CS598 Project Draft: Automatic Sleep Stage Classification

Pradeep Dwivedi
University of Illinois Urbana-Champaign
901 West Illinois Street, Urbana, IL 61801
pkd3@illinois.edu

Michael Renteria
University of Illinois Urbana-Champaign
901 West Illinois Street, Urbana, IL 61801
mrente5@illinois.edu

Jason Kolter
University of Illinois Urbana-Champaign
901 West Illinois Street, Urbana, IL 61801
jkolter2@illinois.edu

Zichun Xu
University of Illinois Urbana-Champaign
901 West Illinois Street, Urbana, IL 61801
zichunx2@illinois.edu

ABSTRACT

Sleep stage classification is important for sleep disorder diagnosis. Emerging deep learning studies in sleep stage classification have demonstrated state-of-the-art performance and can greatly reduce manual effort from human experts. In this draft we present and evaluate our preliminary result from several machine learning models that automatically classify sleep stages. They include convolutional neural network (CNN) model, recurrent neural network (RNN) model, and selected non-deep learning models.

1. INTRODUCTION

Sleep disorders are widespread and can have a large impact on a person's health and quality of life [1]. Sleep stages are critical for diagnosing sleep disorders [2]. There are four stages of sleep that various studies analyze in order to find non regular sleeping patterns, and furthermore pinpoint various sleep related disorders. According to American Academy of Sleep Medicine (AASM), the four stages [3] are broken down into two categories: rapid eye movement (REM) sleep and non-REM sleep. Non-REM sleep is composed of stages N1, N2, and N3. Previously N3 was further divided into N3 and N4. REM sleep is a stage and category by itself. A polysomnogram (PSG) is a comprehensive sleep test that records various sensor readings of the body during a subject's sleep. Since, it typically takes 90 minutes to hit the fourth stage of sleep in a normal patient [4], it is needless to say that you are often working with many hours of PSG data on each specific patient. The hours of data that you are working with is then multiplied by each of the various sensors you choose to use to help you predict the specific stages of sleep.

PSG recordings are divided into 30-second intervals called epochs, which serve as a basic time period for data analysis. A human expert then assigns a sleep stage to each of the epochs. It takes hours to label a PSG result for a single patient's one night sleep. This is time and labor intensive because sleeping stages are scored based on a collection of rules given by AASM [3, 5]. They include (but are not limited to) signal intensity, patterns, duration, sequence or co-occurrence of signals and so on.

Deep learning methods have shown great promise in generating sleep stage labels [6-8] and some models have achieved comparable accuracy to human experts. It can greatly aid the process of diagnosing sleep disorder.

Convolutional neural network, usually seen in image recognition, has also shown great success in signal processing like ECG classification, arrhythmia detection and motor-fault detection [9]. It has many advantages. For example, it can be applied to raw signal data without pre-processing. It can learn to extract useful spatial and time-invariant features in the signal without any domain knowledge [8] or hand-crafted features. Previous sleep stage studies [6,8] that built purely on CNN usually start with 1D convolutional layer followed by 1 to 5 consecutive convolutional layers (with ReLU and max pooling), fully connected layers and lastly a 5-class softmax to generate the final sleep stage prediction. Max pooling layers are used to prevent overfitting and to reduce computational cost.

Recurrent neural networks are powerful for sequential data. In the AASM scoring rules we can see that the scoring for a particular epoch can sometimes depend on the label for its previous epoch or its following epoch [8]. Therefore RNN has shown to be well-suited for discovering temporal dependencies in PSG data [6,7,10].

In this draft, we experimented with a basic CNN and RNN model, as well as some non-deep learning algorithms as baseline.

2. APPROACH AND METRICS

2.1 Dataset

For this project, we used sleep recording data from the Sleep-EDF Expanded database publicly available at PhysioNet[11, 12].

