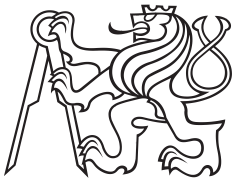
Bachelor Project



Czech Technical University in Prague

F3

Faculty of Electrical Engineering
Department of Cybernetics

Time-Series Classification for Action Detection in Imitation Learning

Michal Mikeska

Supervisor: Mgr. Karla Štěpánová, Ph.D
May 2022

Acknowledgements

I would like to thank Mgr. Karla Štěpánová, PhD as a supervisor for leading and guiding me during my bachelor thesis. I would like to also thank Mgr. Gabriela Šejnová, Ing. Petr Vanc and Kateřina Kubecová for providing the data for this work and precious advices.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

In Prague, May 20, 2022

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze, 20. Května, 2022

Abstract

In this work, we compare different methods for time series classification. The methods cover KNN classification using distances computed by Dynamic Time Warping (DTW), Long-Short Term memory (LSTM), and GRU method. We compare the quality of the classifiers on 4 different datasets, covering hand movements, gestures, movements of robotic manipulator, and audio recordings. We evaluate the results for various settings and parameters of the methods. The difficulty of individual datasets is compared.

Keywords: time series classification, DTW visualisation LSTM, GRU, DTW

Supervisor: Mgr. Karla Štěpánová, Ph.D
Český institut informatiky, robotiky a kybernetiky, CIIRC,
Jugoslávských partyzánů 1580/3,
160 00 Dejvice

Abstrakt

V této práci porovnáváme různé metody klasifikace časových řad. Metody pokrývají klasifikaci algoritmem KNN pomocí vypočítaných vzdáleností Dynamic Time Warping (DTW), Long-Short Term memory (LSTM) a metodou GRU. Porovnáváme kvalitu klasifikátorů na 4 různých datasetech, které zahrnují pohyby rukou, gest, pohyby robotického manipulátoru a audio nahrávky. Výsledky vyhodnocujeme pro různá nastavení a parametry metod. Porovnává se obtížnost jednotlivých datasetů.

Klíčová slova: klasifikace časových řad, vizualizace DTW, LSTM, GRU, DTW

Překlad názvu: Klasifikace časových řad pro rozpoznání akcí v imitačním učení

Contents

Figures

Tables

Chapter 1

Introduction

1.1 Motivation

Time series data can be found all around the world without even noticing. Changes in the weather, flow of money on the stock market or airlines delay statistics. These all are data collected during a time and show the progress of some variable. This representation can serve us to find out more information about the process behaviour affecting the variable changes. Examples of the usage are variable value prediction or variable classification. Both can be very useful and can improve our lives. In this thesis, we will look only into the classification problems.

The application of the time series classification can be seen as action detection in imitation learning. In the not so distant future, it will be necessary to make a robot understand the human more easily than nowadays. One way to improve this is to learn a machine to recognise human gestures, actions and activities. For example, a worker in a factory shows a robot how to work using simple gestures, and the machine starts to work.

If we want to achieve this idea, we need first to compare the classification algorithms. Some of them can be used only in one case, meanwhile others in the second case. Which type of program is suitable for all cases? Is there even the best one? In the last few years, neural networks have risen and become very popular because of their performance.

One type of network is a recurrent neural network (RNN), which suits for work with the time series. These networks can require a large amount of training data and are demanding to train. On the other hand, there are still old fashioned methods for measuring the distance, similarity and other features of the time series. In our thesis, we will work with 2 RNN networks called Long short-term memory (LSTM) and Gated recurrent units (GRU) and with the method Dynamic time warping (DTW), which measures the distance between 2 samples of time series.

Can a program based on those features be better than RNN in the classification problem? We will try to answer this question by comparing those classification methods on four datasets.

1.2 Goals

For a start, we need some data to work with. Our first goal is to create, generate or obtain the datasets. We have to describe and understand data behaviour and see how they perform. We will also prepare them for the algorithms by preprocessing, if it is necessary.

The time series data can be sometimes hard to visualise since there can be more than one dimension (value) to observe. For example, if there were 2-dimensional sequences, it would be the same as displaying 2 different time series. Unfortunately, in most cases, these two values are not independent. Therefore the second goal of the thesis is to visualise the time series data in the way that correlation between individual series can be easily observed. Ideally, we would like to observe the classification algorithms that could cluster the samples in the datasets.

The last and the main goal of the thesis is to compare results of individual classification methods. Performance of the methods with different parameter settings will be evaluated on different prepared datasets. We will discuss if some of the features of the datasets work better than the other ones and try to figure out the reasons. In the end, we try to select the best method for each dataset in the classification problem.

1.3 Related work

This section will focus on previous related work in based-distance measured and deep-learning models, especially applications on classification problems. The development of deep-learning algorithms has been increasing last decades. Studies from Wang et al. [1] contain eight different representations of time series, nine similarity measures and their variants, and testing their effectiveness on 38 time series data sets from various application domains. Dynamic time warping (DTW) and its variants were also included.

Conclusions of DTW mention its effectiveness and simpler implementation compared to the LCSS, EDR and ERP [2]. Jiang's article [2] went even further and compared 1NN classifiers with eight different distance measures and three state-of-the-art deep learning models on 128 time series datasets. Their works show no significant difference between using the 1NN algorithm using DTW metric or edit distance with the real penalty and deep-learning models in their measurement.

1.3.1 Gestures, actions and human moves classification

Another study, from Jenny Cifuentes et al. [3] tries to apply LSTM to hand gestures. Recognition results were based on gesture dynamics, and a comparison of gesture trajectories between novice the expert motion was presented. The LSTM was performed with 99.1% accuracy. *"Exploiting LSTM-RNNs and 3D Skeleton Features for Hand Gesture Recognition"* [4] shows similar results on the 3D skeleton data acquired by the Kinect sensor.

Accuracy led to 92.1%. Shenglin Zhao et al. [5], using the DTW algorithm and recurrent neural network (BiLSTM and GRU), mentions that RNN networks have better classification than DTW. By all those related studies, we can expect better or at least the same accuracy of the RNN network compared to DTW in our project.

In work from Phat Nguyen Huu et al. [6] we see how RNN can improve classification problems by extracting the features from the OpenPose algorithm. His team used the mobilenetV2 backbone and LSTM networks to get corresponding labels of the indoor human poses. In the end, the result led to 99% accuracy.

1.3.2 Audio signals classification

The study from Yinhui Yi et al. [7] focused on music genre classification with the LSTM network. LSTM was used to produce in-depth features from the preprocessed audio signal, and the output was fed to SVM [8] and the KNN classifier. The best result without using LSTM was 61.7 % when combining 2 features extracted before, like zero-crossing rate and Mel Frequency Cepstral Coefficients (MFCC). Meanwhile, with the LSTM, it raised to 98.9 %. This difference shows how the RNN can handle extracting deep features from the sound wave.

DTW was used in D. McGibney's et al. study [9], where he uses KNN with DTW metric for MFCC feature. The KNN boosted the classification system by 24 %, indicating the usefulness of the combination of KNN and DTW.

1.3.3 Prior work at FEE, CTU

In the work of Petr Vanc [10], there is a comparison of the same dataset which we are going to use in our thesis. He used the DTW for computing the distance between the "mean", computed by Probabilistic movement primitives [11] sampler and compared it with the test sample. He does not suggest computing DTW distance to all samples since it is computationally demanding. Our thesis will use this slow computing because we would like to compare the DTW metric performance with the neural networks. Since the metric does

not classify independently, we will build a simple classifier based on the KNN algorithm.

1.4 Outline

First, we will describe the classification algorithms, materials and other methods which we will use in Chapter 2.

In the next Chapter 3, we will describe the data we will use and how we will prepare them for the experimental part.

Then we will perform the experimental part, describe and discuss the results in Chapter 4.

We will compare the classifications and try to suggest the best one per dataset. We will open the topic of future work. This will be done in Chapter 5.

Chapter 2

Material and methods

This chapter will describe which methods, materials and experiments we will use. We will start with the classifications algorithm and their usages. Next, we will focus on the method of audio signal preprocessing since the dataset will consist of sounds. Then we tell which experiments we will run on the data. In the end, we explain how to read the results and compare them with each other.

2.1 Classification algorithms

Generally, we could say that we use two types of methods for classification in our work.

The first one is the method based on employing different time series measures. These measures provide us information like similarity, distance or number of equal samples, depending on the selected method. This information can be then used for clustering or classifying the data. In our thesis, we will work only with one method called Dynamic Time warping (DTW), which measures the distance between 2 samples of time series sequences. The DTW itself is just a metric - it provides with the degree of similarity between series. To use it for classficiation, we will employ K-nearest neighbours algorithm (KNN) with DTW as a metric.

The second method is based on recurrent neural networks (RNN). Unlike traditional neural networks, these networks seem very useful for working with time sequence data. RNN uses the output from the previous state as the input to the new one, which provides better learning through the all sequence. Applications of RNNs cover, for example, speech recognition, music composition and time series prediction. This thesis will compare long short-term memory (LSTM) and gated recurrent units (GRU) networks.

2.1.1 Dynamic Time Warping

Dynamic time warping, also known as DTW, counts the optimal alignment [2] between two time series. If we imagine 2 sequences $X = \{x_0, x_1, \cdots, x_{N-1}\}$ and $Y = \{y_0, y_1, \cdots, y_{M-1}\}$ then DTW algorithm could be count as it is in equation 2.1, where N and M represents length of sequences, $d(x_0, y_0)$ means euler distance between vector x_0, y_0, $\text{Rest}(X) = X \setminus \{x_0\}$ and $\text{Rest}(Y) = Y \setminus \{y_0\}$.

$$\text{DTW}(X,Y) = \begin{cases} \text{if } M = N = 0: \\ 0 \\ \text{else if } M = 0 \text{ or } N = 0: \\ \infty \\ \text{otherwise}: \\ d(x_0, y_0) + min(\text{DTW}(\text{Rest}(X), \text{Rest}(Y)), \\ \text{DTW}(\text{Rest}(X), Y), \text{DTW}(X, \text{Rest}(Y))) \end{cases} \tag{2.1}$$

2.1.2 K-nearest neighbours algorithm

This algorithm, mainly used in statistics, can also be very useful in data classification. K-nearest neighbours algorithm (KNN) holds the labelled data and counts the distances between these data and testing sample with selected metrics. The distances across the whole dataset then decide which class should be the testing sample. The label's decision is made from the K smallest distances from them all. The algorithm selects the label from the majority of the labels represented in the K distances. The pseudocode of 1NN is described below in algorithm 1.

```
Input Training data, test sample
Output label of the testing sample
distance ← -1
for sample in TrainData do
    distance ← DTW(train sample,test sample);
    if distance ≥ result then
        result ←distance;
        result label ← train sample label
    end
end
```

Algorithm 1: 1NN, metric set to DTW, pseudocode.

2.1.3 Long short-term memory

The neural network called Long short-term memory (LSTM) has shown reasonable solutions in classifying time series. The network consists of blocks whose quantity is equal to the length of the sequence. Suppose we have a time series as $X = \{x_0, x_1, \cdots, x_{N-1}\}$. Every block m then takes the output of the previous block $M - 1$ and states x_M as input.[12] Figure 2.1 shows LSTM block architecture.

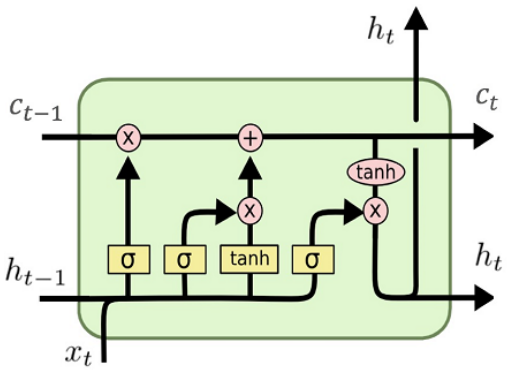


Figure 2.1: LSTM cell with 3 input values c_{t-1}, h_{t-1} from previous state and actual state x_t, where σ is sigmoid and $tanh$ is hyperbolic tangent. Outputs are c_t, h_t which are used in the next step. Source: https://upload.wikimedia.org/wikipedia/commons/9/98/LSTM.png.

2.1.4 Gated recurrent units

GRU is another type of recurrent neural network. It is similar to LSTM, described higher but includes only the update gate and reset gate. GRU was proposed in 2014 by Kyunghyun Cho [13]. Like LSTM, the block of GRU takes previous output with time series value as input and computes the result. Due to a more straightforward architecture, GRU is faster training than LSTM [14]. The Block of GRU is shown in figure 2.2.

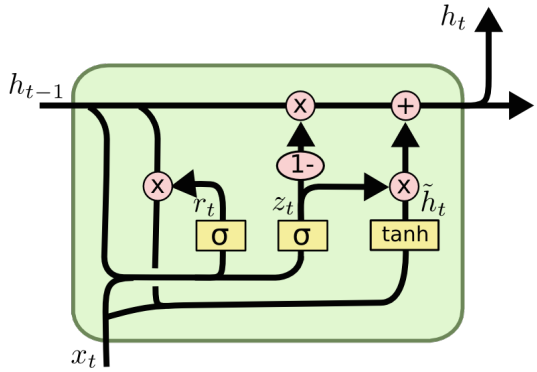


Figure 2.2: GRU cell with only 2 input values h_{t-1} from previous state and actual state x_t, where σ is sigmoid and $tanh$ is hyperbolic tangent. Output is h_t which is used in the next step. Source: https://technopremium.com/blog/rnn-talking-about-gated-recurrent-unit/.

2.2 Signal processing

To enable classification of audio signals, it is necessary to extract from the signal some features. This process procedure may consist of more components. However, to make our comparison more simple, we decided to use only one of them, like J. Guerro-Turrubiates et al. in his work [15], where he did pitch estimation using an artificial neural network. This method is described below 2.2.1.

2.2.1 Fast Fourier transform

The algorithm called The Fast Fourier transform (FFT) performs the discrete Fourier transform (DFT) over the input. The Fourier transform is the integral transform converting the data from time depending on frequency depending. Since the data input cannot be represented in the computer as continuous, we must count discrete transform. Considering the sequence $X = \{x_0, x_1, \cdots, x_{N-1}\}$ as our time series, then the DFT would compute as equation 2.2, where i stands for imaginary unit.

$$y_k = \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{i2\pi}{N}kn} \tag{2.2}$$

The result of the equation is data $Y = \{y_0, y_1, \cdots, y_{N-1}\}$ showing the dependence of frequency. We can use this new data as an input to our algorithms. The example of the FFT is shown in figure 2.3.

