Institute of Cognitive Science
Wachsbleiche 27
49090 Osnabrück

Human movement classification using a 1D-Convolutional Network

Tackling the Bremen Big Data Challenge 2019

Implementing ANNs with TensorFlow

Winter Semester 2022/23

Project Report

Marc Zeller
mazeller@uos.de

Michael Rau
mrau@uos.de

Contents

Chapter 1

Introduction

Artificial Neural Networks (ANNs) are powerful tools for the classification of data. So powerful in fact, that they have become crucial in several technologies and applications. Classification with ANNs allow us to draw on raw data and assign abstract data points different classes through a learning process. Due to its versatility, this approach can be applied to a variety of different domains including face recognition, object detection or the classification of human activities. This project addresses the 2019 edition of the Bremen Big Data Challenge, a yearly competition held by the Cognitive Systems Lab at the University of Bremen since 2016. Its 2019 rendition poses a task in the realm of human activity recognition. More precisely, the goal lies in the accurate classification of daily and exercise-related movements.

Human Activity Recognition (HAR) is the field of applications that apply algorithms onto the classification of certain human activities. Most prominently, the tracking of steps or exercise by wearable trackers or smartwatches are one such application that found its way into the daily life of many.

For this challenge, a data set consisting of sensor data from a leg brace is provided with accompanying labels. These recordings represent twenty-two different movements of a total of nineteen subjects. Using a training set encompassing the data of fifteen subjects, the task is to use artificial intelligence techniques to correctly classify the recordings of the four remaining subjects.

We decided to tackle the problem using a 1-dimensional Convolutional Neural Network (1D CNN) architecture as it , while being relatively simple, allows us to capture local features in the provided sensor streams fairly well. It also promises to apply effective smoothing of the data while at the same time providing low interference from noise.

Chapter 2

Related work

Most of our orientation stems from the article "1D Convolutional Neural Network Models for Human Activity Recognition" by Jason Brownlee, who demonstrates the development of a 1D CNN for Human Activity Recognition employing the publicly accessible dataset which tracks movement data with smartphones (Anguita et al., 2013). This data bears a strong similarity to the data which has been provided by the BBDC'19.

Convolutional Neural Networks, opposed to more classical approaches such as Decision Trees and PCA, allow us to forgo the feature engineering step that is needed for more classic approaches to HAR (for example demonstrated by Lara and Labrador in their 2013 paper *A Survey on Human Activity Recognition* (Lara & Labrador, 2013)) and instead helps to achieve accurate results through feature learning (Brownlee, 2018).

Another successful application of a 1D convolutional network for HAR has been produced in the proceedings paper *Human activity recognition from accelerometer data using Convolutional Neural Network* by Lee et al., which achieved a 92.71 percent accuracy rate with their model classifying movement data (Lee et al., 2017).

The effectiveness of CNN architectures for the use of multi input HAR (through wearable sensor data) has been proven repeatedly and even recently (Dua et al., 2021). Dua et al. implemented Gated Recurrent Unit (GRU) Furthermore, Ismail Fawaz et al. (2019) found in their review *Deep learning architectures for Time Series Classification* that CNNs are the most popular architecture for physiological signals classification, which they chalk down to their robustness and relatively short training times compared to other network architectures.

Chapter 3

Implementation

3.1 The model

After carefully exploring and analyzing the provided data and drawing inspiration from the surprisingly vast amount of related literature, we decided to settle on a 1D convolutional network as it presented itself to be a promising approach to solving the task at hand. These sorts of networks rely mainly on extracting non time-dependant local features in the presented data, which we deemed fitting with respect to our dataset.

3.2 The Dataset

The data encompasses 8139 recordings of ordinary as well as athletic movements. In total, the dataset contains twenty-two types of movement:

- Running (*run*) and walking (*walk*)
- Standing (*stand*) and sitting (*sit*)
- Standing up and sitting down (*sit-to-stand*, *stand-to-sit*)
- Walking up and down the stairs ((*stair-up*), *stair-down*)
- Jumping on one or both legs (*jump-one-leg*, *jump-two-leg*)
- Walking in a curve to the left or the right (*curve-left-step*, *curve-right-step*)
- Turning on the spot to the left or the right, starting with the left or right foot (*curve-left-spin-Lfirst*, *curve-left-spin-Rfirst*, *curve-right-spin-Lfirst*, *curve-right-spin-Rfirst*)
- Shuffling laterally to the left or right ((*lateral-shuffle-left*), (*lateral-shuffle-right*))
- Changing the direction, starting with the left or right foot (*v-cut-left-Lfirst*, *v-cut-left-Rfirst*, *v-cut-right-Lfirst*, *v-cut-right-Rfirst*)

The number of individual recordings per movement range from 347 to 400. They are provided by comma separated values (CSV) files spanning a range from 800 up to around 8000 consecutive data points. Subjects contributed 255 to 455 distinct observations.

