Spectral Analysis of LFP Recordings from Lizards During Sleep and Wakefulness

Ofir Shechter **Eden May-Tal** **Eden Aldema Tshuva**

Abstract

Sleep patterns, similar to those known in mammals, were found in Australian dragon lizards. Yet, characterization of features during wakefulness state is not well defined, and further investigation of different sub-states of waking was needed. In our project, we used LFP recordings from a Stellagama Stellio lizard and recreated the results found in the Australian lizard regarding the sleep patterns characterization. Then, we used signal processing and autocorrelation analysis, as well as clustering methods. We found two main frequency groups that can indicate two different sub-states of waking state. Finally, we used Multilayer Perceptron neural network in order to create a classification of sleep and waking states.

1 Introduction

Prior local field potential (LFP) recordings from the brain of a lizard, the Australian dragon Pogona Vitticeps, revealed slow-wave (SW) and rapid eye movement (REM) patterns (Shein-Idelson et al. 2016). Before these recordings, such electrophysiological characteristics were described only in mammals and birds, and not in Reptiles. These findings show that the brain-stem circuits accountable for REM and SW sleep are involved in sleep dynamics in reptiles, which implies the probable evolution of these dynamics is more ancient than considered, and is pushed back at least to the emergence of amnions (Shein-Idelson et al. 2016). Therefore, a debate about whether or not those results are specific to the Australian lizards has emerged, so the first problem we had to solve was finding REM and SW patterns in a Stellagama Stellio lizard, and by that recreate those results.

Furthermore, while the segmentation of sleep into two distinct states (SW and REM) is known, the variations in neural activity within the waking state remained elusive. Consensus is lacking even on the number of defining features that can distinct sub-states of waking, and better definition and quantification of the different waking states is needed (McGinley et al. 2015). Recent work on behaving mice revealed that bio-markers for waking states can predict changes in animals' capability to represent and respond to stimuli, and has a major part in spontaneous or stimulus-driven behavior (McGinley et al. 2015). For this reason, the second problem was to identify and characterize different neural states during wakefulness.

Due to the difference between sleep and wakefulness states, we wanted to see if it is possible for a simple neural network to classify these states. In addition, we wanted to use machine learning to predict future data from previous data, where the accuracy of the prediction may indicate how strong are the patterns.

2 Related Work

2.1 Prior analysis on sleep in Dragon Lizards

As described in an earlier analysis, LFP recordings can be segmented into two main spectral clusters. The first is δ-band, low-frequency activity (under 4 Hz), and the second is β-band, a broadband activity that is more like the activity during wakefulness (10-30 Hz). Those two clusters oscillate regularly throughout the night, forming a period of E-sleep (Shein-Idelson et al. 2016).

2.2 Wakefulness state findings

Four states of vigilance were defined in the green iguanid lizard: active wakefulness (AW), quiet wakefulness (QW), quiet sleep (QS), and active sleep (AS). Those states yielded different EEG, EOG, and EMG results (Ayala-Guerrero and Mexicano 2008). Quiet and active sub-states of sleep were defined also in rodents and were associated

with low frequencies and cortical activation respectively (McGinley et al. 2015). Further studies showed that the waking state contains continuous transitions between numerous sub-states that are correlated with changes in sensory-motor processing and behavior (McGinley et al. 2015).

3 Methods (our approach)

3.1 Signal Processing

We used **Welch's** spectral density estimation method to move from the voltage domain to the frequency domain. This method is based on a division of the data into bins and calculation of a periodogram for each, using **Discrete Fourier Transform**. The periodograms are then averaged in order to reduce variance in the individual power measurements. The output for this method is the estimated power for each frequency. (Welch 1967). According to the power of the frequencies and previous works (Shein-Idelson et al. 2016), we chose the frequencies' range to focuses on. Then, we normalized each frequency power by its average power to generate a normalized spectrogram. For the bins in this spectrogram, we calculate the **correlation matrix** to find similar bins and identify the frequency characteristics of the highly correlated bins.

