Representation Learning for Gait Analysis in Parkinson's Patients

Tim Löhr and Christoph Popp

Department of Computer Science, University of Erlangen-Nürnberg

***Abstract*—This project aims to quantify how accurately Morbus Parkinson's can be classified by different types of deep learning architecture without preprocessing the original sensor data. For this purpose, four different architectures (LSTM, ResNet, a basic autoencoder and a ResNet autoencoder) were used to evaluate the accuracy. The data was collected from patients at the University Hospital of Erlangen. Different severity levels of Parkinson's were regarded as being diseased. In this regard, this project performed a binary classification task (healthy and diseased). It shows, that a ResNet autoencoder predicts Parkinson with 87% accuracy and can be used as a decision support system for doctors.**

***Index Terms*—Machine Learning Timeseries, Parkinson, Data Analysis, Gait Analysis, Neural Networks, FAU, Department of Computer Science**

1. Introduction

Hospitals in Germany are digitizing their workflows more and more every year [1]. This offers great opportunities to make use of large amounts of collected medical data or being able to collect data in a novel way, for example with different kinds of sensors. Machine Learning and Deep Learning models can be efficiently and cheaply applied by supporting doctors to identify diseases. Morbus Parkinson is the second most common neurodegenerative disease after Alzheimer with around 250.000 to 280.000 affected people in Germany alone [2]. A common way to test if a patient is diseased with Parkinson's is the L-Dopa test. This drug can reduce the complaints of patients with only a high single dosage. On the contrary, the risk of possible side effects rises by the younger the patients are [3]. For this reason, the highest priority in our classification task is to reduce the number of false-positives as much as possible, especially when taking age into consideration. The background of this project is, that we wanted to know how well we can predict Parkinson with sensor data that underwent no special preprocessing, only feature normalization.

The research question for this project can therefore be concluded as:

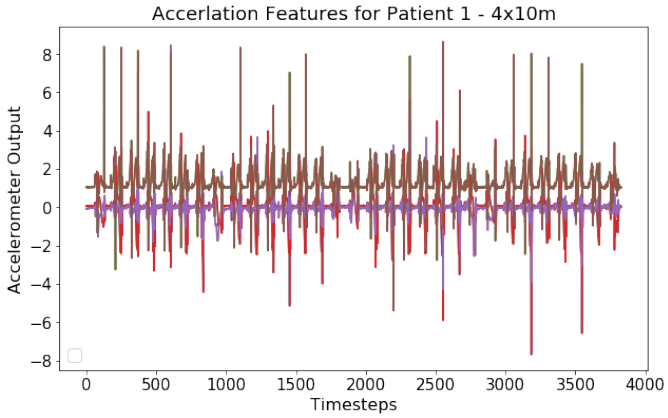

Fig. 1. Timeseries Plot of the acceleration features of patient 1

Can Deep Learning be safely applied as decision support for doctors to identify Morbus Parkinson with normalized sensor data?

2. Methods

For this dataset, the hospital in Erlangen measured 290 patients with three sensors on each foot. The sensors have been attached to the feet of the patients and they were told to walk 4 times 10 meters. In most cases, the time-series data reveals the turns after every 10 meters of the walk as can be seen in Figure 1, where the time-series shows four distinct sections.

It is split up into two-thirds of Parkinson diseased patients and one-third of healthy patients. Based on the *UPDRS* (Unified Parkinson's Disease Rating Scale), there exist seven levels of severity of Morbus Parkinson. For our experiments, we only took into consideration whether or not a patient has Parkinson's, not how severe the disease is. There is also the metric *Hoehn Yahr*, which gives an indication of the severity. We regard patients as healthy if both the UPDRS and the Hoehn Year metric shows a 0. So even if patients have