This database contains 197 samples of whole-night PSG recordings suitable for general purpose sleep analysis. All records contain readings from EEG, EOG, chin EMG and event markers with some samples containing respiration and temperature data. The recordings are available in EDF/EDF+[13] format which is the de facto standard for PSG data. The data from these samples will be used to build the features that will be used to train our scoring models. Along with the sample data, there also is a set of expert-scored hypnograms available that contain ground truth annotations corresponding to each 30-second epoch in the PSG recordings. These patterns are W(Wake), R(REM), 1(N1), 2(N2), 3(N3), 4(N4), M (Movement time) and ? (not scored) [12].

These recordings were obtained from two separate studies [12, 14]. The first contains 153 healthy Caucasian subjects not using sleep related medication. Each of these subjects has 2 PSG recordings of approximately 20 hours each. The EOG and EEG samples were taken at 100Hz, while the event markers were sampled at 1Hz. The second study contains data from 22 subjects who suffer from sleep issues. There are again 2 samples for each subject however this group contains one each of medicated and placebo dosed sleep assistance.

This dataset gives a good representative sample of different sleep patterns and a mix of healthy and unhealthy subjects that we can

use for robustly training the models as discussed in the model implementation sections below.

2.2 Input Data and Feature Selection

Typically EDF files containing PSG and Hypnogram data are manipulated using dedicated software to display and allow human annotation of the sleep recordings (such as Polyman, pictured in Figure 1). However, because the PSG/Hypnogram recordings are stored in the custom binary format, it is necessary to extract the readings into format consumable by the machine learning libraries used for the project. The software allows for exporting the raw sensor readings into a delimited text file format (tab-separated, tsv) and also, separately, can export the scoring data into a delimited format(comma-separated, csv).

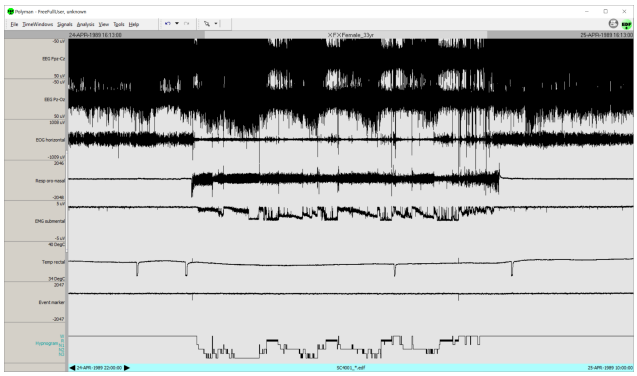

Figure 1: Example of a typical PSG/Annotation display

While this created data usable programmatically, there are two significant issues that need to be addressed. First, the size of the data is very large. From this dataset, the export of a single subject's raw sensor readings sampled at 100Hz resulted in a file approximately 400MB on average. In order to get a large enough pool of data to properly train any deep learning models, we estimate requiring at least twenty subject's worth of readings (two samples per subject), which corresponds to a minimum input data size of 16GB of raw data. The second issue is that each reading is individually captured, and there is a significant amount of pre-processing required to then group each 1/100 of a second of raw sensor reading into 30 seconds epochs, and combine those with the human annotations to generate the feature vectors suitable for our model.

Because of the size and complexity of the work required to perform the necessary aggregations on the raw source data, this preprocessing step would be very challenging to do on a typical consumer-grade laptop in a responsive and timely manner. A further goal of this project is to create a repeatable methodology to quickly create usable data in a generalized model from EDF files to support experimenting with different types of models and further studies. As such, the industry standard *Apache Spark* platform was chosen as the mechanism with which the bulk of the data processing would be done on the raw data exported from the EDF files. The implementation that was built was specifically designed to handle any number of input files and volume of data without any additional manual processing.