The dataset is split into training and test data. The training dataset holds recordings of fifteen subjects and a total of 6401 files. Sixteen of these files were discarded since they represented a label that was not relevant to the task (as part of the challenge). Meanwhile, the test dataset contains 1738 recordings of four subjects.

A singular recording represents values of nineteen different sensors in the knee region, scanned at 1000 Hz. For instance, electromyography was used to document the electrical activity produced by skeletal muscles. The placement of the sensors can be observed in Figure 1.

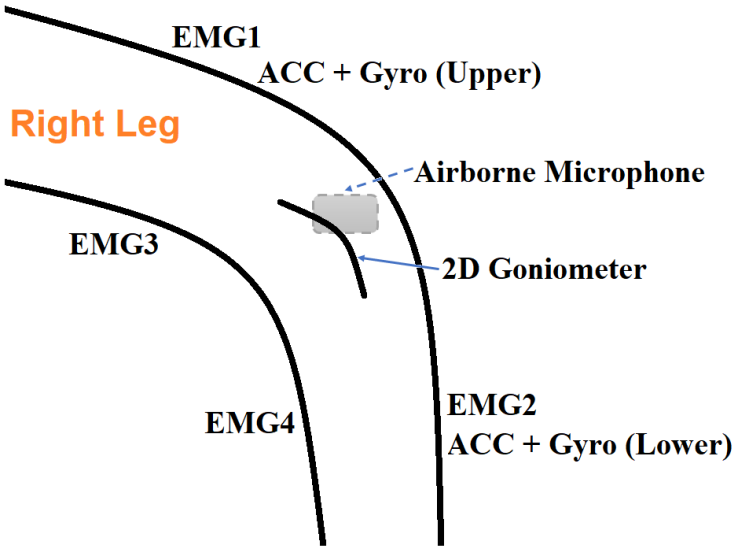

Figure 1: Sensor placement used for each individual movement recording. Taken from the official task description.

3.3 Preprocessing

We start off the preprocessing pipeline by first loading the data into a dataframe locally, categorically encoding the labels using sklearns labelencoder and then extracting and compressing the desired data, i.e. the samples and their corresponding labels. The compressed files are then uploaded to a Google Drive and loaded into the main Colab Notebook. This is done in order to avoid having to upload all provided files into the drive before being able to use them efficiently in conjunction with the main codebase.

3.3.1 Standardization

As mentioned earlier, the BBDC provided us with a test dataset containing around 6400 subject trials, each of which consisted of multivariate timeseries of variable lengths. This lead to us one of the most pressing questions, namely how to approach the standardization of said data.

To gain more insight on how to properly standardize our data we first created a range of plots which showed the total distribution of values per sensor. Most sensor plots appeared to be close to a Gaussian distribution (see Figure 3) whereas others contained values that are almost equally distributed over a relatively broad range (see Figure 2). There were

also vast differences between different labels, but this could not be taken into account as the information wont be accessible when receiving unseen data. A portrayal of these differences can be found in the appendix at Figure 7.

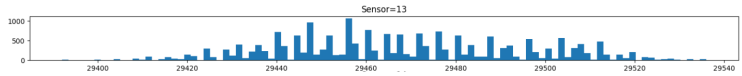

Figure 2: Value distribution of a problematic sensor taken from a "lay"-labeled sample.

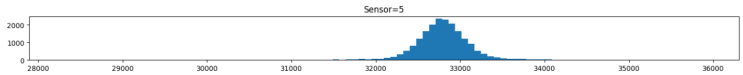

Figure 3: Value distribution of an unproblematic sensor taken from a "lay"-labeled sample.

We decided on applying standardization to each signals sensor individually. To be more precise, we used the median of all standard deviations and the median of all means of each each sensor across the whole dataset and subsequently used these values to apply the standardization. We chose the median instead of the mean with respect to the previously mentioned values, as we did not want outliers to influence our standardization method too heavily. It is worth nothing that we are aware of this not being a perfect solution due to the negative effect standardization has on the more problematic sensor nodes, but based on them being in a minority this solution presented itself as a promising tradeoff. Furthermore, we decided against applying normalization (both sample wise and across the whole dataset), as the several outliers would potentially destroy important underlying relations in the data.

