96.72%
2	Feeding	95.87%	93.67%	94.75%
3	Communication	91.3%	70.59%	79.62%
4	F-Transfer	87.74%	81.4%	84.45%

Table 4.3: Best result of basic movement ten second window using *DNN*

4.1.2 Wrist-Worn Dataset: Specific Movement

As the number of movement types is increased in the specific movement distribution of the UCI dataset, the complexity of recognizing them also increases. In addition, the distribution of data-points is lower for each categories[3] as they are no longer combined as for the basic distribution.

Three second sliding window:

The best result achieved for three second sliding windows for specific movement is an accuracy of 59.93% and a F1 score of 56.59%. This result is achieved by the Adagrad optimizer with a learning rate of 0.01.

Analyzing the confusion matrix in *figure 4.4*, the most difficult movement to recognize is lying down in bed. Considering that it is the movement with the second lowest amount of sliding windows, see the distribution in *figure 3.2*, it is to some extent expected.

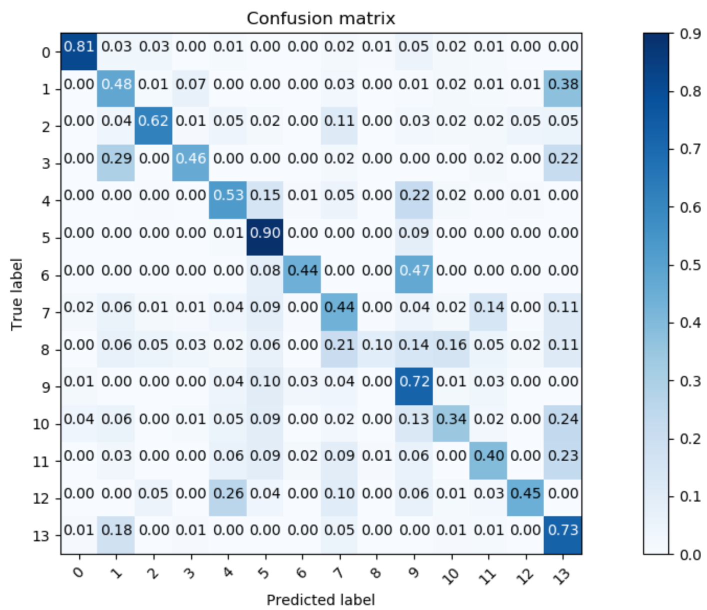


Figure 4.4: Confusion matrix of the best result for specific movement three second window using *DNN*

[3] See the distributions in the figures in section 3.1.1

	Category	Recall	Precision	F1 Score
0	Brush teeth	81.29%	90.0%	85.42%
1	Climb stairs	48.07%	39.55%	43.39%
2	Comb hair	61.82%	79.07%	69.39%
3	Descend stairs	46.15%	57.69%	51.28%
4	Drink	52.74%	64.63%	58.08%
5	Eat w/ knife & fork	90.0%	60.71%	72.51%
6	Eat w/ spoon	44.44%	59.26%	50.79%
7	Get out bed	43.75%	49.46%	46.43%
8	Lie down bed	9.52%	66.67%	16.67%
9	Pour water	72.14%	52.73%	60.92%
10	Sit down	34.12%	46.03%	39.19%
11	Stand up	39.58%	39.58%	39.58%
12	Telephone	44.87%	77.78%	56.91%
13	walk	72.75%	67.41%	69.98%

Table 4.4: Best result of specific movement three second window using *DNN*

The other reasonable miss-classified movement patterns are full body movements. Both climbing and descending stairs are often wrongly interpreted as walking, which is understandable as walking is the largest category in terms of data-points and the movements are comparable. In addition, the two functional transfers, sitting down and standing up, are also interpreted as walking.

Furthermore, descending stairs is either wrongly recognized as either walking or descending stairs. The accelerometer used collects the g-forces, and as it is placed on the wrist. Thus, the patterns on climbing and descending stairs are relatively equal as the axes changes when rotating the hand.

Another conspicuous miss-interpretation is when the algorithm recognizes eating with spoon as pouring water into a glass. The assumption here is that the collected data of pouring water is from a mug. Thus, the resemblance in hand movements affects the algorithm. As an example, when eating soup one slowly moves the spoon from the plate, up towards the mouth before one tilts the spoon inside the mouth. The same pattern goes for pouring water from a mug into a glass. First the mug is lifted, then tilted to pour the water.

Five second sliding window:

Testing the DNN on the five second sliding windows, the highest performing hyper-parameter combination is the Adagrad optimizer and a learning rate of 0.01. Achieving an overall accuracy of 67.83% and an overall F1 score of 62.11%.

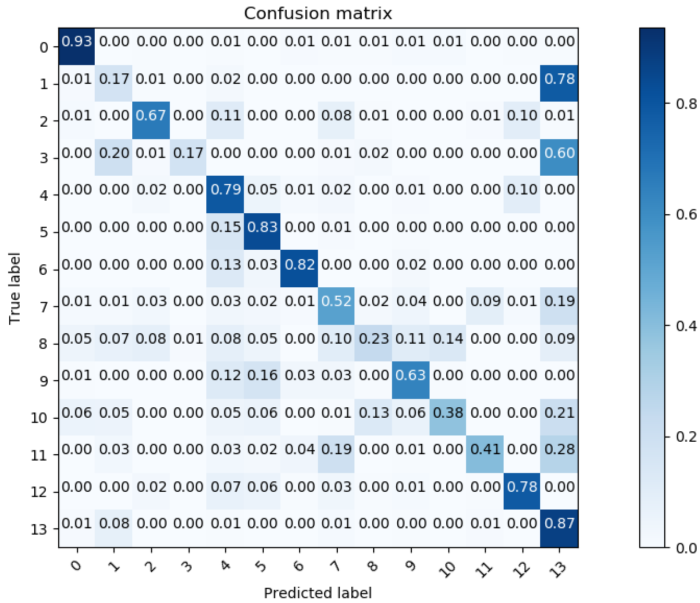


Figure 4.5: Confusion matrix of the best result for specific movement five second window using *DNN*

	Category	Recall	Precision	F1 Score
0	Brush teeth	93.49%	91.56%	92.52%
1	Climb stairs	16.95%	32.72%	22.33%
2	Comb hair	66.78%	82.08%	73.64%
3	Descend stairs	17.29%	79.31%	28.4%
4	Drink	79.06%	59.97%	68.2%
5	Eat w/ knife & fork	83.3%	74.58%	78.7%
6	Eat w/ spoon	81.91%	68.75%	74.76%
7	Get out bed	51.93%	68.18%	58.96%
8	Lie down bed	23.42%	37.68%	28.89%
9	Pour water	63.38%	80.12%	70.77%
10	Sit down	38.06%	68.6%	48.96%
11	Stand up	40.67%	49.19%	44.53%
12	Telephone	78.26%	66.12%	71.68%
13	walk	87.47%	64.0%	73.91%

Table 4.5: Best result of specific movement five second window using *DNN*

The results of the five second sliding windows are similar to the results of the three second sliding windows, where most of the movement categories have a slight increase in their recall score. As mentioned in *section 3.1.1*, each five second sliding window consists of some overlapping data. Allowing the algorithm to more easily recognize the patterns of the movements.

However, the movements which were miss-interpreted as walking are more frequently interpreted incorrect. This, pattern has a correlation to the distribution; see *figure 3.3*, as these movement categories, climbing and descending stairs, have fewer sliding windows for five seconds than for three seconds. Thus, the algorithm has fewer samples to train on, which affects the result.

Ten second sliding window:

For ten second sliding windows, Adagrad combined with a learning rate of 0.01 gives an overall accuracy of 81.29% and an F1 score of 66.75%

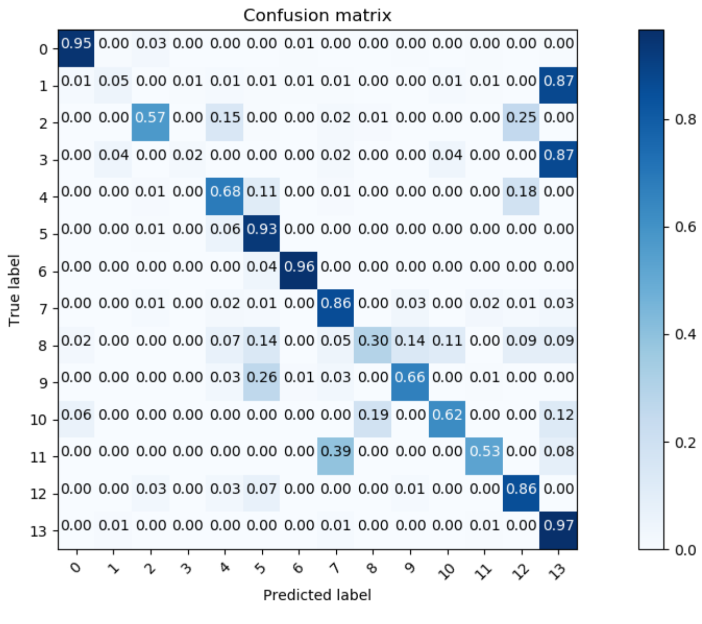


Figure 4.6: Confusion matrix of the best result for specific movement ten second window using *DNN*

Looking at the results shown in the confusion matrix in *figure 4.5*, it follows the same patterns as for the five second sliding window. The recall scores averagely increases, due to the increase of the window size. As the window size increase, the amount of sliding widows decreases for the climbing- and descending- stairs

	Category	Recall	Precision	F1 Score
0	Brush teeth	94.77%	98.82%	96.75%
1	Climb stairs	4.9%	35.0%	8.59%
2	Comb hair	57.26%	84.81%	68.37%
3	Descend stairs	2.17%	33.33%	4.08%
4	Drink	68.28%	60.77%	64.3%
5	Eat w/ knife & fork	93.01%	82.06%	87.19%
6	Eat w/ spoon	95.79%	93.81%	94.79%
7	Get out bed	85.66%	83.27%	84.44%
8	Lie down bed	29.55%	61.9%	40.0%
9	Pour water	66.48%	85.21%	74.69%
10	Sit down	62.5%	60.61%	61.54%
11	Stand up	52.78%	55.88%	54.29%
12	Telephone	85.54%	58.44%	69.44%
13	walk	96.55%	83.3%	89.44%

Table 4.6: Best result of specific movement ten second window using *DNN*

movements. Thus, further decreasing their recall score, as they are more consistently interpreted as walking.

