






# Automatic Classification of Sleep Stages Based on Raw Single-Channel EEG

Kailin Xu<sup>1,2</sup> , Siyu Xia<sup>1</sup> , and Guang Li<sup>2</sup> 

<sup>1</sup> Southeast University, Nanjing 210096, People's Republic of China  
xia081@gmail.com

<sup>2</sup> Zhejiang University, Hangzhou 310027, People's Republic of China  
guangli@zju.edu.cn

**Abstract.** Electroencephalogram (EEG) is a common signal for monitoring people's sleep quality. Manual sleep stage classification on EEG is a time-consuming task. In this paper, we design a model for automatic sleep stage classification based on raw single-channel EEG. This model can preserve the information, broaden the network and enlarge the receptive field as much as possible to extract appropriate time invariant features and classify sleep stage well. For the class-imbalanced problem in sleep stage classification, most of the existing methods rely on cross entropy loss and adjust model hyperparameters by experience, leading to poor performance. We implement a two-step training algorithm. The first is pre-training the model with the hyperparameters obtained by Bayesian Optimization after rebalancing datasets by over-sampling. The second is using feedback loss in model fine-tuning to reduce the impact of class-imbalanced problem. The loss weights dynamically change with the per-class F1-score which is used as feedback information. We evaluate our method on Fpz-Cz channel from the Sleep-EDF dataset. The overall accuracy, macro F1-score, Cohen's Kappa coefficient are 85.53%, 81.18%, 0.80 respectively, showing our method has better classification performance than the state-of-the-art methods and is an efficient tool for automatic sleep stage classification.

**Keywords:** Automatic sleep stage classification · Raw single-channel EEG · Deep learning · Feedback loss

## 1 Introduction

Sleep disorders are common in people and can lead to serious health problems that affect the quality of life [1]. Monitoring people's sleep quality has important implications for medical research and practice [2].

A polysomnography (PSG) records the physiological signals of a subject during sleep at night, which is composed by multiple signals such as electroencephalogram (EEG), electrocardiogram (ECG), electrooculogram (EOG),

---

The first author is a student.

and electromyography (EMG) [3]. According to American Academy of Sleep Medicine (AASM) [4], sleep stages can be divided into wake (W), three non-rapid eye movement (NREM) stages (N1-N3), and rapid eye movement (REM). And stage N3 (also called Slow Wave Sleep) is divided into two distinct stages, N3 and N4 in Rechtschaffen and Kales (R&K) [5]. Most PSG recordings last at least eight hours. For sleep experts, manual sleep stage classification in such a long signal is a tedious task and highly dependent on the appropriate inter-rater agreement. Therefore, it is important to classify sleep stage automatically, which can avoid the human subjective bias in classification.

Automated sleep stage classification algorithms can be divided into two categories: the hand-engineered feature-based methods and the automated feature extraction-based methods. For the first category, methods extract features such as time, frequency and time-frequency domain features [6–10] for training. Because these methods only extracts features, they may lose most of the original information. As a result, these methods do not generalize well, especially given the nature of PSG recordings, where variability effects are caused by a number of factors, including patient and hardware differences, etc. For the second category, methods learn directly from the raw data, which may solve the limitation in handcrafted feature extraction. Recently, because some neural networks can be trained and optimized end-to-end, they are used both as feature extractors and classifiers. For example, [11] build model with stacked sparse autoencoders, [12] build model with convolutional neural networks. [13] build model with convolutional neural network and bidirectional recurrent neural network.

Considering the number of channels for neural network’s input, we use single EEG channel which is cheap and ensures the subjects’ sleep does not be affected by the instruments. In the methods investigated, the accuracy of sleep stage classification is not high for the method based on raw single-channel signal, especially for the Sleep-EDF dataset. Besides, the macro F1-score (MF1) and Cohen’s Kappa coefficient ( $\kappa$ ) do not exceed 0.8 in most studys [11–14], due to the serious imbalance of the dataset and the authors might do not pay more attention to the process of model optimization and use the loss which is not particularly suitable for imbalanced datasets.