3.3.2 Windowing

Sadly, we were unable to create padded whole length sequences for our training. This was mainly due to very unequal distribution of sequence lengths (see Figure 4 in conjunction with the lack of sufficient RAM on both Google Colab and our personal computers. Furthermore, we were hesitant about this approach as it would result in a very low number of training samples and thus would most likely lead the model to heavily overfit to our training data.

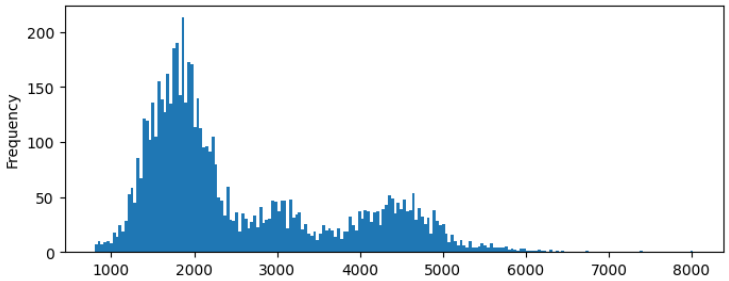

Figure 4: Distribution of sequence lengths in the dataset.

As a solution to the above mentioned problems we decided on windowing each sample and then concatenating all created sub-windows into one large tensor. We also included the ability to add stride to these windows as to raise the chance of capturing local features in our sub-sequences. The creation of said sub-windows did result in a loss of (potentially)

important information though, as the last windows were also dropped instead of being padded. Due to the medium sized window lengths, this should not impact training too much.

3.4 Architecture

For our 1D-CNN, we chose a mixture of Conv1d Layers with an hour glass shaped feature map size progression whereas for our kernels, we started of with a relatively large kernel size and decreased it over the progression of layers. These configuration was chosen as it allows for a better low-to-high level feature extraction process. After each Conv1d layer we applied batch normalization to further aid the learning process. As the ReLU activation function seemed to perform poorly on our architecture, we decided to change it to TanH which achieved better results immediately. We then applied global average pooling and fed the output into a dense layer followed by a Softmax activation function to generate our final prediction.

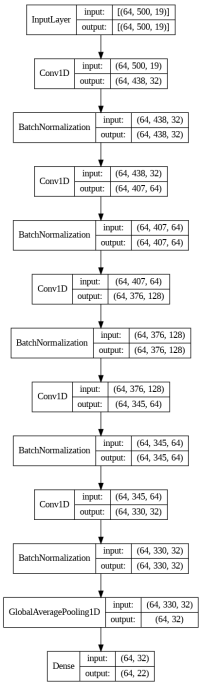

Figure 5: Model pipeline.

To train our model we used a training, test and validation dataset. The training and the test dataset were created by splitting the provided BBDC training data with a 80-20 ratio whereas the validation dataset was adopted using the provided BBDC test data. The training and test dataset were standardized as a whole before being split, whereas the validation dataset was standardized based only on itself (although instance standardization has been tested out as well).

The training process itself included model checkpointing and early stopping with a flexible patience parameter. AdamW (Loshchilov & Hutter, 2019) was chosen as an optimizer in conjunction with a small learning rate of 0.001 and a weight decay of 0.00001. AdamW in specific was chosen in favor of L2 regularization, as it regularizes variables with large

gradients more than L2 regularization would, thus potentially yielding a better performance and generalization which is especially important due to our low sample size. The full parameter list can be found in the Appendix at Table 5.3.

Chapter 4

Results

Our model achieved mixed results. While it had a favorable loss and accuracy development with respects to our test data, it did not perform well on the validation data set.

	Training	Test	Validation
Accuracy	$89 \pm 1.5\%$	$80 \pm 1.5\%$	$4 \pm 0.5\%$
Loss	0.33 ± 0.1	0.67 ± 0.1	8.73 ± 0.4

Table 1: Results of the training process.

From this we can conclude that model is capable of learning the required features, but, while not necessarily overfitting as the test data hasn't been seen by the model, it seemingly was not being able to generalize enough to new data.