The most distinguish pattern in the ten second sliding window confusion matrix, is the miss-interpretation of standing up. It is approximately 40% of the time interpreted as getting out of bed. Indicating that the similarity of getting out of the bed, and standing up from a chair is quiet high when monitoring ten seconds of those movements.

4.2 Recurrent Neural Network

In this section we discuss the best results[4] of the Recurrent Neural Network (**RNN**) experiments. As for the DNN, a detailed explanation of the best combination of hyper-parameters is given for each sliding window, for both the basic- and specific- movement categories. In addition, the results for the secondary dataset, the hip-worn data, are explained.

The hyper-parameters are the same as for the DNN. However, two additional parameters are tested; Number of cells in the RNN, and the size of the cells. The RNN is run with four and eight cells, for each of the learning rates. Furthermore, these were both tested with two different sizes for the cells, 16 and 32 respectively.

Looking at the result overviews in *appendix C*, lower learning rates often perform better than higher. However, Adagrad performs surprisingly bad with the lowest learning rate. The assumption here is that the number of training steps are to low for the Adagrad optimizer to learn the patterns with such a low learning rate.

The same pattern for the SGD optimizer are shown for the RNN; SGD does not learn with the ReLU cells, except it is able to achieve relatively impressive results with the lowest learning rate.

4.2.1 Wrist-Worn Dataset: Basic Movement

In this section the best performing results with regards to the overall F1 score for the basic movements, for each sliding window type, are explained.

Three second sliding window:

Running a RNN, consisting eight cells of size 32, with the Adam optimizer and a low learning rate of 0.001, an overall accuracy of 84.89% and an overall F1 score of 82.56% is achieved. Comparing these results to the DNN results[5], it is an in-

[4] The best result is decided by the overall f1 score
[5] The dnn results are discussed in section 4.1.1

crease of approximately 4%.

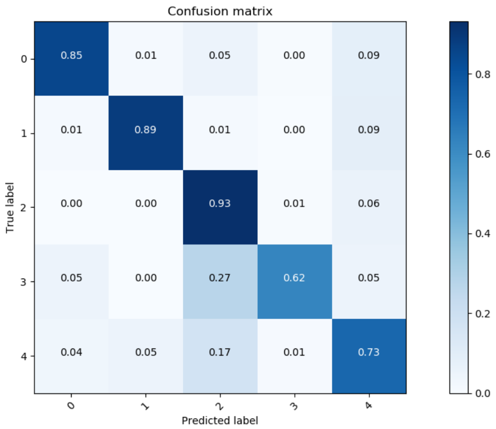


Figure 4.7: Confusion matrix of the best result for basic movement three second window using *RNN*

	Category	Recall	Precision	F1 Score
0	Hygiene	84.81%	88.42%	86.58%
1	Mobility	88.62%	95.61%	91.98%
2	Feeding	93.12%	80.86%	86.56%
3	Communication	62.34%	85.71%	72.18%
4	F-Transfer	73.22%	73.53%	73.38%

Table 4.7: Best result of basic movement three second window using *DNN*

Looking at the confusion matrix in *figure 4.7*, the results again shows that whenever a sliding window is miss-interpreted it is often guessed as a hand movement. As an example, the category with the lowest amount of sliding windows, communication, is often guessed as feeding. This is understandable because of the similarity in movements, and also due to the fact that feeding is the category with the second most sliding windows, see distribution in *figure 3.2*.

The other noticeable result is the interpretation of functional transfers. As discussed in previous sections, the placement of the accelerometer can be used as an explanation. Placing it at the wrist is probably affecting the movement pattern, as a small hand gesture can have an influence on the algorithms. Thus, the assumption is that functional transfers is classified as feeding due to the accelerometer placement.

Five second sliding window:

Using a learning rate of 0.01 and the Adagrad optimizer with the five second sliding windows, achieves an impressive 94.65% overall accuracy and a 93.03% overall F1 score.

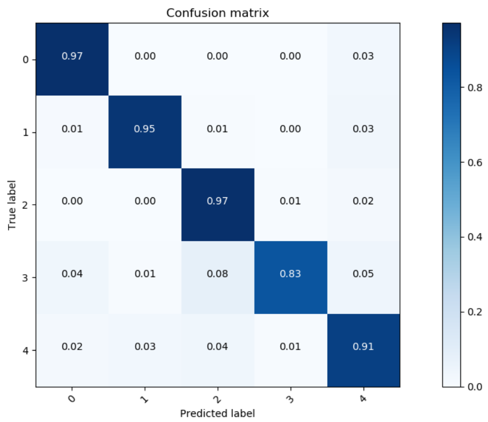


Figure 4.8: Confusion matrix of the best result for basic movement five second window using *RNN*

	Category	Recall	Precision	F1 Score
0	Hygiene	96.95%	93.73%	95.31%
1	Mobility	94.86%	98.36%	96.58%
2	Feeding	97.16%	95.66%	96.41%
3	Communication	83.33%	91.67%	87.3%
4	F-Transfer	90.85%	87.79%	89.29%

Table 4.8: Best result of basic movement five second window using *RNN*

The only result standing out in the confusion matrix in *figure 4.8*, is the communication category. However, the "low" accuracy is explained by looking at the distribution[6]. Consisting of the fewest sliding windows, communication is expected to be the hardest class to predict for the RNN.

Ten second sliding window:

The highest performing combination of hyper-parameters for ten second sliding windows, is also the highest performing result for basic movements in general. Combining the Adam optimizer with a learning rate of 0.001, in a RNN with 4 cells of size 32, an overall accuracy of 98.75% and an overall F1 score of 98.06% is achieved.

Considering the overlapping of data-point in the sliding windows, and the fact that RNN uses prior knowledge to improve. These results are expected. The few miss-interpreted sliding windows is assumed to be caused by the "noise"[7] from the placement of the accelerometer.

[6] The Communication category consists of the fewest sliding windows, see *figure 3.3*

[7] As mentioned individual hand-gestures from participants during data collection might affect the data patterns, we consider this as "noise"

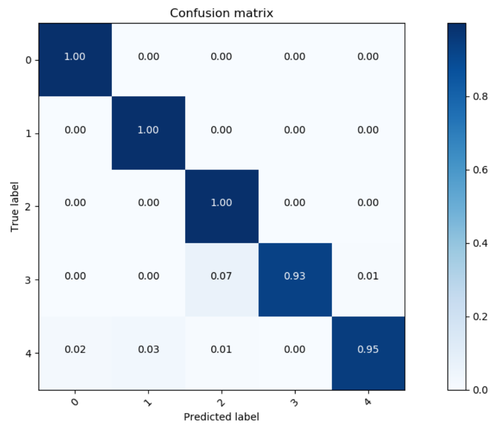


Figure 4.9: Confusion matrix of the best result for basic movement ten second window using *RNN*

	Category	Recall	Precision	F1 Score
0	Hygiene	99.85%	98.79%	99.32%
1	Mobility	99.58%	99.08%	99.33%
2	Feeding	99.66%	98.35%	99.0%
3	Communication	92.93%	99.42%	96.07%
4	F-Transfer	94.71%	98.27%	96.45%

Table 4.9: Best result of basic movement ten second window using *RNN*

4.2.2 Wrist-Worn Dataset: Specific Movement

Throughout this section we explain and discuss the results of the best performing hyper-parameters for all three types of sliding windows for specific movement.

Three second sliding window:

The results of the RNN for three second sliding window, are noticeably better than for the DNN. A RNN with four cells, with size 32, a learning rate of 0.001 and the Adam optimizer, gives an overall accuracy of 70.39% and an overall F1 score of 65.58%.

One of the main differences between the results of the RNN and the DNN, are both descending and climbing the stairs. *Figure 4.10* shows that the RNN have reduced the number of miss-interpretations of stair movements as walking. Thus, we assume that the "memory" of the RNN are able to remember the small differences between walking and moving upwards or downwards.