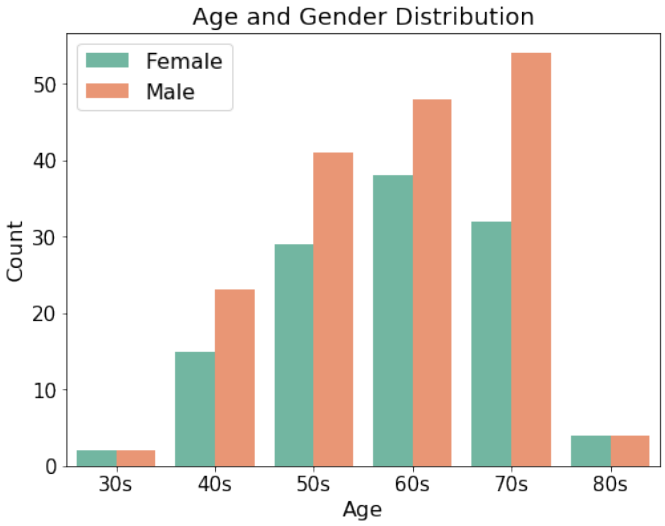

Fig. 2. Age and gender distribution of the 290 patients

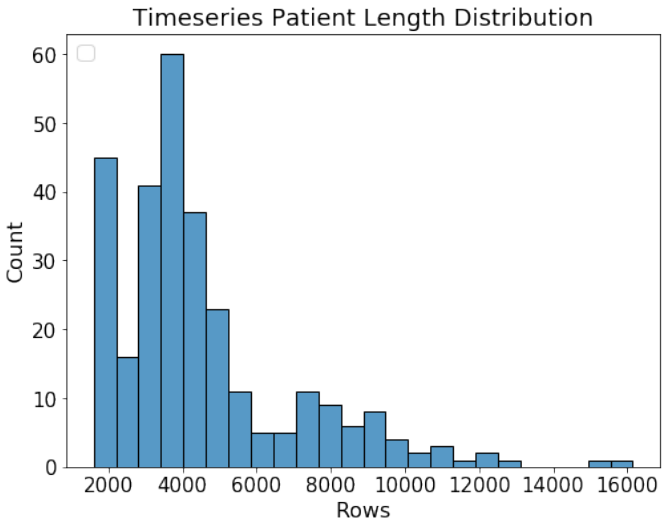

Fig. 3. Distribution of the length of the timeseries data for the 290 patients

Parkinson's on the best or worst possible condition, we regard everything as sick. Therefore, we have a binary classification problem.

The mean age of the patients from this dataset is distributed normally around 60 years (Figure 2). This is also the most common age for people to be affected by this disease [3].

The measurement of the six sensor results in twelve features per patient, because every sensor has an x and y measurement. On some occasions, the left-foot and right-foot data needed to be aligned. Furthermore, one row in the dataset does not necessarily indicate one second. The length of each patient's data varies a lot by length. The shortest measurement has around 2000 rows, whereas the longest has 16000 (Figure 3) with a median value of 4000 rows.

A recurrent neural network does not care about the input length. For the convolutional neural networks, we needed to make a fixed cut at a certain length and padded data that is shorter than the cut with zeros. We tested our binary classification task with four different Deep Learning approaches:

- Long Short Term Memory (LSTM)
- ResNet (CNN)
- Autoencoder Sampler
- Autoencoder Reconstruction Error

The models needed to be prepared in very different manners. Architectures and hyperparameters were tested differently for each model.

2.1 LSTM

The LSTM was the first attempted to model the binary classification task. Recurrent neural networks are often used for time-series tasks. The LSTM could be able to find patterns in the timeframes, that indicates if the person is diseased. Parkinson often leads to a trembling walking [2], so the LSTM possibly captures the trembling in the sensor data. We used a very basic architecture with:

- Input size: 12 (features)
- Hidden size: 256
- Number of Layers: 5

The hidden size and the number of layers were chosen as commonly used default values. Leaky ReLU's are additionally used and fully connected layers, which resolve in a scaler output which is used to compute the binary entropy loss for the binary classification.

