Sleep Stage Classification

Project Final Report

Jean QUENTIN
CentraleSupélec
jean.quentin@student.ecp.fr

Elisabeth DAO
CentraleSupélec
elisabeth.dao@student-cs.fr

Timothée RIO
CentraleSupélec
timothee.rio@student.ecp.fr

Abstract

Diagnosis of sleep disorders, which can cause severe health issues, are usually done using manual annotations of raw polysomnography signals. This task is tedious and error-prone, and the recent advances in machine learning and deep learning have motivated researchers to explore their applications in sleep analysis. In this context, Dreem, a company positioned on at-home sleep diagnosis, had launch a data challenge aimed at automatically and accurately detecting the different sleep stages, which is crucial for analyzing sleep disorders. The data that is provided for the challenge comes from the output of a headband developed by the company. In the present work, we propose a review of the main model architectures used for sleep stage classification and more broadly for multivariate timeseries analysis, and apply them to our relatively small dataset. We show that a rather simple CNN-based model performs better on the Dreem Challenge Dataset than complex LSTM and Attention-based ones, which are dragged down by the lack of training data. All our implementations are available on our GitHub repository ***timrio/deep_learning_project***.

1. Introduction

1.1. Motivation

Sleep is crucial for every individual: it allows the body to work normally and to perform our daily activities. Ensuring a restful and healthy sleep is a condition for well-being. Sleep disorders however can cause many issues and are referred as psychiatric illnesses. Thus, for those who suffer somnipathy, it has been vital to find solutions.

Sleep progresses in cycles: wake, light sleep, deep sleep and rem sleep, which are associated to different physiological functions. Understanding the different sleep stages and being able to monitor them are important steps to diagnose sleep disorders. Automatic detection of these different steps constitutes a large domain of research that until now has been conducted in hospitals or sleep labs.

Designing an efficient and reliable automated sleep staging solution could save clinicians countless hours of work and thus make sleep assessment and diagnosis more widely available. Accurate and cost effective monitoring of sleep not only has great medical value but could also enable individuals to self-assess and self-manage their sleep and could enable earlier diagnostic of incipient sleep conditions.

In this context, Dreem, a company that is positioned on At-Home Sleep Diagnosis, has developed a headband that can perform polysomnography, which consists in acquiring biological signals while a patient is asleep. This headband is able to measure physiological signals from the brain activity at home thanks to three kinds of sensors : electroencephalogram (EEG), pulse oximeter and accelerometer signals.

The company has proposed a Kaggle challenge [1] to perform sleep stage scoring thanks to 30 seconds epochs of biophysiological signals on headband data. Each 30 seconds time window is made of 8 signals: 5 electroencephalogram time series sampled at 50 Hz (1500 measures per time window) and 3 accelerometer series sampled at 10 Hz (300 measures per time window).

1.2. Problem Definition

The American Academy of Sleep Medicine (AASM) segments sleep in five stages [12, 2] : wake, N1, N2, N3 and R.

- W (Wakefulness): The wakefulness stage is characterized by high-amplitude muscle contractions and eye blinking.

- N1 (Non-REM 1): ”Dozing off” stage, it normally lasts only a couple minutes. Body and brain activities are starting to slow with periods of brief movements (twitches).
- N2 (Non-REM 2): Stage N2 is associated with a drop in body temperature, relaxed muscles and a slowed breathing and heart rate while the remaining eye movement stops. N2 stages ususally lasts 10 to 25 minutes and make up to half the total sleep time.
- N3 (Non-REM 3): Also known as ”deep sleep”, breathing, heart rate and muscles activity decrease even further. N1 stages last for about 20 to 40 minutes and occur mainly during the first half of the total sleep phase.
- R (REM): The ’rapid eye-movement’ stage is defined by much higher brain activity, nearly reaching wake levels, while all muscles except for eye and breathing muscles are paralyzed. REM stages can last from a few minutes to an hour and make up about a quarter of the total sleep time, mainly during the second half.

During sleep, a person will alternate between all the phases as shown in Figure 1, and the sequence of these phases as well as the time spent in each phase characterizes healthy and unhealthy sleep patterns.

