MIXING DIFFERENT SIGNAL REPRESENTATIONS: A STUDY OF A HYBRID NEURAL NETWORK ARCHITECTURE FOR SLEEP STAGE CLASSIFICATION USING SINGLE-CHANNEL EEG

Leonardo Tessarolo

University of São Paulo

ABSTRACT

Sleep stage identification is an essential tool for diagnosing and treating sleep disorders. However, it may be very expensive, time consuming and prone to human errors due to its dependence on visual assessment from well-trained experts. To alleviate the burden on this process, several machine learning-based automatic sleep stage classifiers have been developed. Among others, two possible approaches to creating these classifiers include the direct use of temporal EEG signals in ML models and the explicit extraction of features from EEG signals prior to usage in classifiers. In the proposed work, a mixture of both was evaluated in a hybrid multiple-input Deep Neural Network architecture. The hybrid architecture performed better than similar control architectures, but fell short of current state of the art models for sleep stage classification. Codes available at: https://github.com/leonardotessarolo/eeg-sleep-stage-detection

Index Terms— Sleep Stage Identification, Machine Learning, Artificial Neural Networks, Hybrid Architectures

1. INTRODUCTION

Sleeping plays an essential role in a broad range of tasks related to one's cognitive and motor performance [1]. Humans spend around one third of their lifetimes asleep, and sleep-related conditions such as Obstructive Sleep Apnea (OSA) and insomnia can considerably affect one's health [2]. Particularly, what are typically referred to as *sleep stages* hold importance in biological functions such as memory, muscle activity and blood pressure [3], [4].

There exist two main types of sleep states: non-rapid eye movement (NREM) and rapid eye movement (REM) sleep, which alternate cyclically throughout the sleeping process [5]. According to guidelines developed by Rechtstaffen and Kales (R&K) in 1968, NREM sleep can be further divided into 4 stages (also referred to as S1, S2, S3 and S4) [2]. However, a more recent rule proposed in 2007 by the American Academy of Sleep Medicine (AASM) proposes the division of NREM into three stages N1, N2 and N3 [6]. N1 and N2 (also referred to as transitional and light sleep, respectively) directly follow S1 and S2 stages from the 1968 guideline, while N3 (deep sleep or slow wave sleep) may be obtained from merging S3 and S4 stages [2], [7].

Sleep stage scoring is the current best practice for analyzing human sleep, particularly for diagnosing and basing treatment for sleep disorders. [2], [5]. Polysomnographic (PSG) recordings are typically used, with experts visually labeling Electroencephalogram (EEG), Electrooculogram (EOG), Electromyogram (EMG) and Electrocardiogram (ECG) recordings acquired from patients while they sleep overnight at the hospital [8].

Given the need for visual examination of multiple channels by an expert, sleep stage scoring is task which is expensive, tedious, time consuming and prone to human error [2], [5], [7], [9]. As such, a number of machine learning-based automatic sleep stage detectors have emerged to aid in the process [2], [5]. Generally stating, the EEG is the most often used signal for sleep stage scoring, be it manual or automated [2]. In this work, single-channel EEG will be used, as it makes for a broader usability. This is due to multi-channel EEG requiring several electrodes to be placed, which is financially onerous, restricts patients' movements and requires subjects to sleep in the health facility for recording [10].

Regarding automated sleep stage detection via machine learning systems, two possible approaches to creating classifiers for recognizing sleep stages from EEG signals include [2], [5], [9]:

1. The explicit engineering and derivation of features from EEG signals, which undergo feature selection and are introduced into classification models such as Random Forests or Support Vector Machines [6], [10], [11].

2. Feeding a less processed, more raw representation of the EEG signals (such as the original time-domain EEG or its spectrogram, with normalizing preprocessing transformations) into a Deep Learning model equipped with architectural components such as 1D Convolutions, 2D Convolutions or LSTMs [12]–[14].

In Deep Learning, convolutional layers (be them 1D or 2D convolutions) can be seen as to derive learned represen-

tations ("features") from input objects [15]. Albeit through a different mechanism, the same can be said about recurrent networks, be them "vanilla" Recurrent Neural Networks or gated recurrent architectures such as GRU or LSTM [15]. In this work, an investigation was done on whether there may be some advantage in performance on sleep stage detection by combining learned representations from specific Deep Learning architectures with designed representations obtained from manual feature engineering. The combination of both types of features in a single multiple-input Deep Neural Network architecture was be evaluated and compared in performance against either individual approach.

The created hybrid architecture has the following general specifications, which will be discussed in greater depth in subsection 2.3:

- There are two feedforward parallel input branches which later on concatenate to form a final (also feedforward) processing flow. Each branch receives a specific type of input: one receives the time-domain EEG signal itself after minor preprocessing, while the other receives the engineered and selected features.
- The branch which processes the "raw" EEG signal is composed of units consisting of 1D Convolution layers, max pooling layers, dropout layers and batch normalization layers;
- The branch which processes the engineered features is composed of units consisting of fully-connected layers, dropout layers and batch normalization layers;
- The final processing flow is also composed of units consisting of fully-connected layers, dropout layers and batch normalization layers;

The remainder of this work is structured as follows: Section 2 gives further details on the dataset used, feature selection process performed, on the neural architectures trained and specifications for training them and on the computational and programming infrastructure used. Section 3 presents the results obtained in sleep stage classification, performing comparisons between the three architectures trained in this work and also between the hybrid "full" architecture here obtained with state of the art models. Finally, Section 4 gives conclusions and possible next steps for the work performed.

2. MATERIALS AND METHODS

2.1. Dataset

In the experiments performed, the Sleep-EDFX-78 dataset from Physionet.org [16] was used. The dataset contains PSG recordings from two studies:

- Sleep Cassette (SC* files) studied age effects on sleep and was conducted on healthy participants with no sleep-related medication use aged from 25 to 101 years.
- Sleep Telemetry (ST* files) addressed the effects of temazepam medication on sleep in 22 Caucasian adults without any other medication in use.