2.2 ResNet

The second approach for the classification task was to use a convolution architecture to find patterns in the data. A common architecture is the ResNet, which is usually used in image classification tasks. Besides the ResNet architecture, there are many different ones, which could be used like the Inception architecture or the classic AlexNet. All of those architectures are relatively old and because of that we also considered a newer architecture called EfficientNet. The main idea here is to let the network learn the size of the convolution operation instead of fixing it. Yet, after some runs this architecture did not seem to work for our problem, providing significantly worse results than the simple ResNet. This is the reason we concentrated on the improvement of the ResNet architecture. We ended up with the usage of four Resblocks consisting of two convolution layers, two batchnorm layers, two relu activation functions and one one-by-one convolution layer in the order: convolution, batchnorm, relu, convolution, batchnorm, relu, 1x1 convolution.
Before the Resblocks one convolution layer was used with a relu activation function and a maxpooling layer. After the last Resblock one averagepooling layer was used and the flattened output was fed into three fully-connected layers with exponential linear unit (ELU) activation functions and a dropout chance of 20%. To get the binary classification result, as the last layer another fully-connected layer was used together with a sigmoid activation function to map the output values to a range from 0 to 1. The last layers used an ELU activation function instead of a relu to minimize the vanishing gradient problem. All the other building blocks are used as in a

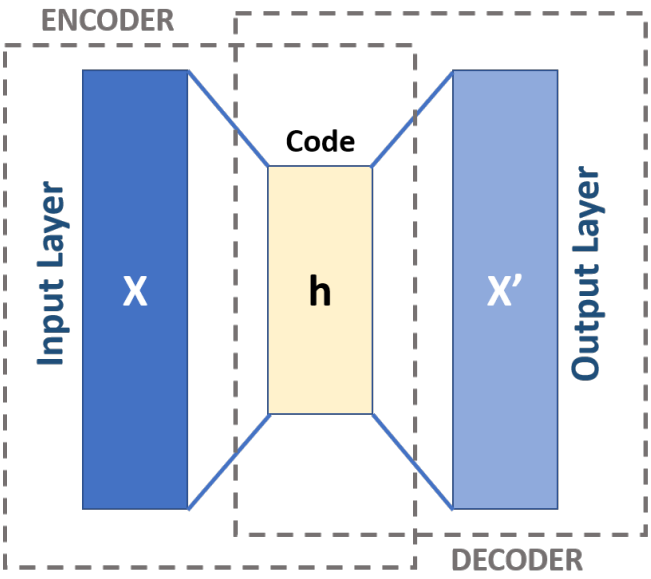

Fig. 4. Basic autoencoder architecture, Image from Wikipedia

classic ResNet. For training, the binary-cross-entropy loss was used, since this is the standard loss for binary classification problems. As an optimizer, Adam was the right choice with a learning rate of 8e-4, because other optimizers, like simple SGD, produced outputs that were always very close to each other.

2.3 Autoencoder

The classical autoencoder consists of two parts as shown in Figure 4:

- Encoder
- Decoder

The encoders architecture can be freely designed as long as the input is downsampled to a bottleneck vector **h** like in Figure 4. The bottleneck vector can be as small as a scaler value, but a higher dimension is also possible. For non-linear activation functions, the autoencoder behaves like a Principal Component Analysis (PCA). The lower-dimensional representation of the data is upsampled again by the decoder. The MSE loss (Mean Squared Error) can be calculated by the difference of the original input and the downsampled and back upsampled input. Ideally, the networks learn how to reduce the dimensionality of the input data in the bottleneck layer and can reconstruct the original image from the downsampled version perfectly without a loss of information. There exist multiple ways in which an autoencoder can be applied and designed. For our project, we implemented two possible ways of classifying the sensor data binary.

We also decided on a fixed-length time-series size of 8192 for both of our autoencoder networks. Due to the median length of data being around 4000 rows long, most data is padded at least a little bit with zeros at the end. Still, the networks have shown to increase their accuracy clearly by doubling the first attempted block size of 4096 to 8192. This makes intuitive sense because more data can be used to train the networks.