Finally, the sleep stage classification problem can be formalized by stating that a multidimensional biological temporal signal $x = (x_1, x_2, ..., x_n)$ with $x_i = (eeg_1^i, egg_2^i, egg_3^i, egg_4^i, eeg_5^i, a_x^i, a_y^i, a_z^i)$. We try to determine the label sequence of sleep stages $(y_1, y_2, ..., y_n)$ with $y_i \in \{1, 2, 3, 4, 5\}$.

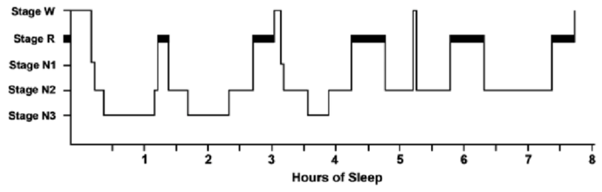


Figure 1: EEG extract taken from [6]

2. Related Work

In recent years, time series analysis has attracted the attention of many researchers [10]. In earlier research, machine learning methods are used to classify sleep stages but these methods rely generally on manually-engineered features that require a lot of prior knowledge. Therefore, with the recent development of deep learning techniques, researchers have started to explore the benefits of designing a completely automated sleep staging technology. Furthermore, given that the acquired biological signals are very classical time series, there is already a huge literature dealing with its analysis. In particular, the existence of increasingly large public data sets online (for example, PhysioNet [11] and the National Sleep Research Resource (NSRR) [24]) has enabled exploitation of deep learning, to create very powerful automated sleep staging algorithms that leverage the vast amount of training data that has been gathered in the past few years.

Nowadays, deep learning methods based on CNNs or RNNs are being commonly used for sleep staging. Hybrid models with CNNs followed by recurrent neural networks (RNN) have demonstrated state-of-the-art results that have reached the level of medical specialist, as does, for instance, Deep-SleepNet [20] which employs CNNs to extract time-invariant features and LSTMs to learn the time-dependant features.

Obayya and Abou-Chadi [15] leverage spectral analysis, wavelet transforms and fuzzy clustering based on fuzzy c-means algorithm (FCM) to perform classification, while [3] presents a novel method to identify sleep stages that is light enough to be implemented in an embedded hardware device. It uses manually designed feature vectors extracted from the time series signal, that are then plugged into a classifier.

3. Methodology

The key part of our work will focus on how to use raw data and manually designed pre-preprocessings, as well as novel deep learning models to create the best performing classification architecture on our dataset.

3.1. Data Processing

The EEG data collected by the headband are signals which represent the combined activity of multiple brain areas that might oscillate at different frequencies. A powerful tool for analyzing this oscillatory structure has been spectral analysis, a quantitative approach for describing a waveform signal in terms of its underlying oscillations (sinusoids) at different frequencies, by decomposing complex signals in mono-frequency sinusoidal waves. This is particularly useful in the analysis of EEG data, where the signal represents the combined activity of multiple networks of neurons throughout the brain that oscillate at different frequencies.

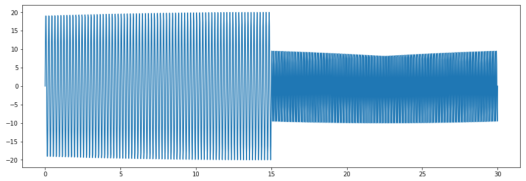
3.1.1 Fast-Fourier Transform

The simplest and most common method used for performing spectral estimation on EEG data is called the periodogram and can be obtained with an algorithm called the