Each record in both datasets contains EEG recordings from two channels (Fpz-Cz, Pz-Oz) and EOG recordings. Labels given follow the 1968 R&K guidelines, there being a wake stage, a REM sleep stage and four non-REM sleep stages. Following [13] and [14], in this work only the 78 subjects in the Sleep Cassette study were considered, and the Fpz-Cz channel in each recording was used as the single-channel EEG. As done in [13], [14] and [6], 30 second samples were obtained by segmentation of the single channel EEG for generating the inputs to the models. As the EEG measurements are performed at a sampling rate of 100Hz, this corresponds to samples of length 3000. Figure 1 presents one such 30 second segment, corresponding to a wake sleep state.

Also, according to the latest 2007 AASM guidelines, stages 3 and 4 were merged into a single stage [7]. Thus, the sleep stage detection performed is a 5-class classification between classes W (wake), R (REM sleep), and N1, N2, N3 (non-REM sleep stages 1,2 and 3). Furthermore, awake periods before and after sleep periods were limited to 30 minutes each, to give more focus on periods during which subjects were asleep [13].

The result is a dataset containing a total of 192017 samples, whose class distribution is presented in Table 1. One may observe that the dataset obtained is highly unbalanced.

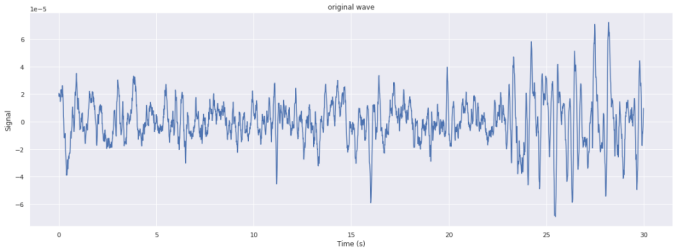

Fig. 1. 30 second sample from an Fpz-Cz channel in Sleep-EDFX-78, corresponding to a wake sleep state.

2.2. EEG Sample Preprocessing

As the aforementioned hybrid architecture will both pursue learned representations and receive manually created features, there must be two types of preprocessing: one for the "raw" time-domain EEG signal and another which engineers and selects features for characterizing the EEG signal. The former is presented in 2.2.1, while the latter in 2.2.2.

Sleep Stage	Samples
W	62489
N1	21522
N2	69132
N3	13039
R	25835

Table 1. Number of samples generated per sleep stage.

2.2.1. Time Signal Input Preprocessing

For each 30 second signal such as the one showed in Figure 1, the preprocessing for time signal input is fairly simple. The operation performed is that of standardization by subtracting the sample mean and normalizing by the sample standard deviation. If X is the original 3000-long signal, μ is its mean, σ is its standard deviation and $\tilde{X}$ is the transformed standardized signal, it may be written:

$$\tilde{X} = \frac{X - \mu}{\sigma} \tag{1}$$

2.2.2. Feature Engineering and Selection

FREQUENCY-DOMAIN DECOMPOSITION

Signal decomposition is a useful step for extracting useful information from a complex signal such as the EEG [6]. In this work, a frequency-domain decomposition is performed on the signals, in order to obtain characteristic brain waves which are important for experts to classify sleep stages [17]. The five basic brain waves and their considered frequency ranges are as follows [6], [17]:

1. Delta (δ) wave (0.5-4Hz): usually appears in slow-wave sleep (N3).
2. Theta (θ) wave (4-8Hz): deep relaxation/drowsiness.
3. Alpha (α) wave (8-15Hz): relaxation.
4. Beta (β) wave (15-31Hz): active thinking.
5. Gamma (γ) wave (>31Hz): concentration.

Beyond the presented 5 basic waves and their associated frequency bands, from [6] the following bands are added for a total of seven distinct frequency decompositions:

1. Dividing the (β) wave into two subwaves: a β_1 wave (14-22Hz) and a β_2 wave (22-31Hz).
2. Spindle waves: a train of distinct waves with most common frequency in the range 12-14Hz.

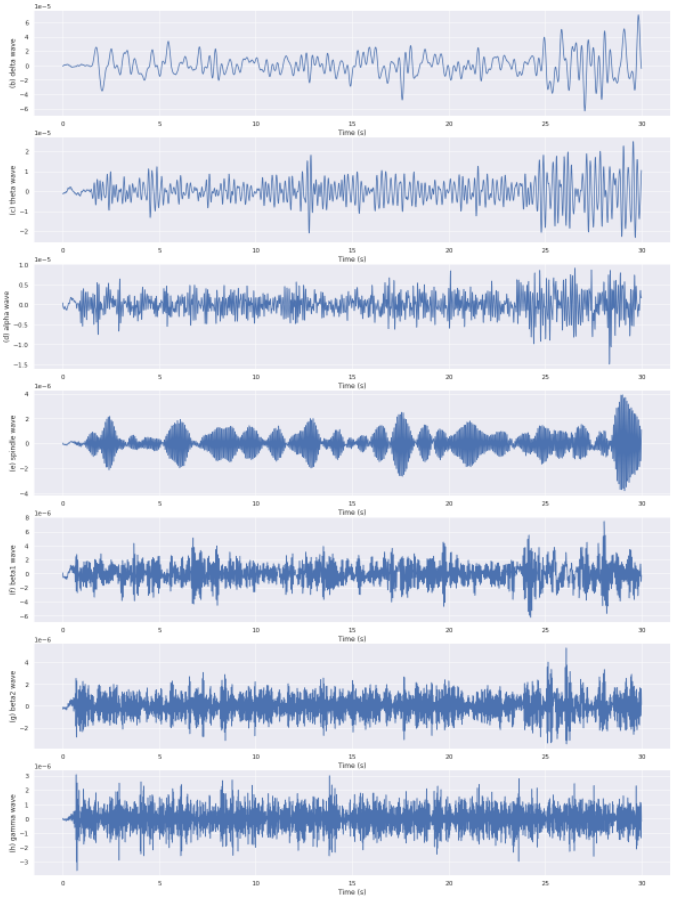

Fig. 2. Frequency decomposition of 30 second sample presented in Figure 1, corresponding to a wake sleep state.