Many of the miss-interpreted movements are movements of similar types, mostly different hand movements. Thus, some sliding windows are interpreted incorrectly as another hand-movement. Examples of such miss-interpretations are: drinking interpreted as pouring water, Eating with a spoon interpreted as

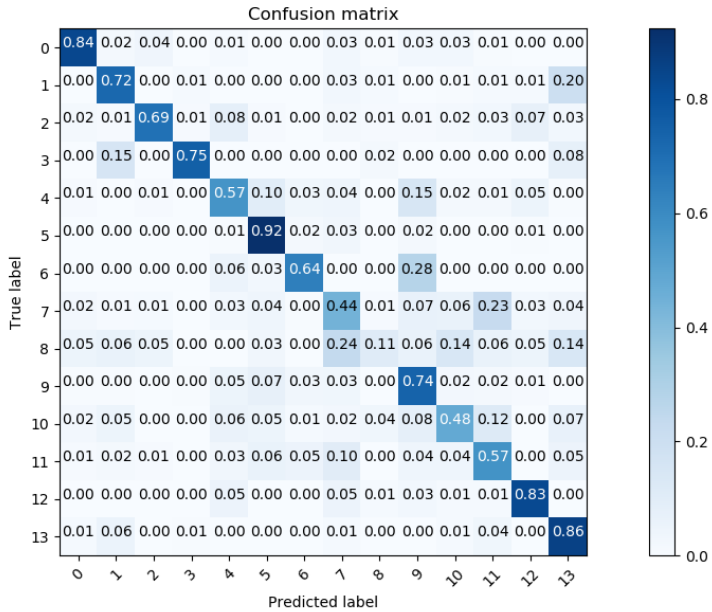


Figure 4.10: Confusion matrix of the best result for specific movement three second window using *RNN*

	Category	Recall	Precision	F1 Score
0	Brush teeth	83.87%	86.09%	84.97%
1	Climb stairs	72.38%	71.2%	71.78%
2	Comb hair	69.09%	83.52%	75.62%
3	Descend stairs	75.38%	87.5%	80.99%
4	Drink	56.72%	73.08%	63.87%
5	Eat w/ knife & fork	92.35%	73.02%	81.56%
6	Eat w/ spoon	63.89%	50.0%	56.1%
7	Get out bed	44.23%	57.86%	50.14%
8	Lie down bed	11.11%	36.84%	17.07%
9	Pour water	74.13%	65.35%	69.46%
10	Sit down	48.24%	45.56%	46.86%
11	Stand up	57.29%	37.67%	45.45%
12	Telephone	83.33%	66.33%	73.86%
13	walk	86.15%	84.3%	85.22%

Table 4.10: Best result of specific movement three second window using *RNN*

pouring water. These movements are all hand-gestures which understandably can be confused with each other.

The bad results of body movements such as lying down, sitting down and standing up has a low precision score due to their low data distribution, shown in *figure 3.2*. Considering the low amount of sliding windows, these classes are miss-interpreted as a few different categories. However, they are mostly interpreted as similar movements such as: lying down as either getting out of bed, sitting down or walking, sitting down as standing up, and standing up as getting out of bed.

Five second sliding window:

An overall accuracy of 85.6% and an overall F1 score of 81.67% is the highest performing result of five second sliding windows with RNN. The RNN used to accomplish these results is a network consisting on 4 cells of size 32. This network uses Adam as its optimizer and a learning rate of 0.01.

The result of the five second sliding windows are overall increased compared to the three second sliding windows. This is probably due to the overlap in the sliding windows, which further allows the network to recognize the differences in the movement patterns.

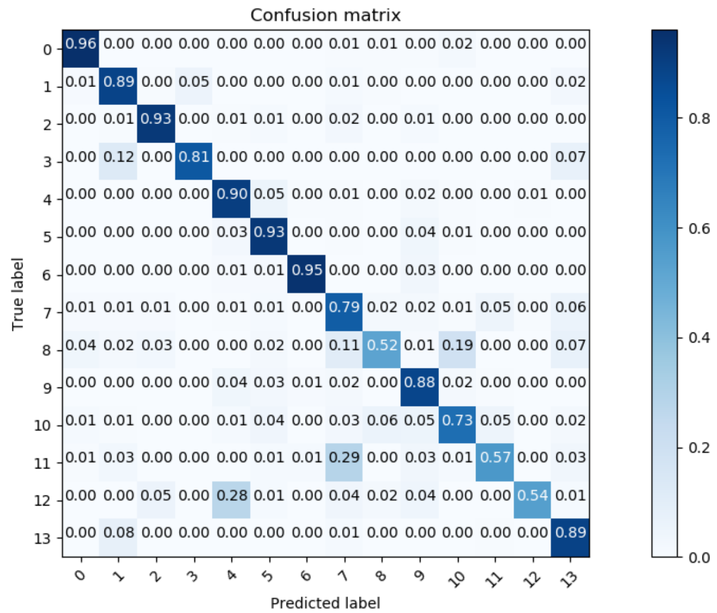


Figure 4.11: Confusion matrix of the best result for specific movement five second window using *RNN*

Again, the worst performing categories are the ones with the lowest distribution of sliding windows. Thus, they are interpreted as movements with similar patterns to its own.

	Category	Recall	Precision	F1 Score
0	Brush teeth	96.01%	96.21%	96.11%
1	Climb stairs	88.78%	75.15%	81.4%
2	Comb hair	92.54%	92.23%	92.39%
3	Descend stairs	81.2%	81.82%	81.51%
4	Drink	90.42%	81.36%	85.65%
5	Eat w/ knife & fork	92.9%	87.77%	90.26%
6	Eat w/ spoon	94.68%	89.9%	92.23%
7	Get out bed	79.23%	77.34%	78.27%
8	Lie down bed	52.25%	63.04%	57.14%
9	Pour water	88.03%	84.84%	86.41%
10	Sit down	72.9%	68.48%	70.62%
11	Stand up	57.33%	69.92%	63.0%
12	Telephone	54.11%	95.73%	69.14%
13	walk	89.14%	93.84%	91.43%

Table 4.11: Best result of specific movement five second window using *RNN*

Ten second sliding window:

Ten second sliding windows are again, the highest performing distribution of the data-points. Achieving an overall accuracy of 96.52% and an overall F1 score of 93.43%, when using Adam as the optimizer, a learning rate of 0.001 on a RNN consisting of 8 cells of size 32.

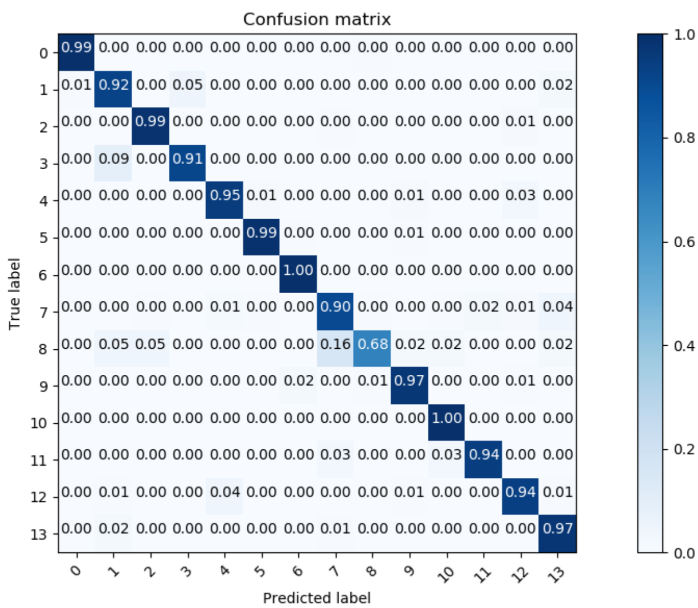


Figure 4.12: Confusion matrix of the best result for specific movement ten second window using *RNN*

	Category	Recall	Precision	F1 Score
0	Brush teeth	99.32%	99.77%	99.54%
1	Climb stairs	92.31%	85.71%	88.89%
2	Comb hair	98.72%	98.72%	98.72%
3	Descend stairs	91.3%	84.0%	87.5%
4	Drink	95.16%	94.65%	94.91%
5	Eat w/ knife & fork	98.94%	99.36%	99.15%
6	Eat w/ spoon	100%	95.0%	97.44%
7	Get out bed	90.16%	94.02%	92.05%
8	Lie down bed	68.18%	85.71%	75.95%
9	Pour water	97.25%	95.68%	96.46%
10	Sit down	100%	88.89%	94.12%
11	Stand up	94.44%	85.0%	89.47%
12	Telephone	93.98%	93.98%	93.98%
13	walk	97.49%	98.42%	97.95%

Table 4.12: Best result of specific movement ten second window using *RNN*

Most miss-interpretations are eliminated with the exception of lying down, the category with the fewest sliding windows. There are some concerns to these results as the small shift of one second for each ten seconds sliding window might cause the algorithms to overfit during training. Thus, achieving such impressive results. The problem however is the size of the dataset, increasing the shift of the sliding window drastically reduces the number of sliding windows in the dataset. Leading to poor training as the size of the dataset would be low.

4.2.3 Hip-Worn Dataset

In this section we discuss the highest performing results for the RNN on the newly collected dataset, provided by participants associated with UiA Kristiansand. This dataset was tested for each type of sliding window, which are separately discussed throughout this section.