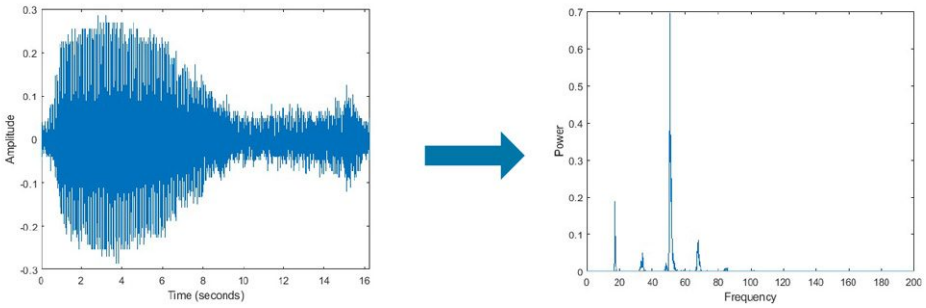


Figure 2.3: The illustration of Fast Fourier transform (FFT). There is an original signal on the right. The new one is shown on the left side. Source: https://www.mathworks.com/discovery/fft.html.

2.3 Experiments description

Our main goal is to compare the algorithms for the classification of our datasets. Then we will also provide a visualisation of datasets where we can observe if it is even possible to use some of the classifiers.

We will compute the DTW distance matrix for all subdatasets (sets) of datasets, which we will prepare in Section Preprocessing 3.2. This matrix can visualise how DTW can hold the datasets and separate them into classes. We will calculate the distances between each sample in the set. An example of this can be seen in figure 2.4. The brighter space means a bigger distance. Meanwhile, the darker one represents close samples. Ideally, we should see the diagonal squares of the single classes, which should be dark, while the rest of the matrix should be bright. Thanks to this visualisation, we can observe the interference between the classes across the whole dataset.

LSTM and GRU neural networks have a couple of hyperparameters that need to be tuned based on the working dataset. The manual tuning does not make sense because it is very time-consuming and random selection is not possible thanks to the high variability of the performance based on the selected parameters. As the solution, we decide to perform the grid search that tries to find the best hyperparameters to fit the training data. We will use the accuracy of the testing data as the index of network perfection. That information can provide us with the best hyperparameters and the best accuracy of the network on our testing data.

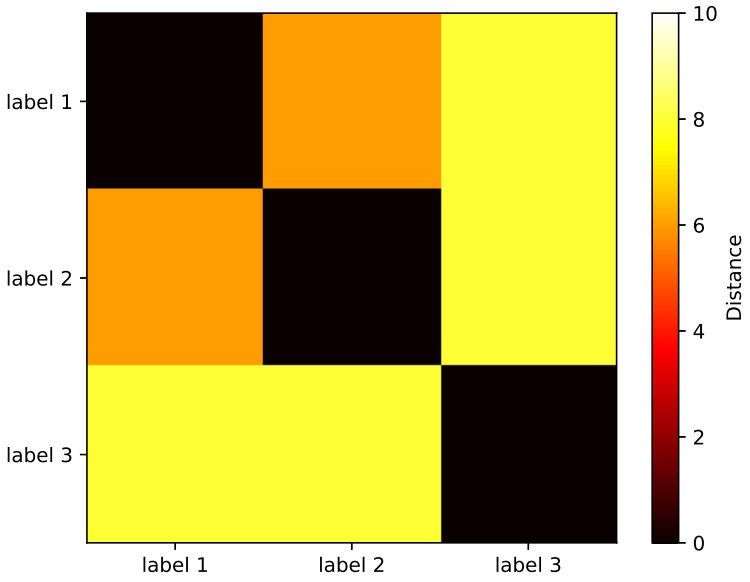


Figure 2.4: The example of DTW distance matrix. The brighter space means a bigger distance. The darker one represents close samples.

2.4 Reading and comparing the results

The confusion matrix seems to be the best way to observe the classification behaviour of the current algorithm. We can efficiently compute that matrix for the LSTM, GRU network and KNN (DTW). We will use the information from the grid searches to build a new model of the neural networks. Then we apply it to the testing data and plot the confusion matrix from the classification results per sample. We also apply testing data to the KNN algorithm, and we will plot the confusion matrix.

2.4.1 Confusion matrix

This matrix can tell us the interference between real labels and the predicted ones. It contains true positive(TP)/negative(TN) and false positive(FP)/negative(FN) values of the classification. There are many variants of this matrix. An example is shown in figure 2.5. The matrix can also contain the information of recall and precision per class. Description is in equation 2.3 and 2.4.

$$Precision = \frac{TP}{TP + FP} \tag{2.3}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.4}$$

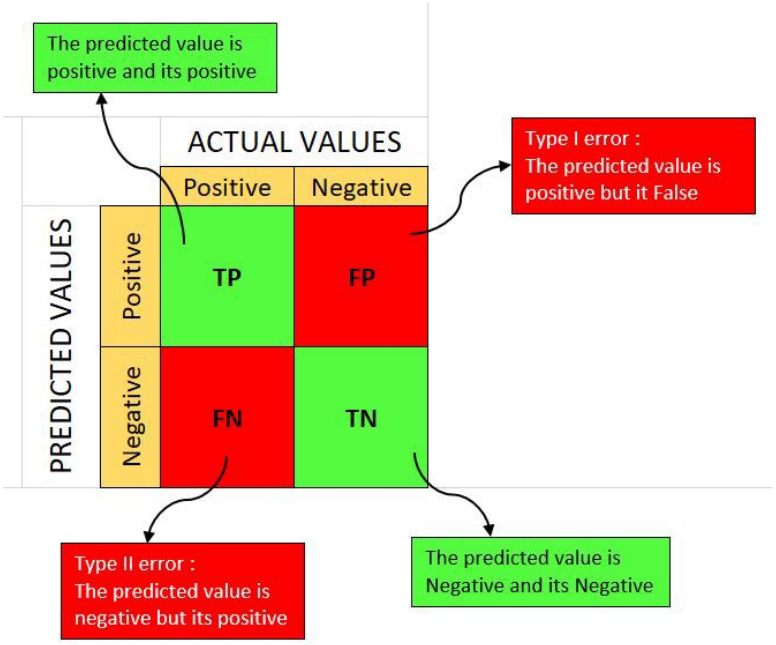


Figure 2.5: The example of confusion matrix. Source:https://medium.com/analytics-vidhya/what-is-a-confusion-matrix-d1c0f8feda5.

Chapter 3

Data collection, preprocessing and dataset creation

3.1 Datasets

We use four datasets for our thesis. Two of them were obtained from the previous works. One is from G. Šejnová [16], produced by an OpenPose algorithm [17] from the pepper humanoid robot's camera data. We call it the Human skeleton dataset (Skeleton). The other one, that we named the Hand gesture dataset (Hand), comes from P. Vanc [18], which consists of hand gestures. Then we generated one, called CoppeliaSim dataset (Coppelia) in the simulator CoppeliaSim via the code provided by K. Kubecová. This dataset contains information about the robot moving the object. We created the fourth one. It is named Bottle moving dataset (Bottle), describing the movement of the bottle using the sound records. The summary of the datasets and their prepared subdatasets (sets) can be seen in table 3.1

Dataset	Set	Dim	Train	Test
Hand	50:50	4	150	150
Hand	60:40	4	181	119
Hand	70:30	4	211	89
Hand	80:20	4	238	62
Hand	90:10	4	270	30
Bottle	Max	2	500	75
Bottle	Fft	2	500	75
Coppelia	Forces	7	500	75
Coppelia	ObjOri	3	500	75
Coppelia	ObjPos	6	500	75
Coppelia	ObjOriPos	3	500	75
Coppelia	TipPos	3	500	75
Coppelia	TipMinObjPos	3	500	75
Skeleton	Skeleton	8	60	30

Table 3.1: Review of the all prepared datasets. Set means part or prepared dataset for our algorithms. Dimension (Dim) stands for a number of the values in each timestamp. The Train represents the amount of all training data. The Test shows the number of all testing data in the set of the dataset.

3.1.1 Hand gesture dataset

In our project, we operated with a hand gestures dataset created by Petr Vanc [18] recorded via Leap Motion sensor. One gesture is represented by time series with length of 101. The number of values in one state equals 4. These values are normalized coordinates of palm in Cartesian coordinate system x, y, z and the last one is the euclidean distance d from palm to point fingertip. Data are described in equation 3.1.

$$\begin{aligned} &\text{Hand} = \{H_i\} && i = 1, ..., 300 \\ &H_i = [\vec{h}_i^1, ..., \vec{h}_i^T] && T = 100 \\ &\vec{h}_i^t = [x_{i,t}, y_{i,t}, z_{i,t}, d_{i,t}] \end{aligned} \tag{3.1}$$

Hand stands for all data, H_i represents a single sample. T means the biggest time t of the samples. The t is time, therefore concrete $\vec{h}_i^t$ is one timestamp. The last line shows the meaning values (Cartesian coordinates and euclidean distance from palm to point fingertip) in each timestamp.

Data contains dynamic and static gestures. For our case, we decided to work only with dynamic gestures. They are *called pin, rotate, touch, swipe left/right* and *swipe up/down*. For distribution of the gestures see table 3.2. Dynamic gestures can be also seen as a movement of the hand when the user is moving it across the whole working space. On contrary, in the case of static gestures, the user moves only with his fingers, and the palm is located in one place, moving minimally. Therefore, dynamic gestures are better for

comparing our algorithms because the palm values change significantly in time.

The dataset contains 301 samples for dynamic gestures. There are seven types of motion with a variable amount of samples. This instability could lead to the suppression of less represented data while classifying or clustering. Detailed data distribution is shown in table 3.2.

Label	No. of samples	No. of dim.
pin	28	4
rotate	33	4
touch	38	4
swipe left	51	4
swipe right	54	4
swipe up	38	4
swipe down	59	4

Table 3.2: Distribution of individual dynamic gestures in the dataset.

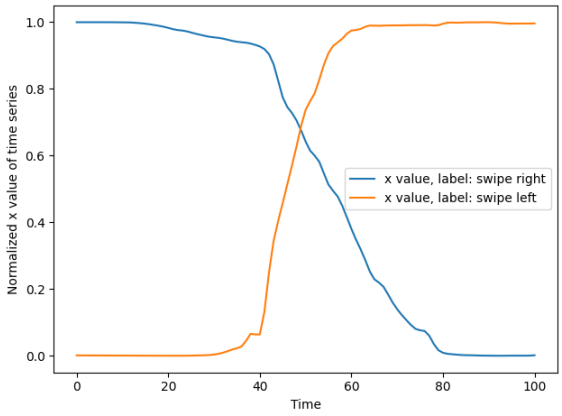


Figure 3.1: Example of time series for two dynamic gestures *swipe right,swipe left*. The x value is shown.

The demonstration of this dataset could be problematic due it is four-dimensional. Therefore we decided to plot only one value, which differs across the classes. We can see the x value comparison of two different labels in figure 3.1. We decided to show left and right-swiping. We observe the most significant difference, because *swipe right* and *swipe left* are opposite gestures.

We can recognize the same motion represented by two sequences in figure 3.2. The time series are not the same, but the patterns in time are very similar.

3.1.2 Bottle moving audio dataset

The world can give us lots of information about its behaviour, which can be used to classify some phenomena. The previous dataset describes the

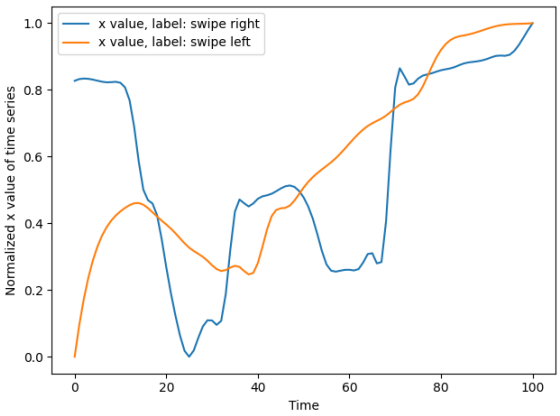


Figure 3.2: Comparison of two different samples of the same dynamic gesture *pin*. The x value is shown.

movement of the hand by spatial coordinates. If this hand gesture would create the sound, for example, while moving some object, we can use this additional information for classifying. Therefore, we created a dataset containing sounds of bottle movements on the wooden table to see how sound can be helpful in classification.

The description of the dataset can be seen in equation 3.2. We also created a new set of datasets in the Preprocessing section 3.2.2 for better performance of classification. They are described in equation 3.3 and named fft and max, based on their features. We used a laptop microphone with two channels for recording. We recorded for 1.5 seconds with a frame rate of 1000 frames per second leading to the samples with the length 1500 frames. Dataset consists of 100 training and 15 testing samples for each class. Data distribution is described in table 3.3.

$$\begin{aligned} &\text{Original} = \{D_i\} && i = 1, ..., 575 \\ &D_i = [\vec{d_i^1}, ..., \vec{d_i^T}] && T = 1500 \\ &\vec{d_i^t} = [ch_{i,t}^1, ch_{i,t}^2] \end{aligned} \tag{3.2}$$

D_i represents a single sample. T means the length of the longest sample. The t is time, therefore concrete $\vec{d_i^t}$ is one timestamp. The last line shows the meaning values (microphone channels) in each timestamp.

$$
\begin{aligned}
&\max = \{M_i\} && i = 1, ..., 575 \\
&M_i = [\vec{m}_i^1, ..., \vec{m}_i^T] && T = 500 \\
&\vec{m}_i^t = [\max(ch_{i,t}^1, ch_{i,t+1}^1, ch_{i,t+2}^1), \max(ch_{i,t}^2, ch_{i,t+1}^2, ch_{i,t+2}^2)] \\
&\\
&\text{fft} = \{F_i\} && i = 1, ..., 575 \\
&F_i = [\vec{f}_i^1, ..., \vec{f}_i^T] && T = 750 \\
&\vec{f}_i^t = [\text{fft}(ch_{i,t}^1), \text{fft}(ch_{i,t}^2)]
\end{aligned}
\tag{3.3}
$$

fft stands for all Fast Fourier transform data. Max stands for all max set data. M_i, F_i represents a single samples. The $\vec{m}_i^t, \vec{f}_i^t$ are one timestamps of the data. T means the length of the longest sample. The t is time.

