AUTOMATIC SLEEP STAGE SCORING WITH CONVOLUTIONAL AND RECURRENT NEURAL NETWORKS

Anders Geil, Alma Lindborg
Code and supplementary material: https://github.com/ageil/deepsleep

ABSTRACT

Sleep stage scoring is an important component both in sleep research and as a clinical diagnosis tool for various sleep disorders. Currently, sleep scoring is done by manual inspection of polysomnograms (PSGs), which is an expensive and time-consuming procedure, but several studies have demonstrated the applicability of various machine learning approaches for automatic sleep stage scoring. We expand on a previously implemented convolutional neural network model using a pre-trained image classification network for classifying spectrogram images of electroencephalograms (EEG), which is an important component of the PSG. By using a recurrent neural network on the image features, we significantly increase classification accuracy compared to the non-recurrent model.

Index Terms— Sleep stage scoring, recurrent neural networks, convolutional neural networks, transfer learning

1. INTRODUCTION

Sleep stage scoring plays an important role in sleep research [1] as well as in medical diagnosis of sleeping disorders such as insomnia, sleep apnea or narcolepsy [2]. By tracking the progression of sleep stages during the course of the night, the researcher or clinician gets insights into the sleeping patterns of an individual. Sleep scoring is conventionally done by a trained specialist who reviews a polysomnogram (PSG). A PSG consists of an electroencephalographic (EEG) recording of brain activity, as well as an electrooculogram (EOG) recording of eye movements and an electromyogram (EMG) of muscle tension. With EEG equipment becoming increasingly cheap and accessible on the market, there is a growing interest in scoring sleep on EEG alone. This would give a possibility for extending sleep studies to more people, but since manual sleep stage scoring is very time consuming - requiring a specialist to review the PSG for the whole night and score each epoch of 30 seconds individually - it is of interest to build an automatic system which can reliably classify the PSG in a human-like manner.

Studies on automatic sleep stage scoring have applied artificial neural networks on various features in the EEG, mostly focusing on hand-crafted spectral features [3, 4]. However, a recent study proposed an altogether different approach, using one of the widely successful convolutional neural networks (CNNs) used for image recognition - the VGGnet - to extract features from spectrograms of the EEG [2]. The comparatively high performance of the CNN architecture for classifying spectrograms indicates successful transfer learning from the natural image domain to the spectrograms [2].

However, the CNN model does not fully exploit the structure in the sleep EEG data. Treating each spectrogram as an individual sample to be classified separately, the CNN model is oblivious to the fact that the sleep stages should be thought of as a Markov process with unequal transition probabilities, rather than as a number of independent random draws. The previous implementation attempted to solve this problem by including two epochs directly before and two epochs directly after the current (current-2 to current+2) in each spectrogram image [2], but since this approach requires compressing 5 epochs into one image, it leads to a potential loss of information as well as a less interpretable representation of the data. Therefore, a natural development of the CNN architecture would be to apply a recurrent neural network (RNN) to the features extracted by the CNN. This allows the model to weigh in information from neighboring time epochs.

2. METHODS

We implemented three models to classify sleep stages from spectrogram images: a CNN model, a CNN-RNN model with three time steps (RNN3) and a CNN-RNN model with five time steps (RNN5). All models had at its base the convolutional layers of the 16-layer VGGNet [5], which has well-documented high performance on natural images. The 16-layer VGGNet has the following architecture:

$$\mathrm{ccm}_{64}\ \mathrm{ccm}_{128}\ \mathrm{cccm}_{256}\ \mathrm{cccm}_{512}\ \mathrm{cccm}_{512}\ \mathrm{fcr}_{4096}\ \mathrm{fcr}_{4096}\ \mathrm{fcs}_{1000},$$