For simplicity and better phase behavior, a filter bank of FIR filters was constructed for executing the frequency decomposition. The filter coefficients for each subband were obtained by using the Remez algorithm. The decomposition of the sample in Figure 1 in the presented frequency bands is portrayed in Figure 2.

CREATED FEATURES

Having obtained the aforementioned decomposed components, features are constructed as to represent the characteristic differences between EEG signals from distinct sleep stages. Following from [6], [10], [11], [18], [19], 4 general types of features are created:

- **General statistics:** maximum, minimum, skewness and kurtosis. As time-domain signals are standardized before calculating features, mean and variance are not calculated.
- **Relative Energy in Characteristic Waves:** the relative concentration of energy along the seven compo-

nents obtained from frequency-domain decomposition are calculated.

- **Time-domain Complexity/Waveform measures:** Shannon Entropy [20] is calculated as a measure of the "uncertainty" contained in the distribution of the EEG signals. Another type of useful measure for describing signal complexity are Fractal Dimensions [6], [18]. In this work, Katz, Petrosian and Higuchi fractal dimensions are calculated. Also, Hjorth parameters of mobility and complexity, which are commonly used to characterize EEG signals, are obtained [19].
- **Spectral Features:** for characterizing the frequency content of the EEG signals, Spectral Centroid, Bandwidth, Roll-Off and Flatness [10] are calculated, as well as Zero-Crossing Rate [11]. Each measure in this group yields two features, as the measures are obtained for five subsequent sub-windows in the 30 second EEG sample. The mean and standard deviation along the five sub-windows are calculated for each sub-window.

Further details on each calculated measure are given:

1. *Maximum:* maximum point of EEG signal.
2. *Minimum:* minimum point of EEG signal.
3. *Skewness [10]:* skewness (γ) is a measure of the degree of assymetry of a distribution around its mean.

$$\gamma = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{x_i - \mu}{\sigma}\right)^3 \tag{2}$$

4. *Kurtosis [10]:* kurtosis (κ) is a measure of the degree to which a distribution has is long-tailed.

$$\kappa = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{x_i - \mu}{\sigma}\right)^4 \tag{3}$$

5. *Relative Energy of the ith characteristic wave [6]:* the relative energy E_i may be defined as:

$$E_i = \frac{\sum_{n=1}^{N}|x_i(n)|^2}{\sum_i(\sum_{n=1}^{N}|x_i(n)|^2)} \tag{4}$$

6. *Entropy [6]:* the entropy H(**X**) of a random variable **X** is the average level of "information" or "uncertainty" relative to the variable's possible outcomes [20]. Given a discrete random variable **X** with possible outcomes ξ It is defined as:

$$H(X) = -\sum_{x\in\xi} p(x)\log p(x) \tag{5}$$

As the EEG does not a discrete-valued distribution, an approximation was made by binning each sample's distribution. 10 bins were used.

7. *Katz Fractal Dimension [6]:* Katz Fractal Dimension (KFD) is defined as:

$$KFD = \frac{\log N}{\log N + \log(d/L)} \tag{6}$$

where N is the total number of samples in the signal, L refers to the sum of distances between two successive points, and *d* represents the maximum Euclidean distance between the first point and any other point on the waveform.

8. *Petrosian Fractal Dimension [6]:* Petrosian Fractal Dimension (PFD) is defined as:

$$PFD = \frac{\log N}{\log N + \log(N/(N+0.4M))} \tag{7}$$

where N is the total number of samples in the signal, and M is the number of sign changes in the signal that results from differentiating the original signal.

9. *Higuchi Fractal Dimension [6], [18]:* the original signal *x(n)* is regrouped into *k* new series. For any parameter *m* smaller than *k*, the *k*th series is represented as

$$x_m^k = \{x(m), x(m+k), x(m+2k), ...\} \tag{8}$$

for *m* from 1 to *k*. Then, the length of the curve by connecting the *k*th series x_m^k is:

$$L_m(k) = \frac{1}{k}\sum_{i=1}^{\lfloor (N-m)/k\rfloor}\frac{|x(m+ik) - x(m+(i-1)k|}{\lfloor (N-m)/k\rfloor \cdot k/(N-1)} \tag{9}$$

The average of $L_m(k)$ across all possible parameters *m* is

$$L(k) = \frac{1}{k}\sum_{m=1}^{k} L_m(k) \tag{10}$$

The Higuchi Fractal Dimension (HFD) is defined as the slope of the line which best fits the point pairs $(-\ln(k), \ln(L(k)))$ for all values of *k* smaller than seven [18].

10. *Hjorth Mobility [6], [19]:* Hjorth Mobility (HM) represents the proportion of standard deviation of the power spectrum,

$$HM = \frac{\sigma'}{\sigma} \tag{11}$$

where σ' is the standard deviation of the first derivative of the analyzed signal.

11. *Hjorth Complexity [6], [19]:* Hjorth Complexity (HC) represents the similarity between the signal and a sine wave,

$$HC = \frac{\sigma'' \sigma'}{(\sigma')^2} \tag{12}$$

where σ'' is the standard deviation of the second derivative of the analyzed signal.