Three second sliding window:

The highest performing result for the three second sliding window gets an overall accuracy of 85.5%, and a F1 score of 84.04%. Examining the results shown in the confusion matrix in *figure4.13*, the wrongly interpreted sliding windows are reasonable.

Examples are laying down which is interpreted as sitting relaxed, which in some cases might be a person almost lying in a sofa. Walking is sometimes confused with walking fast, which might be explained by differences in walking speed between participants. One persons normal walking speed might be the same speed as another persons speed when walking fast. Thus, confusing the algorithm.

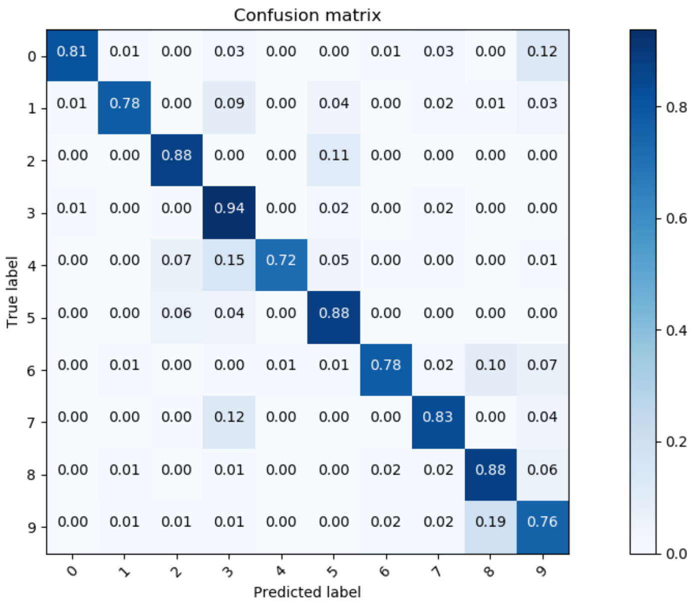


Figure 4.13: Confusion matrix of the best result for three second window of hip-worn data using *RNN*

	Category	**Recall**	**Precision**	**F1 Score**
0	Cycling	81.33%	82.99%	82.15%
1	Jogging	77.67%	94.28%	85.17%
2	Laying still	87.8%	86.82%	87.31%
3	Sitting	93.73%	83.56%	88.36%
4	Sitting in a vehicle	72.09%	98.53%	83.26%
5	Sitting relaxed	88.19%	85.55%	86.85%
6	Walking stairs	77.96%	83.04%	80.42%
7	Standing	83.27%	82.45%	82.85%
8	Walking fast	87.97%	81.27%	84.49%
9	Walking normal	75.63%	77.2%	76.41%

Table 4.13: Best result of three second window of hip-worn data using *RNN*

Looking at the categories, shown in *table 4.13*, and their individual performances, shown in *figure 4.13*. The lower performing category is sitting in a vehicle. As there are multiple vehicle options it can be hard to predict this category. For instance, if one of the participants where sitting in a buss, then the three second sliding window have a possibility to be when the buss is at a stop. Thus, making the algorithm believe it is a person who is sitting still.

The results explained above are achieved through a RNN with 4 cells, with cell sizes of 32, a learning rate of 0.1 and Adagrad as its optimizer.

Five second sliding window:

Using Adagrad as the optimizer for a RNN with four cells, of size 32, combined with a 0.1 learning rate, an accuracy of 88.48% and a F1 score of 85.29% is achieved. Again the category of sitting in a vehicle is among the lowest performing categories, as different vehicles have different driving patterns which might confuse it with other categories.

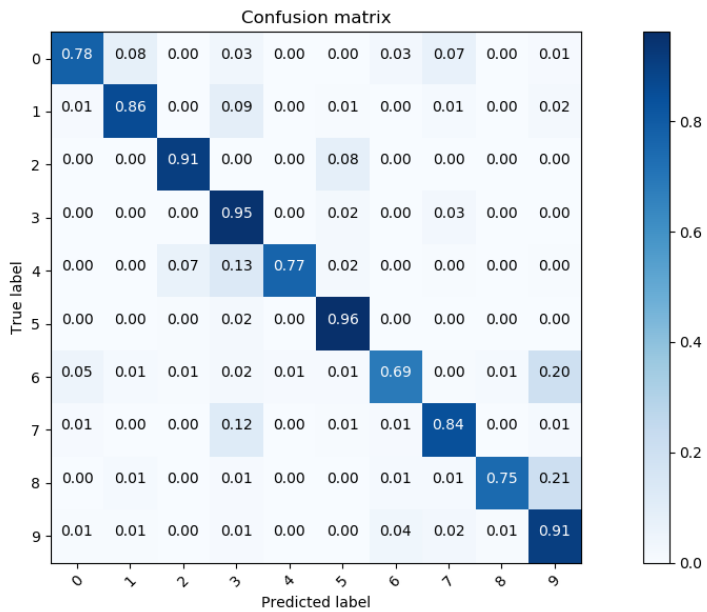


Figure 4.14: Confusion matrix of the best result for five second window of hip-worn data using *RNN*

However, looking at the results of both the category for walking stair and walking fast, shown in the confusion matrix in figure 4.14, their recall score decreased compared to the three second sliding window. We assume that the patterns between walking stairs and walking in normal speed are easier to distinguish when

	Category	Recall	Precision	F1 Score
0	Cycling	78.08%	78.74%	78.41%
1	Jogging	85.82%	89.39%	87.57%
2	Laying still	90.61%	94.86%	92.69%
3	Sitting	94.84%	86.27%	90.35%
4	Sitting in a vehicle	76.81%	97.51%	85.93%
5	Sitting relaxed	96.19%	90.63%	93.33%
6	Walking stairs	68.61%	78.83%	73.37%
7	Standing	84.24%	80.89%	82.53%
8	Walking fast	74.51%	98.56%	84.86%
9	Walking normal	90.78%	70.02%	79.06%

Table 4.14: Best result of five second window of hip-worn data using *RNN*

using three seconds of data compared to five. Thus, the results gets worse for five second sliding window within this category.

Ten second sliding window:

A clear pattern in the results is that ten second sliding windows performs better compared to the three- and five- second sliding windows. Testing the RNN with the hip-worn dataset is no exception. When combining a learning rate of 0.01 with a RNN with four cells, with a size of 32, and using Adam as the optimizer, an accuray of 89.31% with a F1 score of 89.36% is achieved.

	Category	Recall	Precision	F1 Score
0	Cycling	87.55%	89.67%	88.6%
1	Jogging	89.41%	96.54%	92.84%
2	Laying still	98.52%	82.88%	90.02%
3	Sitting	94.27%	87.44%	90.73%
4	Sitting in a vehicle	79.01%	98.36%	87.63%
5	Sitting relaxed	85.97%	96.12%	90.76%
6	Walking stairs	93.18%	94.41%	93.79%
7	Standing	85.09%	81.83%	83.43%
8	Walking fast	77.3%	98.55%	86.64%
9	Walking normal	94.69%	76.51%	84.63%

Table 4.15: Best result of ten second window of hip-worn data using *RNN*

The results shown in the confusion matrix, see figure 4.15, are impressively high. Most categories are interpreted correctly more than 85% of the time, with the exception of sitting in a vehicle and walking fast. Other incorrectly interpreted sliding windows are confused with related categories, such as standing interpreted as sitting, and sitting relaxed as laying still.

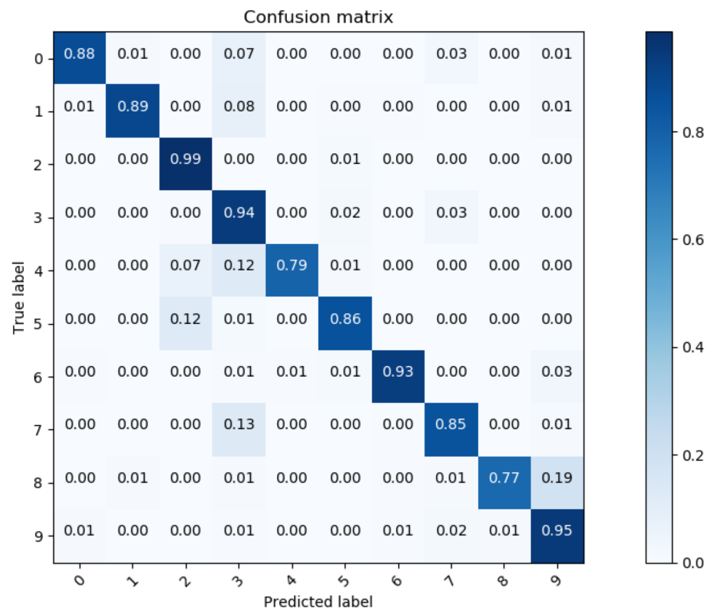


Figure 4.15: Confusion matrix of the best result for ten second window of hip-worn data using *RNN*

4.3 Summary of Results, and Discussion

This section summarizes the results of the experiments, and discuss their performances compared to the state-of-the-art algorithm.

To summarize the results of the DNN, the results are as expected. Recognizing ADLs, few categories, are accomplished with good success. The DNN achieve accuracies between 80-93% and F1 scores between 80-90% for basic movements, see *table 4.16*.