The vector $[\max(ch_{i,t}^1, ch_{i,t+1}^1, ch_{i,t+2}^1), \max(ch_{i,t}^2, ch_{i,t+1}^2, ch_{i,t+2}^2)]$ represents downsampling and selecting the maximum of the channels values in time t in sample i. The vector $[\text{fft}(ch_{i,t}^1), \text{fft}(ch_{i,t}^2)]$ stands for Fast Fourier transform of the value of the channel in time t in sample i.

label	No. of training samples	No. of testing samples	No. of dim.
P	100	15	2
D	100	15	2
I	100	15	2
Z	100	15	2
O	100	15	2

Table 3.3: Distribution of the Bottle dataset.

There are five classes called by individual letters *P, D, I, Z* and *O*. The letters represent the shape of the movement. We can imagine this as writing a letter with a pencil, but we use the bottle instead of the pencil. In our conditions, human ears could not recognise the difference between similar gestures like pushing forward and backwards. This feature depends on the material of the bottle and surface. Visualisation of the raw data is shown in figure 3.3. We can hardly observe differences between single classes except for the letter *I*, which protrudes. We can also see dissimilarity between the letters *P* and *D* compared to *Z* and *O*.

3.1.3 CoppeliaSim dataset

Experiments in the real world are still increasingly inefficient, expensive and time-consuming. On this side, the simulation comes to the scene. We created the new time series dataset with the simulation program called CoppeliaSim.

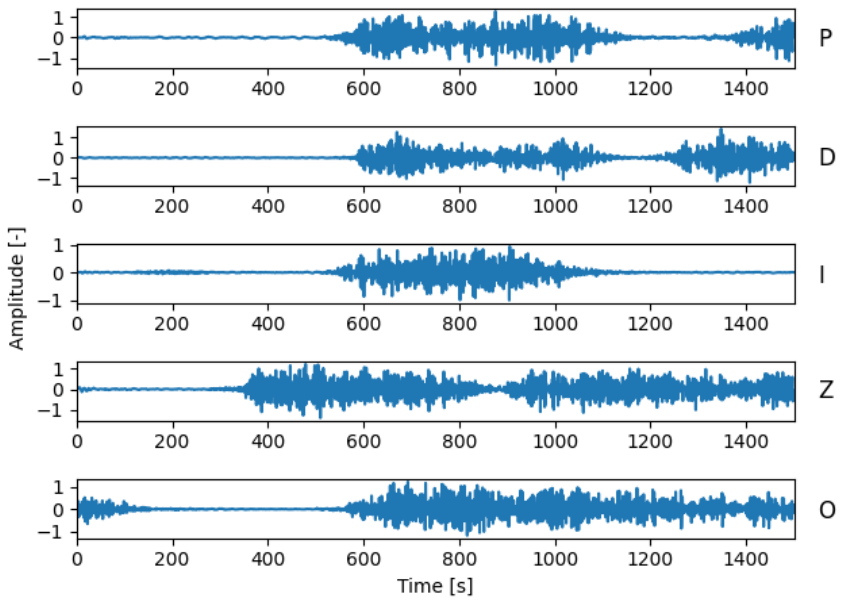


Figure 3.3: Example of raw Bottle dataset, all letters. The first channel is shown.

Our concrete simulation was provided by simulation workspace miracle_sim [19], which simulates 7DoF-axis mechanical arm robot. We used the work of K. Kubecová [20], who programmed the simulation to create the data. We used her code and generated a new data collection.

feature	No. of values
tip position	3
joints velocities	7
joints forces	7
joints angles	7
object orientation	3
object position	3
gripper info	2

Table 3.4: Dimensions of each feature in the CoppeliaSim dataset.

The dataset consists of the robot motion and action with the block. There are three labels (actions): *push, bump* and *lift*, describing the interaction between the robot's arm and the object, in our case, the cube. We collected 115 samples per class and split the data into the training (100 samples) and the testing (15 samples) data. The same division we made in the Bottle dataset 3.1.2. The creation of the one sample can be seen in figure 3.4.

The Bottle dataset (see Sec. 3.1.2) held one type of data with different dimensions. The simulator allowed us to measure more information in time. Thanks to this feature, we collected time series of gripper force and position,

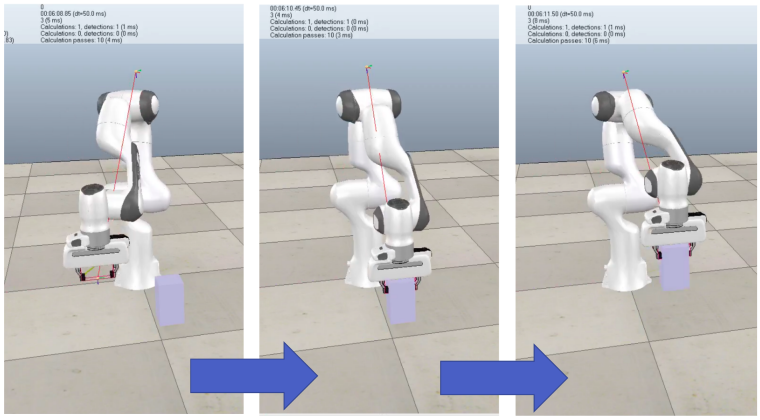


Figure 3.4: Example of the object lifting in the CoppeliaSim simulator.

tip position, joint velocities, angles and forces, object orientation and position. Dimensions for each feature are described in table 3.4. This may lead to combinations of those features like adding or computing new data, which can help with the classification and show us the importance of the original single information. We created sets of dataset called: Forces, ObjOri, ObjPos, ObjOriPos, TipPos, TipMinObjPos. The detailed information can be seen in the equations 3.4. The creation of the new prepared datasets is described in section 3.2.3. Table 3.1 shows the final datasets' dimensions and numbers of the testing and training data.

$$
\begin{aligned}
&\text{Forces} = \{A_i\} && i = 1, ..., 345 \\
&A_i = [\vec{a}_i^1, ..., \vec{a}_i^T] && T = 47 \\
&\vec{a}_i^t = [f_{i,t}^1, f_{i,t}^2, f_{i,t}^3, f_{i,t}^4, f_{i,t}^5, f_{i,t}^6, f_{i,t}^7] \\
\\
&\text{ObjOri} = \{B_i\} && i = 1, ..., 345 \\
&\vec{b}_i = [\vec{b}_i^1, ..., \vec{b}_i^T] && T = 47 \\
&\vec{b}_i^t = [angle_{i,t}^x, angle_{i,t}^y, angle_{i,t}^z] \\
\\
&\text{ObjPos} = \{C_i\} && i = 1, ..., 345 \\
&C_i = [\vec{c}_i^1, ..., \vec{c}_i^T] && T = 47 \\
&\vec{c}_i^t = [x_{i,t}^{obj}, y_{i,t}^{obj}, z_{i,t}^{obj},] \\
\\
&\text{ObjOriPos} = \{D_i\} && i = 1, ..., 345 \\
&D_i = [\vec{d}_i^1, ..., \vec{d}_i^T] && T = 47 \\
&\vec{d}_i^t = [x_{i,t}^{obj}, y_{i,t}^{obj}, z_{i,t}^{obj}, angle_{i,t}^x, angle_{i,t}^y, angle_{i,t}^z] \\
\\
&\text{TipPos} = \{F_i\} && i = 1, ..., 345 \\
&F_i = [\vec{f}_i^1, ..., \vec{f}_i^T] && T = 47 \\
&\vec{f}_i^t = [x_{i,t}^{tip}, y_{i,t}^{tip}, z_{i,t}^{tip}] \\
\\
&\text{TipMinObjPos} = \{E_i\} && i = 1, ..., 345 \\
&E_i = [\vec{e}_i^1, ..., \vec{e}_i^T] && T = 47 \\
&\vec{e}_i^t = [abs(x_{i,t}^{obj} - x_{i,t}^{tip}), abs(y_{i,t}^{obj} - y_{i,t}^{tip}), abs(z_{i,t}^{obj} - z_{i,t}^{tip})]
\end{aligned}
\tag{3.4}
$$

The T means the length of the longest sample. The t is time, therefore concrete $Dataset_i^t$ is one timestamp of the data. The last line shows the meaning values in each timestamp. The vector $[f_{i,t}^1, f_{i,t}^2, f_{i,t}^3, f_{i,t}^4, f_{i,t}^5, f_{i,t}^6, f_{i,t}^7]$ stands for the forces torque sensor readings from the individual robot joints in sample i at the time t. The vector $[angle_{i,t}^x, angle_{i,t}^y, angle_{i,t}^z]$ means the object orientation angles measured around the axis in sample i at the time t. The vector $[x_{i,t}^{obj}, y_{i,t}^{obj}, z_{i,t}^{obj}]$ represents object position in the euclidean coordinates system in sample i at the time t. The vector $[x_{i,t}^{tip}, y_{i,t}^{tip}, z_{i,t}^{tip}]$ stands for the tip position in the euclidean coordinates system in sample i at the time t.

3.1.4 Human skeleton dataset

This dataset was produced by the OpenPose algorithm [17] from data coming from Pepper humanoid robot's camera, scanning the person who performs different moves. The goal is to recognise the move from the time series where each timestep consists of 8 values. These values are computed from the RGB picture that the robot scanned. The characters in order are Left Shoulder Roll, Left Elbow Roll, Right Shoulder Roll, Right Elbow Roll, Left Arm Direction, Right Arm Direction, Left Hand Open, and Right Hand Open. Roll values stand for joint angles in degrees. Arm directions are a binary

Figure 3.5: Example of the picture from Pepper humanoid robot's camera with the skeleton structure, which was produced by the OpenPose algorithm

value since we cannot extract an angle. The value 100 stands for arm pointing upwards meanwhile 0 means pointing downwards. We apply the same rules for open hand information where value 0 is a closed hand, and 100 is an open one. The detailed information can be found in equation 3.5. There are three moves, called *dance, fly* and *wave*. The dataset consists of 20 train samples and 10 test samples per class. The ratio of the training and testing data can be seen in table 3.1.

$$
\begin{aligned}
&\text{Skeleton} = \{D_i\} && i = 1, ..., 300 \\
&D_i = [\vec{d}_i^{\,1}, ..., \vec{d}_i^{\,T}] && T = 33 \\
&\vec{d}_i^{\,t} = [LeftShoulderRoll_{i,t}, LeftElbowRoll_{i,t}, RightShoulderRoll_{i,t}, \\
&RightElbowRoll_{i,t}, LeftArmDirection_{i,t}, RightArmDirection_{i,t}, \\
&LeftHandOpen_{i,t}, RightHandOpen_{i,t}]
\end{aligned}
\tag{3.5}
$$

D_i represents a single sample. T means the length of the longest sample. The t is time, therefore concrete $\vec{d}_i^{\,t}$ is one timestamp. The last line shows the meaning values in each timestamp in sample i.

3.2 Data Preprocessing

Processes in nature are rough, varied and unstable, which leads to the raw measured signals. We can use these data, but we may also extract some features using algorithms or create a data selection for better performance of our classifiers.

3.2.1 Hand gesture dataset

This dataset does not contain training and testing data separately, so we must split it. The suggested ratio we will use to compare the classification will be 70:30, which means that we split our data into training data (70% for each class) and testing data (30% for each class). We also decided to create datasets with different ratios. We can see how algorithms will manage the different sizes of the training and testing data by those combinations. The new datasets distribution is shown in table 3.5. The algorithm provided this splitting takes N, described in equation 3.2.1, random samples from each class and sets them as new testing data. Meanwhile, the rest of the samples are training one.

$$N \doteq \frac{(\textit{wanted percentage of testing data}) \cdot (\textit{current class size})}{100} \tag{3.6}$$

ratio:	50:50		60:40		70:30		80:20		90:10	
label	train	test	train	test	train	test	train	test	train	test
pin	14	14	17	11	20	8	22	6	25	3
rotate	16	16	20	13	23	10	26	7	30	3
touch	19	19	23	15	27	11	30	8	34	4
swipe left	26	26	31	20	36	15	41	10	46	5
swipe right	27	27	32	22	38	16	43	11	49	5
swipe up	19	19	23	15	27	11	30	8	34	4
swipe down	30	30	35	24	41	18	47	12	53	6

Table 3.5: New hand gesture datasets from splitting the original data into training and testing. The ratio of each dataset is shown with the data distribution for each class. Numbers represents the amount of the training and testing samples.

3.2.2 Bottle moving audio dataset

As we said before in the description of the data 3.1.2, we cannot observe big contrast between each sound. That is the first reason we have to change the data for the computer, so it works better. The second reason is the

computational time which we may decrease by shrinking the data while also extracting some features.

We started by computing the median, maximum and mean of the amplitude of the defined calculating window. For example, we can set the window to 3 points which leads to shortening the signal 3x times and selecting the feature mentioned before from those points. In figure 3.6 we observe the letter *D* modified.

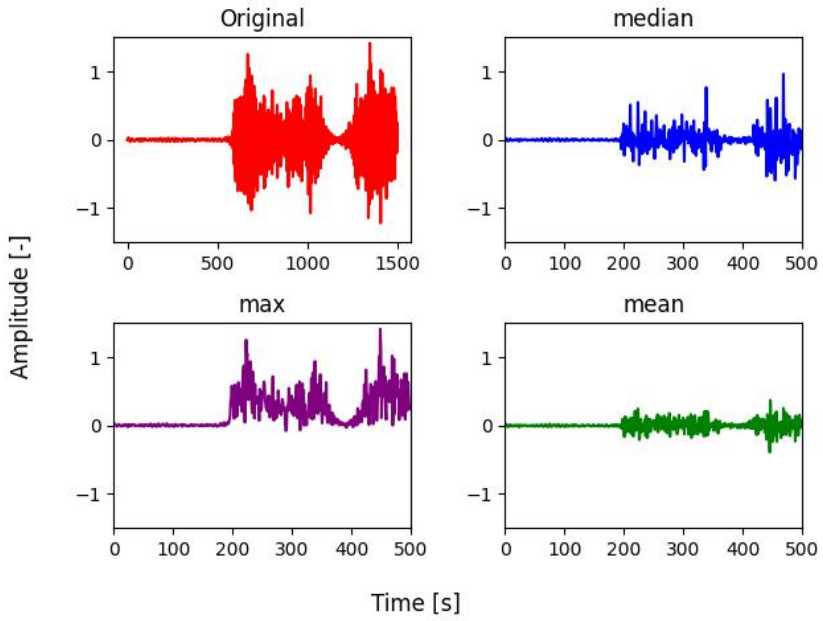


Figure 3.6: Audio signal for letter *F*. Example of the raw and modified data. The first channel is shown.