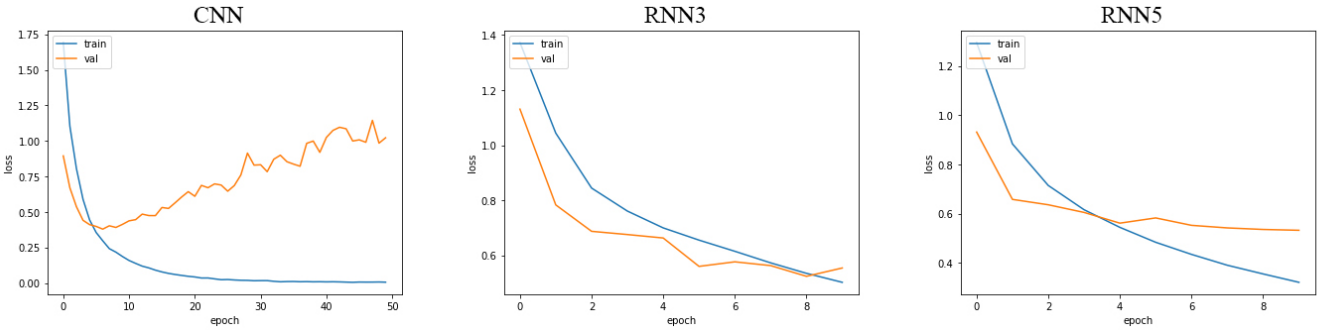

Figure 1. Training and validation loss for Fold 4 by training epoch for each model.

where c is a 3 x 3 convolutional filter of stride 1 using a ReLU activation function, m is a 2 x 2 maxpooling layer with a stride of 2, fcr and fcs correspond to fully-connected layers with ReLU and soft-max activations, respectively, and sub-indexed values represent the number of channels in each block. The weights of the convolutional layers of our models were set to the values of the ImageNet pre-trained weights of VGGNet, whereas we used different architectures for the top layers. Our CNN model used the fully connected layers of VGGNet, but now trained on the spectrogram data instead of natural images. The RNN3 and RNN5 models, on the other hand, do not use the fully connected layers but instead two bidirectional long-short term memory (LSTM) layers of 32 and 64 units, respectively, before applying a softmax layer. The difference between RNN3 and RNN5 is that RNN3 uses three consecutive epochs (current-1 to current+1) whereas RNN5 uses five consecutive epochs (current-2 to current+2).

The models were evaluated on EEG recordings from the publicly available Sleep-EDF database [6]. This dataset continues recordings of 20 different subjects, of which 19 were recorded for two nights and one was recorded for one night only. The Fpz-Cz sensor was selected and spectrograms were extracted using multitaper spectral analysis [7]. Following the preprocessing approach in the previous CNN implementation, the spectrograms were subsequently converted into 224 x 224 pixel RGB images for human interpretability by applying the 'jet' colormap in MatLab. The classifier was trained to distinguish between 5 sleep stages: **W** (wake), **N1** (non-REM 1), **N2** (non-REM 2), **N3** (non-REM 3) and **R** (REM).

All models were trained by minimizing the categorical cross-entropy loss function, using the Adam optimizer with a learning rate of 10^{-5} and decay rate of first and second moments set to 0.9 and 0. 999. The CNN model was trained on minibatches of size 70, whereas the RNN3 and RNN5 models were trained on batches of size 16 due to memory constraints.

In order to avoid biases introduced by the class imbalance in the data, the contribution of each class to the loss function was weighted such that the total contribution of each class to the loss function was the same. That way, we were able to train our models on all available training data, instead of a class-balanced subsample as in the previous CNN implementation [2]. Dropout with a rate of 0.5 was applied on the fully connected layers of the CNN model, and on the LSTM layers of RNN3 and RNN5. A 10-fold cross-validation over subjects was used; training on 14 subjects, validating on 4 subjects and testing on 2 subjects in each fold. Folds were randomly generated, but identical between the models to allow for a fair model comparison. Model selection within each training loop was performed based on the validation loss, which was weighted in the same way as the training loss.

Finally, we performed an exploratory sensitivity analysis to gain insight into the input elements affecting the models' decision making during classification. We did this by extracting the gradients of the model outputs with respect to their input images, and depicted the gradients by the mean values across all color channels with the 'jet' colormap.