Increasing the complexity of the dataset, by using specific movement categories, the accuracies significantly decreases. This is expected as there are more categories to learn and recognize. The achieved accuracies for the specific movement types, using DNN, are between 60-80%, while the F1 socres are between 55-65%, also shown in *table 4.16*. Considering each experiment trained the model for 2000 training steps, then providing it with unseen data to classify, the DNN is able to classify "unkown" data, answering *research question 1b*.

As expected, different combinations of hyper-parameters perform better depending on which categorization of the movement patterns were used. For the broad categories, ADLs, low learning rates and the Adam optimizer achieves the highest F1 scores; 78.5% for three second-, 82.4% for five second- and 89.4% for ten second- sliding windows. Recognizing the specific movements, Adagrad with a learning rate of 0.01 gets the highest results; 56.6%, for three seconds, 62.1% for five seconds and 67.8% for ten seconds. The answer to *research question 1a* according to our hyper-parameter search is Adam and a learning rate of 0.0001 for ADLs, and Adagrad and a learning rate of 0.01 for specific movements.

Dataset	**Sliding Window**	**Accuracy**	**F1 Score**
Wrist Worn: Basic Movement	3 Second	80.94%	78.46%
	5 Second	86.01/%	82.37%
	10 Second	92.4/%	89.43%
Wrist-Worn: Specific Movement	3 Second	59.93%	56.59%
	5 Second	67.83%	62.11%
	10 Second	81.29%	66.75%

Table 4.16: Summary of the *DNN* results and Discussion

It is difficult to determine whether the proposed DNN model performs better than the state-of-the-art algorithm. The state-of-the-art algorithm performs better at some categories; Drinking from a glass, climbing stairs, pouring water into glass and standing up from a chair, while the DNN model performs better at others; Getting out of bed, sitting down on a chair, and walking. Additionally the DNN model is trained to recognize all of the categories in the wrist-worn dataset, not just a selection.

Analyzing *Table 4.18*, the proposed DNN model achieves relatively good results considering the complexity of the classification. Comparing it to the state-of-the-art, which classifies seven categories, the DNN achieves comparable results using all 14 categories. Thus, we argue that the proposed DNN model at the very least matches the performance of the state-of-the-art algorithm, answering *research question 1c*.

Compared to the DNN, the accuracies are consistently higher for RNN. Achieving accuracies between 85-99% and F1 scores between 83-98% for the basic movement types. For the specific movement the accuracies are between 70-97%, while the F1 scores are between 65-94%, shown in table 4.17.

The intuition at the start of this research was that the RNN model would perform better on time-series data, compared to the feed-forward DNN model. The results shown in *table 4.18* confirms this intuition, and also answers *research question 2*, as the recognition of each category improves when using the RNN model. The results of the RNN are either improvements or similar to the highest achieved percentages of the state-of-the-art, even with all categories. Thus, it is fair to argue that the proposed RNN model is better than both the state-of-the-art algorithm and the DNN model.

Dataset	**Sliding Window**	**Accuracy**	**F1 Score**
Wrist Worn: Basic Movement	3 Second	84.89%	82.56%
	5 Second	94.65/%	93.03%
	10 Second	98.75/%	98.06%
Wrist-Worn: Specific Movement	3 Second	70.39%	65.58%
	5 Second	85.6%	81.67%
	10 Second	96.52%	93.43%
Hip-Worn	3 Second	85.5%	84.04%
	5 Second	88.48%	85.29%
	10 Second	89.31%	89.36%

Table 4.17: Summary of the *RNN* results

In addition to the ADLs and specific movement types, for the wrist-worn dataset, we tested the hip-worn dataset with the RNN. The results of this dataset are impressive considering that it consists of movements of similar type, with different intensities. Achieving accuracies between 85-90% and F1 scores of 84-90%. Thus, showing that the RNN model is able to perform at a high level on new datasets, answering *research question 3*.

Table 4.18 shows a comparison of the results of the two proposed deep learning models and the state-of-the-art algorithm. The comparable percentages are the true positives of the state-of-the-art and the recall score of the deep learning models.

	DNN results			**State-of-the-art**		**RNN results**		
Category	***Recall***	***Precision***	***F1 score***	***True positives***	***True negatives***	***Recall***	***Precision***	***F1 score***
Brushing teeth	94.77%	98.82%	96.75%	-	-	99.32%	99.77%	99.54%
Climbing stairs	4.9%	35.0%	8.59%	20%	93.34%	92.31%	85.71%	88.89%
Comb hair	57.26%	84.81%	68.37%	-	-	98.72%	98.72%	98.72%
Descend stairs	2.17%	33.33%	4.08%	-	-	91.3%	84.0%	87.5%
Drinking	68.28%	60.77%	64.3%	100%	83.34%	95.16%	94.65%	94.91%
Eat w/ fork and knife	93.01%	82.06%	87.19%	-	-	98.94%	99.36%	99.15%
Eat w/spoon	95.79%	93.81%	94.79%	-	-	100%	95.0%	97.44%
Getting out of bed	85.66%	83.27%	84.44%	60%	66.67%	90.16%	94.02%	92.05%
Lying down on the bed	29.55%	61.9%	40.0%	-	-	68.18%	85.71%	75.95%
Pour water into glass	66.48%	85.21%	74.69%	100%	80%	97.25%	95.68%	96.46%
Sitting down on a chair	62.5%	60.61%	61.54%	0%	93.34%	100%	88.89%	94.12%
Standing up from a chair	52.78%	55.88%	54.29%	60%	83.34%	94.44%	85.0%	89.47%
Using the telephone	85.54%	58.44%	69.44%	-	-	93.98%	93.98%	93.98%
Walking	96.55%	83.3%	89.44%	40%	70%	97.49%	98.42%	97.95%

Table 4.18: Comparison of thesis results and state-of-the-art

Chapter 5

Conclusion and Future Work

This chapter concludes the research done throughout this thesis. Additionally, we discuss a possible road map for future research based on this thesis.

5.1 Conclusion

Determining whether DL classification models are suitable for classifying movement from raw data, we answer the subordinate research questions. Applying different combinations of hyper-parameters; optimizers, and learning rates, we discovered that depending on the categorizations of the movement patterns it differs. However, the proposed DNN model is able to recognize "unlabeled" data with an acceptable overall recall percentage of 64%, using 14 categories, which is a 10% increase compared to the state-of-the-art algorithm which has a 54% overall recall score for seven categories.

The proposed RNN model is shown to perform significantly better on these time-series data, reaching an overall recall score of 94%, which is a 40% increase compared to the DNN. The results of the RNN model surpasses the percentages for most categories compared to the state-of-the-art, even when classifying all categories in the dataset. Furthermore, the RNN model is able to recognize different movement pattern from a new dataset, consisting of movement types of different intensities. Thus, the proposed RNN model is the best suited algorithm, of the discussed algorithms in this thesis, for movement pattern recognition. With this in mind, we believe that we meet the goals defined at the start of this thesis.

5.2 Future Work

The research throughout this thesis proposed two deep learning algorithms for recognition of movement patterns, with relatively high accuracies. However, there are room for improvements.

Possible research areas based on this thesis could be:

- Updating the structure of the deep learning models
- Extensive hyper-parameter search for the proposed deep learning models

Updating the structure:

As shown in this thesis, deep learning models are easily able to differentiate the broad categories for movement pattern, while sometimes struggling with the specific categories. Thus, restructuring the proposed deep learning models to first recognize these broad ADL categories, then classifying which specific movement type within the ADL it is. For instance, the deep learning models first recognizes a pattern as walking, then determines the intensity or direction; walking normal, fast, upstairs or downstairs.

Extensive hyper-parameter search:

The hyper-parameter search of this thesis, consists of testing different optimizers in combination with different learning rates. Howere, as mentioned in the results the activation function we used is not suited for all optimizers, thus gaining poor results. In future researches, testing different activation functions might improve these results

Appendices

Appendix A

Explanation Tables

A.1 Hip-Worn Movement Types

Index	Motion Primitives
0	Cycling
1	Jogging
2	Laying still
3	Sitting
4	Sitting in vehicle
5	Sitting relaxed
6	Walking stairs
7	Standing
8	Walking Fast
9	Walking Normal

Table A.1: Index to motion primitives explanation table for the hip-worn dataset

A.2 Wrist-Worn ADL Categories

Index	**Activities-of-Daily-life**
0	Personal hygiene
1	Mobility
2	Feeding
3	Communication
4	Functional transfers

Table A.2: Index to ADL explanation table for the wrist-worn dataset

A.3 Wrist-Worn Movement Types

Index	**Motion Primitives**
0	Brush teeth
1	Climb stairs
2	Comb hair
3	Descend stairs
4	Drink from glass
5	Eat with knife and fork
6	Eat with spoon
7	Get out of bed
8	Lie down in bed
9	Pour water into glass
10	Sit down on chair
11	Stand up from chair
12	Use telephone
13	Walk

Table A.3: Index to motion primitives explanation table for the wrist-worn dataset

Appendix B

Overall Deep Neural Network Results

B.1 Wrist-Worn Dataset - Basic Movements

Learning Rate	Sliding window	Recall	Precision	F1 Score
0.0001	10 second	85.93%	85.53%	85.73%
	3 second	67.4%	71.27%	69.28%
	5 second	73.83%	76.27%	75.03%
0.001	10 second	88.48%	89.07%	88.77%
	3 second	69.95%	76.65%	73.15%
	5 second	80.47%	82.13%	81.29%
0.01	10 second	83.73%	88.13%	85.87%
	3 second	72.68%	79.98%	76.16%
	5 second	81.01%	82.4%	81.7%
0.1	10 second	20.0%	7.23%	10.62%
	3 second	72.26%	77.17%	74.63%
	5 second	70.41%	76.17%	73.18%