The exported sensor readings and hypnogram scores were uploaded to *Azure Data Lake Storage* and were then processed using *Azure Databricks*. The score files were used to generate 30 second epochs of time and a corresponding sleep stage label per subject/sample. The raw sensor readings that were collected during each epoch were then aggregated into a vector per epoch, resulting in a data model as seen in Figure 2.

	subject	annotation	EEG_Fpz-Cz_uV	EEG_Pz-Oz_uV	EOG_horizontal_uV
0	SC4001	W	[-20.4893772893773]	[-8.033699633699964]	[161.8835164483517]
1	SC4001	W	[-5.860805860680586, -27.9912087912088, -5.9545...	[-2.27545787545788, -6.210256410256642, -5.2505...	[127.38778998779, 170.26105006105, 120.9814407...
2	SC4001	W	[8.86153846153846, 10.643223443223234, 9.0490842...	[2.331135531135552, 3.098901098901109, 1.0835164...	[-27.8429792429791, -57.90354090354098, -40.655...
3	SC4001	W	[0.23443223443234, 4.45421245421245, 1.078388...	[0.21978021978213, -1.60366300366301, 3.67472...	[-8.62393162393152, 21.4366300366301, 3.203174...
4	SC4001	W	[3.32893772893773, 17.582417582417 6, 17.019780...	[1.65934065934065, 0.315750915750909, -3.81098...	[4.68156288156299, 10.595115995115 61, 1.2319902...
...	...	...	...	...	...
2665	SC4172	R	[-8.75238095238096, -2.58412698412699, 4.13650...	[-4.75824175824177, -3.45909645909647, -5.4078...	[-28.3103785103785, -25.71111111111111, -24.844...
2666	SC4172	R	[-21.8253968253968, -21.1809523809524, -20.720...	[2.47985347985347, 1.923076923076 91, 0.6239316...	[-27.0107448107448, -30.90964590964659, -31.342...
2667	SC4172	R	[-2.95238095238096, -3.2285714285714 4, -2.1238...	[0.995115995115982, 2.57264957264956, -8.815873 01587304, -11.41514044151404, -10.981...
2668	SC4172	R	[0.730158730158 72, -6.26666666666668, -6.91111...	[-0.02564102564103 89, -13.48107448107 45, 25.02...	[-9.24908424908428, -14.88083028083303, 0.71477...
2669	SC4172	R	[-18.8793650793651, -15.4730158730159, -17.038...	[-13.2026862026862, -13.29548229548923, -13.017...	[-4.48376068376071, -13.58119658119966, -21.812...

Figure 2: Sample of processed output data

Due to size of the data, and a desire to maintain the vector format for each epoch, standard delimited text file was not a good option, so the processed dataset was stored in Parquet file format. *Apache Parquet* is an open-source column oriented file format that very efficiently serializes large semi-structured data like in our model. It also works very well with Spark as the Spark output writer is heavily optimized to use Parquet format. Typically, processed datasets are larger than the raw data due to using different data structures or augmenting with additional data as in our case, however, our 16GB raw input dataset resulted into a processed dataset of only 934 MB.

Using a Spark cluster of five nodes of modest size (8 cores, 32GB ram), the total time to read, process and write the output was three minutes, twenty seconds. Execution time should scale up or down approximately linearly based on the number of nodes and size of the input data. Parquet is easily usable by standard data science tools, and is much more efficiently read into memory than textual files, and is especially appropriate for this case where we are initially only using a single output channel. Parquet is able to read only the columns needed while maintaining the full dataset. Furthur customizations to this general data model would then be very easy to apply to fine-tune the input data for specific models.

Specifically, the preprocessed dataset output from the Spark cluster had annotations for the sleep stages corresponding to awake, rem, N1, N2, N3, N4, eye movement and some unscored epochs. We further refined this input in the following ways:

- Combined N3 & N4 stages to N3.
- Removed the epochs related to eye movement stage.
- Removed the unscored epochs.
- Selected data of only one channel - EEG_Fpz-Cz_uV - for our model building.
- Removed any incomplete epochs which don't have data for full 30 seconds.
- Annotated the string values of sleep stages to numeric values. The epochs corresponding to sleep stages of R, N1, N2, N3 & W became 0, 1, 2, 3 & 4, respectively.

Sleep stages N3 and N4 were combined into N3 to align with new AASM scoring rules. We removed M (Movement time) and ? (not scored) epochs since they don't belong to any of the sleep stages defined by AASM. Raw EEG data is used without any further

signal transformations. We removed epochs at the beginning and the end of each recording where they only had one reading.