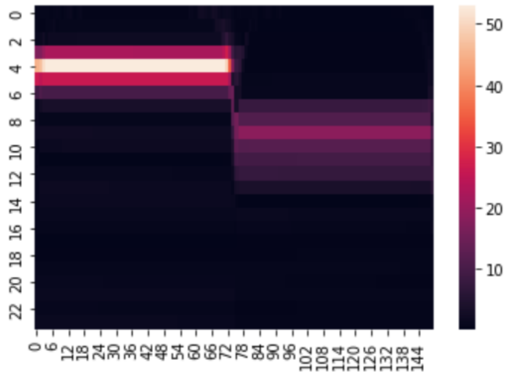
fast Fourier transform (FFT), which is a method that decomposes a time-domain signal into a series of pure sine waves of different wavelengths. However, methods such as the periodogram provide inaccurate (biased) and noisy (variable) estimates of the power spectrum and spectogram. Spectra estimated can be difficult to interpret, with noisy spectral peaks that may obscure important features for sleep EEG dynamics. A common approach to reducing variability in these estimattions is to average these spectra across time, as in Welch's method for instance, which is a time-average method. Although these methods produce high-resolution, low-variance spectra, the temporal resolution is greatly reduced due to the additional smoothing across time.
We have thus used and evalutated the performance of the regular FFT data in our work.

3.1.2 Multitaper Spectral Analysis

After reviewing the different ways to extract frequencies from a finite signal, we chose to try the multitaper spectral estimation approach [4], since it exhibited very good performance and produces clear, accurate, high-resolution spectral estimates, without having to average over frequency or time. It outperforms conventional spectral estimators, producing high-resolution EEG with significantly reduced biais and variance. [5] provides a deeper explanation of the multitaper technique that we applied. Figures 2 shows an example of frequencies represented with a multitaper.



(a) Original signal before the multitaper



(b) Multitaper output on the previous signal

Figure 2: Example of the multitaper approach

3.2. Models and algorithm

Given the limited amount of data we have, we decided to implement solutions of increasing complexity in order to ensure that we do not overfit our model.

Most of the architectures we have read about were adapted to single channel time series. However, the time series in our data set were made of 5 eeg channels. Thus, we slightly adapted to our problem the models we worked on. Our approach was to stack the features of the different channels (either raw time series, FFT signals or multitaper data) and to adapt the kernel size, stride, padding etc. of the models to the new format of the inputs. However we did not modify the overall architectures of the models.

Another adaptation that we implemented was for the simple CNN and the multitaper CNN architectures to duplicated the convolutional networks so as to dedicate the first one to the eeg data and the other one to the accelerometer data. We concatenated the outputs of those two subnetworks before feeding it to the dense layers. However, since we did not notice any major improvement in the performances of the models, we did not implement this approach for the more complex models. All the implementations mentioned hereafter are available on our project's GitHub repository **timrio/deep_learning_project**.

3.2.1 Simple CNN

To address this classification task we first implemented a deep learning approach based on a convolutional neural network [13, 10, 16]. These models leverage one-dimensional convolutions two create multivariate time series from a single time series in order to learn multiple discriminative features that will then be useful for classification. The raw data is passed through convolutional layers before subsequent classification into one of five sleep stages. This residual network comprises four block layers of four bottleneck residual blocks each. A single bottlenecked residual block contains three triplets of a batch normalization layer, a rectified linear unit (ReLU) activation layer, and a conv layer. This pre-activation configuration has shown benefits with regards to trainability and generalization compared to vanilla residual blocks. Before the bottleneck blocks, the tensor X was passed through an initial convolutional layer consisting of filters and then through a maximum pooling layer effectively reducing the time-resolution. The output tensor from the block layers was subsequently passed to a final batch normalization and ReLU activation layer, followed by a mean pooling layer to reduce the tensor. Finally, a fully connected layer with 5 output units corresponding to the sleep stages resulted in the following output tensor. Figure 3 illustrates is the type of architecture we have implemented based on [16] which demonstrated very impressive results.