The shape of the signal is significant in our work which means if the shapes of each class diverges, the classifier will work better. Because of that, the selection of maximal amplitude seems to be the best decision. We can also try median or mean, but their shapes are very similar to the original signal. We may also try the raw data as the input and see the result. We decided to use the max feature since the shape difference is the most significant. We will name this subdataset Max.

Another practical algorithm which we have decided to use is called Fast Fourier transform 2.2.1 which changes our data from amplitude-time depending to amplitude-frequency depending. This transformation may bring us a new dataset where all labels are shaped in significant contrast. The original length of the signal is 1500 points. We compute only real numbers as input. Then the output will be half (750 points) of the original length because we omit the negative frequencies since they are complex conjugates of the corresponding positive-frequency terms. We will name this subdataset Fft. Thanks to this attribute, the computing speed will also increase. An example of the Fft is shown in figure 3.7.

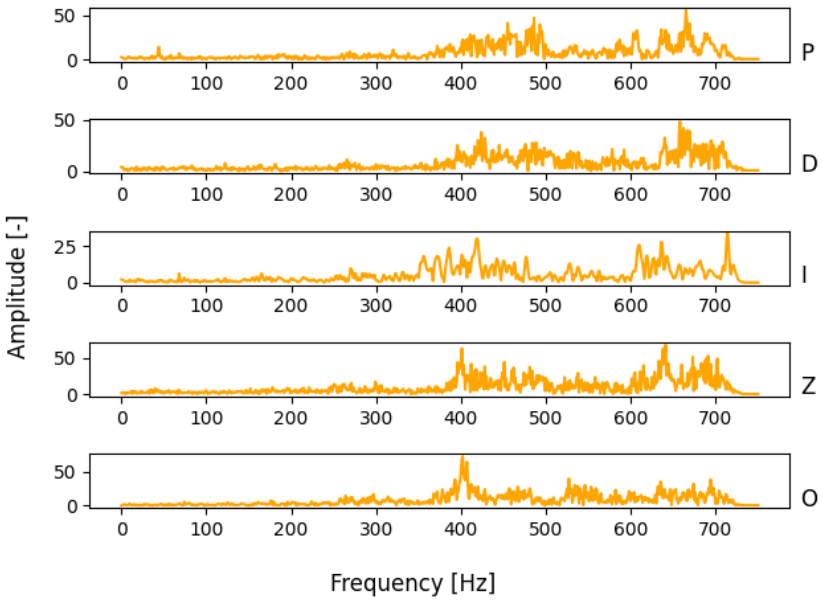


Figure 3.7: Example of the Fast Fourier transform of all letters. The first channel is shown for 1 sample per class.

3.2.3 CoppeliaSim dataset

Since there is a complex data structure, we must decide which type of information we will use. The actions are based on the object movement, which will lead us to use the position information of the object since there are actions called *push* and *bump*, which can be similar. We create one dataset containing only the object's orientation and position separately. We call them ObjOri and ObjPos. Those datasets should work better if we join them since they give the total movement information about the object. We name this new dataset ObjOriPos.

If we imagine we have to work without the moving object's proximity sensor, we may want to use other available information. From this point, we also decide to create separate datasets from the joints torque sensor signals of the forces. We named it Forces. Then there is also tip position which should provide us with better position information, we call it TipPos.

The last one we made was based on the dataset's author's recommendation. It was done by computation between the tip's position and the object. We produce the difference between those coordinations and create new information. We will call it TipMinObjPos. All those datasets: Forces, ObjOri, ObjPos, ObjOriPos, TipPos, TipMinObjPos are detailed and described in the previous section 3.1.3 in equations 3.4.

Chapter 4

Results

This chapter will display the results of the classification and visualisation of the dataset sets. There will also be a discussion about them. We did not perform all algorithms for all sets because some had lousy visualisation or wrong grid search results. In table 4.1 we observe the best accuracy of the selected dataset sets primarily from the confusion matrices. We chose those sets to represent the datasets since they could be performed by most of the algorithms with valuable results.

Dataset	Set	KNN (DTW)	LSTM	GRU
Hand	80:20	95%	87%	90%
Bottle	Max	-	83%	84%
Bottle	Fft	-	63%	76%
Coppelia	Forces	93%	62%	64%
Coppelia	ObjOriPos	89%	51%	56%
Coppelia	TipMinObjPos	100%	41%	66%
Skeleton	Skeleton	100%	77%	83%

Table 4.1: Summary of the datasets sets accuracy for the chosen algorithms. If we create the confusion matrix, its accuracy is written. If there is none, we take the number from the grid search or non-displayed confusion matrix for KNN (DTW).

4.1 DTW

We performed for all sets of the dataset DTW distance matrix, which can visualise the distance interference between each sample. The most contrast between all samples can be seen for the Hand gesture dataset in figure 4.1 and for the Human skeleton dataset in figure 4.8. If the distance matrix can display the difference between the classes then we also did a confusion matrix for KNN algorithm with the DTW metric.

4.1.1 Hand gesture dataset

We created one DTW distance matrix in figure 4.1 for all samples of this dataset. The contrast between the classes can be observed, so we also performed KNN (DTW) confusion matrix. The result may be seen in figure 4.2 for set 80:20. The accuracy raised to 95%. We also performed the KKN (DTW) for the rest of the dataset sets to see how the accuracy changed with the splitting. The result is in table 4.2.

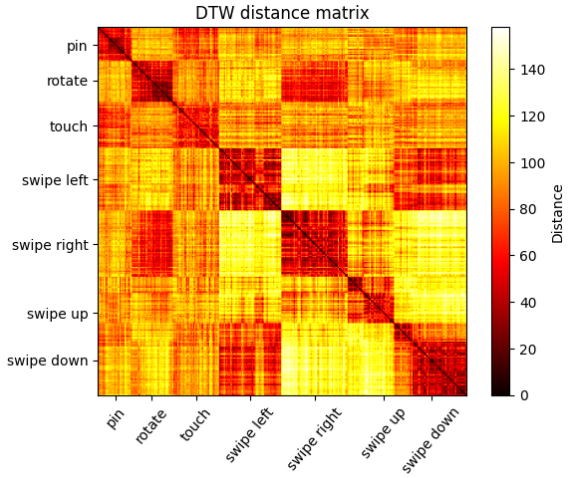


Figure 4.1: DTW distance matrix where each cell represents the distance between two samples of the Hand gesture dataset.

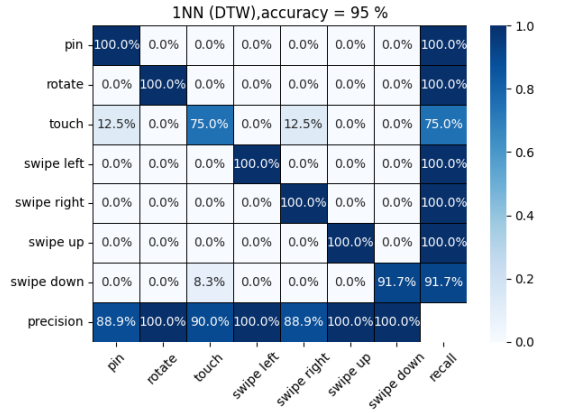


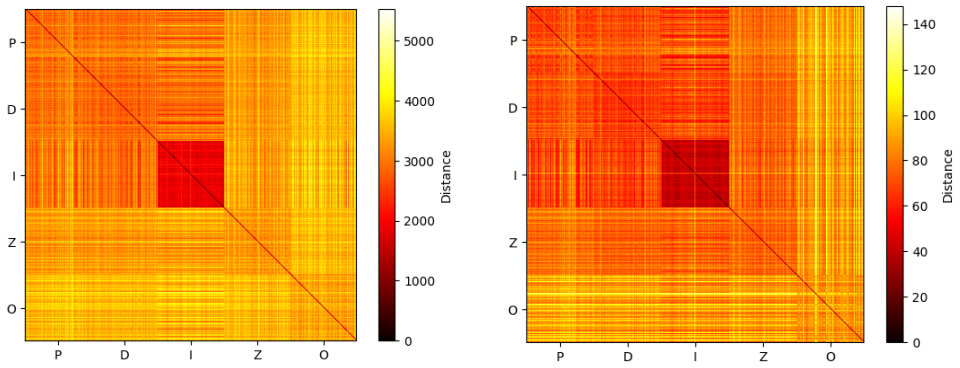
Figure 4.2: Confusion matrix of the Hand gesture dataset, processed by KNN (DTW). Dataset set 80:20 is shown, $K = 1$.

Dataset	Set	Accuracy
Hand	50:50	93%
Hand	60:40	93%
Hand	70:30	94%
Hand	80:20	95%
Hand	90:10	95%

Table 4.2: Table of the KNN (DTW) accuracy for different splitting in the Hand gesture dataset, $K = 1$.

4.1.2 Bottle moving dataset

We did two confusion matrices for this dataset. The first one is for set max (figure 4.3b) and the other for the fft (figure 4.3a) feature. We did not evaluate KNN with DTW metric as for this size of dataset (500-700 timestamps per sequence) the computational time is so high (several hours) that it cannot be used in any real application.

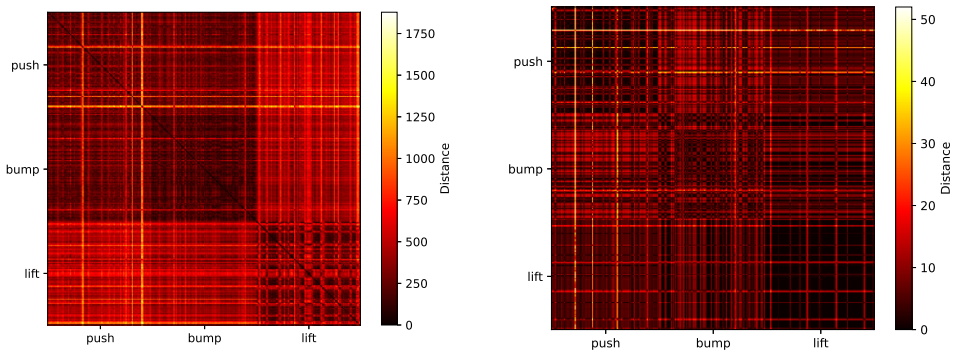


(a) : Fft set **(b) :** Max set

Figure 4.3: DTW distance matrices of the Bottle moving max dataset sets. Fft set is on the left and Max set is on the right. Each cell represents the distance between two samples.

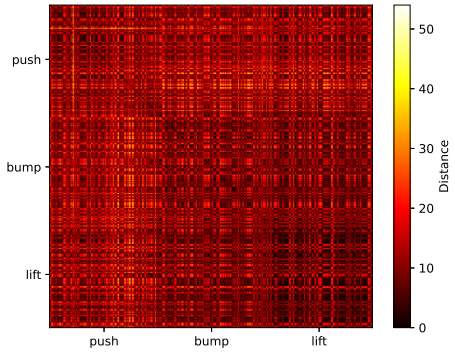
4.1.3 CoppeliaSim dataset

For these dataset sets, we created only visualisation. Since the dataset size is big, it is inefficient to use KNN (DTW). We plotted matrices 4.4a-4.4f for the Forces, ObjOri, ObjPos, ObjOriPos, TipPos and TipMinObjPos. The best contrast for all classes can be seen in 4.4f. The worst one was is in fig. 4.4d because, we have to set a maximum limit of the distance not to observe one big black square. The rest of the results are similar except for 4.4a, where two squares (labels) can be seen. We also performed DTW (KNN) algorithm and plotted its confusion matrix for sets Forces, ObjOriPos and TipMinObjPos, which can be seen from figure 4.5 to figure 4.7.

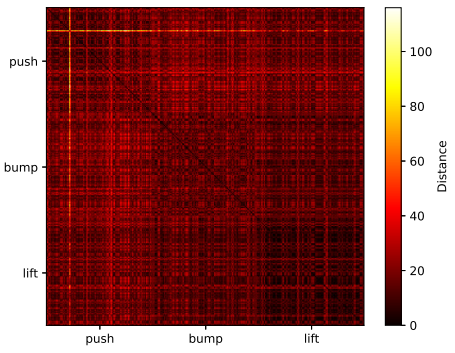


(a) : Forces set

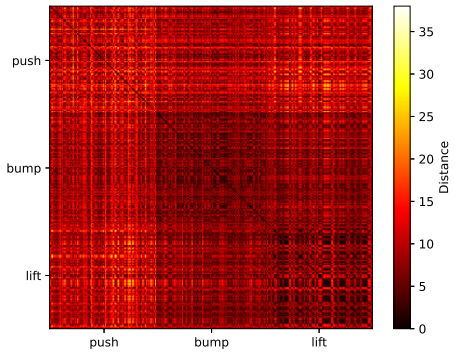
(b) : ObjOri set



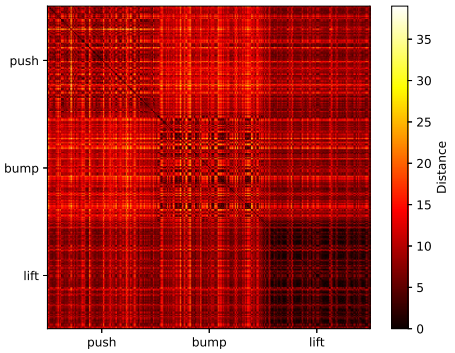
(c) : ObjPos set



(d) : ObjOriPos set



(e) : TipPos set



(f) : TipMinObjPos set

Figure 4.4: The GRU grid search graphs of the CoppeliaSim dataset sets. A higher score means better accuracy of the testing data in the current configuration. Colour represents the hidden size. Meanwhile, the line pattern stands for the num. of layers.