2.3.a Autoencoder Sampler: The autoencoder sampler is trained both on the healthy and diseased patients. This enables the network to both classify and also sample new healthy and diseased data from the model. For this architecture, three networks have to be trained:

- Encoder
- Decoder
- Classification Network

The training data is sampled down by two fully connected layers into a 4D bottleneck layer.

- Fully Connected Layer from 8192*12 into 128
- ReLU activasion function
- Fully Connected Layer from 128 to 4

The dimensionality reduction from 12 features into 4D is then used to simultaneously train a small classification network, consisting also of fully connected layers and ReLU activation functions. The networks try to make a binary decision based on the 4D dimensionality reduced data. The downsampled 4D data is then decoded back and upsampled into the original shape and the loss is computed by adding up both the decoder loss and the classification network's binary cross-entropy loss (BCE).

2.3.b Autoencoder Reconstruction Error: The autoencoder sampler is trained only on the healthy patients' data. In this case, we would only be able to sample healthy patient's data, which has no benefit for sampling tasks. The main purpose of the network is therefore only the binary classification.

For this autoencoder, a deeper architecture has been chosen, because the sampling autoencoder appeared to be not complex enough with too few parameters. The classical ResNet from chapter 2.2 is used as the encoder architecture, and the reversed ResNet is used as the decoder to upsample the bottleneck vector back into its original input shape. This huge network has 47.1 million parameters to train on. Since the ResNet alone worked very well, it appeared that the ResNet architecture could work even better when applied to an autoencoder.

The reconstruction loss is ideally zero when training and predicting only healthy patients. The idea is to use the reconstruction loss after the training for making novel predictions on Parkinson's diseased patients. Since the network has not seen any Parkinson data yet, the reconstruction loss should be greater than zero and a threshold can be set to classify between healthy and Parkinson.

3. Results

3.1 LSTM

After training the network for 100 epochs, the network predicted every patient as being Parkinson's diseased. Due to unbalancing factor is two-thirds of sick patients, the accuracy was 66%. No matter with which learning rate or deeper architectures of the LSTM architecture, the network did not start learning how to classify properly.

