Using Recurrent Neural Networks on the WAY-EEG-GAL dataset

Roman C. Podolski
roman.podolski@tum.de

Dominik Irimi
dominik.irimi@tum.de

Christoph Dehner
dehner@in.tum.de

Manuel Nickel
manuel.nickel@tum.de

Philipp Bergmann
philipp.bergmann@tum.de

Abstract

The *WAY-EEG-GAL* dataset is designed to allow testing of techniques to decode sensation, intention and action from scalp EEG recordings in humans who perform a grasp-and-lift task. This paper demonstrates a methodology to analyse the data and how the problem of classifying various stages of these tasks by using a RNN on EEG and EMG data can be tackled. Particular findings in the EEG data and successes as well as pitfalls concerning the RNN runs are presented.

1 About the WAY-EEG-GAL dataset

In order to allow for critical evaluations of the utility of EEG signals for prosthetic control of object manipulation, the *Wearable interfaces for hAnd function recoverY - EEG - Grasp And Lift (WAY-EEG-GAL)* dataset was created. In particular, it can be used to examine to what extent it is possible to identify the intention to reach and grasp, hand positions and velocities or that the object's properties have unexpectedly changed. The data was collected during a series of experiments, for which twelve right-handed individuals, both male and female, were selected as participants. Each participant was seated at a desk with his hand resting on the table. A LED signalled the person to reach for a small object, grasp it using his index finger and thumb, lift it up a few centimetres, hold it for a brief moment and finally return the object as well as his hand to their initial positions. In this paper, this sequence of events and actions will from now on be refered to as one *trial*.

Each participant conducted 328 such trials under slightly and to the participant unpredictably changing circumstances. That is, the object's weight to be lifted was augmented through an electromagnet, resulting in weights of 165, 330 or $660g$. Furthermore, the contact surface changed to be made of sandpaper, suede or silk. Depending on the specific trial, either only the surface, only the object's weight, both or none changed. In order to record kinematics, forces, muscle activations and brain activity, four types of sensors were used during the experiments. In particular, an electroencephalography (EEG) cap, illustrated in figure 1, with 32 electrodes was used to record the brain activity during the trials, sampling at $500Hz$. Furthermore, five electromyography (EMG) sensors, as depicted in figure 2, were placed on the participant's right arm to measure muscle activity at a sample rate of $4kHz$.

The resulting dataset groups the trials of each participant in nine series. For each trial within one series, the EEG and EMG signals are concatenated in matrices respectively with a time stamp decoding the exact recording time for each measurement. Moreover, the time of the LED being turned on and off, the trial's start and end time, the type of surface and the object's weight are provided for each trial, as well. [1]

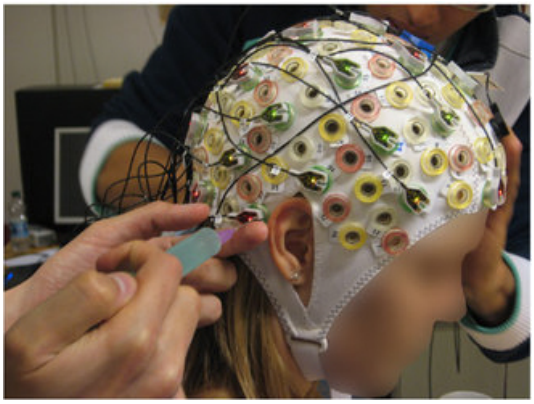

Figure 1: An EEG cap measures the participant's brain activity [1]

Figure 2: EMG sensors record the muscle activity of the right arm [1]

2 Analysing the data using t-SNE

In order to gain insight in how the EEG as well as EMG datasets are inherently structured, they were meant to be visualized. How the notion of *grasping an object* is captured in the data was of particular interest. That is, does an individual measurement by itself carry enough information allowing to conclude that the participant is right now grasping an object or not? Or is this notion rather encapsulated within a series of measurements, with datapoints that change over time?