3. RESULTS

The CNN network was trained for 50 epochs, and the considerably slower RNN models for 10 epochs each. Figure 1 shows training and validation loss as a function of epoch for each model. It is easy to see that the CNN starts overfitting after ~7 epochs, although this should not constitute a major problem for the final results due to the fact that the weights producing the lowest validation error are selected and thus all the later updates are ignored. However, in response to the overfitting we also tried adding *l2* regularization to the fully connected layers (lambda = 0.01). This did not lead to an improvement of the overall results, probably due to a too large choice of regularization constant. Further studies could investigate whether a more moderate regularization can improve classification performance of the CNN. Overfitting does not seem to be a problem for the RNN models, although they were run for considerably fewer epochs due to time constraints. Contrary to the CNN, these models look like they have not fully converged, and thus there could be a potential for improved performance by simply training the models for more epochs.

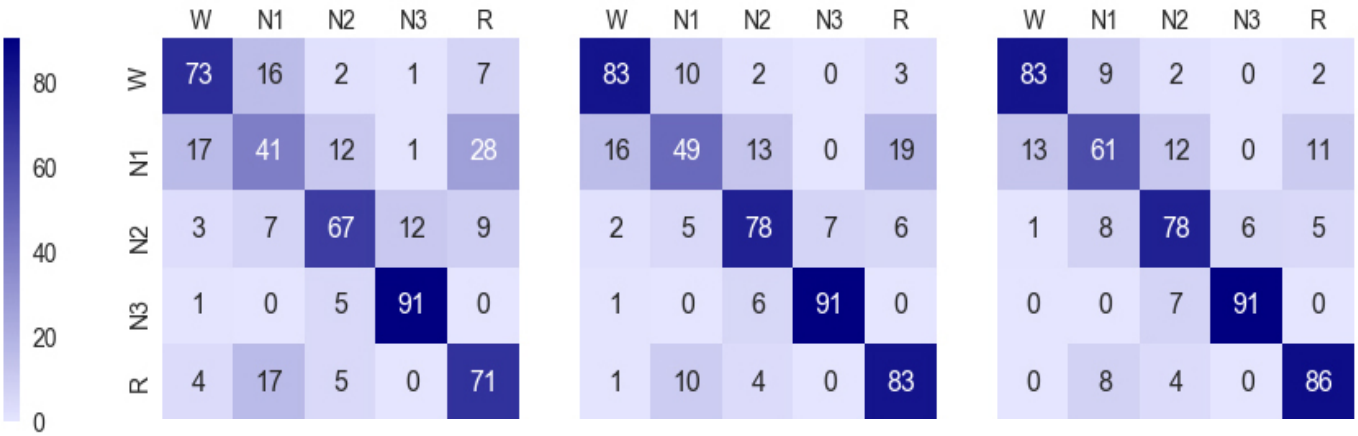

Figure 2. Real (rows) vs. predicted (columns) class labels by model - proportion in %.

The confusion matrices in figure 2 summarize the classification performance of our models. Overall, the CNN predicts most classes correctly, and especially the N3 sleep stage is well-classified by this simple model. As such, the performance of our CNN is only slightly below or on par with the previously implemented CNN, utilizing five stitched-together images to emulate a temporal dimension. Both models show some difficulty distinguishing between the W, N1 and N2 stages, and severe problems when discriminating between N1 and REM sleep. Since N1 and REM sleep tend to be well separated in time, however, simply incorporating temporal dependencies in our models could be expected to help alleviate some of this confusion.

The RNN3 model shows significant improvements in prediction accuracy for all classes except the N3 sleep stage, where it performs on par with the CNN. Of particular interest is the drop by one third in N1 sleep stages misclassified as REM sleep. At this point, our model exceeds the performance of the previously implemented CNN for all classes, despite using fewer input images for each prediction.

This difference is only further exaggerated by the RNN5 model. In this case, our model scores on par with the RNN3 model's best prediction accuracies in the W, N2, and N3 sleep stages. The added temporal information of the RNN5 model, however, further contributes to the prediction accuracy of N1 and REM sleep, where the amount of N1 sleep stages misclassified as REM is reduced by nearly two thirds when compared to the CNN. In this way, the RNN5 model significantly exceeds the performance of our own baseline CNN model, and even surpasses the performance of the previously implemented time-emulating CNN model [2], despite having direct access to the same amount of input images during classification.