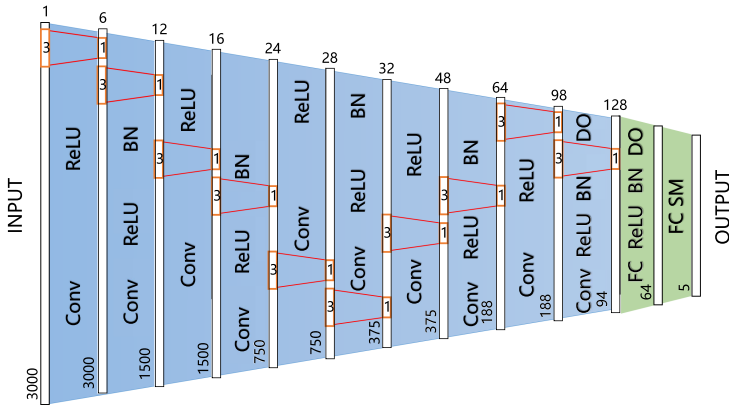
At present, there is little methods using Bayesian Optimization [15] in sleep stage classification. However, Bayesian Optimization can find the optimal hyperparameter according to the previous hyperparameter adjustment results, which has been successfully applied in the machine learning methods [16–19]. In addition, using the idea of back propagation in network training as reference, we propose feedback loss by employing the per-class F1-score of the model as feedback information to adjust the penalty weight of the loss function constantly.

Based on the two points above, we design a deep learning model with a two-step training method, including pre-training network with hyperparameters found by Bayesian Optimization and fine-tuning with feedback loss. This model can preserve the information, increase the number of network channels and enlarge the receptive field to the utmost extent while extracting time invariant features and being trained to classify the sleep stage. Through experiments, the two-step training method can effectively train our model end-to-end through back propagation, and feedback loss can make MF1 and  $\kappa$  exceed 0.8, decreasing bias towards the majority class caused by imbalanced datasets. Moreover, the model can be automatically trained without any hand-engineered features. It is important to note that feedback loss can be used in multiple fields where imbalanced dataset exists, besides sleep stage classification.

## 2 Methods

### 2.1 Model Architecture

The architecture of our model (see Fig. 1) consists of ten convolutional layers and two fully-connected layers. Small kernels of size  $3 \times 1$  are used in every convolutional layer. In view of the rapid fluctuation characteristic of EEG signal, small convolution kernels can extract the subtle information, and realize the suppression of noise through the convolutional layer combination, because the bigger kernels such as  $5 \times 1$  and  $7 \times 1$  can be replaced by the combination of small kernels [20]. Moreover, the use of small convolution kernels can reduce the parameters and increase the nonlinearity of model while ensuring the receptive



**Fig. 1.** Architecture of our model. Conv, Relu, BN, DO, FC and SM mean convolutional layer, rectified linear unit, batch normalization, dropout, fully-connected layer and softmax severally. The use order of the units in each layer is from bottom to top. The number at the bottom represents feature size, and the number at the top represents the number of channels.

field. Every two adjacent convolutional layers are a block, and perform five operations sequentially: convolution, applying the rectified linear unit (ReLU) [21] activation, convolution, applying ReLU activation, batch normalization [22]. The first convolution applies stride 1 and padding to ensure the input feature has the same size compared with output feature when the network width increases. The second convolution applies stride 2 and padding to take place of pooling [23]. In a block, using convolutional layers with stride 1 and stride 2 together can effectively increase the receptive field and broaden the network without losing much information. The adoption of batch normalization and ReLU activation can accelerate model convergence and prevent vanishing gradient. A regularization technique named dropout [24] is used to alleviate overfitting problems and will be removed from the model during testing to provide certain outputs.

## 2.2 Pre-training

In pre-training, the first step is replicating the minority class samples in the original training set until all sleep stages have the same number of samples to reduce the harm of skewed distribution. Then we train model on balanced dataset using a gradient-based optimizer called Adam [25] with hyperparameters found by Bayesian Optimization. Bayesian Optimization is a method for hyperparameter tuning, which searches for the optimal hyperparameter based on the previous results, leading to better performance than human expert-level for many algorithms [15]. In our algorithm, this method is used to adjust the hyperparameters of two dropout layers which means the rate at randomly dropping units along with their connections from the neural network during training. This hyperparameter is very important. If the hyperparameter is too large, only a small part of the model is left in the training process. If the hyperparameter is too small, the regularization effect is not obvious, and the model is still serious overfitting.

## 2.3 Fine-Tuning with Feedback Loss

In sleep stage classification, the class-imbalanced problem is serious but most methods do not choose to improve the loss function for solving it, resulting in low values of MF1 and  $\kappa$ . For methods test on Sleep-EDF dataset, the values mostly lower than 0.8 [11–14]. As a result, we specifically design a new algorithm to calculate loss named feedback loss, based on the thought of back propagation. The following is a detailed description (see Algorithm 1).