12. *Spectral Centroid [10]:* the Spectral Centroid (SC) is defined as the frequency-weighted sum of the magnitude spectrum of the signal normalized by its unweighted sum:

$$SC = \frac{\sum_{m=0}^{N-1} m \cdot |X(m)|}{\sum_{m=0}^{N-1} |X(m)|} \tag{13}$$

13. *Spectral Bandwidth [10]:* the Spectral Bandwidth (SB) can be seen as the variance of the magnitude spectrum around the spectral centroid:

$$SC = \frac{\sum_{m=0}^{N-1} (m - SC)^2 \cdot |X(m)|}{\sum_{m=0}^{N-1} |X(m)|} \tag{14}$$

14. *Spectral Roll-Off [10]:* the Spectral Roll-Off (SRO) is the frequency sample below which $c\%$ of the coefficients of the magnitude spectrum of a signal are concentrated. In accordance with [10], in this work a value of $c = 0.95$ was used.

15. *Spectral Flatness [10]:* Spectral Flatness (SF) is a measure that quantifies how noise-like a sound is, as opposed to being tone-like. It is the ratio of the geometric mean to the arithmetic mean of the magnitude spectrum of a signal.

$$SF = \frac{\prod_{m=0}^{N-1} |X(m)|^{\frac{1}{N}}}{\frac{1}{N} \sum_{m=0}^{N-1} |X(m)|} \tag{15}$$

16. *Zero-Crossing Rate [11]:* the zero-crossing rate (ZCR) of a signal may be defined as the fraction of times adjacent samples in a signal have opposite signs.

For each of the seven frequency-domain decompositions obtained, as well as for the original wave, the measures described in feature groups General Statistics, Time-domain Complexity/Waveform Measures and Spectral Features are obtained. As each measure in the spectral features group yields two features, this results in the creation of $8 \cdot 20 = 160$ features. By including relative energies as well, there are a total of **167 possible features** created.

FEATURE SELECTION

The selection of a subset of the aforementioned features for entering the machine learning models is done by analyzing the total Gini Impurity decrease caused by each feature in the construction of Random Forests [6]. The procedure held is portrayed in Figure 3, and consists of iterative elimination of the worst feature considering the aggregate importance across all frequency-domain decompositions made. Also, selection is performed on a sampled dataset of 19 of the total 78 subjects (approximately one fourth of the entire data) contained in the dataset described in 2.1.

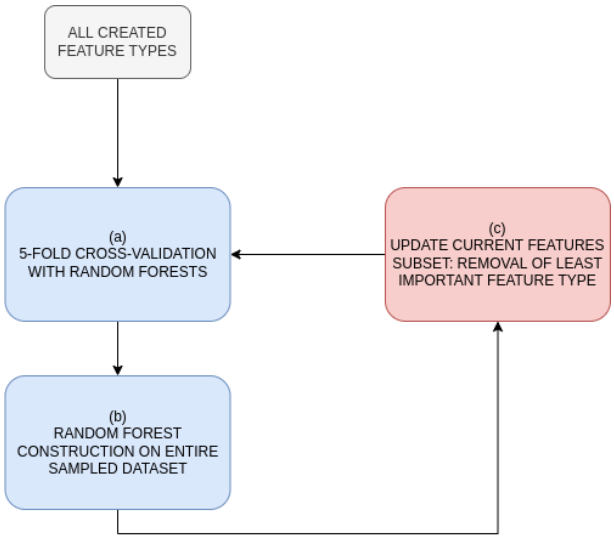

Fig. 3. Feature selection procedure held. **(a)** A 5-Fold cross-validation procedure is performed, also using Random Forest classifiers, to estimate the performance (accuracy) of the current feature set. All samples for a given subject are contained in a single fold for there not to be train-test leakage.**(b)** A Random Forest is constructed on the entire (sampled) dataset for obtaining feature importances for current feature set. **(c)** The worst feature considering aggregate importance across all frequency-domain decompositions made is removed, and the procedure repeats itself until all features are eliminated.

Features are selected as the group after which feature removal yields monotonic decrease in classification accuracy. By these criteria, the resulting selected subset consists of the following features (in decreasing order of importance):

1. *Mean of Spectral Bandwidth.* As mentioned previously in "created features", spectral features are calculated in 5 sub-windows of the 3000-long EEG sample. The mean here refers to the mean of the spectral bandwidth taken across the 5 sub-windows.

2. *Mean of Spectral Roll-Off.* Same logic as the mean taken for Spectral Bandwidth.