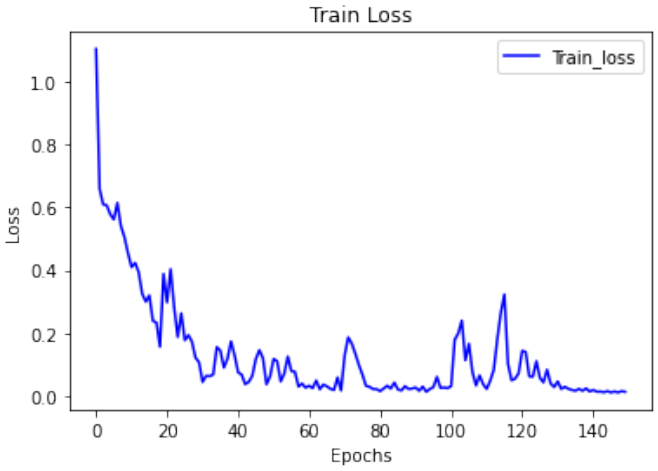

Fig. 5. Training loss of the binary cross entropy function

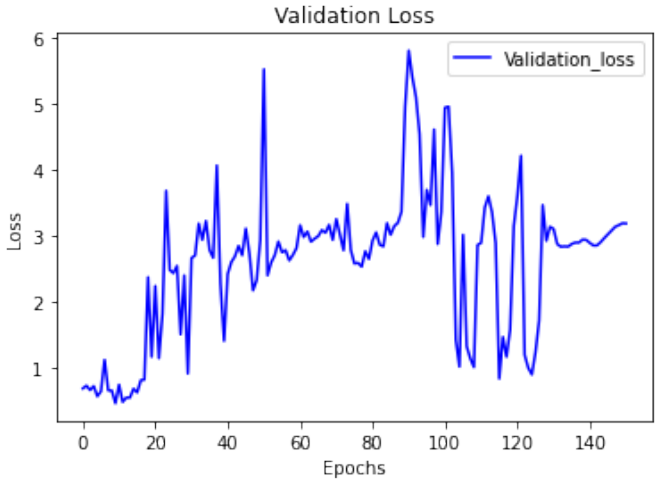

Fig. 7. Validation loss of the binary cross entropy function

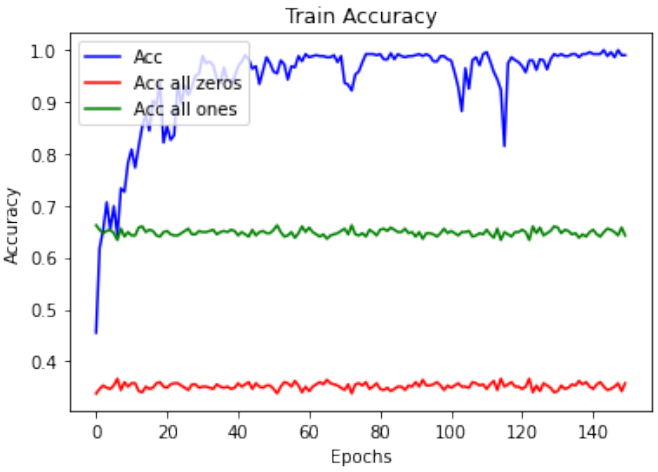

Fig. 6. Training accuracy of the ResNet

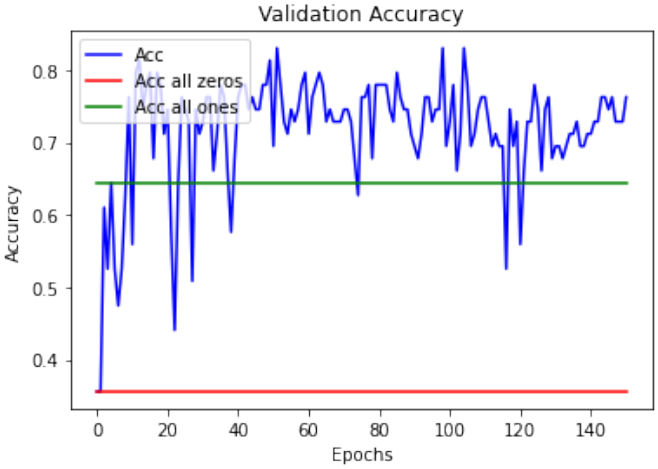

Fig. 8. Validation accuracy of the ResNet

3.2 ResNet

As already mentioned in the ResNet subsection we focused on the ResNet, because it gave us the best output from all other architectures. To get a feeling for what a good output is we have to notice that our dataset was imbalanced, containing 2/3 of Parkinson's and only 1/3 healthy data. Because of that imbalance, an accuracy of around 66% can already be achieved by classifying each input as Parkinson's disease. First, let us have a look at the training loss and training accuracy of the model.

The reconstruction loss is ideally zero when training and predicting only healthy patients. The idea is to use the reconstruction loss after the training for making novel predictions on Parkinson's diseased patients. Since the network has not seen any Parkinson data yet, the reconstruction loss should be greater than zero and a threshold can be set to classify between healthy and Parkinson. It can be seen that the loss curve goes down as expected and the accuracy always gets closer to 100%. This is good because that way we can be sure that our model is learning something and is not just predicting random things. The negative part is that once the training accuracy gets close to 100% it is very likely that our model is overfitting to the training data, which results in a worse result in the validation data. Next comes the validation loss.

Here it can be seen that the curves are not as smooth as the ones for the training. The accuracy here is varying around 75% with a small downward trend for higher epoch numbers. This downward trend can be caused by the overfitting of the network. Also, it is not clear what role the padding with zeros in the data plays. This could have been also a reason why the accuracy is not increasing more.
All in all the accuracy is way above the threshold of 66% with peaks of 82% or even 88% depending on the initialization, which shows that the network is learning a representation of the data and not just the classification of every input to one or zero.