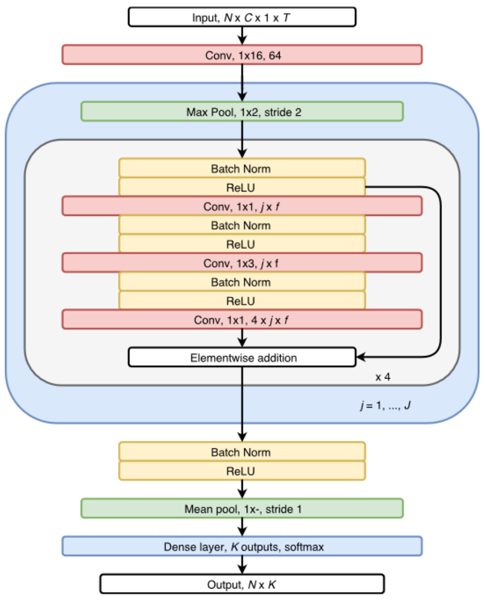


Figure 3: Example of a sleep staging CNN architecture [16]

Regarding the output of the results, it is done with a classical softmax activation as shown in equation 1.

$$p_{n,k} = \frac{exp(z_{n,k})}{\sum_k^K exp(z_{n,k})} \tag{1}$$

In this equation $p_{n,k}$ is the softmax activation of the output from the fully connected layer from the nth subject and the kth sleep stage. The predicted class for the nth subject can be calculated as done in equation 2.

$$y_n = \underset{k}{\operatorname{argmax}} p_{n,k} \tag{2}$$

This very common approach will be used in all our other implementations.

3.2.2 Advanced CNN

Then, we decided to implement an FFT based architecture. Since the FFT outputs a signal that has the same size as the input signal, it can easily be processed in parallel with the original raw data to extract both time-based and frequency-based features. Our implementation is based on the work of [23], available on Kaggle and which demonstrated an outstanding performance on a sensor time series data classification challenge. We reckoned that it would be very interesting to use classification approaches that have been successful on other types of time series classification but not yet experimented with time series classification.

As shown in Figure 4, the architecture we used leverages the efficiency of CNNs to extract features from both the original signals and their Fast-Fourier Transforms. The time and frequency features that we extracted from each signal are then concatenated and the two linear layers will extract the relevant information to classify our signal and output a sleep stage prediction.

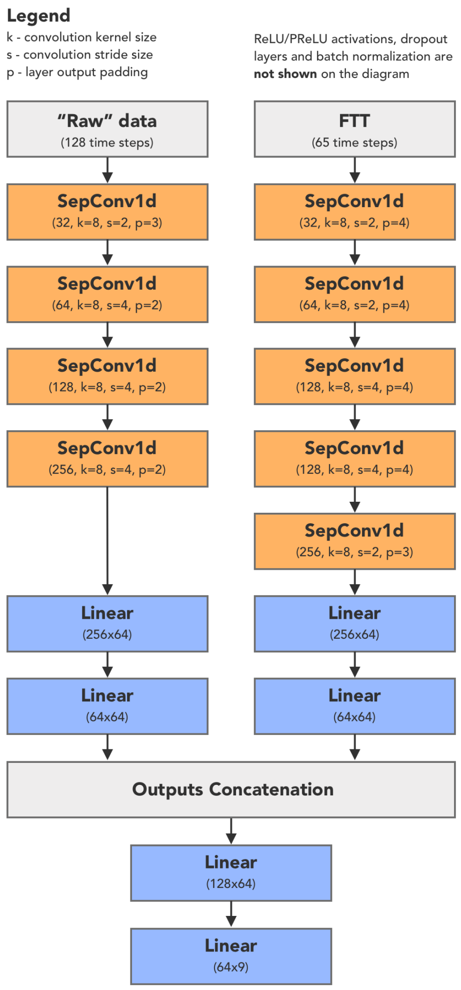

Legend
k - convolution kernel size
s - convolution stride size
p - layer output padding
ReLU/PReLU activations, dropout layers and batch normalization are not shown on the diagram
"Raw" data (128 time steps)
FTT (65 time steps)
SepConv1d (32, k=8, s=2, p=3)
SepConv1d (64, k=8, s=4, p=2)
SepConv1d (128, k=8, s=4, p=2)
SepConv1d (256, k=8, s=4, p=2)
SepConv1d (32, k=8, s=2, p=4)
SepConv1d (64, k=8, s=2, p=4)
SepConv1d (128, k=8, s=4, p=4)
SepConv1d (128, k=8, s=4, p=4)
SepConv1d (256, k=8, s=2, p=3)
Linear (256x64)
Linear (64x64)
Linear (256x64)
Linear (64x64)
Outputs Concatenation
Linear (128x64)
Linear (64x9)

Figure 4: Advanced CNN architecture [23]

3.2.3 Multitaper CNN

As a second step, we decided to feed the multitaper ouput presented in 3.1.2 into the simple CNN classifier in order to improve the performance. We kept the same architecture but we replaced the 1D convolutional layers by 2d convolutional layers since multitaper signals represent a time serie in a 2d space (time, frequency). After several experiments we decided to only use the multitaper data and not the raw data as in the advanced CNN since it led to too much overfitting.