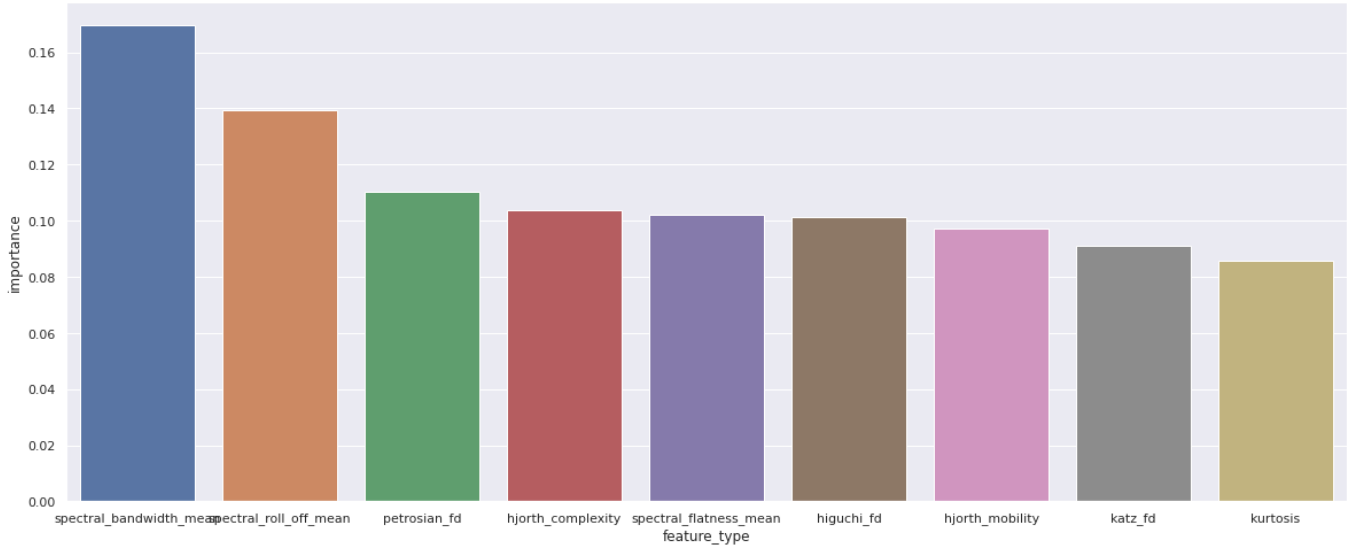

Fig. 4. Aggregate importance across all wave types for each feature type.

3. *Petrosian Fractal Dimension.*
4. *Hjorth Complexity.*
5. *Mean of Spectral Flatness.* Same logic as the mean taken for Spectral Bandwidth and Spectral Roll-Off.
6. *Higuchi Fractal Dimension.*
7. *Hjorth Mobility.*
8. *Katz Fractal Dimension.*
9. *Kurtosis.*

The 9 selected features across the 8 possible wavetypes (original wave and the seven characteristic waves obtained by frequency-domain decomposition) yield a total of **72 features** to be inputed into the Neural Network models. Figure 4 displays the aggregate importance of each selected feature type considering all 8 wavetypes.

2.3. Neural Network Architectures

2.3.1. Building Blocks

Figures 5 and 6 present a *convolutional unit* and a *fully-connected unit* which are repeated throughout the constructed architectures.

The *convolutional unit* (Figure 5) contains a 1D convolutional layer with 64 3x1 filters, a batch normalization layer, a dropout layer with 1% dropout rate, and a 1D max pooling layer with a pooling factor of 3. The *fully-connected unit* (Figure 5) contains a fully-connected layer with 150 neurons, a batch normalization layer and a dropout layer with 1% dropout rate.

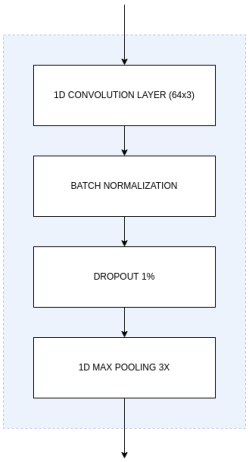

Fig. 5. Convolutional unit (1D convolution + batch normalization + dropout + 1D max pooling).

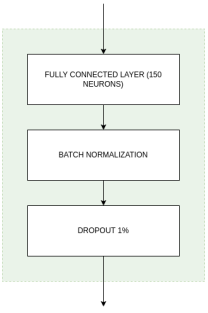

Fig. 6. Fully-connected unit (Fully-connected layer + batch normalization + dropout).

2.3.2. *Time-domain EEG + Derived Features Architecture*

The ”full” hybrid 2-input, 2-branch architecture used for classifying sleep stages from both the EEG time-domain signal and its engineered features is portrayed in Figure 7. The processing of the EEG signal is done in a branch composed of five of the convolutional units in Figure 5. The 72 input features selected in subsection 2.2.2, instead, are processed by three of the fully-connected units in Figure 6. Resulting representations from the two parallel branches are concatenated and further processed by three more fully-connected units. 5 convolutional units are used for processing the EEG time-domain signal so that the number of flattened features resulting from the convolutional branch are not too many more (832) than the number of activations produced by the fully-connected layer (150), and also for not producing representational bottlenecks by performing excessively aggressive max-pooling operations. A final softmax layer generates the 5-class probabilities, resulting in a model with 304299 parameters.

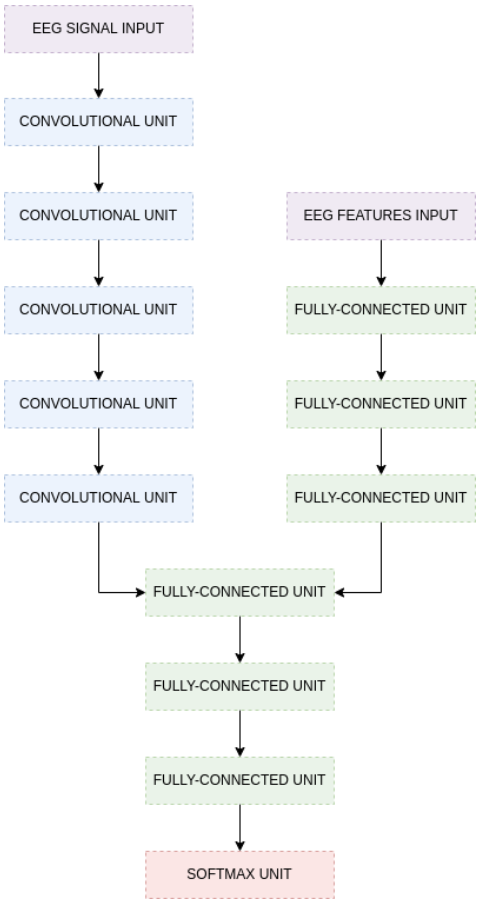

Fig. 7. Full hybrid 2-input and 2-branch architecture.

2.3.3. *Time-domain EEG Architecture*

For performance comparison with respect to the ”full” architecture described in subsection 2.3.2, a convolutional model that takes only the time-domain EEG signal is also composed, its architecture being obtained by removing the three initial fully-connected units in the full architecture. The resulting deep neural network has a total of 223749 parameters and is presented in Figure 8.