Table B.1: Results for basic movement using *Adagrad* with *DNN*

Learning Rate	Sliding window	Recall	Precision	F1 Score
0.0001	**10 second**	**90.99%**	**87.93%**	**89.43%**
	3 second	**76.79%**	**80.21%**	**78.46%**
	5 second	**80.97%**	**83.82%**	**82.37%**
0.001	10 second	84.29%	87.78%	86.0%
	3 second	71.38%	77.16%	74.16%
	5 second	71.39%	81.32%	76.03%
0.01	10 second	81.99%	81.54%	81.76%
	3 second	63.93%	67.93%	65.87%
	5 second	72.89%	70.16%	71.5%
0.1	10 second	20.0%	7.23%	10.62%
	3 second	20.0%	6.77%	10.12%
	5 second	20.0%	6.97%	10.34%

Table B.2: Results for basic movement using *Adam* with *DNN*

Learning Rate	Sliding window	Recall	Precision	F1 Score
0.0001	10 second	20.0%	3.99%	6.65%
	3 second	20.0%	2.57%	4.55%
	5 second	20.0%	3.09%	5.35%
0.001	10 second	20.0%	3.99%	6.65%
	3 second	20.0%	2.57%	4.55%
	5 second	20.0%	3.09%	5.35%
0.01	10 second	20.0%	3.99%	6.65%
	3 second	20.0%	2.57%	4.55%
	5 second	20.0%	3.09%	5.35%
0.1	10 second	20.0%	3.99%	6.65%
	3 second	20.0%	2.57%	4.55%
	5 second	20.0%	3.09%	5.35%

Table B.3: Results for basic movement using *SGD* with *DNN*

B.2 Wrist-Worn Dataset - Specific Movement

Learning Rate	Sliding window	Recall	Precision	F1 Score
0.0001	10 second	54.24%	54.43%	54.33%
	3 second	35.24%	37.62%	36.39%
	5 second	47.55%	48.34%	47.94%
0.001	10 second	62.88%	64.96%	63.9%
	3 second	43.7%	48.49%	45.97%
	5 second	57.61%	62.8%	60.09%
0.01	**10 second**	**63.95%**	**69.8%**	**66.75%**
	3 second	**52.95%**	**60.76%**	**56.59%**
	5 second	**58.71%**	**65.92%**	**62.11%**
0.1	10 second	7.14%	2.08%	3.22%
	3 second	23.62%	17.24%	19.93%
	5 second	7.14%	1.68%	2.72%

Table B.4: Results for specific movement using *Adagrad* with *DNN*

Learning Rate	Sliding window	Recall	Precision	F1 Score
0.0001	10 second	64.49%	65.94%	65.21%
	3 second	48.8%	53.78%	51.17%
	5 second	58.64%	65.66%	61.95%
0.001	10 second	56.88%	64.13%	60.29%
	3 second	53.29%	54.66%	53.97%
	5 second	57.96%	62.44%	60.12%
0.01	10 second	34.35%	28.11%	30.92%
	3 second	38.15%	56.95%	45.69%
	5 second	37.66%	37.58%	37.62%
0.1	10 second	7.14%	2.08%	3.22%
	3 second	7.14%	1.54%	2.53%
	5 second	7.14%	1.68%	2.72%

Table B.5: Results for specific movement using *Adam* with *DNN*

Learning Rate	Sliding window	Recall	Precision	F1 Score
0.0001	10 second	7.14%	0.96%	1.69%
	3 second	7.14%	0.53%	0.99%
	5 second	7.14%	0.67%	1.23%
0.001	10 second	7.14%	0.96%	1.69%
	3 second	7.14%	0.53%	0.99%
	5 second	7.14%	0.67%	1.23%
0.01	10 second	7.14%	0.96%	1.69%
	3 second	7.14%	0.53%	0.99%
	5 second	7.14%	0.67%	1.23%
0.1	10 second	7.14%	0.96%	1.69%
	3 second	7.14%	0.53%	0.99%
	5 second	7.14%	0.67%	1.23%

Table B.6: Results for specific movement using *SGD* with *DNN*

Appendix C

Overall Reccurent Neural Network Results

C.1 Wrist-Worn Dataset - Basic Movements

C.1.1 Results for three second sliding window

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	35.95%	26.16%	30.28%
		32	40.58%	43.15%	41.83%
	8	16	32.83%	17.73%	23.03%
		32	50.85%	41.87%	45.93%
0.001	4	16	67.28%	74.87%	70.87%
		32	73.57%	82.89%	77.95%
	8	16	57.75%	55.39%	56.55%
		32	74.3%	80.44%	77.25%
0.01	4	16	78.36%	80.3%	79.32%
		32	79.29%	83.57%	81.37%
	8	16	78.66%	80.78%	79.71%
		32	79.75%	82.59%	81.15%
0.1	4	16	78.19%	81.85%	79.98%
		32	77.0%	83.15%	79.96%
	8	16	74.81%	79.55%	77.11%
		32	77.54%	81.69%	79.56%

Table C.1: Results for three second window of basic movement using *Adagrad* with *RNN*

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	71.27%	75.93%	73.53%
		32	80.81%	84.14%	82.44%
	8	16	71.49%	76.9%	74.1%
		32	76.23%	77.7%	76.96%
0.001	4	16	79.44%	83.86%	81.59%
		32	78.6%	82.81%	80.65%
	8	16	75.82%	81.68%	78.64%
		32	**80.42%**	**84.82%**	**82.56%**
0.01	4	16	48.58%	52.19%	50.32%
		32	76.4%	79.36%	77.85%
	8	16	20.0%	2.57%	4.55%
		32	20.0%	6.77%	10.12%
0.1	4	16	20.0%	6.77%	10.12%
		32	20.0%	6.77%	10.12%
	8	16	20.0%	6.77%	10.12%
		32	20.0%	2.57%	4.55%

Table C.2: Results for three second window of basic movement using *Adam* with *RNN*

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	77.38%	83.52%	80.33%
		32	77.59%	81.31%	79.41%
	8	16	75.82%	80.91%	78.28%
		32	77.27%	79.26%	78.25%
0.001	4	16	20.0%	2.57%	4.55%
		32	20.0%	6.77%	10.12%
	8	16	20.0%	2.57%	4.55%
		32	78.45%	82.46%	80.41%
0.01	4	16	20.0%	2.57%	4.55%
		32	20.0%	2.57%	4.55%
	8	16	20.0%	2.57%	4.55%
		32	20.0%	2.57%	4.55%
0.1	4	16	20.0%	2.57%	4.55%
		32	20.0%	2.57%	4.55%
	8	16	20.0%	2.57%	4.55%
		32	20.0%	2.57%	4.55%

Table C.3: Results for three second window of basic movement using *SGD* with *RNN*

C.1.2 Results for five second sliding window

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	26.55%	19.72%	22.63%
		32	42.27%	46.94%	44.48%
	8	16	20.0%	6.97%	10.34%
		32	57.88%	45.53%	50.97%
0.001	4	16	75.72%	81.82%	78.65%
		32	82.33%	85.29%	83.78%
	8	16	78.55%	84.15%	81.25%
		32	83.32%	88.27%	85.72%
0.01	**4**	16	90.78%	91.1%	90.94%
		32	**92.63%**	**93.44%**	**93.03%**
	8	16	85.16%	89.57%	87.31%
		32	91.53%	91.89%	91.71%
0.1	4	16	81.33%	80.02%	80.67%
		32	90.87%	92.24%	91.55%
	8	16	85.58%	84.08%	84.82%
		32	20.0%	3.09%	5.35%

Table C.4: Results for five second window of basic movement using *Adagrad* with *RNN*

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	81.71%	86.92%	84.23%
		32	89.55%	90.33%	89.94%
	8	16	78.39%	78.56%	78.47%
		32	85.45%	89.53%	87.44%
0.001	4	16	91.14%	91.37%	91.25%
		32	92.1%	93.62%	92.85%
	8	16	81.18%	86.5%	83.76%
		32	90.85%	90.33%	90.59%
0.01	4	16	86.94%	92.54%	89.65%
		32	20.0%	3.09%	5.35%
	8	16	20.0%	6.97%	10.34%
		32	85.23%	85.72%	85.47%
0.1	4	16	20.0%	6.97%	10.34%
		32	20.0%	6.97%	10.34%
	8	16	20.0%	6.97%	10.34%
		32	20.0%	3.09%	5.35%

Table C.5: Results for five second window of basic movement using *Adam* with *RNN*

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	82.89%	84.32%	83.6%
		32	91.35%	90.34%	90.84%
	8	16	84.75%	86.32%	85.53%
		32	87.2%	86.44%	86.82%
0.001	4	16	20.0%	3.09%	5.35%
		32	31.09%	38.33%	34.33%
	8	16	20.0%	3.09%	5.35%
		32	88.37%	88.59%	88.48%
0.01	4	16	20.0%	3.09%	5.35%
		32	20.0%	3.09%	5.35%
	8	16	20.0%	3.09%	5.35%
		32	20.0%	3.09%	5.35%
0.1	4	16	20.0%	3.09%	5.35%
		32	20.0%	3.09%	5.35%
	8	16	20.0%	3.09%	5.35%
		32	20.0%	3.09%	5.35%