To provide a better intuition of the performance differences between these models, figure 3 summarizes and compares the predictions of the baseline CNN and the RNN5 model for the first night of one test subject (subject 11). Although the CNN generally predicts most sleep stages correctly, the figure also reveals that the predictions, when viewed in their immediate temporal context, are very erratic and volatile. In comparison, the RNN5 model predictions tend to be more consistent and stable with fewer sudden fluctuations and abrupt deviations.

These differences in the models' predictive behavior are particularly evident near the end of the night, between the 7- and 8-hour marks, where the CNN misclassifies most of the predominantly N2 sleep as REM, and the RNN more consistently classifies the period correctly. To investigate this critical segment further, we performed a sensitivity analysis highlighting the most influential elements in the input images contributing to the output predictions of our two models.

Figure 4 taps into a particularly indicative transitory stage near the 7.5-hour mark, where the test subject is just reentering N2 sleep after a short period of N1 sleep. As the figure reveals, the CNN model locks onto some anomalous features in the top half of the image and erroneously classifies the image as REM sleep. Meanwhile, the RNN5 model hedges its bets by mostly ignoring the ambiguous features of the current image, instead focusing on the more unequivocal features in the middle and bottom parts of the images to correctly predict the N2 sleep stage.

Not shown in the figure, however, is the predictive behavior of the models in the surrounding images, which is equally symptomatic of their inner architecture. In the very transition from N1 to N2 sleep, for instance, the RNN5 model incorrectly predicts the first few images as N1 by wrongly emphasizing the images prior to the transition in its prediction. After a short delay of misclassifications, however, the RNN5 model quickly locks onto the new sleep stage and consistently predicts N2 correctly for the majority of the remainder of the period.