3.2.4 CNN and LSTM

Then, we experimented with a combination of CNN and LSTM on the raw data since CNN are efficient in extracting relevant features from a signal but they lack a proper inner representation of the sequence that an RNN has, and thus often fall behind RNN-based classifiers in terms of performance. After several experiments based on an extention of our Simple CNN approach, we chose to implement the same architecture as TinySleepNet [20], shown in Figure 5, since it is relatively less complex than other CNN-LSTM based approaches (it is already a lighter and more performant version of DeepSleepNet [21]) and might lead to less overfitting than a bigger model.
The architecture is very promising since we hope to capture the information contained in the sequence nature of the data, which a CNN-only model does relativerly poorly.

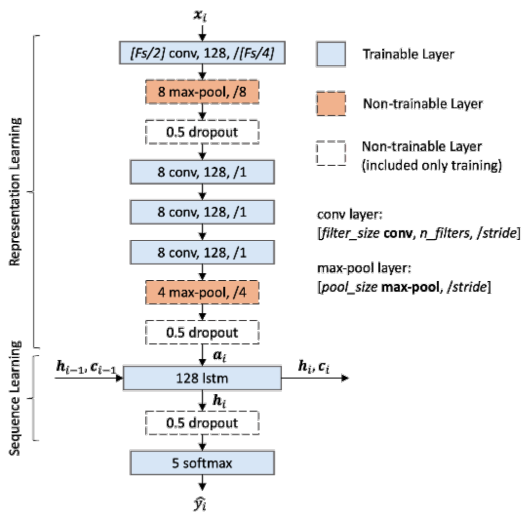


Figure 5: CNN/LSTM architecture [20]

3.2.5 Attnsleep

Finally, we have seen that attention-based models have received a lot of interest lately since they exhibit outstanding performance in many Natural Language Processing [22, 7] and in Computer Vision [8] tasks. Thus, we reckoned that evaluating the performance of these new models on our dataset would be very interesting, and have thus decided to review the attention-based solutions that perform sleep stage classification. In particular, Attnsleep [9] was released last year and exhibited very interesting performance.
Thanks to the authors, the original code used for Attnsleep is available on their GitHub which enabled us to direclty run their model on our dataset, after a conversion to the international PSG file format of our data. Figure 6 illustrates the architecture used by Attnsleep.

This architecture for automatic sleep stages classification proposes a novel feature extraction module based on multi-resolution CNN (MRCNN) and adaptive feature re-calibration (AFR). The MRCNN extracts features corresponding to low and high frequencies from different frequency bands, and the AFR models the features interdependencies to enhance the feature learning.
Attnsleep also proposes a novel temporal context encoder (TCE) that deploys a multi-head attention with causal convolutions to efficiently capture the temporal dependencies in the extracted features. It also designs a class-aware loss function to effectively address the data imbalance issue without additional computations. The main blocks of this method are thus: a feature extraction step, the temporal context encoder and finally the sleep stage classification step as shown in Figure 6.

Let's get into more details about the principle of Attnsleep. First, the MRCNN (detailed in Figure 7) with two branch CNN architectures is exploited to extract the features from a 30-second EEG signal. In particular, it extracts high-frequency features by the small kernel convolutions and low-frequency features by the wide kernel convolutions. Then the AFR module models the inter-dependencies among the features extracted by MRCNN. Moreover, AFR can select and highlight the most important features, which helps to enhance the classification performance. It then develops a TCE module to capture the long-term dependencies in the input features. The core component of TCE is the multi-head attention supported by causal convolutions. Third, the classification decision is done by a fully connected layer with a softmax activation function. We also leverage a class-aware cost-sensitive loss function to handle the data imbalance issue.