3.2 Clustering - Ward's linkage algorithm

First, we tried several known clustering methods (k-means, affinity propagation, spectral clustering, and different kinds of hierarchical agglomerative clustering) on both raw data and spectrogram. After some consideration, we have performed hierarchical agglomerative clustering with **Ward's linkage**. Ward's linkage is a hierarchical clustering method which minimizes the variance within each cluster. Like in any agglomerative hierarchical clustering method, it starts with each sample in its own cluster. In each iteration two of the clusters are fused. The decision of which clusters to fuse is based on a linkage function. Ward's linkage is the increase of total ESS (error sum of squares) that would be caused by fusing each pair of clusters, where for cluster C with centroid $\vec{\mu} = \frac{1}{|C|}\sum_{i=1}^{|C|} x_i$, $ESS(C) = \sum_{i=1}^{|C|} |x_i - \vec{\mu}|^2$. The pair of clusters that minimizes the linkage function is chosen. (Sharma, Batra, et al. 2019).

3.3 Auto-correlation

Autocorrelation is a statistical method for finding repeating patterns in time series data. For each shift of k samples, the correlation between the time series and the k-shifted time series is calculated. Eventually, this builds a new time series describing the similarity between the original data and the shifted data for every shift size. Formally, for a time-series $S_1, ..., S_n$ where $S_{i,j}$ stands for $S_i, ..., Sj$ the Auto-correlation in shift of k samples is defined as follows:

$$Autocorr(k) = Corr(S_{0,n-k}, S_{k,n})$$
$$= \frac{E[(S_{0,n-k} - \bar{S}_{0,n-k})(S_{k,n} - \bar{S}_{k,n})]}{\sigma_{S_{0,n-k}}\sigma_{S_{k,n}}}$$

(Sokal and Oden 1978). In our context, this helps initiate the search of patterns that are unique for the waking state, as it can reveal patterns of periodicity in the signal. To further the research of the low frequencies in the autocorrelation, we have calculated a spectrogram of the autocorrelation. This shows more clearly whether or not the autocorrelation is similar to a simple combination of basic sine waves.

3.4 Multilayer Perceptron (MLP)

MLP is a feedforward artificial neural network consists of three layers: an input layer, a hidden layer, and an output layer. On the layer outputs we use Relu - nonlinear activation function (Gardner and Dorling 1998). MLP utilizes a supervised learning technique called backpropagation for training (Gardner and Dorling 1998). We used this network to distinguish between the sleep and waking states of the lizaed according to its' electrophysiological recording.

3.5 Saliency Map

A saliency map is a heatmap that highlights pixels of the input image that were the most significant for the classification. This map is calculated using the gradient of the output class score with respect to the input image pixel values. The pixels for which this gradient is largest (either positive or negative) are the pixels that small changes in them would affect the class score the most (Rastogi n.d., Khalid n.d.). Therefore, we used it to gain a better understanding of our data and its most significant features.

3.6 Long Short Term Memory (LSTM)

LSTM is a special kind of recurrent neural network (RNN), capable of learning long-term dependencies (Hochreiter and Schmidhuber 1997). While the standard RNN contains repeating modules with a single layer, LSTM contains four layers, and the interaction between them in the cell regulates the ability to remove or add information (Olah n.d.). The gates are as follows:

- **Forget gate:** a layer which decides, using a sigmoid, what information is thrown away.
- **Input gate:** a layer which decides what new information will be stored in the cell state using a sigmoid layer that decides which values will be updated and a tanh layer that creates a vector of new candidate values.
- **Output gate:** in this cell the selection of the parts of the cell states that will be outputted is made.

The above is the general concept of LSTM (Olah n.d.), the network we used is broadly described in the appendix (A.8). The use of LSTM for this kind of prediction is inspired by similar uses for stock price predictions (Saldanha n.d.).

3.7 R^2 score

In the case of time series problems, evaluation of the accuracy of the network is a hard task. In addition to the MSE loss value, we used R^2 score to determine whether the model is accurate or misleading. Best possible score is 1.0 and it might be negative (because the model can be arbitrarily worse) . It's calculated as follows: note y_i as the target value at step i and f_i as the model prediction. $SS_{res} = \sum_i y_i - f_i$ and $SS_{tot} = \sum_i y_i - \hat{y}$. We get $R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$ (Rowe n.d., *Coefficient of determination* n.d.).

4 Results

4.1 Spectral Analysis

We Created spectrograms (A.1) for both sleep state (Figure 1) and waking state state (Figure 2) and observed patterns that can distinguish two different states (in both spectrograms). It is clear that low frequencies have significantly higher power than high frequencies, but by applying logarithm on all powers, we can see some interesting patterns in higher frequencies.