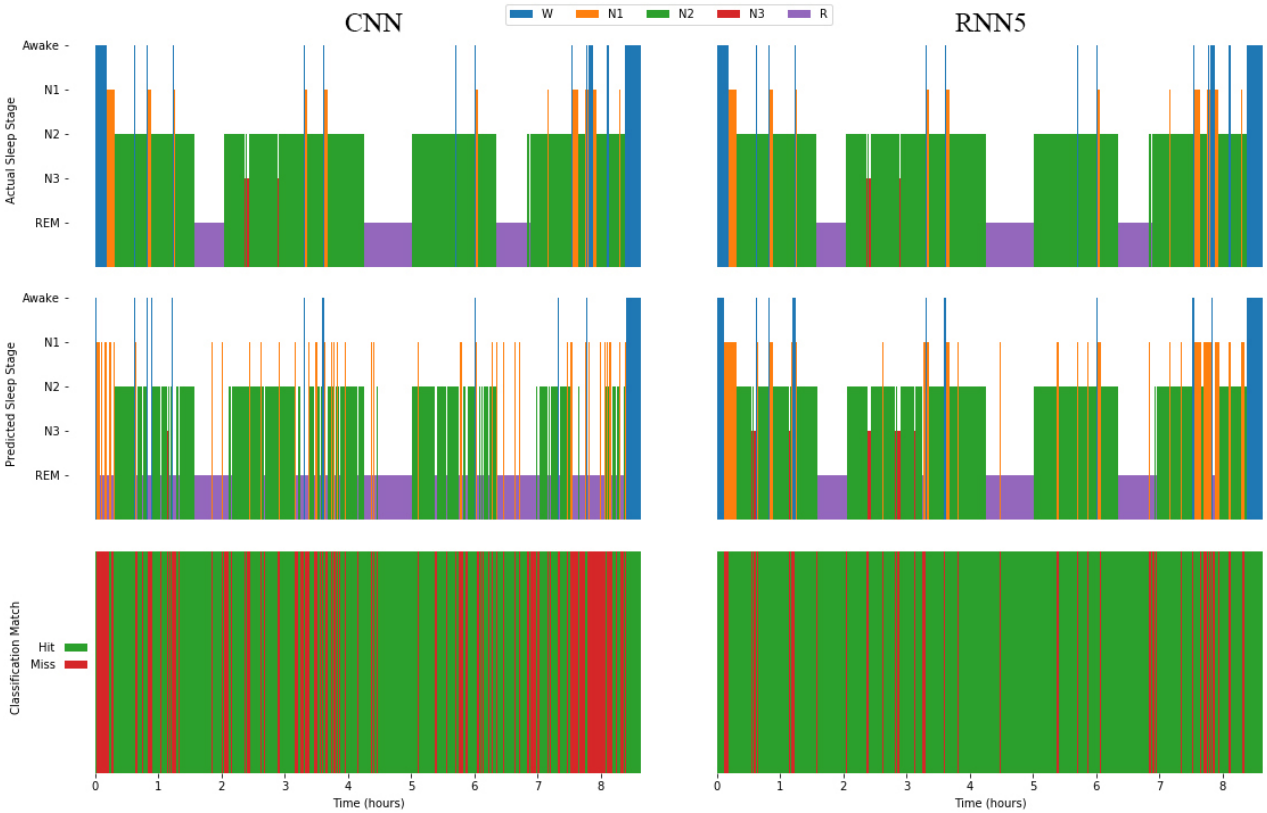

Figure 3. Real (top row) vs predicted (middle row) sleep stage for subject 11, night 1, for CNN (left column) and RNN5. The bottom row summarizes the classification match between the predicted and real sleep stages for the corresponding model and night.

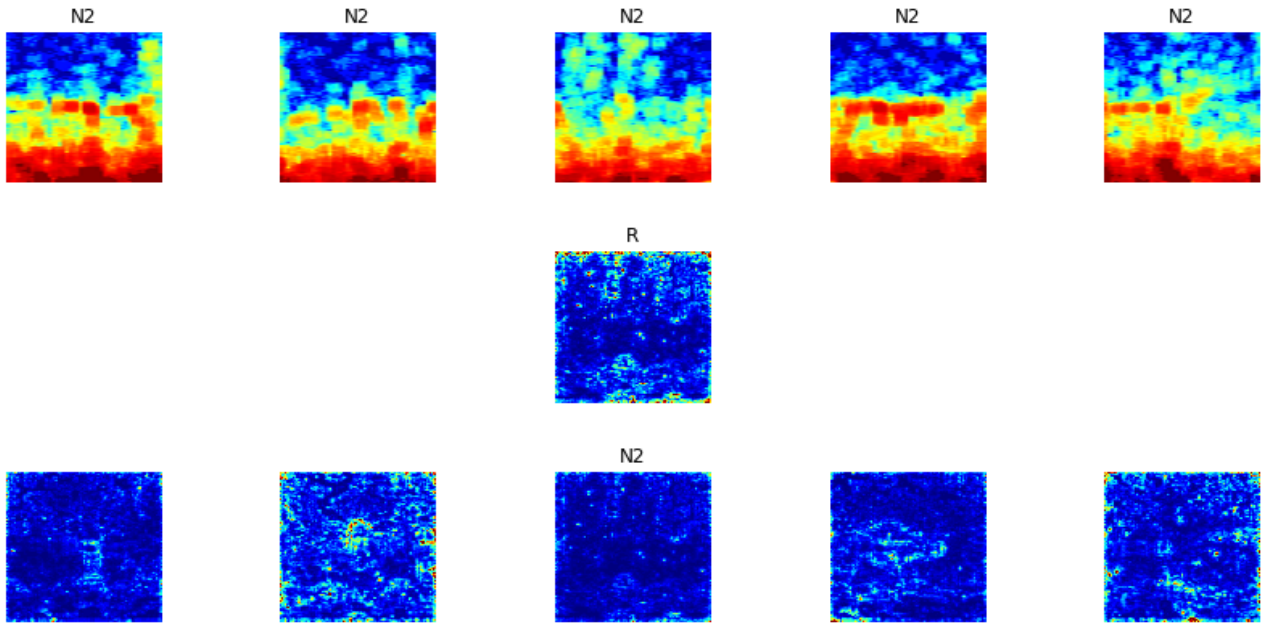

Figure 4. Spectrogram images of five consecutive epochs of N2 sleep (top row), and classifications of the middle image by the CNN (middle row) and RNN5 (bottom row). The classifications are shown with their respective sensitivity maps - the CNN only looking at one image and the RNN5 using all five images.