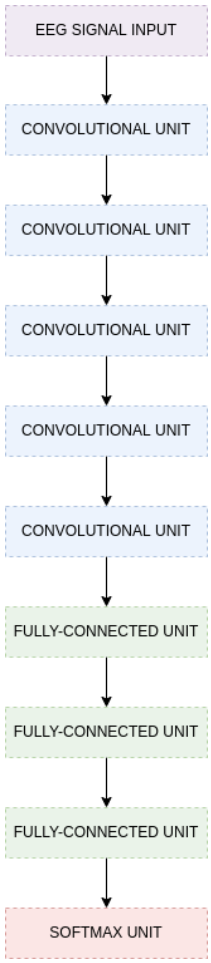

Fig. 8. Convolutional architecture for processing time-domain EEG signal.

2.3.4. *EEG Features Architecture*

Also for performance comparison with respect to the ”full” architecture described in subsection 2.3.2, a multilayer perceptron model that takes only the features selected in subsection 2.2.2 is also composed, its architecture being obtained by removing the five convolutional units in the full architecture. The resulting deep neural network has a total of 128555 parameters and is presented in Figure 9.

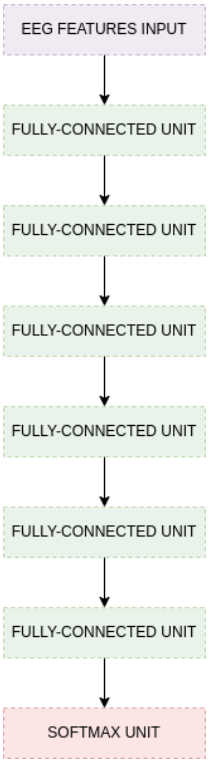

Fig. 9. Multilayer perceptron architecture for processing EEG features.

2.4. Computational infrastructure and programming framework

The models described in subsection 2.3 were built, trained and tested using the Tensorflow Keras Functional API [21], which allows for building more complex multi-input and multi-output neural network architectures. Another Tensorflow framework, the TFRecord dataset, was also used for creating the files used in model training and evaluation.

Spectral features described in subsection 2.2.2 were obtained by using methods from the Librosa package [22]. The code for creating the remaining features was created by the author of this work, by leveraging mainly methods and objects from Numpy [23] and Scipy [24] libraries. Code for Random Forests used for feature selection was obtained from the Scikit-Learn [25] library.

All experiments, including mainly feature selection, dataset creation and model training/evaluation were executed on a 12-CPU machine.

2.5. Model Training and Evaluation Specifications

Following other studies such as [6], [13], [14], a subject-independent cross-validation test was held to evaluate performance aspects of the models presented in subsection 2.3. The 78 subjects in the Sleep-EDFX dataset Sleep Cassette were divided into 5 groups, which were then used in a 5-fold cross-validation for each one of the three architectures presented. Folds used across the three different types of models held constant, and all of the samples relative to each subject in the dataset were contained in one fold. For each fold, a new model was trained, having its validation classification performance logged with confusion matrices, as well as class-specific precision, recall and f1-score. For each fold, training loss and validation loss, as well as total training time were also logged in all cases. Final metrics were the average of the metrics across the 5 training folds. For each class *i*, it is considered that:

$$Precision_i = \frac{TP_i}{TP_i + FP_i} \tag{16}$$

$$Recall_i = \frac{TP_i}{TP_i + FN_i} \tag{17}$$

$$F1_i = 2 \cdot \frac{Precision_i \cdot Recall_i}{Precision_i + Recall_i} \tag{18}$$

where TP_i, FP_i and FN_i are true positives, false positives and false negatives for class *i*, respectively. Overall accuracy is calculated as the ratio between the total number of samples classified correctly and the total number of samples, for all classes.

In all cases, an Adam optimizer was used with a categorical cross-entropy loss function. Following [13], models were trained with a learning rate of 0.001 for 10 epochs, and then the remainder of training epochs with a learning rate of 0.0001. After the initial 10 epochs, an early stopping criterion was set with a patience of 25 epochs on the non-improvement of validation loss. Also following [13], a batch size of 128 samples was used. ReLU activations were tested, but using tanh nonlinearities had slightly better performance and were kept.

3. RESULTS AND DISCUSSION

3.1. Comparison of Architectures Built

Table 3 contains per-class F1 score alongside accuracy, average elapsed time per fold and average number of epochs per fold. The full hybrid architecture performed best in prediction performance against the two control architectures, having greater F1-score for all classes and also greater overall classification accuracy. Although a more definitive conclusion would also require controlling for total number of model parameters (one may note general accuracy in the three presented architectures follows directly with total model size), the obtained results are evidence that there may be an advantage to combining the two forms of representation in a single architecture.

One may also note that the full hybrid architecture has a similar training time per epoch when compared to the Raw EEG model, but converges in fewer epochs. This may be due

	Confusion Matrix					Per-Class Metrics		
	W	N1	N2	N3	R	Precision	Recall	F1
W	58877	1553	976	110	973	84.1%	94.1%	88.8%
N1	5709	4246	8602	204	2761	45.9%	19.9%	27.6%
N2	1896	1582	59613	3620	2421	76.8%	86.3%	81.3 %
N3	121	27	3611	9270	10	70.2%	70.3%	69.4%
R	3249	1937	4808	139	15702	71.8%	60.7%	65.8%

Table 2. Confusion matrix and per-class metrics for the full hybrid model.