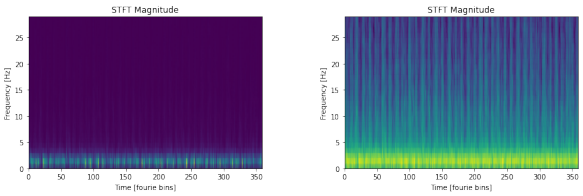

Figure 1: A spectrogram of an hour of LFP recording in **sleep state** (left is the original spectrogram and right is after applying log on all values).

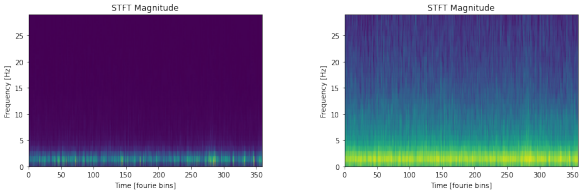

Figure 2: A spectrogram of an hour of LFP recording in **wakefulness state** (left is the original spectrogram and right is after applying log on all values).

We wanted to characterize these patterns, but we noticed that regular clustering did not give good results - only the lower frequencies who had significantly higher power, were taken into consideration. To eliminate this effect we created a normalized spectrogram (A.2) (Figure 3 , Figure 4).

4.2 Trying Different Clustering Algorithms

Before further analysis of the data, we tried popular clustering methods (as detailed in the methods section). To check their success, we compared the results to the known literature (Shein-Idelson et al. 2016). Also, we checked the homogeneity of each cluster by putting it together in the same picture and check what are the dominant frequencies that caused the separation. No meaningful results were found.

4.3 Clustering Using Ward Method

For the normalized spectrogram (sleep state), we calculate the correlation matrix (A.3). On the matrix, we performed **ward clustering** and got two distinct clusters that indicate two sub-states in sleep (Figure 5).

For each cluster, we calculated the average power in each frequency (A.4) (Figure 6). It can be seen that one of the clusters has high power at 0-4 Hz and the other at 5-30 Hz. These frequency ranges are known in the literature as δ (0-4 Hz) and β (10-30 Hz) (Shein-Idelson et al. 2016).

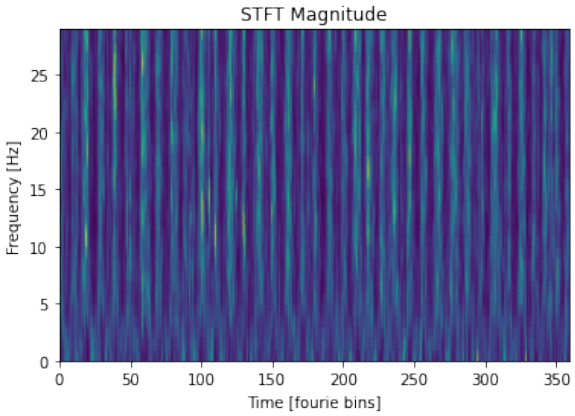

Figure 3: A normalized spectrogram of an hour of LFP recording in **sleep state**

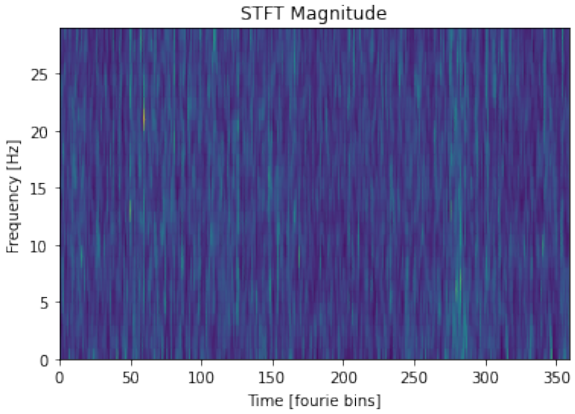

Figure 4: A normalized spectrogram of an hour of LFP recording in **waking state**

4.4 Finding Patterns in Wakefulness

Next, we attempted to characterize unique frequencies behavior in the waking state. In order to do so, we first created a spectrogram and correlation matrix using the same method as for sleep state. We got similar but less distinct results (Figure 7).

Then, we created a spectrogram using smaller segments (A.1) to detect higher frequencies separation. By using the same techniques used above for lower frequencies (A.3, A.4). We found that in contrast to the result above (similar frequencies characterization for sleep and wakefulness states), the division of the average power of frequencies was different for sleep state and waking state (Figure 9). In the sleep state, one cluster has higher power at all the frequencies that were examined (0-300 Hz), and the other has lower power at all the frequencies (Figure 8). In wakefulness, one of the clusters has high power at 0-170 Hz and the other at 170-300 Hz.