2.1 Choosing Barnes Hut t-SNE as a visualization tool

As the individual datapoints are high-dimensional ($\mathcal{D}_{EEG} \in \mathbb{N}^{32}$ and $\mathcal{D}_{EMG} \in \mathbb{R}^5$ respectively), a particularly suitable method of visualization seemes t-distributed stochastic neighbourhood embedding (t-SNE), which allows to represent such data as a two-dimensional mapping by grouping similar state vectors close together and separating them from less similar ones. For the upcoming analysis an efficient Barnes Hut t-SNE implementation, created by L.J.P. van der Maaten [2], was used which allows the creation of plots with a vast number of data points, as it is the case for the EEG as well as the EMG data.

2.2 Reasoning with t-SNE

The following focuses on a subset of the WAY-EEG-GAL dataset. In particular, participant 1's first series, a dataset composed of a total of 34 trials, which are collected in `WS_P1_S1.mat`, is exemplarily examined. Each of the 34 trials were marked with a different color in the resulting plots, e.g. states of trial 1 might be colored in green while states of trial 2 in red. Furthermore, states that were measured within the time span of the LED being lid up are drawn with a black border in order to separate them visually from states with the LED being off. An important property of t-SNE is that it cannot consider time-relations between individual data points, instead it sees each data point as its own entity. However, in order to make sure that the used implementation of t-SNE is not somehow influenced by the ordering of the states, all data points were shuffled and placed in random order before being fed to t-SNE.

Having the above properties and settings in mind, one of these possible outcomes is expected to be observed:

- The LEDon-states are grouped together in one place while the LEDoff-states are grouped in another place: This may be interpreted as a sign that it is possible to extract information about the intention to grasp from individual data points. As t-SNE does not *understand* relations in time, even a simple Feedforward Net, for which the same applies, might suffice to classify grasping.

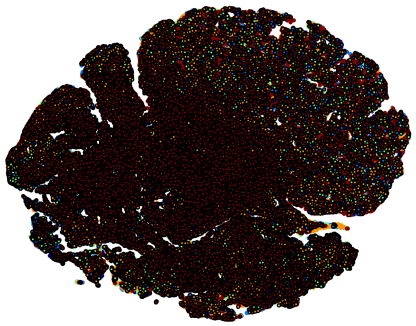

Figure 3: Randomly distributed data points in the case of EMG data when plotted through t-SNE.

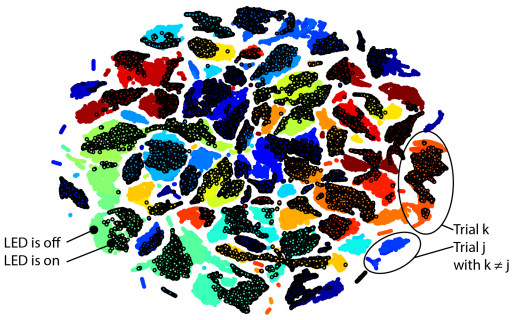

Figure 4: Significant clustering of trials in the case of EEG data when plotted through t-SNE. Different colours indicate different trials.

- All data points are distributed randomly and there is no apparent structure in the plot: This would signal that if there is a way to extract information about the intention to grasp, then it is not sufficient to use Feedforward Nets only. Instead, one needs to at least incorporate time relations and thus consider Recurrent Neural Nets.

Figure 3 shows the t-SNE plot for the 34 trials of participant 1, series 1. Clearly, all data points are distributed randomly and no particular grouping is to be seen. Following the reasoning described above, it can be concluded that using a Feedforward Neural Net cannot suffice to extract the grasping notion out of the EMG data.

The same t-SNE implementation was applied to the EEG data. The result can be viewed in figure 4. Here, the plot shows none of the expected characteristics. Instead, data points belonging to the same trial clearly form clusters while these clusters, each representing a specific trial, are then separated from each other. This indicates, that the EEG data somehow comprises a notion of time, encoding which measurement was taken during which trial. This is, however, not reasonable as the individual measurements are expected to be independent from each other. The correct function of the implementation was yet confirmed through extensive verfications.