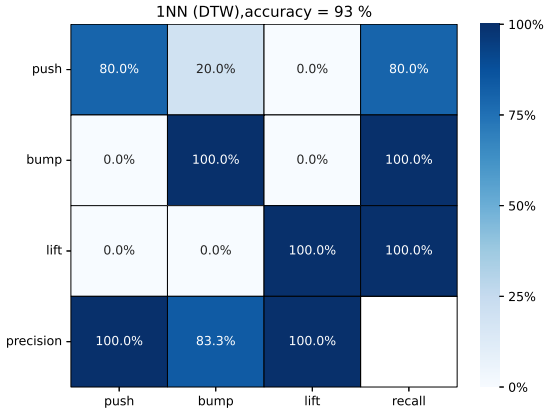


Figure 4.5: Confusion matrix of the CoppeliaSim dataset set Forces, processed by KNN (DTW). There are 15 testing samples per class, $K = 1$.

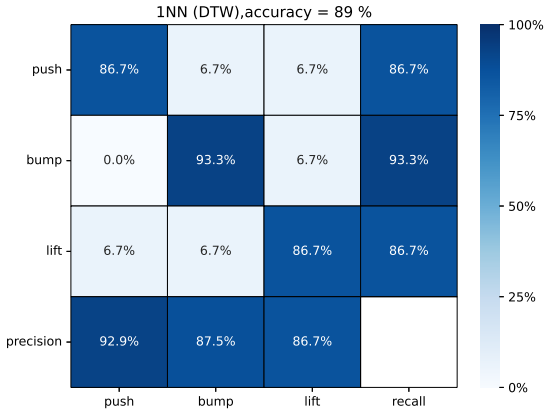


Figure 4.6: Confusion matrix of the CoppeliaSim dataset set ObjOriPos, processed by KNN (DTW). There are 15 testing samples per class, $K = 1$.

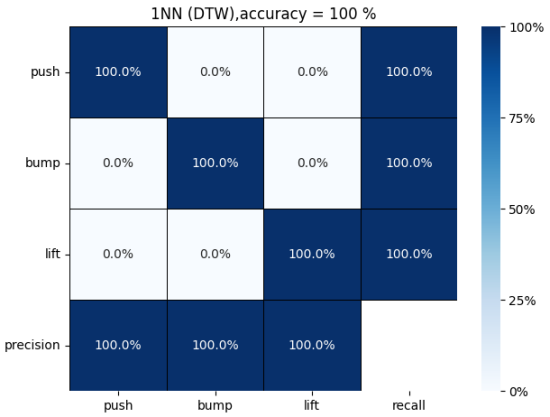


Figure 4.7: Confusion matrix of the CoppeliaSim dataset set TipMinObjPos, processed by KNN (DTW). There are 15 testing samples per class, $K = 1$.

4.1.4 Human skeleton dataset

This dataset is small, therefore both DTW visualisation of distances (see fig. 4.8) as well as results of KNN (see fig. 4.9) are visualised.. Accuracy raised to 100%, and the contrast in the distance matrix is impressive.

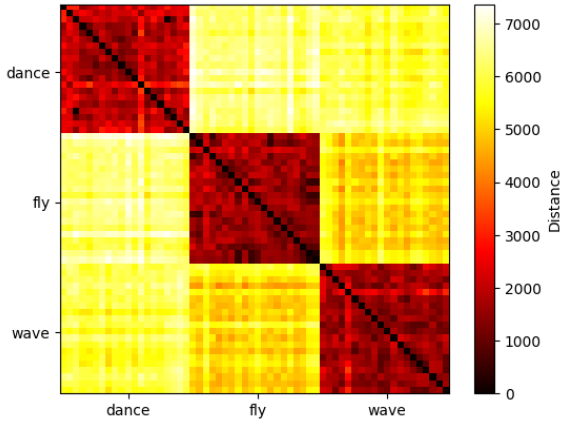


Figure 4.8: DTW distance matrix of the Human skeleton dataset. Each cell represents the distance between two samples.

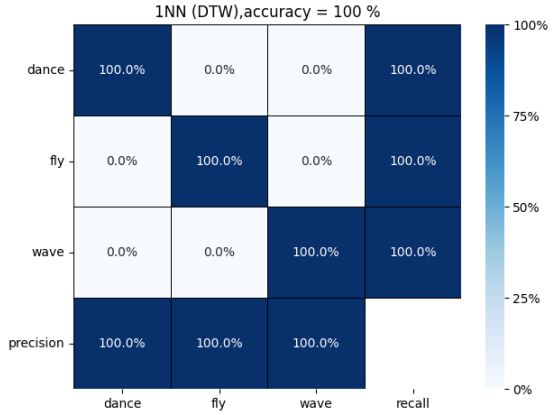


Figure 4.9: Confusion matrix of the Human skeleton dataset, processed by KNN (DTW). There are 10 testing samples per class, $K = 1$.

4.1.5 Discussion

DTW visualisation

We visualised all datasets with the distance matrix, where each cell correspondent to the DTW distance between two samples in the selected dataset. Therefore we can observe how the DTW metric can be helpful if we decide to

use it in some chosen algorithm. This performance has no accuracy since this matrix only shows clustered classes by DTW.

In figure 4.1 we observe seven dark squares that represent non-identical gestures in the Hand gesture dataset. The square size differs because we performed the visualisation through the whole data. Since the Hand gesture dataset is not evenly distributed, the squares hold the divergent number of samples. We can note that some of the gestures can be connected by DTW, whereas others are cleared. That can be observed as the darker places in the rest of the matrix. Gesture *pin* is closer by the distance to the gesture *touch* therefore, we assume misclassification by those two classes if we use the DTW metric. We inspect the exact similarity of the distances between pairs of the labels *rotate, swipe right/left, swipe down*.

Meanwhile, we could see the working visualisation on the Hand gesture dataset. We observe something different in figures 4.3a and 4.3b. Those figures stand for the Bottle moving dataset. We recognise only one clear square, the letter *I*. Complex data points can cause this performance, and since the letter I stands out even in figure 3.3 then DTW marks it. The distances measured on the Fft set are much higher compared to the values in the maximum amplitude.

We also see that in figure 4.3b letters *P,D,I* and Z are close; meanwhile, in figure 4.3a only the first three of them are. We can use this information in future when we build some classification algorithms. For example, by mixing the feature max. amplitude (set Max) and Fft set together.

We also performed the DTW visualisation for the CoppeliaSim dataset. We have six sets, and we did the matrices for all of them. They are represented in figure 4.3. As we see in those figures, there are no single squares for the first view, which we can follow. In figure 4.4a we see that the DTW algorithm can create only two labels. The first one is mixed with *push* and *bump*. This can signify that the DTW would classify *lift* separately for this set. Meanwhile, *push* and *bump* would be joined together. There are not any significant squares from figures 4.4b to 4.4e. Excpet for set ObjOriPos, where *lift* seems to have its square.

For set ObjOri, one distance came so high that the colour bar was not enough to observe anything. Therefore we had to set a maximum distance to 50. This anomaly shows us that this set can contain an inaccurate measurement or a rare sample. In the last figure 4.4f for DTW matrices, we can observe evidence of some squares. The action *lift* has its square, which means the classification algorithm, built on the DTW metric, would recognise *lift* with a high probability.

DTW visualisation came up clear for the Human Skeleton dataset. We detect three squares and no interference between the samples in figure 4.8. This result could lead to functional classification using DTW.

KNN (DTW)

To keep our algorithm simple, we use $K = 1$. Because this algorithm can be used only for the smaller datasets, we performed it only on the Hand gesture dataset, CoppeliaSim dataset and its variations and on the Human Skeleton dataset. Figure 4.2 shows us the confusion matrix for set 80:20. The accuracy is 95% and for 4 gestures: *rotate,swipe left,swipe up,swipe down*, the precision came to 100%. Those gestures were always classified correctly, and none of the other gestures was misclassified as one of them. Gestures *touch* and *pin* were mixed, as it can also be seen in the DTW matrix in figure 4.1. The same came up with the labels *touch* and *swipe down*.

We also did the KNN (DTW) for the other sets. Their accuracy can be seen in table 4.2. There is no big difference between the accuracy across the sets. It can be caused by KNN, which does not learn but only counts the distances across the whole dataset. Since the samples are close distanced, as we see in fig 4.1 there is no requirement for a larger amount of training samples. The quality of the training sample is important. Ideally, use one computed average sample per class.

We created the KNN (DTW) confusion matrix also for In figures 4.5 to 4.7 we observe that even DTW visualisation (see fig. 4.4) seems to have bad performance, the KNN (DTW) could classify set TipMinObjPos with accuracy 100%. Then we assume that this set could be handy while using KNN (DTW) for future work.

We also generate the confusion matrix using KNN (DTW) for the Human skeleton dataset. The result is shown in figure 4.9, where accuracy came to 100%, which means that all moves were classified correctly. This performance could be predicted from the DTW distance matrix in figure 4.8, where all samples are split into the clear dark squares.

4.2 LSTM

We have performed for all modifications of datasets grid-search with the largest possible batch size. The seed was always set to random, leading to some disturbance in comparing the hyperparameters. An example of this disturbance can be seen in figure 4.17. The learning rate decreased logarithmically from 0.01 to 0.00007. All those and other parameters can be seen in table 4.3. We selected the best grid search results and trained the LSTM networks. We plotted the confusion matrices for these networks. The results can be seen below.

Hyperparameter	Values
Learning rate	0.01, ..., 0.00007
Hidden size	32, 64, 128, 256, 512
Layers	2,4
Batch size	max. possible

Table 4.3: Grid search parameters.

4.2.1 Hand gesture dataset

We performed grid searches for all dataset sets. The best accuracy of all sets from the grid search is in table 4.4. For comparison with the other data, we trained again and plotted the confusion matrix only for the set 80:20. The greatest parameters were: batch size = 300, hidden size = 64, num. of layers = 2, learning rate = 0.005. The confusion matrix is in figure 4.11. With those parameters, we got the accuracy equal to 81%.

Grid search

To compare with the other dataset's grid searches, we plot the grid search for set 80:20. Generating took several hours. It is shown in figure 4.10. We also add the table 4.4, which shows the best accuracy for all sets from its grid search.

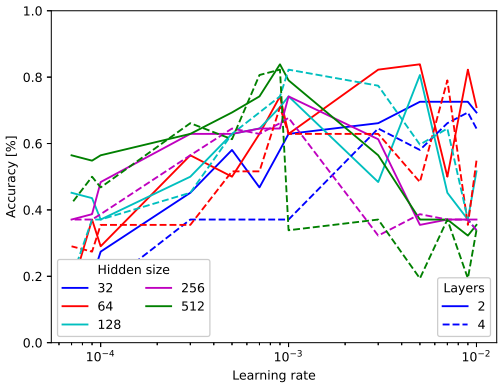


Figure 4.10: Graph of the LSTM grid search of the Hand gesture dataset, ratio 80:20 is shown. A higher score means better accuracy of the testing data in the current configuration. Colour represents the hidden size. Meanwhile, the line pattern stands for the num. of layers.

Dataset	Set	Accuracy
Hand	50:50	82%
Hand	60:40	85%
Hand	70:30	85%
Hand	80:20	83%
Hand	90:10	96%

Table 4.4: The best accuracy from the LSTM grid search per each set in Hand Gesture dataset.

Confusion matrices

We took the configuration from the previous grig search (see fig. 4.10), the trained new model of the network and created the confusion matrix, which is shown in figure 4.11.

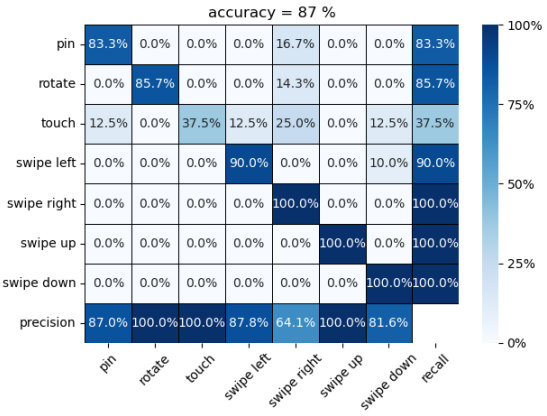


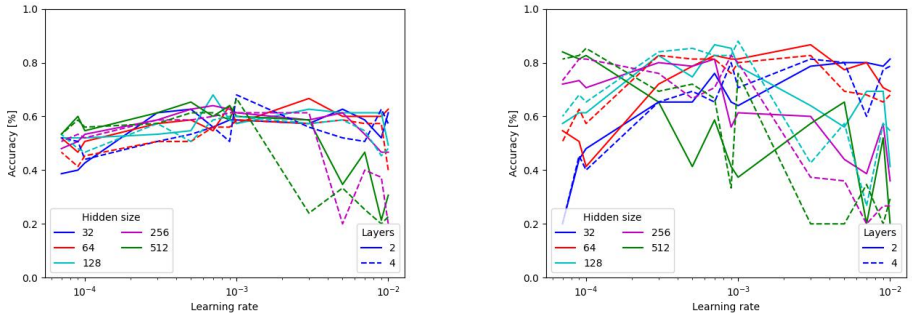
Figure 4.11: Confusion matrix of the LSTM classification of the hand gesture dataset, ratio 80:20. Hyperparameters: batch size = 300, hidden size = 64, num. of layers = 2, learning rate = 0.005. There is 20% of the testing samples per class from the whole dataset.

4.2.2 Bottle moving dataset

We did grid searches for all 6 sets of datasets. Then we trained the networks for the best results in each set. The best accuracy in the confusion matrix went out with the set max in figure 4.13b, which leads to 83%. Meanwhile, the set fft (fig. 4.13a) was only 63%. The concrete parameters are detailed and written in the description of the figures.

Grid search

We performed the grid searches for all two sets of the dataset. Those two were the most time consuming (several hours per one set) compared to the other datasets. The results are shown in figure 4.12.