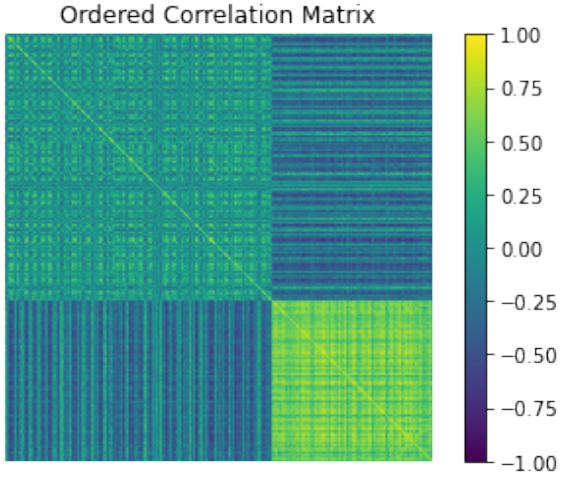

Figure 5: A Ward's linkage clustered correlation matrix calculated from the normalized **sleep state** spectrogram in figure 3

4.5 Sleep and Wakefulness patterns

As mentioned above, β and δ waves reflect two clusters regardless of the lizard's current vigilance state. Meaning, in one cluster β is the dominant band, and on the other one δ is the dominant band. We calculate the ratio between average normalized δ waves power, to average normalized β waves power and revealed a repetitive pattern in both sleep and wakefulness state. We discovered that at sleep state the alternations are slower and sharper than in wakefulness state, which establishes the rate of the alternations as a pattern depend on the vigilance state (Figure 10).

4.6 Autocorrelation and Powerful Low Frequencies

The autocorrelation analysis shows oscillations between 0.3 and -0.3 correlation, with picks around every second. This was not unique to either vigilance state and was lasted during the entire recorded period. (Figure 11, Figure 12).

This is consistent with the fact that both during sleep and during wakefulness, the most powerful frequency was of 1 Hz. (Figure 1, Figure 2). A spectrogram of the autocorrelation, focused on low frequencies (higher frequencies were not significant), revealed strong 0.5 Hz and 0.75 Hz frequencies. (Figure 13) These frequencies are also strong in the data itself throughout the recorded time period (Figure 14).

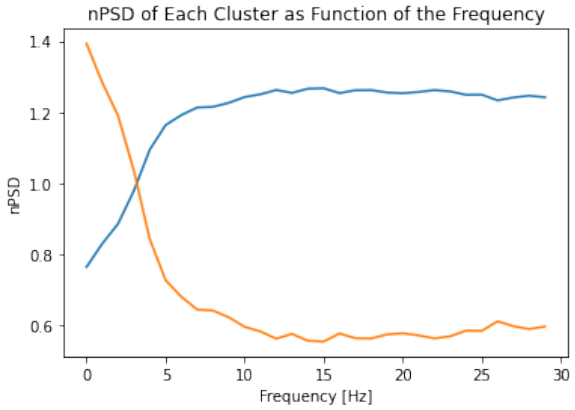

Figure 6: The average normalized power for each frequency for each of the found clusters in figure 5 during sleep.

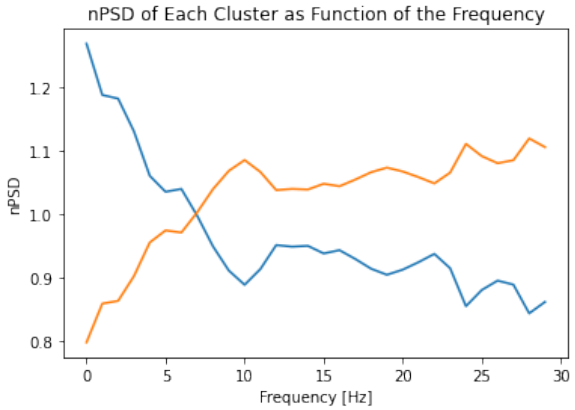

Figure 7: The average normalized power for each frequency for each of the found clusters during wakefulness.