4. Experiments

4.1. Datasets

For this Kaggle challenge, we are provided with the different signals recorded by the *Dreem* headband. The signals are comprised of 30 seconds windows extracted from 61 subject records. The idea thus is to develop an algorithm of sleep staging able to differentiate between Wake, N1, N2, N3 and REM.

More specifically the signals are made of 5 electroencephalogram (denoted as eeg1 to eeg7 but two are actually

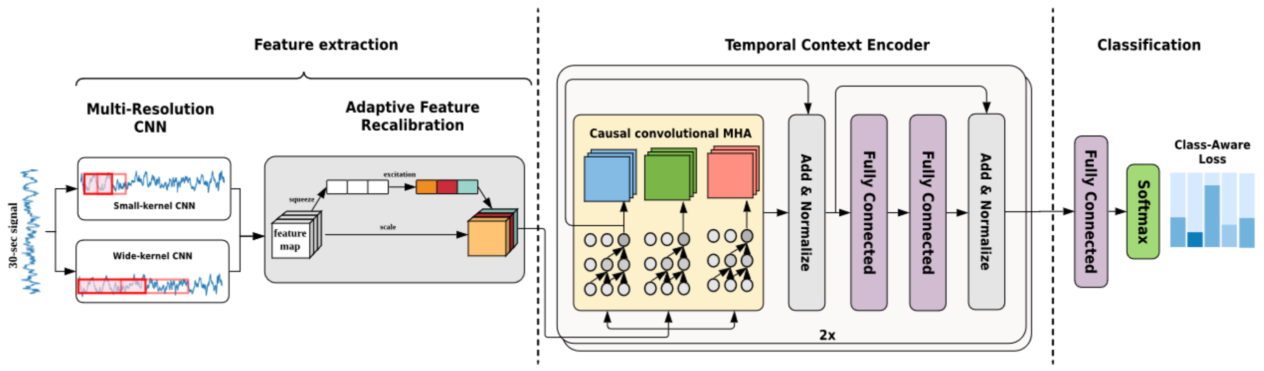


Figure 6: Attnsleep: an attention-based model [9]

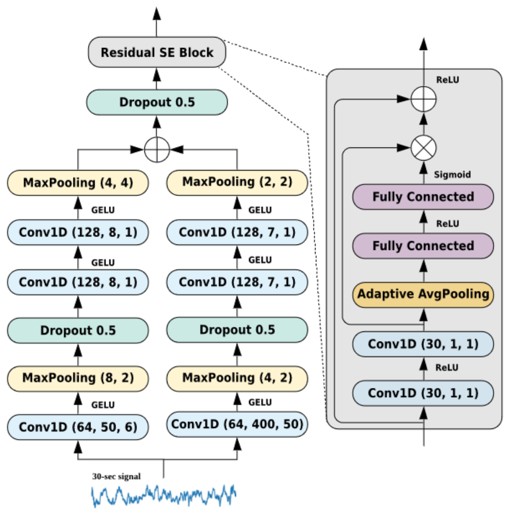


Figure 7: Attnsleep: Multi-resolution CNN

Dataset name	# of 30s epochs	Sampling rate
Dreem Challenge (ours)	10,000	50Hz
Sleep-EDF-20 [14]	42,308	100Hz
Sleep-EDF-78 [14]	195,479	100Hz
SHHS [25, 18]	324,854	125Hz
MASS [17]	58,600	256Hz

Table 1: Comparison of different sleep EEG datasets

Architecture	Average F1-Score
Baseline	0.520
Simple CNN	0.552
Advanced CNN	**0.715**
Multitaper CNN	0.674
CNN/LSTM	0.655
Attnsleep	0.554

Table 2: Results of all the architectures on a 5-fold cross validation

missing) channels which are sampled at a 50 Hz frequency and of 3 accelerometer channels (x,y and z) sampled at a 10Hz frequency. For each accelerometer 30 seconds time window 300 measures are present ($30 * 10\ Hz$) and for each eeg 30 seconds time window 1500 measures are present ($30 * 50\ Hz$).