The output dataset after implementing the above steps had a total of 66538 epochs, with 7993, 2869, 18240, 5497, 31939 epochs corresponding to sleep stages R, N1, N2, N3 & W.

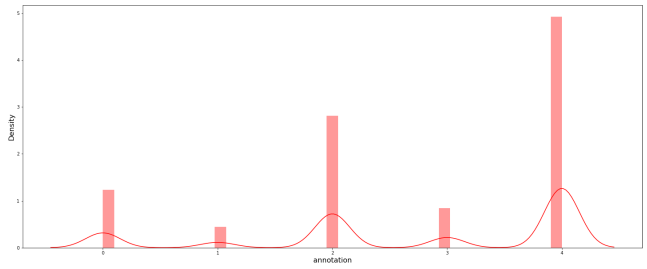

Figure 3: Unbalanced Sleep Stages in Input Dataset

As shown in Figure 3, the classes in the dataset are not balanced. Stage W and N2 have significantly more epochs than the others. This might cause bias in the model output towards the majority classes.

To address the above issue, we removed the every other epoch of the awake stage (i.e., 'W') from the input dataset. There were 31939 epochs of sleep stage W (i.e. 4). We removed 15970 W stage epochs. Since the awake stage usually is longer, removing some epochs from that shouldn't distort the quality of our dataset. Also, it will result in significantly balanced classes for the input dataset for sleep stages, as shown in Figure 4 below.

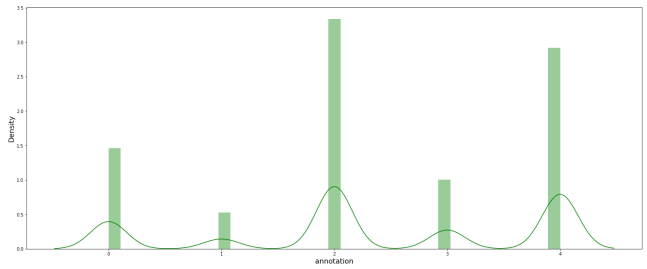

Figure 4: Balanced Sleep Stages in Input Dataset

Apart from the data reduction above, we also experimented with class-balanced random sampling so that each class has the same number of epochs. Since N1 has the least number of epochs, 2869, we resampled all the other classes to have 2869 epochs.

The sleep signals corresponding to our selected channel appear pretty much normally distributed, as shown in Figure 5 below. This is a desirable state of the input data to reduce the possibility of bias and variance in the resultant data models.

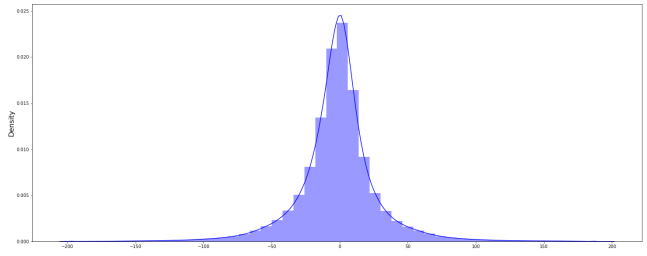

Figure 5: Bell Curve Shaped Sleep Signals in Input Dataset

The sampled dataset is then randomly split into 70-30 for test and validation purposes.

2.3 Model Architecture

2.3.1 Non-Deep-Learning Architectures

We implemented Logistic Regression (LR), Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA) for the input dataset. All 3000 features for a 30 seconds epoch for single channel data ware used. We implemented standard LR, LDA and QDA algorithms so that we'll have enough comparison statistics to use against deep learning models.