4.7 Classify sleep and wakefulness state

Due to the intensive pre-processing of data (as described above), we were able to get an **outstanding success rate (100%)** using a deep learning algorithm Multilayer Perceptron (A.6) as a pictures classifier. Alongside the great classification ability, the network can help get a better understanding of the necessity of the signal processing performed. **For the same record test and train data (split using cross-validation) and the same network architecture, we got the following result:**

- **Spegtrogram:** for an input consisting of 36 bins from the non-normalized spectrogram, the network has performed poorly. we got training accuracy: 62.50% and test accuracy of 50%. Further investigation reveals that the network converged to the majority of the train data and return its' label constantly, so no meaningful learning is performed.

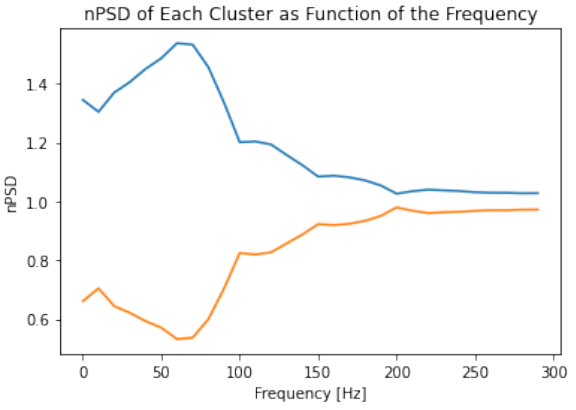

Figure 8: The average normalized power for each frequency for each of the found clusters in **sleep state** focusing on high frequencies.

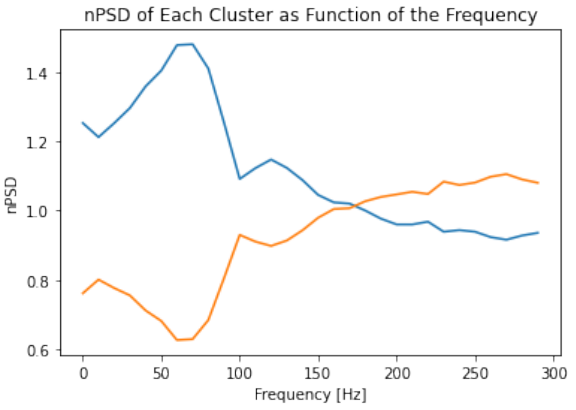

Figure 9: The average normalized power for each frequency for each of the found clusters in **wakefulness state** focusing on high frequencies.

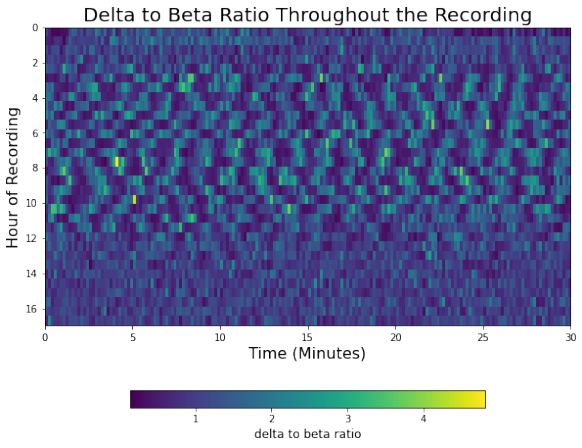

Figure 10: $\frac{\delta}{\beta}$ ratio for 16 hours of record. The region with the slower alternations is when the lizard is in sleep state.

- **Normalized Spectrogram:** for an input consisting of 36 bins from the normalized spectrogram, both training accuracy, and test accu-

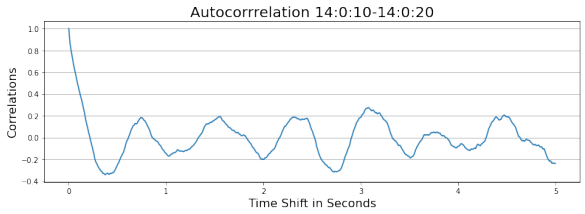

Figure 11: Example for auto-correlation behavior in **waking state**.

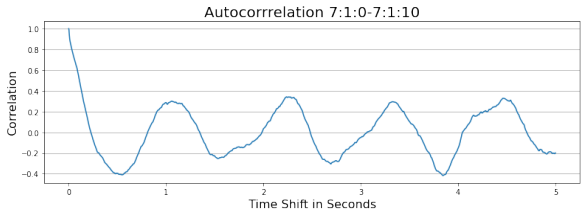

Figure 12: Example for auto-correlation behavior in **sleep state**.