2.3 Confirming the t-SNE results using Feedforward Neural Nets

Assuming figure 4 is correct, the plot suggests that it should be possible to determine for a given data point, i.e. measurement, during which trial it was taken. This comes down to a classification problem which can be solved using a simple Feedforward Neural Net.

Thus, a Neural Net was implemented, using one hidden layer, 32 input neurons and 300 hidden neurons. Furthermore, a tanh activation function and gradient descent optimization were used. The EEG data from participant 1, series 1 contains 146982 samples in 34 trials, which were put in random order and partitioned into a training-, validation- and test-set containing $4/9$, $2/9$ and $1/9$ of the original samples respectively. The dataset was normalized to contain values between -1 and $+1$. In particular, each trial was normalized separately in order to remove a possible offset in the data, which might have caused the clustering in figure 4. However, the Neural Net confirmed the t-SNE results: Using above settings, a test error of mere 2.38% was achieved for participant 1, series 1. The Neural Net was applied to a variety of other subsets of WAY-EEG-GAL, too. There, at times even better test errors of just about 0.01% were achieved. Overall, for any participant and series, the specific trials could be classified very easily by simply looking at a single measurement.

2.4 Consequences for the EEG data

The above results suggest, that indeed information about the time of measurement is included within the measurements themselves. There seems to exist a continuous change in value for at least a subset of the electrodes measuring the brain activity, leading to characteristic patterns within each trial's data. Various normalization and filtering methods were applied in an effort to remove these unwanted characteristics. However, none of these succeeded. In fact, the $-1/+1$-normalization

applied in above example even improved upon the Neural Net test error significantly, from a test error of 19.69% to one of 2.38%. Moreover, classification of trials turned out to be possible not only within a single series but over many more, too. For this, a Neural Net run was performed on the entire EEG data of participant 1, comprising 294 trials in total. This resulted in a test error of 15.65%. Thus, of whatever kind the structure within the EEG data distinguishing the trials may be, it is not bounded by individual series but seems to be of global nature.

As unexpected as these results may be, they do not neccessarily mean that the EEG dataset is entirely corrupted, as will be shown in the following sections. Furthermore, the clustering of LEDon and LEDoff-states respectively in figure 4 suggests that classification of *grasping* and *not grasping* could be possible through a Feedforward Neural Net, too. For participant 1, series 1 indeed a test error of 9.09% was achieved.

3 Methodology

The time-dependent experiment setting on the one hand and the results from the t-SNE in chapter 2 on the other hand, motivate to use a recurrent neural network (RNN). In comparison to a feed forward neural network, such a network contains recurrent connections within the hidden layer and thus saves a state between different inputs. By this way, the output does not only depend on the actual input but gets time-sensitive an is also influenced by previous inputs.

3.1 Recurrent Neural Network design

For implementation of a recurrent neural network Breze [6] is used. Based on python [3], Theano [4] and Climin [5], this library provides useful functionality to implement and train neural networks. The implemented network consists of one hidden layer with $\tanh$ activated neurons and sigmoidal output neurons. For the EMG data 100 inner neurons are used, for the EEG data 200. The weights are initialized by a random uniform distribution with expectation 0 variance 0.1. The network is trained in minibatches of size 50 with the optimizer Adadelta with 0.9 memory decay, 0.9 momentum, $1e-6$ offset and a step rate of 0.1. Another optimizer that performs well in this constellation is RmsProp.

During the learning progress, the network performs an error back-propagation after 300 data points are fed into the network. In combination with the subsampling rate, which will be discussed in the next chapter, this data slice size proved to be a good compromise between the depth of the network during the back-propagation and the period of time, the network can look in the past.

Furthermore 150 important weights are used. It means that the first half of the 300 samples of each back-propagation step are used to initialize the state of the RNN and do not contribute to the gradient as it takes some time until initial oscillations decay. Only the data points of the second half are then used to perform the error back-propagation and train the network.