3.3 Autoencoder Sampler

The sampling autoencoder was the first approach for the autoencoder architecture. It appeared that even after 150

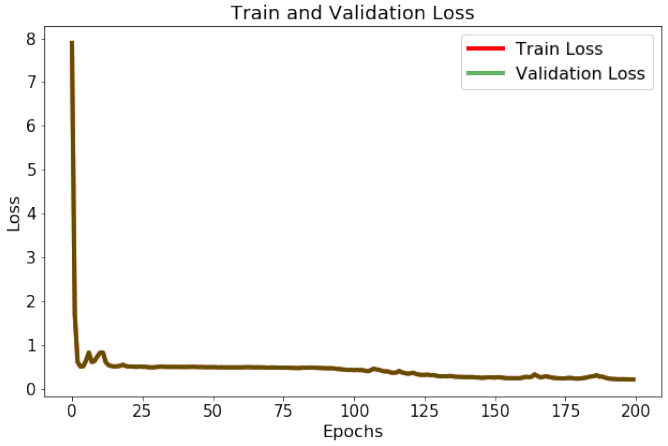

Fig. 9. Train and Validation loss of the reconstruction loss autoencoder

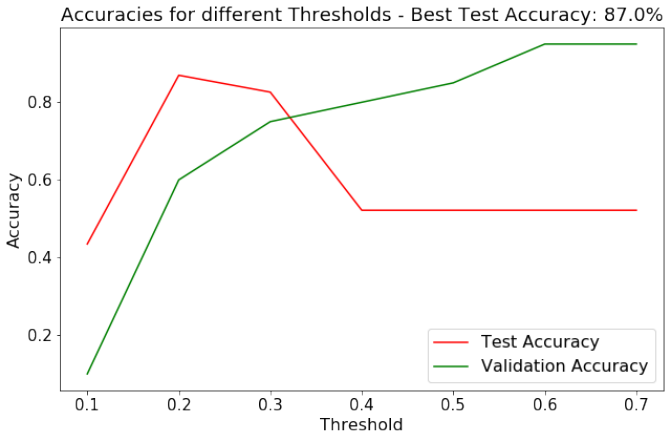

Fig. 10. Train and Validation accuracy of the reconstruction loss autoencoder

epochs of training, the simple fully connected architecture was not capable of training a model that can simultaneously train on healthy and sick data. The reconstruction loss for both the healthy and sick patient's data was completely arbitrary and no proper classification could be done.

3.4 Autoencoder Reconstruction Error

Since the classical ResNet worked well, an autoencoder with the same kind of architecture could work as well. When training only on the same label (healthy), the network is expected to heavily overfit the healthy patient's data. After 200 epochs of training the loss converged to a fixed 0.23 value (Figure 9).

A test dataset with 13 healthy and 10 diseased patients was used to make predictions based on the reconstruction loss. It showed, that the best accuracy of 87% was achieved with a threshold value of 0.2, as shown in Figure 10.

When looking at Figure 10, the validation accuracy quickly rises to 100%, because the validation dataset consists only out of healthy like the training dataset. Therefore, the accuracy must rise to 100% when further enhancing the threshold. The test accuracy has its peak at 0.2, but the greater the threshold is increased, the more false-positive are reduced.