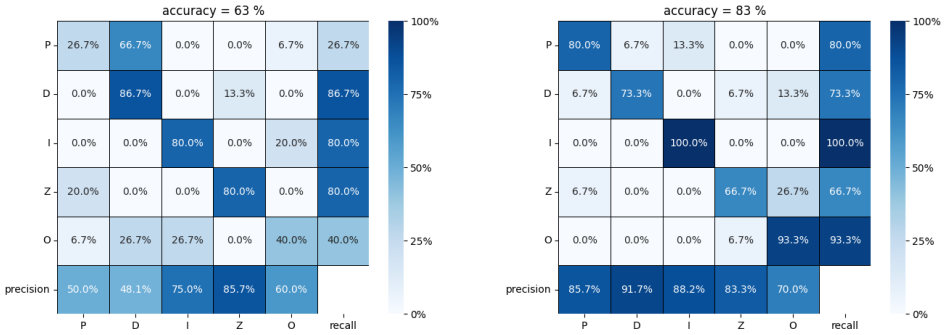


(a) : Fft set **(b) :** Max set

Figure 4.12: Graph of the LSTM grid searches of the Bottle moving dataset sets. Fft set is on the left and Max set is on the right. A higher score means better accuracy of the testing data in the current configuration. Colour represents the hidden size. Meanwhile, the line pattern stands for the num. of layers.

Confusion matrices

We took the best configuration from the grid searches (see fig. 4.12) and plotted the confusion matrix for the Bottle moving audio dataset sets, which is shown in figure 4.13.



(a) : Fft set **(b) :** Max set

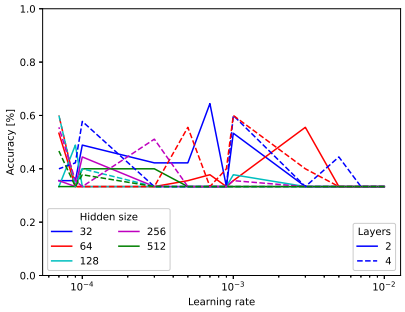
Figure 4.13: Confusion matrices of the GRU classification of the Bottle moving dataset. Set Fft is shown on the left, Max on the right. Hyperparameters for network trained on Fft set: batch size = 100, hidden size = 32, num. of layers = 4, learning rate = 0.001. Hyperparameters for network trained on Max set: batch size = 100, hidden size = 128, num. of layers = 4, learning rate = 0.001. There are 15 testing samples per class.

4.2.3 CoppeliaSim dataset

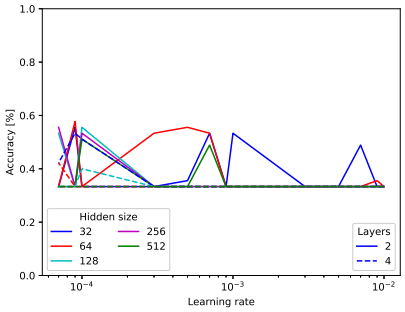
For all dataset sets, we did a grid search. This case shows us how hard it can be for the network to learn something from the data. For most of them, the network cannot learn. The best result came up with the forces 4.15a with parameters: batch size = 300, hidden size = 32, num. of layers = 2, learning rate = 0.0007. We achieved accuracy 62%. We also made a confusion matrix for ObjOriPos 4.15b where we can see that accuracy 51% is lower, but precision is higher in the case of labels "push" and "pump".

Grid search

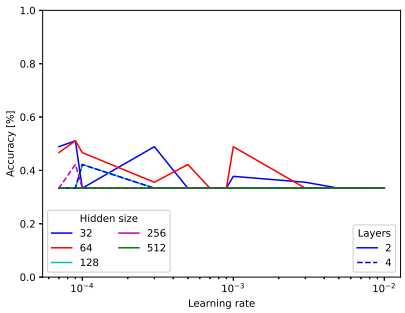
We performed grid searches for all six sets. The computational time took several days. The results can be seen in figure 4.14.



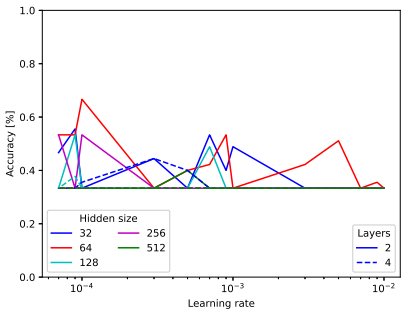
(a) : Forces set



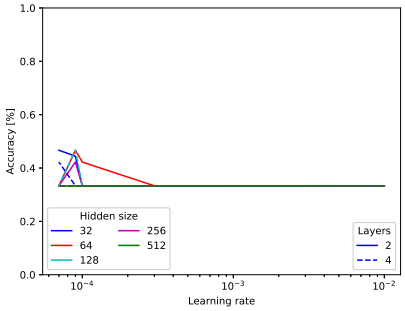
(b) : ObjOri set



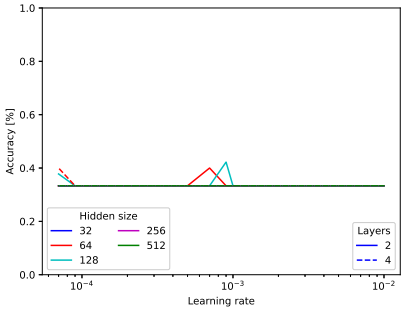
(c) : ObjPos set



(d) : ObjOriPos set



(e) : TipPos set

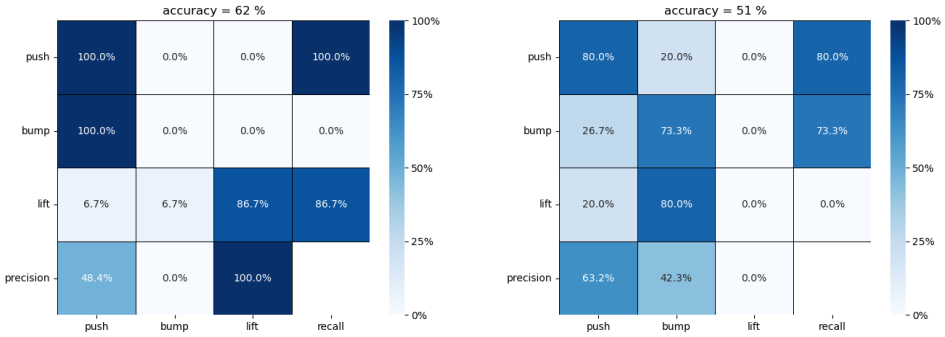


(f) : TipMinObjPos set

Figure 4.14: The LSTM grid search graphs of the CoppeliaSim dataset sets. A higher score means better accuracy of the testing data in the current configuration. Colour represents the hidden size. Meanwhile, the line pattern stands for the num. of layers.

Confusion matrices

For the CopelliaSim dataset, we decided to show two confusion matrices (see fig. 4.14) for sets Forces and ObjOriPos, which can be compared with the others. Decisions were based on the accuracy and performance of the sets grid searches 4.15.



(a) : Forces set **(b) :** ObjOriPos set

Figure 4.15: Confusion matrices of the LSTM classification of the Bottle moving dataset. Set Forces is shown on the left, ObjOriPos on the right. Hyperparameters for network trained on Forces set: batch batch size = 300, hidden size = 32, num. of layers = 2, learning rate = 0.0007. Hyperparameters for network trained on ObjOriPos set: batch size = 300, hidden size = 64, num. of layers = 2, learning rate = 0.0001. There are 15 testing samples per class.

4.2.4 Human skeleton dataset

In this dataset, we performed the grid search 4.16, where the network increased the highest accuracy (96.6%) in one configuration of hyperparameters. We did another grid search with more seeds and calculated the average and the deviation. The results can be seen in the figure 4.17. As we can see, the average accuracy is too much below the highest one. Therefore, we could not create the same performance again, so we trained a new network with an accuracy of 77%. Confusion matrix is in figure 4.18. The parameters are: batch size = 300, hidden size = 512, num. of layers = 2, learning rate = 0.009.

Grid search

We plot two grid searches for the Human Skeleton dataset. The first one (see fig. 4.16) is the same as for the other datasets. The other one (see fig. 4.17)represents the more advanced grid search, which consists of more iterations in the selected configuration. Therefore average accuracy can be computed, and better comparison can be observed.

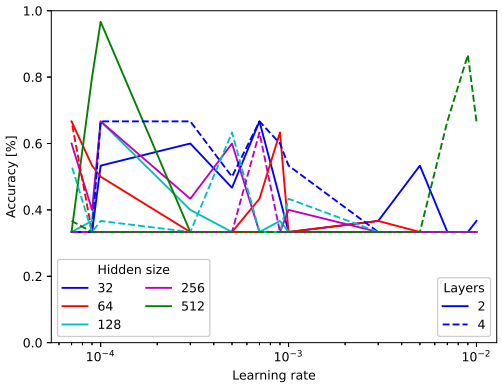


Figure 4.16: Graph of the LSTM grid search of the Human skeleton dataset. A higher score means better accuracy of the testing data in the current configuration. Colour represents the hidden size. Meanwhile, the line pattern stands for the num. of layers.

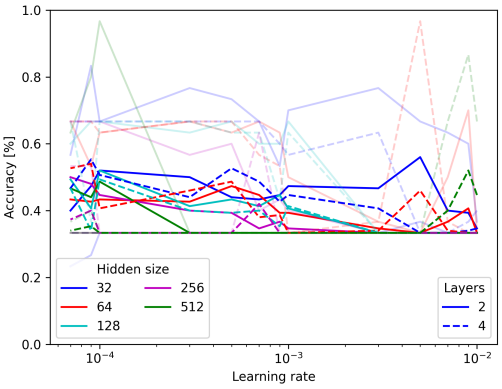


Figure 4.17: Graph of the average LSTM grid search of the Human skeleton dataset. A higher score means better accuracy of the testing data in the current configuration. Colour represents the hidden size. Meanwhile, the line pattern stands for the or num. of layers. Transparent lines represent the maximum/minimum accuracy for five different seeds. Meanwhile, the full line represents the average.

Confusion matrices

We took the best configuration from the grid search (see fig. 4.16) and plotted the confusion matrix for the Human skeleton dataset, which is shown in figure 4.18.

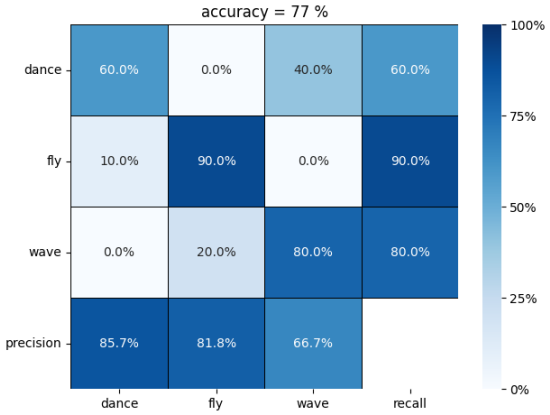


Figure 4.18: Confusion matrix of the LSTM classification of the Human skeleton dataset. Hyperparameters: batch size = 300, hidden size = 512, num. of layers = 2, learning rate = 0.009. There are ten testing samples per class.

4.2.5 Discussion

Hand gesture dataset

We produced for all sets of datasets grid search. The parameters can be seen in table 4.3. For the Hand gesture dataset, we displayed only one grid search for set 80:20 in figure 4.19. We observe the network's accuracy linear grow from the lowest learning rate for almost all configurations to the point, where the learning rate is equal to 10^{-3}. After that point, we could say that the best behaviour of the configurations was for parameters, where hidden size = 32, layers = 2 (the full red line). The rest of the configurations behave chaotically. We took the best result from the grid search and trained the network again. We run it on the test data from set 80:20 and plot the confusion matrix, which can be observed in figure 4.11.

The accuracy of the newly trained network came to 87%, which is higher than the grid search best accuracy (81%) for this set. This disagreement is caused by random selection of the seed when initialising the network. The gesture *swipe rigth* has the smallest precision and the other gestures like *touch, rotate* and *pin* are mistaken for this gesture. However the accuracy of *swipe rigth* is 100% which signifies that the network has more problems with the *touch, rotate* and *pin* gestures.

The most problematic gesture is *touch*, which has an accuracy of only 37.5%. The same gesture has the lowest accuracy in the KNN (DTW) confusion matrix (see fig. 4.2). However, in the GRU confusion matrix, it came up with 75% accuracy. In table 4.4 can be seen, the best accuracy for all sets of the Hand gesture dataset from the grid search. The percentages are similar except for dataset 90:10, which leads to 96%. This peak can be the possible improvement of more training samples or coincidence by selecting the right seed for training in a grid search.

Bottle moving dataset

We have done a grid search for set Max and Fft. They can be seen in figures 4.12a and 4.12b. Configurations in set Fft behave similarly, and they were close except for the configuration, where the hidden size was 512 and the one where hidden size was equal to 256, with 4 layers. The grid search for set Max was chaotic, and the configurations were spread. We plotted the confusion matrices 4.13a and 4.13b for the new trained networks.

The Fft set came to 63% accuracy. The lowest accuracy (26.7%) is for the letter *P*, which was mistaken with the letter *D* by 66.7%. Letters *D*,*I*,*Z* has similar accuracy around 80%. Letter *O* was also problematic since it was misclassified as letters *P*,*D*,*I*. Its accuracy goes to 40%. Much better progress came up with the Max set, leading to 83% accuracy. The accuracy of all letters is above or equal to 80%. The outstanding result is for the letter *I*, which has an accuracy of 100%. This outstanding can also be seen in the original dataset visualisation in figure 3.3.

CoppeliaSim dataset

We created the grid searches (see fig. from 4.14a to 4.14f) for the CoppeliaSim dataset sets. For sets TipPos and TipMinObjPos, the network does not learn at all. For the rest of the sets: Forces, ObjOri, ObjPos and ObjOriPos network was not able to across 66% accuracy.

We decided to show the confusion matrix of Forces in figure 4.15a and ObjOriPos in figure 4.15b. We observe the accuracy of 61% for Forces and 51% for ObjOriPos. The network trained on the set ObjOriPos cannot recognize the action *lift*, however, it can see the difference between the labels *bump* and *push*. This feature is missing in the trained network on the Forces set, as we can see in figure 4.15a.

However, network trained on Forces set can classify clearly with an accuracy of 86.7% the *lift* action. Those two networks can be modified together to complement each other in the future.