On the other hand, the CNN never successfully locks onto the correct class, but continuously cycles between N1, N2 and REM in its predictions, depending on the particular aberrations in the current input image. In this way, these fundamental differences in the models' inner architectures may provide a possible explanation to the remarkable divergence in predictive stability and accuracy between the two models when viewed in their temporal context.

4. CONCLUSION

We have implemented three models for automatic sleep stage scoring, using the pre-trained VGGNet for feature extraction of EEG spectrogram images. The first model (CNN), which only uses the features from one epoch at a time, reaches a reasonable performance on some classes but is particularly bad at classifying the N1 stage. Using this model as our baseline, we have investigated the performance gain of adding a recurrent structure to the model, using either three (RNN3) or five (RNN5) consecutive spectrograms to classify one epoch. RNN3 gives a considerable performance boost for four of the five classes, with the fifth class (N3) possibly being at ceiling performance with an accuracy of 91%. RNN5 is the best performing model, matching RNN3 on four of five classes, and significantly increasing classification performance on the troublesome N1 class (accuracy of 49% for RNN3 vs. 61% for RNN5). Besides beating our own models, our RNN5 implementation also constitutes a major improvement compared to the previously published implementation using the pre-trained VGGNet weights [2].

Sensitivity analysis as well as inspection of the predictions made by the respective models on test data shows that the recurrent implementations, and specifically RNN5, give more stable predictions than the CNN, which better capture the time-dependent structure of the data. Recurrent models thus seem to be well-suited for this classification problem. We believe that even more performance gains are possible. A promising option would be to train the convolutional layers and not only the top layers, a strategy that produced substantial performance gains on the troublesome N1 class in the previously published implementation [2]. Apart from this, some hyperparameters - such as learning rate, number of LSTM units in the recurrent layers, and the number of training epochs - could be tuned to further improve performance.

In conclusion, we have demonstrated that combining a CNN designed for image classification can be combined with a RNN to improve on existing models for automatic sleep stage scoring, both with regards to overall classification accuracy and to the plausibility of predictions produced by the model. We believe that our RNN models set out a promising route for automatic sleep scoring models, and that they can be further improved by tuning of the bottom layers as well as by hyperparameter tuning.

5. REFERENCES

[1] T. Andrillon, D. Pressnitzer, D. Léger, and S. Kouider, "Formation and suppression of acoustic memories during human sleep," *Nat. Commun.*, vol. 8, no. 1, Dec. 2017.

[2] A. Vilamala, K. H. Madsen, and L. K. Hansen, "Deep Convolutional Neural Networks for Interpretable Analysis of EEG Sleep Stage Scoring," *ArXiv171000633 Cs Stat*, Oct. 2017.

[3] M. Ronzhina, O. Janoušek, J. Kolářová, M. Nováková, P. Honzík, and I. Provazník, "Sleep scoring using artificial neural networks," *Sleep Med. Rev.*, vol. 16, no. 3, pp. 251–263, Jun. 2012.

[4] F. Ebrahimi, M. Mikaeili, E. Estrada, and H. Nazeran, "Automatic Sleep Stage Classification Based on EEG Signals by Using Neural Networks and Wavelet Packet Coefficients," *Conf. Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. IEEE Eng. Med. Biol. Soc. Conf.*, vol. 2008, pp. 1151–4, Feb. 2008.

[5] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *ArXiv14091556 Cs*, Sep. 2014.

[6] B. Kemp, A. H. Zwinderman, B. Tuk, H. A C Kamphuisen, and J. J L Oberyé, "Analysis of a sleep-dependent neuronal feedback loop: The slow-wave microcontinuity of the EEG," *Biomed. Eng. IEEE Trans. On*, vol. 47, pp. 1185–1194, Oct. 2000.

[7] M. J. Prerau, R. E. Brown, M. T. Bianchi, J. M. Ellenbogen, and P. L. Purdon, "Sleep Neurophysiological Dynamics Through the Lens of Multitaper Spectral Analysis," *Physiology*, vol. 32, no. 1, pp. 60–92, Dec. 2016.