3.2 Targets

The targets for the RNN are defined by specific time events given in the dataset. For the experiments of this paper, the following five targets are used:

1. Move hand to target phase: Event "Hand starts to move" until event "First digit touches the object"
2. Lift object phase: Event "Object lifts off" until event "Object reaches final lift position"
3. Hold object phase: Event "Object reaches final lift position" until event "LED is switched off"
4. Replace object phase: Event "LED is switched off" until event "object is replaced back"
5. Move hand back to start phase: Event "Both fingers released the object" until event "Hand stopped to move"

The targets for the RNN are represented by a one-hot-encoding scheme. Each target is represented by a separate target vector, stating for each time step if the target is active or not.

4 Data preprocessing

The EMG and EEG data is provided by [1] as *MATLAB *.m*-format files. Those files are structured in series for each participant a windowed format of each trial series, described in 1.

Based on these sources the neural network implementation can be configured to load various combinations of participants and series. Before the network starts to train, the data is preprocessed as described in the following.

4.1 Normalization

In the first step all input data of the same type (e.g. EEG, EMG) is scaled and normalized to zero mean. Single data point records have to be mapped to points in time, as those are provided separately.

4.2 Subsampling

Muscle activity from the five EMG sensors are obtained at 4kHz sampling rate. As obvious by visual inspection of raw signals, significant changes are following more sluggish behaviour. Thus, subsampling the raw data allows to speed up learning by reducing the amount of data and without loosing relevant information for the result. The simplest implementation is create a subset of records by a regular frequency without interpolation. In experiments of this paper, the data is by this way with 10Hz.

4.3 Splitting and reshaping

After normalization and subsampling, the data is split up for cross validation into a set of size 80% for training and two sets of size 10% each for validation and testing.

In the neuronal network implementation, error back-propagation should be performed after 300 data samples. To archive this, Breze requires the test and validation set to be provided as matrix in the shape $(300 \times \#\text{back-propagations in one epoch} \times \#\text{targets})$. Thus, the train and validation set is cut to a multiple of 300 and then reshaped.

5 Results

All RNNs are trained to predict the previously defined targets using EMG or EEG data. In this chapter, at first the results with EMG data and then the results using only EEG data are presented.

5.1 Results using EMG-data

Figure 5 shows the results of a RNN trained with six data series of particpant 1. In the first two plots the expected target vectors and the prediction of the RNN are compared. The third plot illustrates the train, validation and test error during the training epochs.

It can be seen that the RNN can learn the general experiment procedure out of the EMG data greatly. Each predicted sequence of the five targets corresponds to exactly one trial in the target vector. Out of the five targets, the move-hand-to-target phase, the lift-object phase and the move-hand-back-to-start phase are predicted very well. Except for some inaccuracies at the borders, these targets are always predicted correctly. The other two targets, the lift-object phase and the replace object phase, are not predicted very well. A reasonable explanation for this behaviour could be, that the targets are to small be learned properly during the training in this setting with one-hot-encoded targets and the Bernoulli loss.

So far, all this observations are made on results of a RNN, which was trained with data of just one participant. To demonstrate that predicting the targets with data of more than one participant works similarly fine, figure 6 visualizes the results of a RNN trained with data of four different participants. Compared to figure 5, the RNN with data of more participants produces lower amplitudes. However, the structure of the predictions does not change and still fits similarly well to the target vectors.