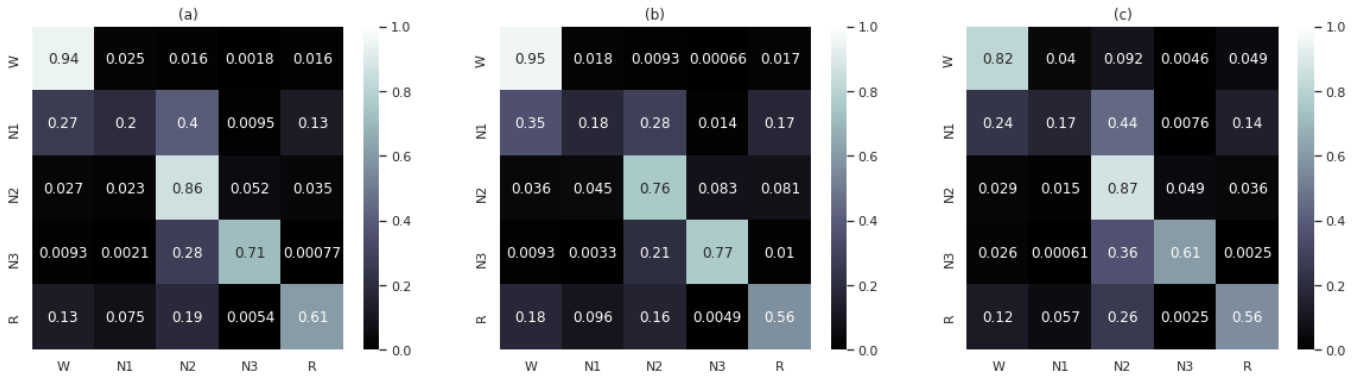

Fig. 10. Confusion matrices scaled by row totals for **(a)** Full Time-domain EEG + Features architecture, **(b)** Raw EEG architecture, **(c)** Features architecture.

	Per-Class F1 Score					Overall Metrics		
Model	W	N1	N2	N3	REM	Accuracy	Avg Time/Fold	Avg Epochs/Fold
Raw EEG	87.0%	22.9%	77.2%	67.9%	57.3%	73.0%	808 min	74.0
Features	81.6%	24.0%	77.3%	62.7%	58.9%	71.5%	**5 min**	47.6
Raw EEG + Features	**88.8%**	**27.6%**	**81.3%**	**69.4%**	**65.8%**	**76.9%**	516 min	**46.6**

Table 3. Comparison between the three architectures trained in this work.

to the need for the Raw EEG architecture to form its feature extractors from scratch in its early convolutional layers, while the full model already has some more processed information available with the addition of the extracted features as inputs. Also, an important result to be observed is the considerably smaller time in training required by the model with no convolutions, and that has only manually designed features as inputs.

A confusion matrix and per-class metrics for the ”full” hybrid architecture is given in Table 2. Figure 10 presents confusion matrices for the three trained architectures, where rows are scaled to sum up to one. It is interesting to note that the three architectures may have considerably different recalls for given classes (W, N2 and N3 diagonal values, for instance). Also, in all cases, classification performance for class N1 was the worst among the 5 classes.

3.2. Comparison Against State of the Art Architectures

Table 4 presents a benchmark of the full hybrid architecture developed in this work against three state-of-the-art algorithms in sleep stage detection: SleepEEGNet [26], AttnSleep [13] and EEGSNet [14]. Comparisons are made based on per-class F1-score and overall accuracy. Despite all algorithms being trained on the same Sleep-EDFX-78 dataset, one may see that the number of samples used differs between algorithms.

The algorithm created in this work underperforms in comparison to state-of-the-art models both in per-class F1 and in overall accuracy. However, observation of design principles present in these works offer helpful insights on ways to improve on performance for the hybrid multiple-input network built.

A type of module that is present in both [14] and [26]

Model	Per-Class F1 Score					Overall Metrics	
	W	N1	N2	N3	REM	Number of Samples	Accuracy
Raw EEG + Features	88.8%	27.6%	81.3%	69.4%	65.8%	192017	76.9%
SleepEEGNet [26]	91.72%	44.05%	82.49%	73.45%	76.06%	222479	80.03%
AttnSleep [13]	92.0%	42.0%	**85.0%**	**82.1%**	74%	195479	81.3%
EEGSNet [14]	**93.19%**	**50.03%**	84.19%	74.41%	**83.48%**	195479	**83.02%**

Table 4. Comparison between the full hybrid architecture developed in this work and state-of-the-art algorithms on sleep stage classification.

is a sequence learner, with the objective of learning more likely sequences of sleep stages. In [14], cascaded bidirectional LSTMs are employed for this task, while [26] employs bidirectional LSTMs in an encoder-decoder architecture. The same principle of optimizing over a sequence of states is also implemented in [6] via a Hidden Markov Model.

A second possible improvement over the hybrid architecture created is the inclusion of two convolutional branches, which is done in [13] and [26]. In both cases, the design principle is having two convolutional branches with different kernel sizes: one with a large kernel for extracting low-frequency features, and the other with a smaller kernel for extracting higher-frequency features. In both works, even the smaller kernels are about 10 times larger than the kernel size of 3 which was employed for convolutional units in this work.

Yet another possible improvement is the use of custom loss functions. As may be seen in Table 1, the dataset used is highly unbalanced, which may cause performance bias towards the more represented classes [13]. A class-aware loss function was implemented in [13] which gives larger weight to the misclassification of less represented classes. As the neural architectures trained in this work had better performance on the most representative classes W and N2 (Table 3), the use of a class-aware loss function may bring better classification performance, especially for class N1.

4. CONCLUSIONS

A multi-input hybrid neural network architecture was created, receiving as inputs both a "raw" time-domain EEG signal and features manually engineered and extracted from it. Training was performed for this architecture and for two control architectures with either only the EEG signal or the manually extracted features. Although controlling for the number of model parameters would be necessary for stronger conclusions, the hybrid architecture outperformed both of the control ones.

However, the multiple-input model still fell short of state-of-the-art algorithms on classification performance for sleep stages. Inpired by design principles present in these state-of-the-art architectures, improvements may be made on the architectures created in this work. One such design principle is the employment of a sequence learner to also consider more likely sequences of sleep stages. Another is the use of different kernel sizes in parallel convolutional branches, in order to better learn low-frequency and high-frequency features. Also, as the dataset used has high class imbalance, the use of a class-aware loss function may also bring better classification performance, especially for the least represented class N1.