Table C.6: Results for five second window of basic movement using *SGD* with *RNN*

C.1.3 Results for ten second sliding window

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	24.86%	18.64%	21.31%
		32	53.74%	49.72%	51.65%
	8	16	29.5%	22.65%	25.63%
		32	51.88%	44.34%	47.81%
0.001	4	16	77.24%	87.36%	81.99%
		32	89.41%	89.99%	89.7%
	8	16	71.57%	67.93%	69.7%
		32	92.87%	91.38%	92.12%
0.01	4	16	95.84%	95.19%	95.51%
		32	96.99%	97.42%	97.2%
	8	16	94.73%	94.17%	94.45%
		32	94.17%	96.32%	95.23%
0.1	4	16	95.9%	97.09%	96.49%
		32	96.94%	97.07%	97.0%
	8	16	20.0%	3.99%	6.65%
		32	95.44%	93.74%	94.58%

Table C.7: Results for ten second window of basic movement using *Adagrad* with *RNN*

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	91.49%	91.37%	91.43%
		32	94.98%	95.94%	95.46%
	8	16	87.86%	91.69%	89.73%
		32	95.79%	96.01%	95.9%
0.001	**4**	16	95.99%	96.43%	96.21%
		32	**97.35%**	**98.78%**	**98.06%**
	8	16	94.93%	96.14%	95.53%
		32	96.98%	96.92%	96.95%
0.01	4	16	94.91%	94.64%	94.77%
		32	20.0%	3.99%	6.65%
	8	16	20.0%	3.99%	6.65%
		32	54.97%	54.65%	54.81%
0.1	4	16	20.0%	7.23%	10.62%
		32	20.0%	3.99%	6.65%
	8	16	20.0%	3.99%	6.65%
		32	20.0%	3.99%	6.65%

Table C.8: Results for ten second window of basic movement using *Adam* with *RNN*

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	90.45%	90.59%	90.52%
		32	96.18%	95.51%	95.84%
	8	16	91.61%	91.04%	91.32%
		32	95.13%	92.68%	93.89%
0.001	4	16	20.0%	3.99%	6.65%
		32	20.0%	7.23%	10.62%
	8	16	20.0%	3.99%	6.65%
		32	82.7%	89.51%	85.97%
0.01	4	16	20.0%	3.99%	6.65%
		32	20.0%	3.99%	6.65%
	8	16	20.0%	3.99%	6.65%
		32	20.0%	3.99%	6.65%
0.1	4	16	20.0%	3.99%	6.65%
		32	20.0%	3.99%	6.65%
	8	16	20.0%	3.99%	6.65%
		32	20.0%	3.99%	6.65%

Table C.9: Results for ten second window of basic movement using *SGD* with *RNN*

C.2 Wrist-Worn Dataset - Specific Movement

C.2.1 Results for three second sliding window

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	9.43%	3.7%	5.31%
		32	9.88%	6.18%	7.6%
	8	16	7.14%	0.68%	1.24%
		32	6.83%	0.7%	1.27%
0.001	4	16	28.91%	28.54%	28.72%
		32	47.23%	53.63%	50.23%
	8	16	23.42%	21.67%	22.51%
		32	45.08%	49.46%	47.17%
0.01	4	16	52.24%	57.33%	54.67%
		32	62.61%	63.28%	62.94%
	8	16	50.05%	55.6%	52.68%
		32	58.78%	58.95%	58.86%
0.1	4	16	61.19%	58.5%	59.81%
		32	60.98%	62.09%	61.53%
	8	16	7.14%	1.54%	2.53%
		32	61.17%	62.91%	62.03%

Table C.10: Results for three second window of specific movement using *Adagrad* with *RNN*

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	43.97%	49.08%	46.38%
		32	54.36%	62.15%	57.99%
	8	16	39.68%	44.78%	42.08%
		32	55.95%	59.01%	57.44%
0.001	**4**	16	47.74%	47.75%	47.74%
		32	**65.58%**	**65.59%**	**65.58%**
	8	16	58.5%	61.17%	59.81%
		32	61.93%	64.37%	63.13%
0.01	4	16	47.5%	53.0%	50.1%
		32	7.28%	8.69%	7.92%
	8	16	14.2%	8.39%	10.55%
		32	59.32%	61.52%	60.4%
0.1	4	16	7.14%	1.54%	2.53%
		32	7.14%	1.54%	2.53%
	8	16	7.14%	1.54%	2.53%
		32	7.14%	1.54%	2.53%

Table C.11: Results for three second window of specific movement using *Adam* with *RNN*

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	49.04%	53.4%	51.13%
		32	53.83%	63.52%	58.27%
	8	16	38.96%	42.81%	40.79%
		32	52.87%	57.07%	54.89%
0.001	4	16	33.36%	41.18%	36.86%
		32	7.14%	0.53%	0.99%
	8	16	37.0%	40.35%	38.6%
		32	45.43%	53.15%	48.99%
0.01	4	16	7.14%	0.53%	0.99%
		32	7.14%	0.53%	0.99%
	8	16	7.14%	1.54%	2.53%
		32	7.14%	0.53%	0.99%
0.1	4	16	7.14%	0.53%	0.99%
		32	7.14%	0.53%	0.99%
	8	16	7.14%	0.53%	0.99%
		32	7.14%	0.53%	0.99%

Table C.12: Results for three second window of specific movement using *SGD* with *RNN*

C.2.2 Results for five second sliding window

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	7.12%	12.04%	8.95%
		32	10.96%	7.16%	8.66%
	8	16	13.89%	8.5%	10.55%
		32	13.3%	9.26%	10.92%
0.001	4	16	28.2%	25.28%	26.66%
		32	58.61%	61.82%	60.17%
	8	16	31.16%	32.31%	31.72%
		32	65.09%	66.78%	65.92%
0.01	4	16	67.65%	68.44%	68.04%
		32	80.04%	80.81%	80.42%
	8	16	53.36%	61.4%	57.1%
		32	79.23%	80.0%	79.61%
0.1	4	16	74.8%	74.09%	74.44%
		32	77.61%	78.49%	78.05%
	8	16	7.14%	1.68%	2.72%
		32	70.07%	72.84%	71.43%

Table C.13: Results for five second window of specific movement using *Adagrad* with *RNN*

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	47.25%	47.31%	47.28%
		32	74.52%	73.55%	74.03%
	8	16	45.98%	45.44%	45.71%
		32	66.9%	68.3%	67.59%
0.001	4	16	74.21%	73.18%	73.69%
		32	79.51%	80.75%	80.13%
	8	16	77.18%	76.98%	77.08%
		32	80.47%	79.79%	80.13%
0.01	**4**	16	66.1%	65.71%	65.9%
		32	**80.68%**	**82.69%**	**81.67%**
	8	16	7.14%	0.67%	1.23%
		32	7.14%	1.68%	2.72%
0.1	4	16	7.14%	0.67%	1.23%
		32	7.14%	1.68%	2.72%
	8	16	7.14%	1.68%	2.72%
		32	7.14%	1.68%	2.72%

Table C.14: Results for five second window of specific movement using *Adam* with *RNN*

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	68.27%	70.48%	69.36%
		32	75.2%	78.34%	76.74%
	8	16	43.22%	47.84%	45.41%
		32	69.16%	72.96%	71.01%
0.001	4	16	53.98%	63.3%	58.27%
		32	7.14%	0.67%	1.23%
	8	16	61.77%	65.64%	63.65%
		32	7.14%	0.67%	1.23%
0.01	4	16	7.14%	0.67%	1.23%
		32	7.14%	0.67%	1.23%
	8	16	7.14%	0.67%	1.23%
		32	7.14%	0.67%	1.23%
0.1	4	16	7.14%	0.67%	1.23%
		32	7.14%	0.67%	1.23%
	8	16	7.14%	0.67%	1.23%
		32	7.14%	0.67%	1.23%

Table C.15: Results for five second window of specific movement using *SGD* with *RNN*

C.2.3 Results for ten second sliding window

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	6.82%	1.46%	2.41%
		32	13.69%	8.89%	10.78%
	8	16	13.41%	6.12%	8.4%
		32	19.89%	12.61%	15.43%
0.001	4	16	39.34%	37.44%	38.37%
		32	73.71%	81.43%	77.38%
	8	16	41.53%	39.42%	40.45%
		32	68.0%	69.1%	68.55%
0.01	4	16	74.44%	77.76%	76.06%
		32	85.92%	85.59%	85.75%
	8	16	74.44%	79.21%	76.75%
		32	82.14%	84.09%	83.1%
0.1	4	16	7.14%	0.96%	1.69%
		32	43.19%	44.36%	43.77%
	8	16	7.14%	2.08%	3.22%
		32	58.33%	65.71%	61.8%

Table C.16: Results for ten second window of specific movement using *Adagrad* with *RNN*