To put our dataset in perspective, the sleep models we studied (Attnsleep, TinySleepNet) where designed to be run on Sleep-EDF-20 [14] and MASS [17], and Sleep-EDF-20, Sleep-EDF-78 [14] and SHHS [25, 18] respectively, which are much larger, both in terms of number of epochs and sampling rate, as shown by Table 1. We will thus have to take this into account when evaluating our performance.

4.2. Model evaluation

To assess the performance of our model, we use an average F1-score metric. More specifically we compute a F1 score for each class (F1 score of the class against all others) and we average the 5 obtained values to get the average F1-score. The main asset of the F1 score is that it takes into account the recall and the precision of the model and thus it is more reliable metric than the accuracy in case of a not perfectly balanced data set.
The results of the models we have evaluated are available in Table 2, all evaluations were performed using 5-Fold cross validation to obtain the best approximation of each model's true classification performance.

To further assess the efficiency of the model we will rely

Architecture	Average F1-Score
Baseline	0.520
Simple CNN	0.532
Advanced CNN	**0.705**
Multitaper CNN	0.661
CNN/LSTM	0.642
Attnsleep	0.561

Table 3: Results of all the architectures on the public Kaggle test set

on the Kaggle leaderboard which contains the scores obtained by 75 teams, as shown in Table 3. However the scores in the leaderboard are obtained with a test data set on which we do not have the ground truth. So the Kaggle leaderboard will only provide us with a good proxy of the performance of our model. As you can see, the performances and quite close to the cross-validated performances which underlines the importance of cross-validation of model performance assessment.

We were somewhat surprised with the results as the models we were expecting to perform better has poorer results than simpler models. The best performing model is clearly the Advanced CNN which leverages the raw signals and their FFT, while the runner ups are the multitaper version of the Simple CNN and the CNN/LSTM architecture based on TinySleepNet.

Our interpretation of this performance ranking is the relative small size of our dataset, which makes complex architectures like Attnsleep and CNN/LSTM quite powerless compared to lighter models which are less prone to overfitting and more able to capture the underlying signal within our recordings.

5. Discussion and Conclusion

Our work has shown through the thorough evaluation of several common time series preprocessing, feature extraction and classification models that using a smaller and lighter model leads to improved performance in spite of our initial inclination to use more complex ones. We reckon that this is due to the relative small size of our dataset compared to the open-source datasets on which our reference models were trained, which explains why they performed poorly on our experiments.

We think that we could get better results by leveraging transfer learning since there are very large open-source sleep studies available (detailed in Table 1), which would allow the creation of large pretrained models like VGG16 [19] or ResNet50 [8], or even more complex ones like Transformer-based ViT [8] or BERT [21] though it might require even more data than presently available.

References

[1] Dreem sleep staging challenge 2021. https://www.kaggle.com/c/dreem-3-sleep-staging-challenge-2021/overview. Accessed: 2022-02-01. 1

[2] Stages of sleep. https://www.sleepfoundation.org/stages-of-sleep. Accessed: 2022-02-01. 1

[3] Khald Aboalayon, Miad Faezipour, Wafaa Almuhammadi, and Saeid Moslehpour. Sleep stage classification using eeg signal analysis: A comprehensive survey and new investigation. *Entropy*, 18(9):272, Aug 2016. 2

[4] Khald Ali I. Aboalayon, Miad Faezipour, Wafaa S. Almuhammadi, and Saeid Moslehpour. Sleep stage classification using eeg signal analysis: A comprehensive survey and new investigation. *Entropy*, 18(9), 2016. 3

[5] Behtash Babadi and Emery N. Brown. A review of multitaper spectral analysis. *IEEE Transactions on Biomedical Engineering*, 61(5):1555–1564, 2014. 3

[6] Ahmed Bahammam, Divinagracia Gacuan, Smitha George, Karen Lorraine Acosta, Seithikurippu R. Pandi-Perumal, and Ravi Gupta. *POLYSOMNOGRAPHY I: PROCEDURE AND TECHNOLOGY*, pages 443–456. 09 2016. 2