References

[1] F. S. Luyster, P. J. Strollo Jr., P. C. Zee, and J. K. Walsh, "Sleep: A health imperative," *Sleep*, vol. 35, no. 6, 2012.

[2] K. A. I. Aboalayon, M. Faezipour, W. S. Almuhammadi, and S. Moslehpour, "Sleep stage classification using eeg signal analysis: A comprehensive survey and new investigation," *Entropy*, vol. 18, no. 9, 2016.

[3] J. Tank *et al.*, "Relationship between blood pressure, sleep k-complexes, and muscle sympathetic nerve activity in humans," *Amer. J. Physiol.-Regulatory, Integrative Comparative Physiol*, vol. 285, no. 1, 2003.

[4] G. Rauchs, B. Desgranges, and F. Eustache, "The relationship between memory systems and sleep stages," *J. Sleep Res*, vol. 14, no. 2, 2005.

[5] H. W. Loh *et al.*, "Automated detection of sleep stages using deep learning techniques: A systematic review of the last decade (2010–2020)," *Applied Sciences*, vol. 10, no. 24, 2020, ISSN: 2076-3417. DOI: 10.3390/app10248963. [Online]. Available: https://www.mdpi.com/2076-3417/10/24/8963.

[6] D. Jiang, Y.-n. Lu, M. Yu, and W. Yuanyuan, "Robust sleep stage classification with single-channel eeg signals using multimodal decomposition and hmm-based refinement," *Expert Systems with Applications*, vol. 121, pp. 188–203, 2019.

[7] K. D. Tzimourta *et al.*, "Eeg-based automatic sleep stage classification," *Biomed J*, vol. 1, no. 6, 2018.

[8] S. A. Keenan, "Chapter 3 an overview of polysomnography," in *Handbook of Clinical Neurophysiology*, ser. Handbook of Clinical Neurophysiology, C. Guilleminault, Ed., vol. 6, Elsevier, 2005, pp. 33–50. DOI: https://doi.org/10.1016/S1567-4231(09)70028-0. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1567423109700280.

[9] A. Craik, Y. He, and J. L. Contreras-Vidal, "Deep learning for electroencephalogram (EEG) classification tasks: A review," *Journal of Neural Engineering*, vol. 16, no. 3, p. 031 001, Apr. 2019. DOI: 10.1088/1741-2552/ab0ab5. [Online]. Available: https://doi.org/10.1088/1741-2552/ab0ab5.

[10] A. R. Hassan, S. K. Bashar, and M. I. H. Bhuiyan, "On the classification of sleep states by means of statistical and spectral features from single channel electroencephalogram," in *2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2015, pp. 2238–2243. DOI: 10.1109/ICACCI.2015.7275950.

[11] S. Motamedi-Fakhr, M. Moshrefi-Torbati, M. Hill, C. M. Hill, and P. R. White, "Signal processing techniques applied to human sleep eeg signals—a review," *Biomedical Signal Processing and Control*, vol. 10, pp. 21–33, 2014.

[12] O. Yildirim, U. B. Baloglu, and U. R. Acharya, "A deep learning model for automated sleep stages classification using psg signals," *International journal of environmental research and public health*, vol. 16, no. 4, p. 599, 2019.

[13] E. Eldele *et al.*, "An attention-based deep learning approach for sleep stage classification with single-channel eeg," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 29, pp. 809–818, 2021. DOI: 10.1109/TNSRE.2021.3076234.

[14] C. Li, Y. Qi, X. Ding, J. Zhao, T. Sang, and M. Lee, "A deep learning method approach for sleep stage classification with eeg spectrogram," *Int. J. Environ. Res. Public Health*, vol. 19, no. 10, 2022. DOI: 10.3390/ijerph19106322.

[15] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[16] B. Kemp, A. Zwinderman, B. Tuk, H. Kamphuisen, and J. Oberye, "Analysis of a sleep-dependent neuronal feedback loop: The slow-wave microcontinuity of the eeg," *IEEE Transactions on Biomedical Engineering*, vol. 47, no. 9, pp. 1185–1194, 2000. DOI: 10.1109/10.867928.

[17] P. A. Abhang, B. Gawali, and S. Mehrotra, *Introduction to EEG-and speech-based emotion recognition*. Academic Press, 2016.

[18] A. Accardo, M. Affinito, M. Carrozzi, and F. Bouquet, "Use of the fractal dimension for the analysis of electroencephalographic time series," *Biological Cybernetics*, vol. 77, no. 5, 1997. DOI: 10.1007/s004220050394.

[19] B. Hjorth, "Eeg analysis based on time domain properties," *Electroencephalography and clinical neurophysiology*, vol. 29, no. 3, pp. 306–310, 1970.

[20] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[21] Martín Abadi *et al.*, *TensorFlow: Large-scale machine learning on heterogeneous systems*, Software available from tensorflow.org, 2015. [Online]. Available: https://www.tensorflow.org/.

[22] B. McFee *et al.*, *Librosa/librosa: 0.9.2*, version 0.9.2, Jun. 2022. DOI: 10.5281/zenodo.6759664. [Online]. Available: https://doi.org/10.5281/zenodo.6759664.

[23] C. R. Harris *et al.*, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. DOI: 10.1038/s41586-020-2649-2. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2.

[24] P. Virtanen *et al.*, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.

[25] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[26] S. Mousavi, F. Afghah, and U. Acharta, "Sleepeegnet: Automated sleep stage scoring with sequence to sequence deep learning approach," *PLoS One*, vol. 14, no. 5, 2019. DOI: 10.1371/journal.pone.0216456.