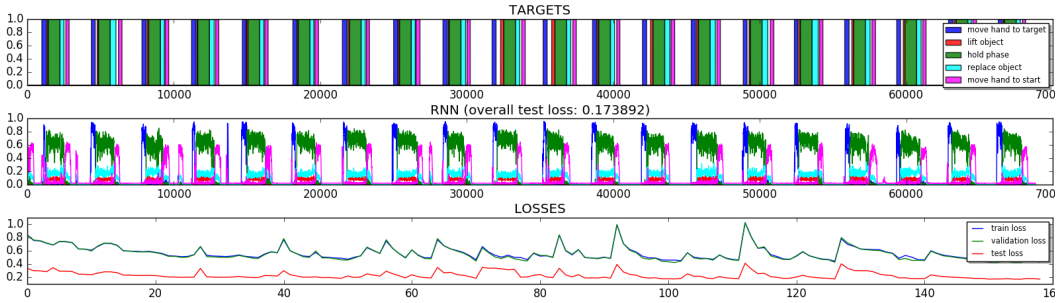

Figure 5: Results of RNN trained with data of participant 1, series 1-6

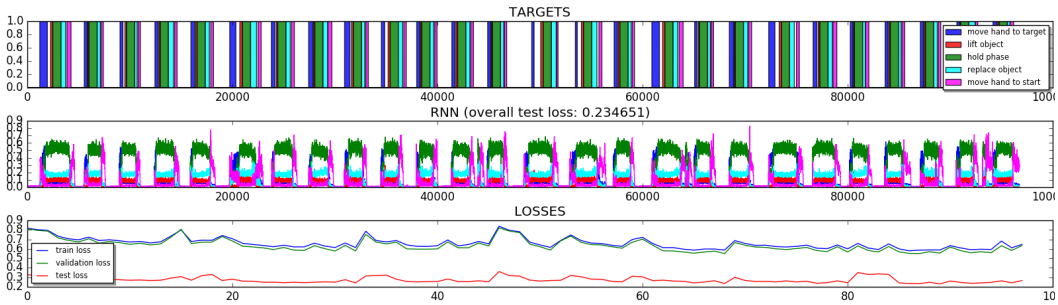

Figure 6: Results of RNN trained with data of participant 1-4, series 1-2

5.2 Results using EEG-data

The same network was used to predict the labels solely based on the 32 channel EEG data. Only the number of hidden units was increased to 200 to account for the the higher dimension of the input data and probable more complex structure of the EEG data compared to the EMG data.

Figure 7 shows the RNN trained on one person with the move hand to target label only, as its performance was good on previous experiments. It can be seen that the RNN is capable to find a structure in the EEG data to predict the labels correctly. It almost always shows a reaction on the right time, but the outputs do not have the right shape anymore and thus prediction of the timespan of the event is not possible. On the other side the RNN now is prone to false responses and reacts more often than it should. It seems that on a few test samples the performance is especially poor, as its output is quite high on the whole trial and correct prediction of the event becomes infeasible. When trained on more than one person the performance of the RNN decreases. It still can be seen that it reacts on the right times of the events but the output becomes very noisy.

On Figure 8 the RNN was trained with all 5 targets. Like before with the EMG data the lift-object phase and the replace object phase could not be predicted very well. The other 3 targets are also very noisy, but one can see that the RNN is able to find a structure in the data to predict these labels.

6 Summary

Using t-SNE it was shown that EEG measurements from the same trial form clusters in the stochastic neighbourhood embedding. A Feed-Forward-Artificial-Neural-Network was used to classify single trials from the EEG data and on this task an error rate of under 2% was archive. The contribution of this work to the analysis of the WAY-EEG-GAL dataset it the proof that there is not-sequential order in the dataset present. The provided plots and experiments yield the assumption that this structuring is based on the time of measurement, since measurements from the same trial cluster together. To find the discrete cause for this clustering further research is necessary.

Further this work shows that the WAY-EEG-GAL dataset can be used to learn a classifier for a stream of sequential EEG/EMG data to distinguish phases in grasp-and-lift motions. By using a Recurrent