**Algorithm 1.** Fine-Tuning with Feedback Loss

---

**Input:** *model, data, target, multiplier*  
**Output:** *model*

```

for  $i = 1$  to  $n$  do
   $output = model(data)$ 
   $f_{score}, mean, standard\_deviation = compute(output, target)$ 
   $f_{score} = (f_{score} - mean) / standard\_deviation$ 
   $weight = \max(-f_{score} + 1, multiplier \times f_{score} + 1)$ 
   $FeedbackLoss = CrossEntropyLoss(weight)$ 
  for  $j = 1$  to  $n\_below$  do
     $output = model(data)$ 
     $loss = FeedbackLoss(output, target)$ 
     $model = get\_model(model, loss)$ 
  end for
end for
return model

```

---

For the convenience, we introduce the process for one class that is identical to the processes for other classes. (a) Put *data* into *model* to obtain *output*, then compare the *output* and *target* by *compute* to obtain F1-score *f<sub>score</sub>* for the class, and the *mean* and *standard deviation* for all per-class F1-scores; (b) Centralize *f<sub>score</sub>* with zero-mean by the equation as follows:

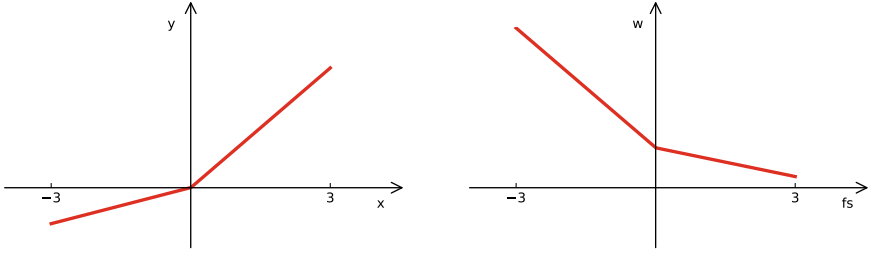
$$f_s = \frac{f_s - m_f}{s_f} \quad (1)$$

where *f<sub>s</sub>*, *m<sub>f</sub>* and *s<sub>f</sub>* are the *f<sub>score</sub>*, *mean* and *standard deviation* respectively; (c) Obtain *weight* by the equation as follows:

$$w = \max(-f_s + 1, k_f f_s + 1) \quad (2)$$

where *w* is *weight* of the class. *k<sub>f</sub>* is *multiplier* of the F1-score, and the value lies between  $-0.25$  and  $0$ ; (d) Reload *weight* as penalty weight on *CrossEntropyLoss* to obtain *FeedbackLoss*; (e) Use *FeedbackLoss* to obtain *loss* with *output* and *target* and optimize model parameters by *get\_model* with *loss* for *n<sub>below</sub>* epochs; (f) Return *model* after running step a,b,c,d,e for *n* times.

Feedback loss has the following characteristics. The first is selecting F1-score as feedback information. In imbalanced dataset, the accuracies of minority classes is easy to be affected by majority classes, resulting in there is not much difference between the accuracies of minority classes and majority classes. For recall and precision, both of them are important and an increase in one leads to a decrease in the other. As a result, F1-score is selected, because it is a combination of recall and precision. The second is using centralizing F1-score with zero-mean (see Eq. 1) and the computing method of weight (see Eq. 2) together. For the F1-scores, the sum of the positive values is equal to the negative value after centralizing. The method of getting weight by F1-score comes from Leaky ReLU (see Fig. 2). For a class with good recognition results, its F1-score is positive after centralizing. The slow weight adjustment with  $\frac{dw}{df_s} = k_f$  leads to the suppression on effects of the class. In contrast, for a class with poor recognition results,



**Fig. 2.** Comparison between Leaky ReLU (left) and the calculation of weight in feedback loss (see Eq. 2) (right). For the important portion, like the positive part of  $x$  and the negative part of  $fs$ , the slope of curve stays the same, the absolute value is 1. For the unimportant portion, like the negative part of  $x$  and the positive part of  $fs$ , the slope of curve is small.

its F1-score is negative after centralizing. The rapid weight adjustment with  $\frac{dw}{df_s} = -1$  and the gradient of weight will not decrease with the iteration, leading to the constant attention on effects of the class. This method can ignore the effects of the classes identified well appropriately and focus on classes identified badly, resulting in the improvement of MF1 and  $\kappa$ . The third is that the penalty weight is adjusted by the feedback repeatedly in the process of training and the latest weight is used to calculate feedback loss for model optimization, ensuring the model has a good performance. By the above statement, we find feedback loss can apply to most class-imbalanced problems.