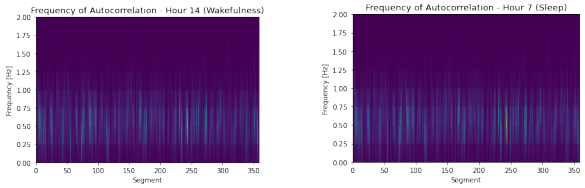

Figure 13: Spectrogram of autocorrelation in **waking state** (left) and **sleep state** (right).

racy are 100%.

Next, we used a saliency map (A.7) to visualize the network's work. As can be seen in the examples (Figure 15, Figure 16), the clusters we characterized in the previous sections (δ dominance vs. β dominance) are the meaningful features for the classification.

4.8 Time series prediction

We ran an encoder-decoder LSTM network (A.8). Some prediction seemed to be very accurate (Figure 17, Figure 18).

We used R^2 score (3.7) and MSE to investigate how good is the model. Unfortunately, the histograms of the R^2 score (Figure 20) and the MSE (Figure 19) revealed poor results.

As for the R^2 score, the majority of the data has a negative score for both sleep and wakefulness states, which means that for most inputs, the prediction of the average is better than the prediction of the model. Despite that, interestingly, the model learned the sleep state better than the waking state. This is consistent with our previous results, as it shows in the spectrograms (Figure 1, Figure 2), the sleep state has more distinct patterns, compared to the waking state.

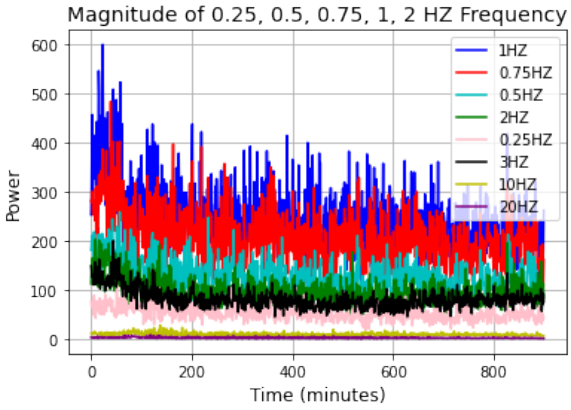

Figure 14: Power of several frequencies throughout the recording.

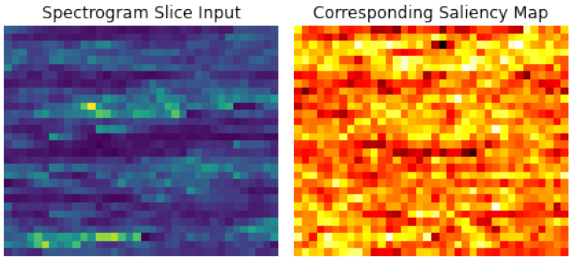

Figure 15: Example of an input of the network from **sleep state** and its' corresponding saliency map

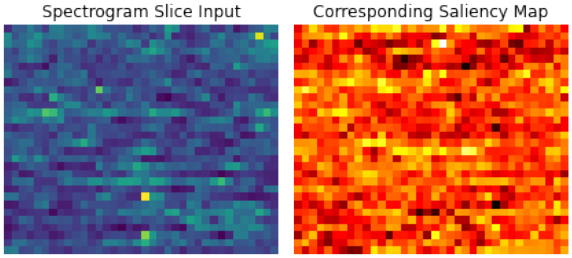

Figure 16: Example of an input to the network from **wakefulness state** and its' corresponding saliency map

5 Acknowledgments

This project was a challenge for us. We had to face academic papers in significantly unfamiliar fields, search online for unique solutions for our problems, implement the code according to the project needs and overcome some major setbacks. We would like to thank Mark shein-Idelson, Nitzan Albeck, Amir Bar and Amir globergson for the guidance and for the opportunity to learn.

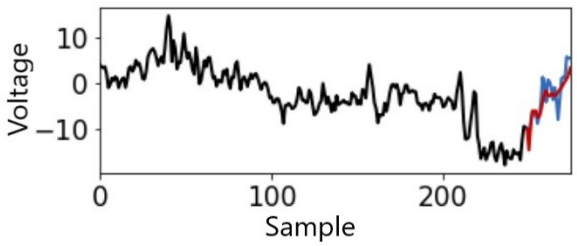

Figure 17: Example of successful prediction (from test data) using record of **sleep state**. The black color is the input, the blue is the target and the red is the prediction.