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. 5

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. 5, 7

[9] Emadeldeen Eldele, Zhenghua Chen, Chengyu Liu, Min Wu, Chee-Keong Kwoh, Xiaoli Li, and Cuntai Guan. An attention-based deep learning approach for sleep stage classification with single-channel eeg. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 29:809–818, 2021. 5, 6

[10] John Cristian Borges Gamboa. Deep learning for time-series analysis, 2017. 2, 3

[11] Ary Goldberger, Luís Amaral, L. Glass, Shlomo Havlin, J. Hausdorg, Plamen Ivanov, R. Mark, J. Mietus, G. Moody, Chung-Kang Peng, H. Stanley, and Physiotoolkit Physiobank. Components of a new research resource for complex physiologic signals. *PhysioNet*, 101, 01 2000. 2

[12] Conrad Iber, Sonia Ancoli-Israel, A.L. Chesson, and Stuart Quan. The aasm manual for the scoring of sleep and associated events: Rules, terminology and technical specifications. *Westchester, IL: American Academy of Sleep Medicine*, 01 2007. 1

[13] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, 2019. 3

[14] B. Kemp, A.H. Zwinderman, B. Tuk, H.A.C. Kamphuisen, and J.J.L. Oberye. Analysis of a sleep-dependent neuronal feedback loop: the slow-wave microcontinuity of the eeg. *IEEE Transactions on Biomedical Engineering*, 47(9):1185–1194, 2000. 6

[15] Marwa Obayya and F. E. Z. Abou-Chadi. Automatic classification of sleep stages using eeg records based on fuzzy c-means (fcm) algorithm. In *2014 31st National Radio Science Conference (NRSC)*, pages 265–272, 2014. 2

[16] Alexander Neergaard Olesen, Poul Jennum, Paul E. Peppard, Emmanuel Mignot, and Helge Bjarup Dissing Sørensen. Deep residual networks for automatic sleep stage classification of raw polysomnographic waveforms. *CoRR*, abs/1810.03745, 2018. 3, 4

[17] C. O'Reilly, N. Gosselin, J. Carrier, and T. Nielsen. Montreal archive of sleep studies: An open-access resource for instrument benchmarking exploratory research. *Journal of Seep Research*, 10.1111/jsr.12169, 2014. 6

[18] S. F. Quan, B. V. Howard, C. Iber, J. P. Kiley, F. J. Nieto, G. T. O'Connor, D. M. Rapoport, S. Redline, J. Robbins, J. M. Samet, and P. W. Wahl. The Sleep Heart Health Study: design, rationale, and methods. *Sleep*, 20(12):1077–1085, Dec 1997. 6

[19] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 7

[20] Akara Supratak, Hao Dong, Chao Wu, and Yike Guo. Deepsleepnet: A model for automatic sleep stage scoring based on raw single-channel eeg. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 25(11):1998–2008, 2017. 2, 5

[21] Akara Supratak and Yike Guo. Tinysleepnet: An efficient deep learning model for sleep stage scoring based on raw single-channel eeg. In *2020 42nd Annual International Conference of the IEEE Engineering in Medicine Biology Society (EMBC)*, pages 641–644, 2020. 5, 7

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. 5

[23] Ilia Zaitsev. Deep time series classification. `https://www.kaggle.com/purplejester/pytorch-deep-time-series-classification`, 2019. 4

[24] Guo-Qiang Zhang, Licong Cui, Remo Mueller, Shiqiang Tao, Matthew Kim, Michael Rueschman, Sara Mariani, Daniel Mobley, and Susan Redline. The national sleep research resource: Towards a sleep data commons. *Journal of the American Medical Informatics Association*, pages 572–572, 08 2018. 2

[25] G. Q. Zhang, L. Cui, R. Mueller, S. Tao, M. Kim, M. Rueschman, S. Mariani, D. Mobley, and S. Redline. The National Sleep Research Resource: towards a sleep data commons. *J Am Med Inform Assoc*, 25(10):1351–1358, 10 2018. 6