2.3.2 CNN Architecture

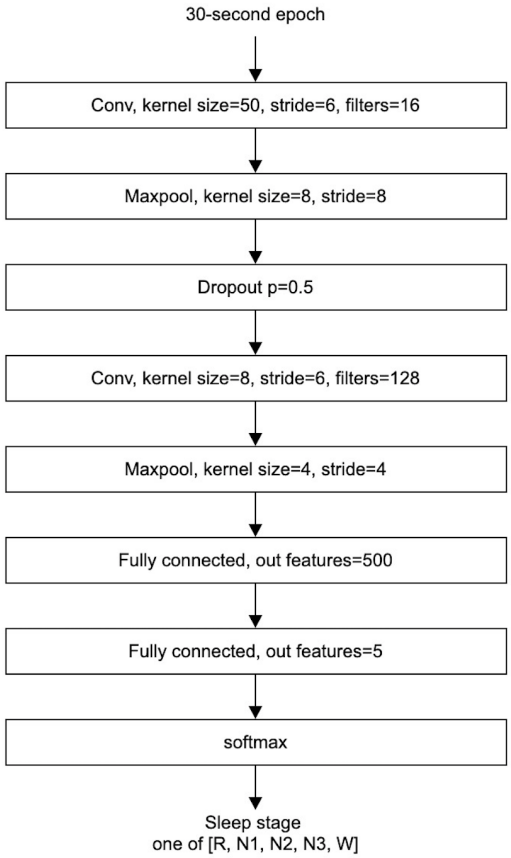

Figure 6: CNN architecture

As shown in Figure 6 above, this model contains 2 convolutional layers, 2 max-pooling layers and 2 fully connected layers. Each convolutional layer performs a 1D-convolution, batch normalization, and rectified linear unit (ReLU) activation. ReLU activation is also used after the first fully connected layer. The last layer is a softmax to output the most likely sleep stage. In the first layer of CNN, we used a relatively small kernel size because smaller filter is better at capturing temporal patterns according to [10]

The model is trained with Adam optimizer with a learning rate of 0.001. We use cross entropy loss as the loss function.

2.3.3 RNN Architecture

RNN and more specifically Long short-term memory (LSTM) is a popular architecture among sleep stage classification papers. LSTM is popular because of its use case to detect patterns in

sequences that can be used as classification. Previous research [6,7,10] focused on inputting a sequence of epochs and predicting a sequence of labels. However we took a different approach and experimented using RNN within the 30-second epoch to predict sleep stage, since it can be considered as a time series within the epoch as well. The EEG data is sampled at 100Hz, so every epoch contains 3000 readings, which is our sequence length. We used LSTM and experimented with different numbers of hidden states. Then we took the last hidden state and put it through a fully connected layer and then a softmax layer to output the final sleep stage.

The model is trained with Adam optimizer with a learning rate of 0.001. We use cross entropy loss as the loss function.

2.4 Model Evaluation

Classification performance of our models is evaluated with per-class and average precision, recall, F1 score and Cohen's Kappa score. Cohen's Kappa score measures inter-rater reliability. It allows us to tell to what extent the labels produced by our model are in agreement with that from a human expert. The overall score from eight European centers is 0.76 [5], and we will be comparing ours with this human level performance.

In our input dataset, N2 is the majority class and corresponds to 36% of the data. Therefore, 36% is our baseline and for any algorithm to demonstrate learning, it should surpass this baseline of 36%.

3. RESULTS

We've summarized below the accuracy score of all the algorithms we've tried so far:

	Logistic Regression	LDA	QDA	CNN	RNN
Accuracy Score	29%	37%	60%	77%	51%

Table 1: Accuracy scores from different models

Now, let's examine the results of some of the most successful models, below:

3.1 LDA

The confusion matrix for LDA doesn't look ideal. It has a lot of big values on off diagonal cells. It will, however, serve as an effective non-deep learning model to compare against deep learning models.

	R	N1	N2	N3	W
R	27	8	1517	23	440
N1	10	6	492	7	156
N2	94	20	3270	100	1151
N3	51	6	884	64	376
W	280	218	1955	222	1265

Table 2: Confusion matrix from validation set for LDA model

3.2 CNN

	Predicted					per class metrics		
	R	N1	N2	N3	W	PR	RE	F1
R	**529**	231	118	2	22	0.77	0.59	0.66
N1	117	**548**	133	9	37	0.62	0.65	0.63
N2	31	42	**700**	55	13	0.69	0.83	0.76
N3	0	1	46	**827**	6	0.91	0.94	0.93
W	10	63	16	13	**735**	0.9	0.88	0.89

Table 3: Confusion matrix from validation set, and per class precision, recall, and F1 score

Table 3 shows the confusion matrix from running the model on the validation test. Data is randomly resampled so that every class has 2869 epochs. Therefore the confusion matrix doesn't need to be normalized. The model has Cohen's Kappa score of 0.712, accuracy score of 0.770, and average F1 score of 0.770.