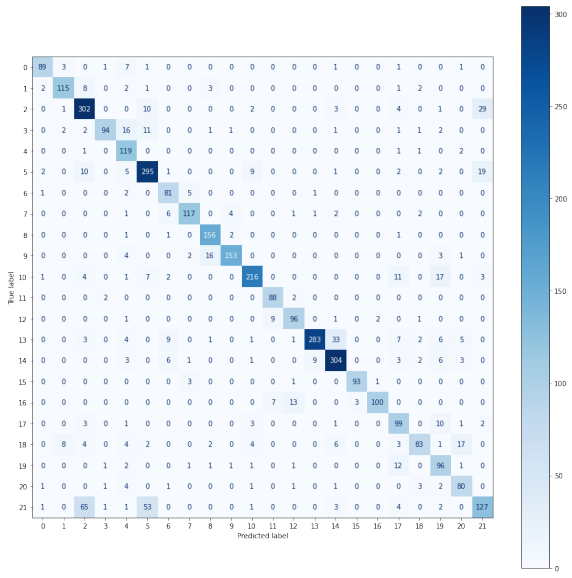

Figure 6: Confusion Matrix taken from the test dataset.

As can be seen in the Confusion Matrix above, our model performed well on the test dataset, classifying most labels correctly. Interestingly, Label 2, Label 5 and Label 21 seemed to show similar features, causing the model to sometimes mix up their classification.

The corresponding Tensorboard logs and model weights can be found in the Appendix at 5.4.

Chapter 5

Discussion

The reason for the bad performance of the model likely stems from the lack of samples of the underlying data distribution, their very diverse nature and the resulting implications when applying the standardization process. We tried tackling these problems using different preprocessing steps as well as tweaking the training process and the model architecture.

While preprocessing the 3 datasets, we, amongst other things, tested out normalization based on the true min and max values of the BBDC training dataset, but this proved to lead to even more undesirable results. we also tried this wrt. to each sensor node individually, but this also did not result in better performance. Furthermore, we tried standardizing the data based on the median standard deviation and mean of the whole data set, opposed to the sensor specific value distributions, which lead to a decrease in accuracy on both the test and the validation data. Additionally, we also experimented with applying a Butterworth filter to the individual sensor streams with the goal of normalizing and smoothing the presented data. This also did not result in better performance.

Another thought was that it might be a bad idea to split the test data into a training and test set, as this further decreased the amount of available samples, but after only using the test and validation dataset the results did not change in a significant way.

Some tweaking of the learning rate, introducing dropout layers and tweaking their drop-rate, relying more on a normalization layer to address the differences in normalization, introducing a gaussian noise layer and using grouped convolutions were among the most significant changes that we implemented with respect to the general model structure, which sadly did not result in a worthwhile change in the validation loss either.

Finally, the lack of performance might also be attributed to a certain problem within fully Convolutional Networks themselves. This being the the failure to pick up on long-distance relationships, something that is often crucial for time series classification. As we intended to explore the possibilities of "pure" 1D CNNs, implementing a combination of our model with different architectures such as GRUs or Transformers which would likely improve the capabilities of our model significantly.

A nice addition to this project would have been a comparison of the impact different kernel

and feature map sizes hold on the models accuracy. While we did some experimentation in this regard, more exploration could have neatly been portrayed in a whisker plot, but was sadly dropped due to time constraints and the more pressing training issues.

List of Figures

List of Tables

Bibliography

Anguita, D., Ghio, A., Oneto, L., Parra Perez, X., & Reyes Ortiz, J. L. (2013). A public domain dataset for human activity recognition using smartphones [Accepted: 2013-12-03T12:13:39Z], 437–442. Retrieved March 21, 2023, from https://upcommons.upc.edu/handle/2117/20897

Brownlee, J. (2018). 1D Convolutional Neural Network Models for Human Activity Recognition. Retrieved March 21, 2023, from https://machinelearningmastery.com/cnn-models-for-human-activity-recognition-time-series-classification/

Dua, N., Singh, S. N., & Semwal, V. B. (2021). Multi-input CNN-GRU based human activity recognition using wearable sensors. *Computing*, *103*(7), 1461–1478. https://doi.org/10.1007/s00607-021-00928-8

Ismail Fawaz, H., Forestier, G., Weber, J., Idoumghar, L., & Muller, P.-A. (2019). Deep learning for time series classification: A review. *Data Mining and Knowledge Discovery*, *33*(4), 917–963. https://doi.org/10.1007/s10618-019-00619-1

Lara, O. D., & Labrador, M. A. (2013). A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys Tutorials*, *15*(3), 1192–1209. https://doi.org/10.1109/SURV.2012.110112.00192