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	48.69%	49.71%	49.19%
		32	75.95%	77.94%	76.93%
	8	16	57.59%	62.79%	60.08%
		32	70.21%	74.76%	72.41%
0.001	4	16	81.07%	81.8%	81.43%
		32	91.46%	92.4%	91.93%
	8	16	89.99%	90.27%	90.13%
		32	**94.09%**	**92.78%**	**93.43%**
0.01	4	16	42.9%	47.2%	44.95%
		32	7.14%	2.08%	3.22%
	8	16	77.81%	83.59%	80.6%
		32	89.94%	89.62%	89.78%
0.1	4	16	7.14%	2.08%	3.22%
		32	7.14%	0.96%	1.69%
	8	16	7.14%	2.08%	3.22%
		32	7.14%	0.96%	1.69%

Table C.17: Results for ten second window of specific movement using *Adam* with *RNN*

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	72.67%	76.99%	74.77%
		32	85.58%	88.94%	87.23%
	8	16	57.57%	59.39%	58.47%
		32	82.52%	85.69%	84.08%
0.001	4	16	46.52%	49.96%	48.18%
		32	7.14%	0.96%	1.69%
	8	16	7.14%	0.96%	1.69%
		32	7.14%	0.96%	1.69%
0.01	4	16	7.14%	0.96%	1.69%
		32	7.14%	0.96%	1.69%
	8	16	7.14%	0.96%	1.69%
		32	7.14%	0.96%	1.69%
0.1	4	16	7.14%	0.96%	1.69%
		32	7.14%	0.96%	1.69%
	8	16	7.14%	0.96%	1.69%
		32	7.14%	0.96%	1.69%

Table C.18: Results for ten second window of specific movement using *SGD* with *RNN*

C.3 Hip-Worn Dataset

C.3.1 Results for three second sliding window

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	16.67%	11.56%	13.65%
		32	14.02%	6.89%	9.24%
	8	16	8.48%	3.9%	5.34%
		32	11.77%	3.69%	5.62%
0.001	4	16	17.12%	8.09%	10.99%
		32	45.85%	53.05%	49.19%
	8	16	32.18%	28.22%	30.07%
		32	43.77%	39.01%	41.25%
0.01	4	16	58.35%	65.49%	61.71%
		32	72.89%	77.26%	75.01%
	8	16	58.91%	57.49%	58.19%
		32	79.62%	82.33%	80.95%
0.1	**4**	16	58.51%	75.14%	65.79%
		32	**82.56%**	**85.57%**	**84.04%**
	8	16	61.89%	71.11%	66.18%
		32	62.84%	67.27%	64.98%

Table C.19: Results for three second window of hip-worn data using *Adagrad* with *RNN*

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	39.32%	34.38%	36.68%
		32	47.33%	48.38%	47.85%
	8	16	48.92%	52.81%	50.79%
		32	51.51%	51.99%	51.75%
0.001	4	16	69.07%	78.73%	73.58%
		32	76.44%	83.44%	79.79%
	8	16	73.89%	76.15%	75.0%
		32	79.16%	84.42%	81.71%
0.01	4	16	77.39%	77.09%	77.24%
		32	17.79%	35.66%	23.74%
	8	16	64.62%	66.84%	65.71%
		32	80.61%	83.65%	82.1%
0.1	4	16	10.0%	0.17%	0.33%
		32	10.0%	0.17%	0.33%
	8	16	10.0%	2.15%	3.54%
		32	10.0%	0.17%	0.33%

Table C.20: Results for three second window of hip-worn data using *Adam* with *RNN*

C.3.2 Results for five second sliding window

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	6.13%	2.84%	3.88%
		32	10.0%	2.14%	3.53%
	8	16	10.98%	7.78%	9.11%
		32	10.0%	2.14%	3.53%
0.001	4	16	24.53%	21.16%	22.72%
		32	40.01%	32.31%	35.75%
	8	16	36.01%	30.26%	32.89%
		32	46.64%	41.13%	43.71%
0.01	4	16	56.68%	56.38%	56.53%
		32	74.1%	78.32%	76.15%
	8	16	69.17%	77.22%	72.97%
		32	74.11%	74.86%	74.48%
0.1	**4**	16	77.07%	86.1%	81.34%
		32	**84.05%**	**86.57%**	**85.29%**
	8	16	62.75%	64.09%	63.41%
		32	65.15%	69.65%	67.32%

Table C.21: Results for five second window of hip-worn data using *Adagrad* with *RNN*

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	28.91%	27.88%	28.39%
		32	51.97%	50.6%	51.28%
	8	16	52.17%	51.05%	51.6%
		32	63.03%	65.73%	64.35%
0.001	4	16	62.83%	67.68%	65.16%
		32	74.19%	77.96%	76.03%
	8	16	67.47%	74.88%	70.98%
		32	77.22%	81.88%	79.48%
0.01	4	16	36.11%	48.36%	41.35%
		32	21.65%	31.26%	25.58%
	8	16	17.29%	12.06%	14.21%
		32	10.0%	0.18%	0.35%
0.1	4	16	15.7%	7.12%	9.8%
		32	10.0%	0.18%	0.35%
	8	16	10.0%	0.18%	0.35%
		32	10.0%	0.18%	0.35%

Table C.22: Results for five second window of hip-worn data using *Adam* with *RNN*

C.3.3 Results for ten second sliding window

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	8.17%	2.05%	3.28%
		32	11.44%	5.81%	7.71%
	8	16	13.64%	5.16%	7.49%
		32	10.0%	3.46%	5.14%
0.001	4	16	22.92%	16.4%	19.12%
		32	41.69%	37.18%	39.31%
	8	16	17.66%	12.18%	14.42%
		32	44.1%	43.89%	43.99%
0.01	4	16	63.56%	66.86%	65.17%
		32	84.28%	85.29%	84.78%
	8	16	59.36%	60.89%	60.12%
		32	72.61%	78.8%	75.58%
0.1	4	16	86.49%	89.18%	87.81%
		32	10.0%	0.17%	0.33%
	8	16	10.0%	0.17%	0.33%
		32	68.35%	76.79%	72.32%

Table C.23: Results for ten second window of hip-worn data using *Adagrad* with *RNN*

Learning Rate	Cells	Cell Size	Recall	Precision	F1 Score
0.0001	4	16	41.61%	41.08%	41.34%
		32	58.04%	57.66%	57.85%
	8	16	44.43%	40.81%	42.54%
		32	58.12%	64.15%	60.99%
0.001	4	16	73.08%	80.97%	76.82%
		32	79.59%	83.51%	81.5%
	8	16	67.51%	76.0%	71.5%
		32	84.72%	84.99%	84.85%
0.01	**4**	16	21.54%	26.25%	23.66%
		32	**88.5%**	**90.23%**	**89.36%**
	8	16	75.98%	78.61%	77.27%
		32	10.0%	0.17%	0.33%
0.1	4	16	10.0%	0.17%	0.33%
		32	10.0%	0.17%	0.33%
	8	16	10.0%	0.17%	0.33%
		32	10.0%	0.17%	0.33%

Table C.24: Results for ten second window of hip-worn data using *Adam* with *RNN*

Bibliography

[1] P. Andersen, "Deep reinforcement learning using capsules in advanced game environments," *CoRR*, vol. abs/1801.09597, 2018.

[2] B. Bruno, F. Mastrogiovanni, A. Sgorbissa, T. Vernazza, and R. Zaccaria, "Human motion modelling and recognition: A computational approach," in *2012 IEEE International Conference on Automation Science and Engineering (CASE)*, Aug 2012, pp. 156–161.

[3] ——, "Analysis of human behavior recognition algorithms based on acceleration data," in *2013 IEEE International Conference on Robotics and Automation*, May 2013, pp. 1602–1607.

[4] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems*, vol. 2, no. 4, pp. 303–314, Dec 1989.

[5] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[6] Y. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller, "Efficient backprop," in *Neural Networks: Tricks of the Trade, This Book is an Outgrowth of a 1996 NIPS Workshop*. London, UK, UK: Springer-Verlag, 1998, pp. 9–50. [Online]. Available: http://dl.acm.org/citation.cfm?id=645754.668382

[7] Y. LeCun, I. Kanter, and S. A. Solla, "Second order properties of error surfaces: Learning time and generalization," in *Advances in Neural Information Processing Systems 3*, R. P. Lippmann, J. E. Moody, and D. S. Touretzky, Eds. Morgan-Kaufmann, 1991, pp. 918–924.

[8] D. S. Procter, A. Page, A. R. Cooper, C. M. Nightingale, B. Ram, A. R. Rudnicka, P. Whincup, C. Clary, D. J. Lewis, S. Cummins, A. Ellaway, B. Giles-Corti, D. G. Cook, and C. G. Owen, "An open-source tool to identify active

travel from hip-worn accelerometer, gps and gis data," in *The international journal of behavioral nutrition and physical activity*, 2018.

[9] A. B. D. R. Victor Fragoso, Chunhui Liu, "Do convolutional neural networks act as compositional nearest neighbors?" *CoRR*, vol. abs/1711.10683, 2017. [Online]. Available: http://arxiv.org/abs/1711.10683

[10] D. E. Warburton, C. W. Nicol, and S. S. Bredin, "Health benefits of physical activity: the evidence," *CMAJ*, vol. 174, no. 6, pp. 801–809, 2006. [Online]. Available: http://www.cmaj.ca/content/174/6/801