It performs the best on classifying N3 and W stages, with 0.93 and 0.89 per class F1 score, respectively. The most misclassified class is N1. The largest off diagonal numbers come from pairs R-N1, N1-N2, followed by R-N2, N1-R. The remaining pairs have a very small chance of being misclassified to each other.

Model trained on data with only W stage downsampling does not show valid results. Although the average accuracy is 0.78, the per class metrics revealed that the model didn't output any labels in the N1 and N3 class.

3.3 RNN

Our test RNN model showed an accuracy score of 0.51 when trained without any class re-balancing, but this data is not representative since the model didn't predict any label of R, N1 and N3. With class-balanced random sampling (each class has 2869 epochs), the accuracy dropped to 0.23, with an average F1 score of 0.19. Different humber of the hidden layer doesn't make a lot of difference in the final output.

4. DISCUSSION

4.1 Non-Deep-Learning Models

Since the input sleep signals have non-linear patterns, it was evident that nonlinear algorithms will result in better accuracy. As we've seen, QDA has the highest accuracy. Logistic regression, which is based on linear decision boundary, performs worse than the baseline accuracy. QDA improves almost 100% over LDA, in terms of accuracy. Even with the improved performance in QDA, some sleep stage classes are under-represented in the confusion matrix. We hope to get better F1 scores for these classes, when using deep-learning algorithms.

4.2 CNN Model Performance

As shown in the results section, the CNN model was not able to achieve a satisfying result after we downsampled half of the epochs in the 'W' stage. The remaining class imbalance still resulted in the model favoring strongly towards the most represented class. In the light of this observation we did class-balanced random sampling.

Using the data for 20 subjects, our model achieved accuracy of 0.770 and average F1 of 0.770. Previous research [8,15] results

reported accuracy in the range of 0.74 to 0.78, F1 of 0.79 and Cohen's Kappa of 0.7. While these results are not directly comparable due to difference in dataset, or possible hand-engineered features, it showed that our model performance is in a good and reasonable range.

While the model on its own has achieved relatively good results, this class-balanced random sampling approach however, does not support any future extension of the CNN model to be connected with an RNN model, since RNN relies on input of consecutive epochs, not randomly sampled epochs. More research is needed to find a better way to account for the class imbalance of data.

4.3 RNN Model Performance

RNN model failed to achieve a satisfying result. This is likely due to the fact that our sequence length of 3000 is too long and LSTM failed to retain much useful information at the very end. We could potentially divide each epoch into smaller sequences and later combine them together, but given that CNN already showed good results in identifying important signal patterns within each epoch, we decide that we will not pursue further into this RNN-within-epoch approach. We plan to however, further explore the RNN with other combinations of the parameters and a relatively class balanced input dataset.

5. OPTIMIZATION AND CONCLUSION

The recent advances in machine learning combined with statistical learning can help distinguish the sleep stages successfully and effortlessly. The automated sleep scoring mechanism proposed in this paper can reduce the burden on clinicians, contribute to the in-home device implementation of a sleep monitoring system and benefit the sleep research community.

Our work so far is based on PSG data from 20 subjects. To make our model robust against variability among subjects, we could use data from more subjects from the Sleep-EDF database to train, validate and test.

In this draft we only used the Fpz-Cz channel from EEG data. Since sleep stage rules also depend on other channel signals, we could add in EEG's PZ-Oz channel and EOG channel, all sampled at 100Hz, to see if it will find other important features and thus improving performance.

At this point, RNN is largely a work in progress. Many papers [6, 7, 10] were able to achieve very high accuracy with a combination of CNN and LSTM. The current plan is to first get a stable RNN, and then explore to see if a cross implementation with CNN can improve accuracy. In order to get the RNN stable, we are going to look into: different methods of maintaining or not maintaining the hidden state, different loss functions, different activation functions, and finally different ways of feeding the data into the RNN model.