Human Skeleton dataset

In the grid search of the Human skeleton dataset, we can see how random seeds can change the results for each configuration. We first created the grid search with the random seeds (see fig. 4.16) where one configuration went with an accuracy above 95%. To show that, in some cases depends only on the network seed initialisation, we plot the averages of the five different seeds for the full grid search. The result is in figure 4.17.

We observe that average accuracy is much lower than its maximum. We trained the network with the best configuration from the first grid search and reached 77% accuracy. The confusion matrix for the test data of the trained network can be seen in the figure 4.18. The problematic label is *dance* which has an accuracy of 60% and was misclassified with *wave* move. The rest of the moves have accuracy above or equal to 80%.

4.3 GRU

We created for all modifications of datasets grid-search with the largest possible batch size. The seed was always random, leading to some disturbance in comparing the hyperparameters. The learning rate decreased logarithmically from 0.01 to 0.00007. All those and other parameters can be seen in table 4.5. We selected the best grid search results and trained the GRU networks. We plotted the confusion matrices for these networks. The results can be seen below.

Hyperparameter	values
Learning rate	0.01, ..., 0.00007
Hidden size	32, 64, 128, 256, 512
Layers	2,4
Batch size	max, based on data

Table 4.5: Grid search parameters

4.3.1 Hand gesture dataset

The best accuracy of all sets from the grid search is in table 4.6. We performed the confusion matrix via GRU for set 80:20. The greatest parameters were: batch size = 300, hidden size = 64, num. of layers = 2, learning rate = 0.009. The confusion matrix is in figure 4.20. Accuracy came to 90%.

Grid search

To compare with the other dataset's grid searches, we created the grid search for set 80:00. Generating took several hours. It is shown in figure 4.20. We also add the table 4.6, which shows the best accuracy for all sets from its grid search.

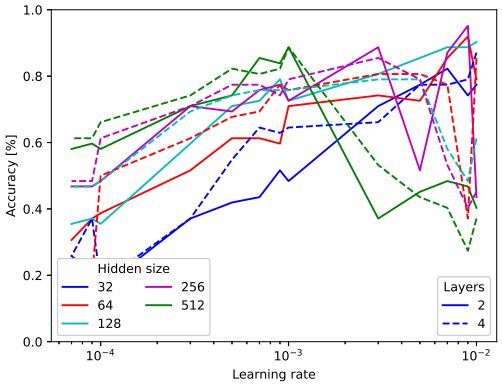


Figure 4.19: Graph of the GRU grid search of the Hand gesture dataset, ratio 80:20 is shown. A higher score means better accuracy of the testing data in the current configuration. Colour represents the hidden size. Meanwhile, the line pattern stands for the num. of layers.

Dataset	Set	Accuracy
Hand	50:50	90%
Hand	60:40	90%
Hand	70:30	92%
Hand	80:20	95%
Hand	90:10	96%

Table 4.6: The best accuracy from the GRU grid search per each set in Hand Gesture dataset.

Confusion matrix

We took the configuration from the previous grid search (see fig. 4.19), the trained new model of the network and created the confusion matrix, which is shown in figure 4.20.

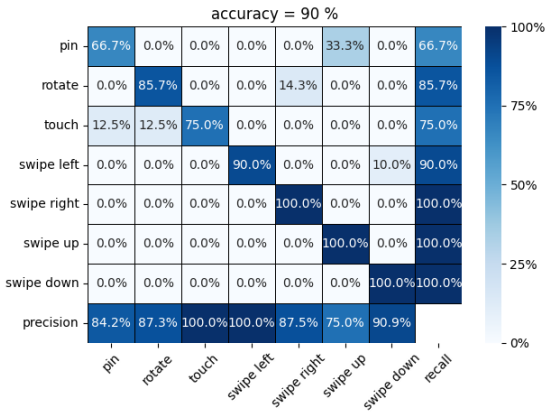


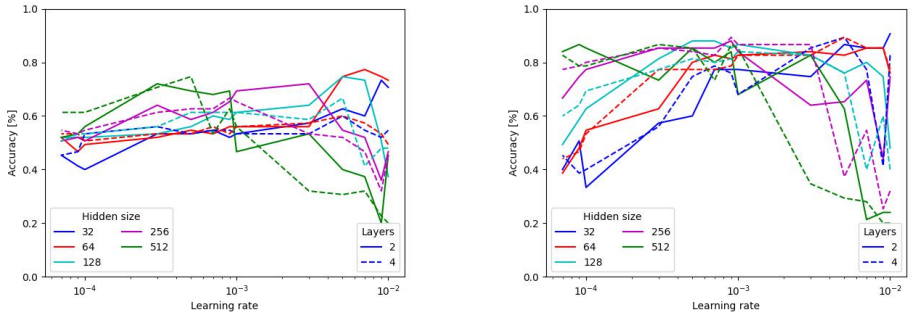
Figure 4.20: Confusion matrix of the GRU classification of the hand gesture dataset, ratio 80:20. Hyperparameters: batch size = 300, hidden size = 64, num. of layers = 2, learning rate = 0.009. There is 20% of the testing samples per class from the whole dataset.

4.3.2 Bottle moving dataset

We have done grid searches for fft and max set. Then we trained the networks for the best results in each set. The best accuracy in the confusion matrix went out with the set max in figure 4.13b, which leads to 84%. Meanwhile, the set fft (fig. 4.13a) was only 76%. The concrete parameters are detailed and written in the description of the figures.

Grid search

We did the grid searches for all two sets of the dataset. The results are shown in figure 4.21.

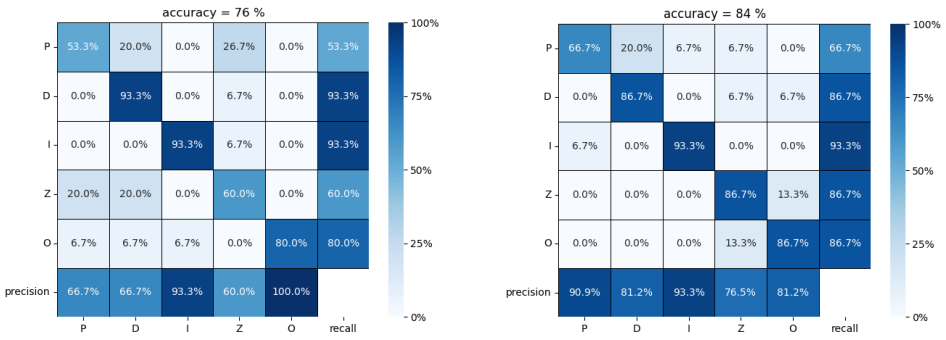


(a) : Fft set **(b) :** Max set

Figure 4.21: Graph of the GRU grid searches of the Bottle moving dataset sets. Fft set is on the right and Max set is on the left. A higher score means better accuracy of the testing data in the current configuration. Colour represents the hidden size. Meanwhile, the line pattern stands for the num. of layers.

4.3.3 Confusion matrices

We took the best configuration from the grid searches (see fig. 4.21) and plotted the confusion matrix for the Bottle moving audio dataset sets, which is shown in figure 4.22.



(a) : Fft set **(b) :** Max set

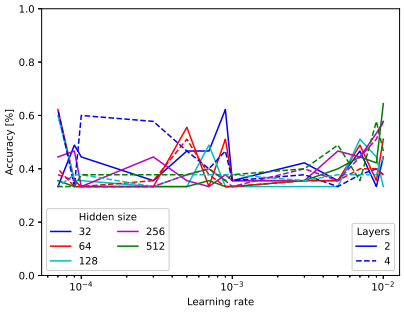
Figure 4.22: Confusion matrices of the GRU classification of the Bottle moving dataset. Set Fft is shown on the left, Max on the right. Hyperparameters for network trained on Fft set: batch size = 100, hidden size = 64, num. of layers = 2, learning rate = 0.007. Hyperparameters for network trained on Max set: batch size = 100, hidden size = 256, num. of layers = 4, learning rate = 0.0009. There are 15 testing samples per class.

4.3.4 CoppeliaSim dataset

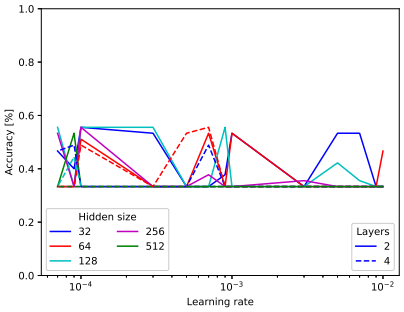
For all dataset sets, we computed a grid search. The results are similar to the LSTM grid search since this network has the same core as GRU. For most of the cases, the network cannot learn. The best result came up with the forces 4.24a with parameters: batch size = 300, hidden size = 512, num. of layers = 2, learning rate = 0.01. We also made a confusion matrix for ObjOriPos to show that the final accuracy can be increased by selecting the right set.

Grid search

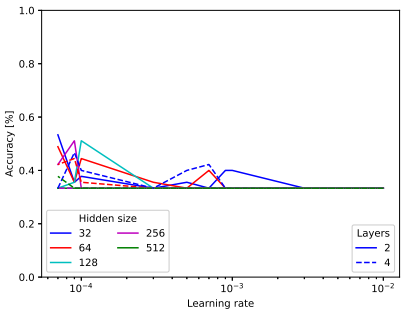
We performed grid searches for all six sets. The computational time took several days. The results can be seen in figure 4.23.



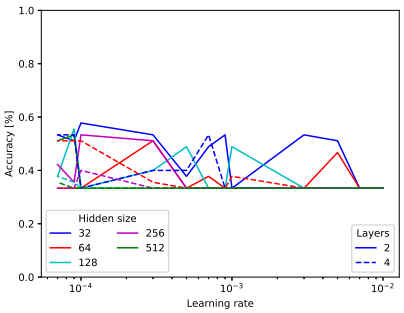
(a) : Forces set



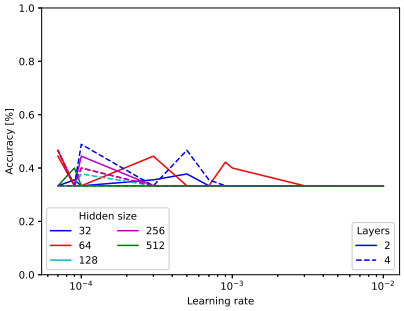
(b) : ObjOri set



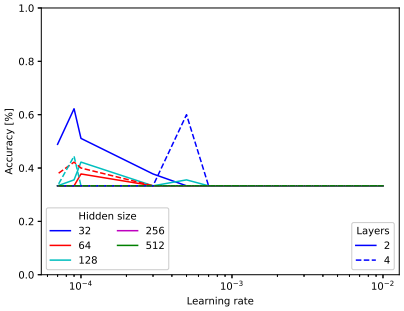
(c) : ObjPos set



(d) : ObjOriPos set



(e) : TipPos set

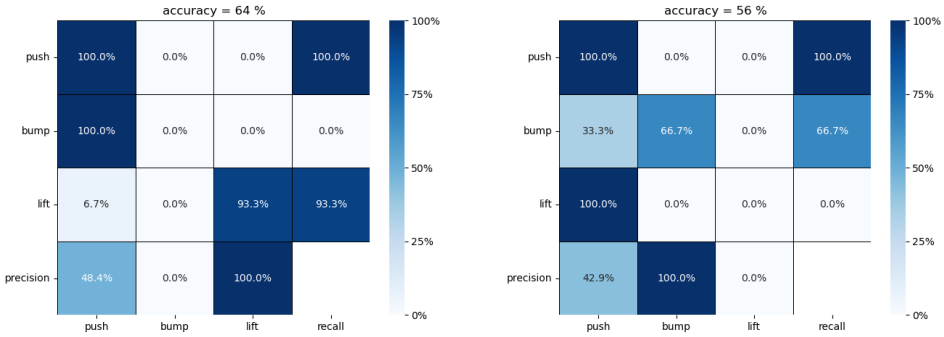


(f) : TipMinObjPos set

Figure 4.23: The GRU grid search graphs of the CoppeliaSim dataset sets. A higher score means better accuracy of the testing data in the current configuration. Colour represents the hidden size. Meanwhile, the line pattern stands for the num. of layers.

Confusion matrices

For the CopelliaSim dataset, we decided to show two confusion matrices (see fig. 4.24) for sets Forces and ObjOriPos, which can be compared with the others. Decisions were based on the accuracy and performance of the sets grid searches 4.23.



(a) : Forces set **(b) :** ObjOriPos set

Figure 4.24: Confusion matrices of the GRU classification of the Bottle moving dataset. Set Forces is shown on the left, ObjOriPos on the right. Hyperparameters for network trained on Forces set: batch size = 300, hidden size = 512, num. of layers = 2, learning rate = 0.01. Hyperparameters for network trained on ObjOriPos set: batch size = 300, hidden size = 32, num. of layers = 2, learning rate = 0.0001. There are 15 testing samples per class.

4.3.5 Human skeleton dataset

In this dataset, we performed the grid search 4.25. We trained the network with an accuracy of 83%. The confusion matrix is in figure 4.26. The parameters are: batch size = 300, hidden size = 64, num. of layers = 2, learning rate = 0.0001.

Grid search

For the GRU, we created only one grid search, which can be seen in figure 4.25.

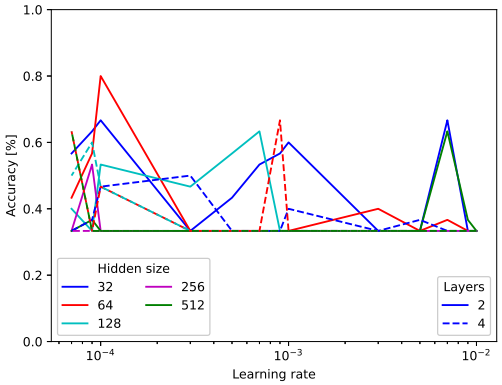


Figure 4.25: Graph of the GRU grid search of the Human skeleton dataset. A higher score means better accuracy of the testing data in the current configuration. Colour represents the hidden size. Meanwhile, the line pattern stands for the num. of layers.

Confusion matrices

We created the confusion matrix (displayed here 4.26)of the network trained on the Human Skeleton dataset with the hyperparameters based on the result from the grid search (see fig. 4.25).