Lee, S.-M., Yoon, S. M., & Cho, H. (2017). Human activity recognition from accelerometer data using convolutional neural network. *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, 131–134. https://doi.org/10.1109/BIGCOMP.2017.7881728

Loshchilov, I., & Hutter, F. (2019). Decoupled Weight Decay Regularization [arXiv:1711.05101 [cs, math]]. https://doi.org/10.48550/arXiv.1711.05101

Appendix

5.1 Meeting report

We held a meeting with Mathis on the 15th of March to discuss our project's direction. At the beginning we confirmed if our approach was suitable, at which point Mathis reminded us to always check our assumptions starting a project with any approach (i.e. why we decided to pick the approach that we picked).

Several questions have been discussed, first and foremost the topic of how we should be structuring our windowing especially in regards to the affinity of 1D CNNs to favor local features. Another question we had was regarding the Normalization. The data involved sensor data and labels. We weren't sure if we should normalise per data and label, which Mathis discouraged, since the normalization wouldn't be usable without label. Most importantly, we should bring the data into an effective numerical range.

One big question was whether or not we could use a multi-headed approach. Mathis instead suggested to use group convolution, with different kernels for each window.

Regarding the Windowing, we discussed different lengths of sequence inputs, which Mathis assured us that it wouldn't be a big problem, as 1D CNNs do well with these, assuming we do not flatten the input but use global pooling instead. If must be, these could be standardized to the same length.

Mathis then suggested that we could take a look at two different approaches and compare the performance of each, regarding short-term- and long-term relationships. We decided that while this is an interesting approach, we wouldn't be able to fully realize it due to time constraints.

5.2 Label distributions

Below one can find two exemplary value distributions belonging to two different labels.

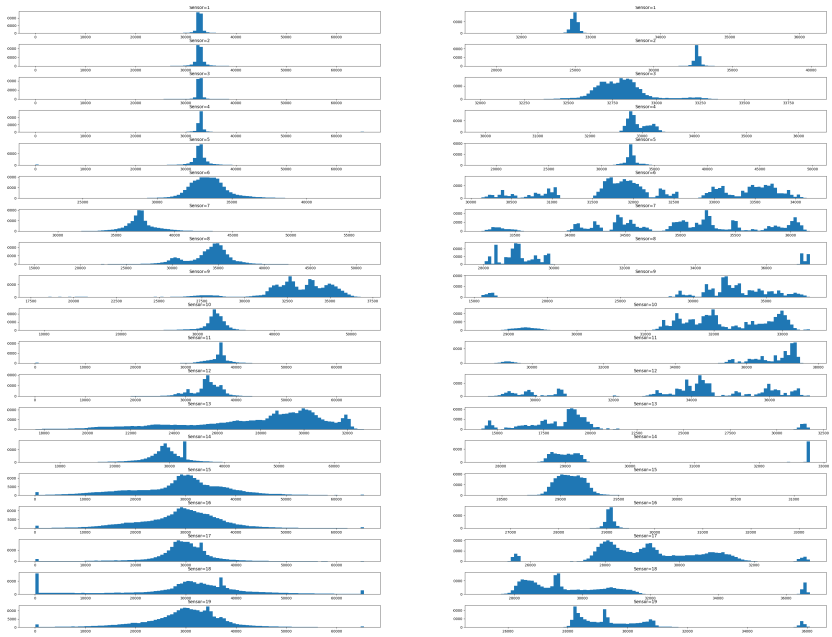

Figure 7: Sensor value distributions belonging to a ”walk” and ”sit”-labeled sample.

5.3 Parameters

The parameters used for training can be found in the table below.

split	0.2
epochs	100
batch_size	64
optimizer	AdamW
lr	0.001
weight_decay	0.0001
window_len	500
stride	450
patience	10

Table 2: Parameter combinations used for training.

5.4 Files

Preprocessing Notebook

Colab Notebook

Tensorboard logs and model weights

Declaration of Authorship

We hereby declare that the paper presented is our own work and that any assistance we received in its preparation is fully acknowledged and has been fully disclosed.

In addition, we affirm that neither we nor anybody else has submitted this paper or parts of it to obtain credits elsewhere before.

We have also cited all sources from which we used visuals, data, ideas or words, either quoted directly or paraphrased. All secondary literature and other sources are marked and listed in the bibliography. We also understand that a grade will not be assigned without the submission of this agreement.

Osnabrück, March 2023