For the CNN component, we could also experiment with using a large filter, as [10] shows, a large filter is better at capturing frequency information.

In conclusion, we experimented with LR, LDA, QDA, CNN and RNN models on PSG data from 20 subjects. CNN has shown promising results and we plan to improve upon it and later introduce a RNN component to build our final model.

6. REFERENCES

[1] Raymond C. Rosen, Mark Rosekind, Craig Rosevear, Wesley E. Cole, William C. Dement, Physician Education in Sleep and Sleep Disorders: A National Survey of U.S. Medical Schools, Sleep, Volume 16, Issue 3, May 1993, Pages 249–254, https://doi.org/10.1093/sleep/16.3.249

[2] Abad, V. C., & Guilleminault, C. (2003). Diagnosis and treatment of sleep disorders: a brief review for clinicians. Dialogues in clinical neuroscience, 5(4), 371–388.

[3] Iber, C., S. Ancoli-Israel, A. Chesson, and S. F. Quan. The AASM Manual for the Scoring of Sleep and Associated Events: Rules, Terminology and Technical Specifications. Westchester, IL:American Academy of Sleep Medicine, 2007.

[4] Miladinović Đ, Muheim C, Bauer S, Spinnler A, Noain D, Bandarabadi M, et al. (2019) SPINDLE: End-to-end learning from EEG/EMG to extrapolate animal sleep scoring across experimental settings, labs and species. PLoS Comput Biol 15(4):e1006968. https://doi.org/10.1371/journal.pcbi.1006968

[5] Danker-Hopfe H, et al. Interrater reliability for sleep scoring according to the Rechtschaffen & Kales and the new AASM standard. J Sleep Res. 2009;18(1):74–84.

[6] S. Biswal, J. Kulas, H. Sun, B. Goparaju, M. Brandon Westover, M. T. Bianchi, and J. Sun. SLEEPNET: Automated sleep staging system via deep learning. 26 July 2017.

[7] L. Zhang, D. Fabbri, R. Upender, and D. Kent. Automated sleep stage scoring of the sleep heart health study using deep neural networks. 2019.

[8] O. Tsinalis, P. M. Matthews, Y. Guo, and S. Zafeiriou. Automatic sleep stage scoring with single-channel eeg using convolutional neural networks. arXiv preprint arXiv:1610.01683, 2016.

[9] Kiranyaz, S. (1), Ince, T. (2), Abdeljaber, O. (3), Avci, O. (3), & Gabbouj, M. (4). (n.d.). 1-D Convolutional Neural Networks for Signal Processing Applications. ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, 2019–May, 8360–8364.

[10] A. Supratak, H. Dong, C. Wu and Y. Guo, "DeepSleepNet: A Model for Automatic Sleep Stage Scoring Based on Raw Single-Channel EEG," in IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 25, no. 11, pp. 1998-2008, Nov. 2017, doi: 10.1109/TNSRE.2017.2721116.

[11] Goldberger, A., et al. "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. Circulation [Online]. 101 (23), pp. e215–e220." (2000).

[12] B Kemp, AH Zwinderman, B Tuk, HAC Kamphuisen, JJL Oberyé. Analysis of a sleep-dependent neuronal feedback loop: the slow-wave micro-continuity of the EEG. IEEE-BME 47(9):1185–1194 (2000)

[13] B Kemp, J Olivan. European data format 'plus' (EDF+), an EDF alike standard format for the exchange of physiological data. Clinical Neurophysiology 114:1755–1761 (2002).

[14] MS Mourtazaev, B Kemp, AH Zwinderman, HAC Kamphuisen. Age and gender affect different characteristics

of slow waves in the sleep EEG. Sleep 18(7):557–564 (1995).

[15] I. Al-Hussaini, C. Xiao, M. B. Westover, and J. Sun. Sleeper: inter- pretable sleep staging via prototypes from expert rules. In Machine Learning for Healthcare Conference, pages 721–739, 2019.