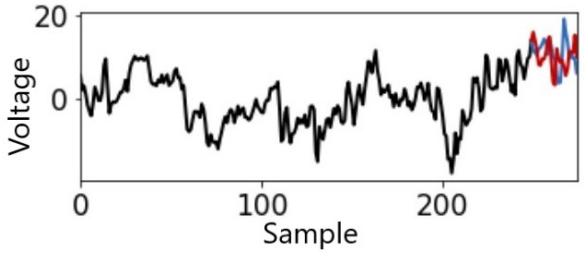

Figure 18: Example of successful prediction (from test data) using record of **wakefulness state**. The black color is the input, the blue is the target and the red is the prediction.

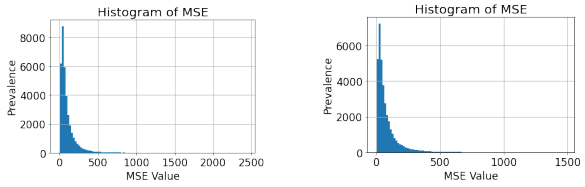

Figure 19: The left histogram is the **sleep state** MSE for the test inputs, the right histogram is the **wakefulness state** MSE for the test inputs.

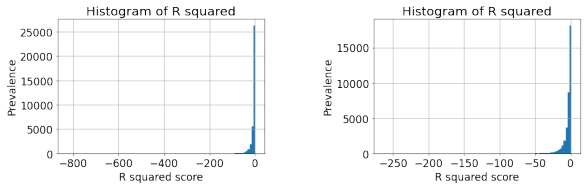

Figure 20: The left histogram is the **sleep state** R^2 score for the test inputs, the right histogram is the **wakefulness state** R^2 score for the test inputs.

All relevant code can be found in:

`https://github.com/OfirShechter/ML_Neuroscience_Project`

6 References

References

[] *Coefficient of determination*. URL: https://en.wikipedia.org/wiki/Coefficient_of_determination.

[AM08] F Ayala-Guerrero and G Mexicano. "Sleep and wakefulness in the green iguanid lizard (Iguana iguana)". In: *Comparative Biochemistry and Physiology Part A: Molecular & Integrative Physiology* 151.3 (2008), pp. 305–312.

[GD98] Matt W Gardner and SR Dorling. "Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences". In: *Atmospheric environment* 32.14-15 (1998), pp. 2627–2636.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[Kha] Irfan Alghani Khalid. *Saliency Map for Visualizing Deep Learning Model Using PyTorch*. URL: https://towardsdatascience.com/saliency-map-using-pytorch-68270fe45e80.

[LH16] Ilya Loshchilov and Frank Hutter. "Sgdr: Stochastic gradient descent with warm restarts". In: *arXiv preprint arXiv:1608.03983* (2016).

[McG+15] Matthew J McGinley et al. "Waking state: rapid variations modulate neural and behavioral responses". In: *Neuron* 87.6 (2015), pp. 1143–1161.

[Ola] Christopher Olah. *Understanding LSTM Networks*. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[Ras] Aditya Rastogi. *Visualizing Neural Networks using Saliency Maps in PyTorch*. URL: https://medium.datadriveninvestor.com/visualizing-neural-networks-using-saliency-maps-in-pytorch-289d8e244ab4.

[Row] Walker Rowe. *Mean Square Error R2 Score Clearly Explained*. URL: https://www.bmc.com/blogs/mean-squared-

error-r2-and-variance-in-regression-analysis/.

[Sal] Rodolfo Saldanha. *Stock Price Prediction with PyTorch*. URL: https://medium.com/swlh/stock-price-prediction-with-pytorch-37f52ae84632.

[SB+19] Shweta Sharma, Neha Batra, et al. "Comparative Study of Single Linkage, Complete Linkage, and Ward Method of Agglomerative Clustering". In: *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*. IEEE. 2019, pp. 568–573.

[She+16] Mark Shein-Idelson et al. "Slow waves, sharp waves, ripples, and REM in sleeping dragons". In: *Science* 352.6285 (2016), pp. 590–595.

[SO78] Robert R Sokal and Neal L Oden. "Spatial autocorrelation in biology: 1. Methodology". In: *Biological journal of the Linnean Society* 10.2 (1978), pp. 199–228.

[Wel67] Peter Welch. "The use of fast Fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms". In: *IEEE Transactions on audio and electroacoustics* 15.2 (1967), pp. 70–73.