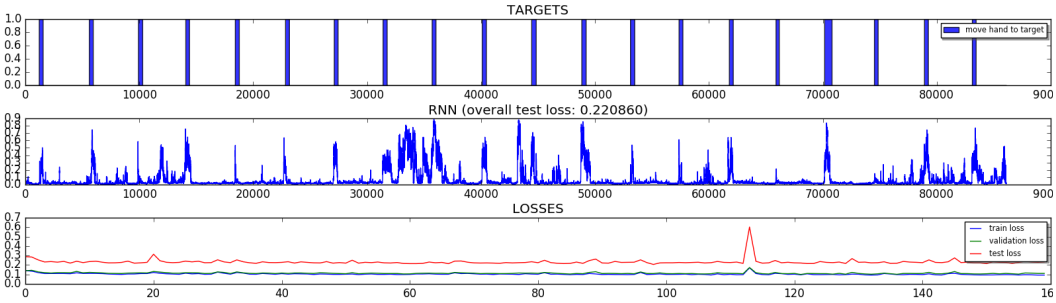

Figure 7: Results of RNN trained with EEG data of participant 1, series 1-6

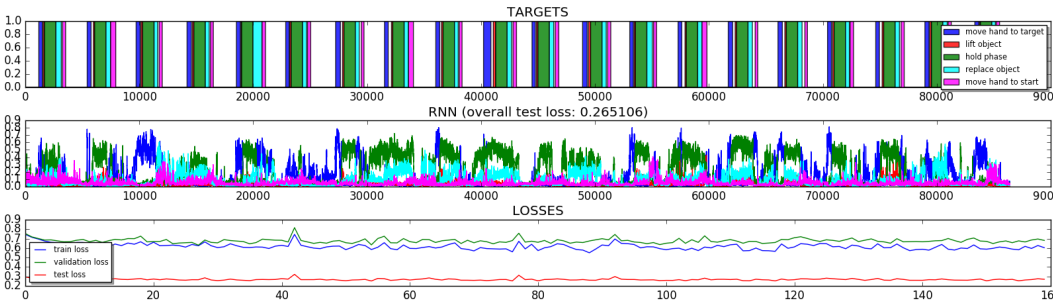

Figure 8: Results of RNN trained with EEG data of participant 1, series 1-6

Artificial Neural Network for the EMG data the classifier were able to distinguish 5 different phases for each trial, while for the EEG data it only detected the hand movement to the object. This work therefore proofs the concept of detecting intention of movement from EEG/EMG data by using a Recurrent Artificial Neural Network based classifier. Nevertheless, this concept faces some downfalls.

Target phases with a long duration where easily detectable by this approach, while short phases remained mostly undetected. Since Recurrent Neural Networks treat data input as a sequential stream of signals, short *spikes* may not be recognised as significant. It is likely that the Network threats those short signals similar to random noise. Therefore the certainty level of the corresponding output class does not grow over a recognisable threshold. A better design of the targets used for training the model or modifications to the model itself, which take this consideration into account, may overcome this problem.

For the EEG data one may criticise the overall design of the targets used to train the model. It is not possible to detect the exact phases were a participant develops the intention to apply movement. The design used is only an approximation based on external events and stimuli which signal the participant to apply movement. For the EMG Data this approach works fine, since discrete muscle contractions are measured and a clear correlation between those contractions and the applied movement can be found. Regarding EEG data it is possible that the model did not use the signals, corresponding to the development of movement, but those which correspond to the processing of stimuli which are associated with the phase of the experiment. In that case the EEG classifier could only be applied in the same scenario, as it was practised in the experiments [1]. To exclude this possibility, the model would have to be tested on the ability to transfer its knowledge to a similar, but not identical domain.

References

[1] Luciw, M. D., Jarocka, E., Landsberger, B. B. (2014): *Multi-channel EEG recordings during 3,936 grasp and lift trials with varying weight and friction*. Available at `http://www.nature.com/articles/sdata201447`, retrieved on July 20, 2016.

[2] van der Maaten, L. J. P. (2016): *t-SNE Implementations*. Available at `http://lvdmaaten.github.io/tsne/`, retrieved on July 24, 2016.

[3] Python, Python Software Foundation: *Python Language Reference, version 2.7*. Available at `http://www.python.org`, retrieved on August 5, 2016.

[4] Theano Development Team: *A Python framework for fast computation of mathematical expressions by Theano Development Team*. Available at `http://arxiv.org/abs/1605.02688`, retrieved on August 5, 2016.

[5] Bayer, J., Osendorfer, C., Diot-Girard, S., Rckstiess, T., Urban, S. (2016): *climin - A pythonic framework for gradient-based function optimization*. TUM Tech Report. Available at `http://github.com/BRML/climin`, retrieved on August 5, 2016.

[6] Breze. Available at `https://github.com/breze-no-salt/breze`, retrieved on August 5, 2016.