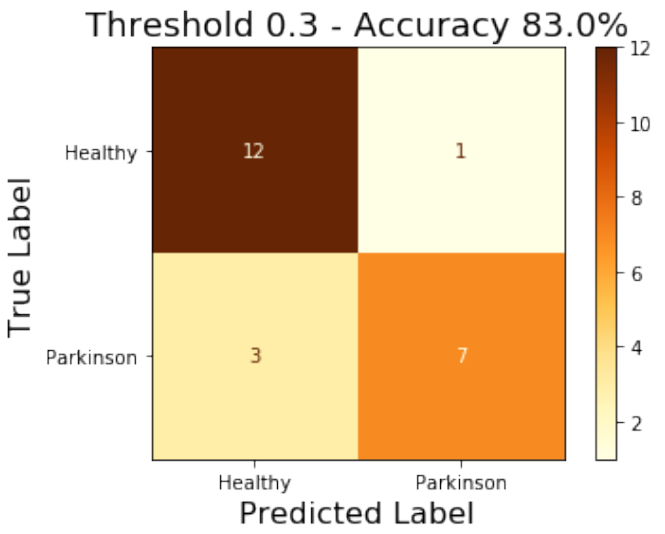

Fig. 11. Confusion Matrix for the treshold 0.2 with the accuracy of 87%

So ideally, when wanting a 95% certainty for not having any false positives, the threshold must be set so high until this constraint is satisfied. This is the advantage of the autoencoder with classification on the reconstruction loss over the classical ResNet classification.

The confusion matrix in Figure 11 reveals exactly this. By further increasing the threshold, the false-positives (lower left) will decrease, whereas the false-negatives (upper right) will increase. People left undetected with Parkinson's increase, but no patient is harmed due to medication for a false treatment.

4. Discussion

The experiments showed, that a convolution-based architecture like the ResNet performs better than recurrent based neural network architectures. The ResNet itself was developed in 2012, so there exist much more modern architectures with more parameters. The results show that CNN based architectures perform reasonably well, but it is not clear what the models have exactly been learning. The big difference in rows for each patient led to many problems for performing the training. Still, there is no clear indication that more data means a patient has Parkinson's, which would be a reasonable assumption since Parkinson's diseased people walk generally worse than healthy people and therefore need more time and rows. Since this was assumption was false, it appears the network learned the difference in some sense. Nevertheless is the autoencoder approach more useful than the basic ResNet architecture, because it enables us to adjust the threshold and accordingly uses this to adjust for a minimum false-positive or false-negative ratio. After considering all evaluation steps, can Deep Learning be safely applied as decision support for doctors to identify Morbus Parkinson with normalized sensor data?

Pros

- Cheap and easy decision support
- Arbitrary threshold enables adjustment of the false-positive rate

- Age can be taken separately into consideration after the prediction
- The accuracy of 87% indicates some level of certainty
- Sampling of healthy patient data is possible

Cons

- The prediction allows no medical evidence at all
- It is unclear what the network exactly learned
- More data is required

The autoencoder with reconstruction loss can be used as cheap decision support for doctors in the future, also for various other diseases, this project can be used as a template, because most sensor data does not vary that much from each other.

5. Conclusion

The conclusion of this project leads to three finals questions:

Can we classify without any special kind of preprocessing? The results have shown, that the Resnet and the autoencoder with reconstruction loss have a performance of 87% accuracy. Without any further bayesian optimization or a deeper network, this indicates that the networks could even improve further. The answer to this question is a clear yes.

Which network architecture performs the best? After the experiments, it showed that recurrent models could not capture the complexity of the task at all. CNN based architecture worked the best for this project.

Does the accuracy and prediction provides evidence that the network clearly understands the difference between Parkinson diseased and healthy patients? No clinical evidence at all. There is a possibility, that the network only learned the padding at the end of each fixed batch block of length 8192, even though the was no clear evidence that larger time-series data indicate Parkinson's disease.

6. Future Work

More adjustment can be made for future experiments:

- Using a more modern network architecture for the autoencoder like the Efficient Net
- Use bayesian optimization for the batch size and learning rate
- Train more epochs

References

[1] B. für Gesundheit, "Krankenhauszukunftsgesetz für die digitalisierung von krankenhäusern," Report, 2020. [Online]. Available: https://www.bundesgesundheitsministerium.de/krankenhauszukunftsgesetz.html

[2] P. Gesellschaft, "Parkinson," Report, 2018. [Online]. Available: https://www.parkinson-gesellschaft.de/

[3] D. G. für Neurologie, "Parkinson-syndrome: Diagnostik und therapie," Report, 2008. [Online]. Available: https://dgn.org/wp-content/uploads/2013/01/ll08kap_009.pdf