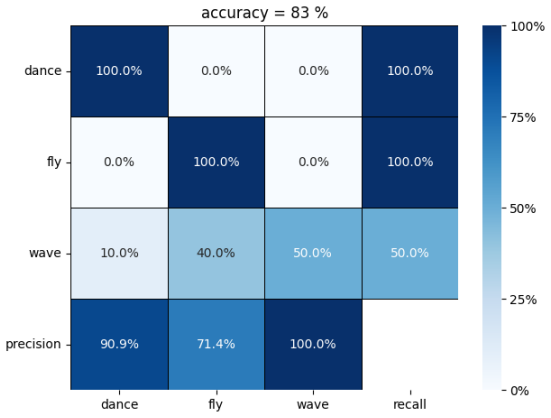


Figure 4.26: Confusion matrix of the GRU classification of the Human skeleton dataset. Hyperparameters: batch size = 300, hidden size = 64, num. of layers = 2, learning rate = 0.0001. There are 10 testing samples per class.

4.3.6 Discussion

Hand gesture dataset

We show only one grid search for set 80:20 in figure 4.19. The grid search is similar to the LSTM one (see fig. 4.10 to compare). The network's accuracy is growing from the lowest learning rate for almost all configurations to the point where the learning rate is equal to 10^{-3}. At this point, the configurations start to behave unpredictably except for one configuration, which continues to grow linearly. It is the one where hidden size = 128 with two layers (full turquoise line).

We trained the GRU network with the grid search configuration with the highest accuracy. We turn it on the test data from set 80:20 and plot the confusion matrix, as shown in figure 4.20. The accuracy of the newly trained network came to 90%, which is lower than the grid search best accuracy (95%) for this set. It is caused by random selection of the seed when initialising the network. The gestures precisions are consistently above or equal to 75%, which is better than in the LSTM confusion matrix (see fig. 4.11) for this set.

The weakest gesture is *pin*, which has an accuracy of 66.7% and is misclassified with the gesture *swipe up*. We observe the best accuracy for all sets of the Hand gesture dataset from the GRU grid search in table 4.6. The percentages are increasing as the number of training samples is increasing. This signifies that the networks learn better with more training samples and fewer testing samples.

Bottle moving dataset

We performed a grid search for set Max and Fft. They can be observed in figures 4.21a and 4.21b. Configurations in set Fft behave similarly, and they were close except for the configuration, where the hidden size = 512, layers = 4. The same behaviour came with the Max grid search, which is different compared to the LSTM Max grid search in figure 4.12b, where it was more chaotic.

The point where all configurations were the highest together is around the learning rate equals 10^{-3}. We took the best configuration based on accuracy and trained the networks. The confusion matrices are shown in figures 4.22a and 4.22b. The accuracy is better for set Max, leading to 84%. Meanwhile, it is 76% for the set Fft. The Max set confusion matrix has precisions and recalls higher than the Fft set confusion matrix. Except for the letter *D* which has higher accuracy in the Fft set confusion matrix.

CoppeliaSim dataset

We did the grid searches, displayed in figures from 4.23a to 4.23f, of the CoppeliaSim dataset sets. The learning was better compared to the learning of the LSTM networks. It depended on the init seed of the network. We performed to show the confusion matrix of Forces in figure 4.24a and ObjOriPos in figure 4.24b. So we can compare them to the LSTM algorithm. We observe the accuracy of 64% for Forces and 56% for ObjOriPos. Those are higher than in the LSTM network (see figures 4.15a and 4.15b).

The problems with recognising the labels are the same as in the LSTM. The network trained on the set ObjOriPos cannot classify the action *lift*, but it can classify the actions *bump* and *push* separately. *Bump* has a precision of 100%, which is higher than in LSTM based network. It is caused by classifying *lift* as *push* action. Meanwhile, in the LSTM confusion matrix for this set, *lift* is divide between *bump* and *push*. The network trained on the set Forces can classify clearly with an accuracy of 93.3% the *lift* action. Those two networks can be modified together to complement each other in the future.

Human Skeleton dataset

In the grid search (see fig. 4.25) of the Human skeleton dataset, we see the peaks of the accuracy. This network learning is mainly based on good seed initialisation. The problem, in general, is shown in figure 4.17, where some of the peaks seem to have potential, but the average accuracy, calculated from more seeds, is below them. We selected the highest peak from the grid search and trained the network based on the configurations. The result is in figure 4.26. We reached 83% accuracy.

The problematic label is *wave* which has an accuracy of 50% and was misclassified with *fly* movement. The rest of the moves have accuracy equal to 100%. In the trained LSTM network *wave* has an accuracy of 80%, which is higher than in the GRU result. In future work, we can theoretically those two networks modified to be together to complement each other in the future.

Chapter 5

Conclusion and future work

In this work, we started by getting closer to the problematics of the time series classification. Then we talked about the methods used for the classification, chose 3 of them named LSTM, GRU and KNN (DTW) and described them. We described four datasets: Hand gesture dataset, Bottle moving audio dataset, CoppeliaSim dataset, and Human skeleton dataset. We created subdatasets for them while preparing the data for our classification algorithms.

We visualised all sets with the DTW distance matrix and discussed them. We performed the KNN (DTW) classification for the Hand gesture, CoppeliaSim and Human skeleton dataset. The results were shown as confusion matrices. We did the GRU and LSTM networks grid search for selected sets and trained the networks based on the best configuration from the search. The testing samples were computed via the networks, and the results were displayed as the distance matrix. We also discussed them.

The best option for the Hand gesture seems to be the KNN (DTW) algorithm with a final accuracy of 95%. The CoppeliaSim dataset is also the KNN (DTW), where accuracy was up to 100% for the set TipMinObjPos. The best accuracy came with GRU with the number 83% for set Max in the Bottle moving audio dataset. The algorithm KNN (DTW) performed well again for the Human skeleton dataset, where it led to 100%. Comparing the classification algorithms, KNN (DTW) stood out because it was the classifier with the highest accuracy for three out of four datasets.

For future work, we suggest to perform the same experiment but with different classifiers. The KNN (DTW) could be improved to see if the high accuracy of the dataset is caused mainly by the metric DTW or by classifier behaviour on the dataset. More preprocessing procedures like MFCC for audio dataset can be done to increase accuracy. The improvement of the neural networks can also be made by mixing the GRU and LSTM since we could see that the networks work better for different labels.

As we can see, the problem with the time series classifications is various. There is still a need to do much work if we want to see a world where computers

fully understand our gestures, actions and activities.

Bibliography

[1] Xiaoyue Wang et al. "Experimental Comparison of Representation Methods and Distance Measures for Time Series Data". In: *Data Mining and Knowledge Discovery* 26 (Dec. 2010). DOI: 10.1007/s10618-012-0250-5.

[2] Weiwei Jiang. "Time series classification: nearest neighbor versus deep learning models". In: *SN Applied Sciences* 2 (Apr. 2020), p. 721. DOI: 10.1007/s42452-020-2506-9.

[3] Jenny Cifuentes et al. "Gesture Classification Using LSTM Recurrent Neural Networks". In: *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. 2019, pp. 6864–6867. DOI: 10.1109/EMBC.2019.8857592.

[4] Heyuan Guo, Yang Yang, and Hua Cai. "Exploiting LSTM-RNNs and 3D Skeleton Features for Hand Gesture Recognition". In: *2019 WRC Symposium on Advanced Robotics and Automation (WRC SARA)*. 2019, pp. 322–327. DOI: 10.1109/WRC-SARA.2019.8931937.

[5] Shenglin Zhao et al. "Hand Gesture Recognition on a Resource-Limited Interactive Wristband". In: *Sensors (Basel, Switzerland)* 21 (2021).

[6] Phat Nguyen Huu, Ngoc Nguyen Thi, and Thien Pham Ngoc. "Proposing Posture Recognition System Combining MobilenetV2 and LSTM for Medical Surveillance". In: *IEEE Access* 10 (2022), pp. 1839–1849. DOI: 10.1109/ACCESS.2021.3138778.

[7] Yinhui Yi et al. "Music Genre Classification with LSTM based on Time and Frequency Domain Features". In: *2021 IEEE 6th International Conference on Computer and Communication Systems (ICCCS)*. 2021, pp. 678–682. DOI: 10.1109/ICCCS52626.2021.9449177.

[8] Corinna Cortes and Vladimir Vapnik. "Support-vector networks". In: *Machine learning* 20.3 (1995), pp. 273–297.

[9] D. McGibney et al. “Cooperative distributed object classification for multiple robots with audio features”. In: *2011 International Symposium on Micro-NanoMechatronics and Human Science*. 2011, pp. 134–139. DOI: 10.1109/MHS.2011.6102174.

[10] Petr Vanc. *Probabilistic Gesture Control for a Robotic Arm*. May 2021. URL: https://dspace.cvut.cz/handle/10467/95271.

[11] Alexandros Paraschos et al. “Probabilistic Movement Primitives”. In: Jan. 2013.

[12] Sumit Kumar et al. “Energy Load Forecasting using Deep Learning Approach-LSTM and GRU in Spark Cluster”. In: *2018 Fifth International Conference on Emerging Applications of Information Technology (EAIT)*. 2018, pp. 1–4. DOI: 10.1109/EAIT.2018.8470406.

[13] Kyunghyun Cho et al. *On the Properties of Neural Machine Translation: Encoder-Decoder Approaches*. 2014. arXiv: 1409.1259 [cs.CL].

[14] Shudong Yang, Xueying Yu, and Ying Zhou. “LSTM and GRU Neural Network Performance Comparison Study: Taking Yelp Review Dataset as an Example”. In: *2020 International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)*. 2020, pp. 98–101. DOI: 10.1109/IWECAI50956.2020.00027.

[15] Jose de Jesus Guerrero-Turrubiates et al. “Pitch estimation for musical note recognition using Artificial Neural Networks”. In: *2014 International Conference on Electronics, Communications and Computers (CONIELECOMP)*. 2014, pp. 53–58. DOI: 10.1109/CONIELECOMP.2014.6808567.

[16] Gábina Šejnová. *Dataset - Humanoid robot Pepper camera’s recording procced by OpenPose algorithm*. 2022. URL: https://drive.google.com/drive/folders/1a5RYDETUa4l31J8BMaSZZzTNBPJmr-gp.

[17] Ginés Hidalgo et al. *OpenPose Algorithm*. 2019. URL: https://github.com/CMU-Perceptual-Computing-Lab/openpose.

[18] Petr Vanc. *Dataset - Hand gestures*. 2021. URL: https://drive.google.com/drive/folders/1lasIj7vPenx_ZkMyofvmro6-xtzYdyVm (visited on 06/02/2022).

[19] *Mirracle_sim*. 2022. URL: https://gitlab.ciirc.cvut.cz/imitrob/mirracle/mirracle_sim.

[20] Kateřina Kubecová. *Action-representationVAE*. 2022. URL: https://gitlab.ciirc.cvut.cz/imitrob/mirracle/mirraclestudents/kubecova-action-representationvae.git.

BACHELOR‘S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Mikeska Michal** Personal ID number: **492182**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Cybernetics and Robotics**

II. Bachelor’s thesis details

Bachelor’s thesis title in English:

Time-Series Classification for Action Detection in Imitation Learning

Bachelor’s thesis title in Czech:

Klasifikace časových řad pro rozpoznání akcí v imitačním učení

Guidelines:

When teaching robots using imitation learning, we encounter the problem of classifying time series. The individual demonstrations do not have to be the same length and might depend on the specific object the actions are performed with (e.g., pushing or banging on objects). In this bachelor thesis we will focus on how time series (actions) can be compared.
Individual steps:
1. Preparation of a dataset of actions performed by a demonstrator. For selected actions (e.g., move, release, lift, bang) and various types of objects, record sequences of audio data and demonstrator hand movement (using Leap motion, HTC Vive tracker or HTC Vive glove).
2. Preparation of a dataset of actions performed by a robot. Prepare a dataset in the CoppeliaSim simulator, where the robotic manipulator performs selected actions on various objects. In addition to the position of the robot's end element (end-effector), the dataset should also contain other data (e.g., values on the force-torque sensor or the position of the object).
3. Implementation of methods for classification of time series based on distances (e.g., Dynamic time warping) and classifiers based on neural networks (e.g., LSTM, spatio-temporal transformer network).
4. Compare clustering results using DTW method for individual datasets and actions.
5. Learning a classifier based on neural networks for acquired time series (e.g., LSTM or spatio-temporal transformer network). Comparison of classification quality against classification using the kNN classifier with DTW as the distance metric.
6. Comparison of the quality of classification of individual actions using different modalities (e.g., sound vs. end-effector movement) and individual types of classifiers.

Bibliography / sources:

[1] Jiang, Weiwei. "Time series classification: Nearest neighbor versus deep learning models." SN Applied Sciences 2.4 (2020): 1-17.
[2] Wang X, Mueen A, Ding H, Trajcevski G, Scheuermann P, Keogh E (2013) Experimental comparison of representation methods and distance measures for time series data. Data Min Knowl Discov 26(2):275–309
[3]Fawaz, Hassan Ismail, et al. "Deep learning for time series classification: a review." Data mining and knowledge discovery 33.4 (2019): 917-963.
[4] Grabocka, Josif, and Lars Schmidt-Thieme. "Neuralwarp: Time-series similarity with warping networks." arXiv preprint arXiv:1812.08306 (2018).
[5] Sun, Yan, Yixin Shen, and Liyan Ma. "MSST-RT: Multi-Stream Spatial-Temporal Relative Transformer for Skeleton-Based Action Recognition." Sensors 21.16 (2021): 5339.
[6] Lohit, Suhas, Qiao Wang, and Pavan Turaga. "Temporal transformer networks: Joint learning of invariant and discriminative time warping." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019.

Name and workplace of bachelor's thesis supervisor:

Mgr. Karla Štěpánová, Ph.D. Robotic Perception CIIRC

Name and workplace of second bachelor's thesis supervisor or consultant:

Date of bachelor's thesis assignment: **12.01.2022** Deadline for bachelor thesis submission: **20.05.2022**

Assignment valid until: **30.09.2023**

Mgr. Karla Štěpánová, Ph.D.	prof. Ing. Tomáš Svoboda, Ph.D.	prof. Mgr. Petr Páta, Ph.D.
Supervisor's signature	Head of department's signature	Dean's signature

III. Assignment receipt

The student acknowledges that the bachelor's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the bachelor's thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student's signature