A Details about the system and setting

This is a data-science project, we had some interesting findings and we pass forward to the hosting lab the finding, the classifier, and easy-to-use interactive code for further investigation. The following describes the details of our work.

A.1 Generation of the Spectrogram

The following has details of the configuration used to generate the spectrograms:

- **Low frequencies spectrogram:** Voltage traces were binned to 10s. The average normalized power spectrum (spectrum in each bin divided by the average over the entire dataset) for each bin was calculated using the Welch method (0.1s windows, 50% overlap)
- **Higher frequencies spectrogram:** Voltage traces were binned to 1s. The average normalized power spectrum (spectrum in each bin divided by the average over the entire dataset) for each bin was calculated using the Welch method (0.1s windows, 50% overlap).

A.2 Generation of the normalized spectrogram

Once the spectogram is generated (A.1), for each bin and frequency, the power was divided by the average across all bins power of that current frequency.

A.3 Generation of the correlation matrix

The correlation matrix was evaluated using Pearson correlation between all bin pairs.

A.4 Average power as function of frequency

Once the normalized spectrogram is created (A.2), and the data is divided into clusters, for each cluster, the average power of each frequency across all bins in that cluster was calculated.

A.5 Autocorrelation

The autocorrelation analysis for wakefulness state was performed on time windows of 10 seconds which are 200,000 samples, with shifts between 1 and 10,0000 samples (5 seconds), with steps of 10 samples between the different shifts.

A.6 Classify states

The input for the layer was 36 bins from the spectrogram, due to the lack of more data, there were significant dependencies between the samples (each sample is a shift of one bin from the previous sample). But it is important to emphasize that there were **no dependencies** between the train and the test data. The network configuration is as follow:

- **Hyperparameters:**
  - Learning rate: 0.1
  - Number of epochs: 10
  - Batch size: 64
- **Architecture:**
  - Input layer: gets the numbers of features as input (1080- the size of the picture as a vector 30*36) and returns a vector of size 128.

- Hidden layer: gets the output of the input layer as input (vector of size 128) and returns a vector of size 256.
- Output layer: gets the output of the hidden layer as input (vector of size 256) and returns a vector of 2 - the number of classes.

A.7 Saliency Map

For the trained network, one picture was passed as an input, and backpropagation was used in order to get the derivative of the output (the output is the maximum score - the score of the predicted label) with respect to the input image. Then, the gradient values of the input image are the saliency values. Finally, the output of the saliency map is put next to the input image to create the "heat map" of the important origins.

A.8 Time series prediction

In this approch, we implement encoder-decoder LSTM, based on the following repository: https://github.com/lkulowski/LSTM_encoder_decodert with our needed modification.

The input to the network was generated as follows:

- **Pre-processing:** The input was down-sampled from 20000Hz to 250Hz for easier model construction and training (smaller input and output gives the same time window).
- **Input-Output size:**
  - **Input window:** the number of samples that according to it the prediction performed- 250 samples.
  - **Output window:** the number of samples that the network should predict- 25 samples.
  - **Overlap:** space between the start of the windows- 5 samples.

The network configuration is as follow:

- **Hyperparameters:**
  - Learning rate: 0.01 using cosine annealing (Loshchilov and Hutter 2016)
  - Teacher forcing ratio: 0.7
  - Number of epochs: 50
  - Batch size: 5000
- **Architecture:**
  - **Encoder:** The encoder gets a sequence of 250 (i.e input window) as an input. The encoder has one layer of LSTM (3.6), the output is an hidden (size 30).
  - **Decoder:** The decoder has one LSTM layer and one linear layer. It gets as an input hidden (size 30) and an indication of the last sample (size 1). This input is passed to the LSTM layer which outputs hidden of size 30 (ideally contains the history) and lstm-output of size 30. The lstm-output is the input to the linear layer and its' output ideally represents the next sample value. The decoder returns a hidden (size 30) and an output (size 1).
  - **Network:** The network itself uses the encoder and the decoder as follows: pass the sequence (i.e input window) to the encoder, then run the decoder that iterates 25 times (i.e output window). For the first iteration, the input to the decoder is the encoder-hidden and the last sample. For other iterations, the input to the decoder depends on the previous step. We used **mixed teacher forcing**. According to the teacher forcing ratio, the input is the previous decoder hidden state and either the previous target sample (with teacher forcing) or the previous decoder output node (without teacher forcing).