## 3 Results

### 3.1 Data

In this study, two versions of the Sleep-EDF dataset [26,27] are used. The first version (Sleep-EDF-13) is contributed in 2013 with 61 polysomnograms, and the second version (Sleep-EDF-18) is contributed in 2018 with 197 polysomnograms. Both of them have two studies about the age effects on sleep in healthy individuals (SC = Sleep Cassette) and the temazepam effects on sleep in individuals with mild difficulty falling asleep (ST = Sleep Telemetry) separately. To get closer to normal, we choose the data in SC studies. Files in SC studies are obtained in healthy Caucasians aged 25–101, without any sleep-related medication. Each PSG lasts nearly 20h, and contains two EEG signals from Fpz-Cz and Pz-Oz electrode locations, one EOG signal, one EMG signal and one event marker. Every two recordings are collected during two subsequent day-night periods on a subject. The EOG and EEG signals are sampled at 100 Hz. According to the R&K standard, sleep experts manually classify the recordings into one of the six classes, W, N1, N2, N3, N4, REM. The N3 and N4 classes are combined in one class named N3 on the basis of AASM. Because each SC file lasts nearly 20h and we mainly focus on the sleep period, only the recordings between half hour

before and after the sleep periods are retained. Table 1 presents the number of 30-s epochs and proportion for each sleep stage in two different versions.

**Table 1.** Number of 30-s epochs for sleep stages in two versions of Sleep-EDF.

Dataset	Type	W	N1	N2	N3	R
Sleep-EDF-13	Number	8,285	2,804	17,799	5,703	7,717
	Proportion (%)	19.58	6.63	42.07	13.48	18.24
Sleep-EDF-18	Number	65,951	21,522	69,132	13,039	25,835
	Proportion (%)	33.74	11.01	35.37	6.67	13.22

For Sleep-EDF-13 dataset, we test our method using  $k$ -fold cross-validation, where  $k$  was set to 14. In each fold,  $n_f \mid k$  recordings are selected for testing model, and the remaining recordings are taken to train model, where  $n_f$  is the number of recordings in the dataset. This process is repeated  $k$  times. Then we combine the predicted sleep stages from all folds and compute the evaluation metrics. Sleep-EDF-18 has 153 recordings, from which 142 recordings are selected for training network, 8 recordings are validation set, and the remaining 3 recordings are test set. It is important to note that the training set, validation set and test set are divided into subjects, ensuring test subjects' epochs do not appear in training set, so we can verify our method on the unknown subjects. If all recordings are mixed before testing, the data of the same subject will appear in the training set and the test set. Although the model's performance improves [15], its practicability reduces. Our division ensures that the data of the same subject will not appear in training set and test set at the same time.

### 3.2 Experimental Design

In the pre-training, the Adam optimizer's learning rate is 0.001, and 30-s epoch is taken as a sample. After shuffling the oversampled dataset, 128 samples are loaded into the model as a mini-batch. The cost function is multiclass cross-entropy. For hyperparameter tuning, we randomly search for five hyperparameter combinations at first, then use Bayesian Optimization, setting (0.3, 0.8) as the search space for hyperparameter of every dropout layer and choosing Gaussian Process [28] as internal regressor which is trained with training set comprised by the previous results. After that, search for multiple hyperparameter combinations by two parts: randomly initializing multiple combinations and searching for more combinations using L-BFGS-B [29]. Next, the mean  $m_p$  and standard deviation  $s_p$  of each combination are obtained by using regressor. Obtain the fitted value  $v$  at each combination by the equation as follows:

$$v = m_p + k_p s_p \quad (3)$$

$k_p$  is enlargement factor. Then return the combination corresponding to the maximum value which is the new hyperparameter combination. We repeat this

search operation 15 times. In the fine-tuning, the Adam optimizer’s parameter learning rate, mini-batch size are set 0.0002, 128 respectively. In addition, *multiplier*, *n\_below*, *n* are  $-0.24$ , 8 and 6 separately.

### 3.3 Evaluation Metrics

Different metrics are used to evaluate the performance of our approach including per-class recall, overall accuracy (ACC), MF1 and  $\kappa$ .

$$ACC = \frac{\sum_{c=1}^N TP_c}{TF} \quad (4)$$

$$MF1 = \frac{\sum_{c=1}^N FS_c}{N} \quad (5)$$

$TP_c$  is the true positive epoches of class  $c$ ,  $TF$  is the number of epoches in the dataset,  $FS_c$  is the F1-score of class  $c$ ,  $N$  is the number of classes.

### 3.4 Sleep Stage Classification Performance and Comparison

Table 2 shows confusion matrices obtained from Sleep-EDF-13 and Sleep-EDF-18 datasets respectively. It can be seen that true positive values in the main diagonals are higher than other values in the same rows and columns, meaning our method can accurately identify each classes in most case. Table 3 shows the comparison of our method with other state-of-the-art methods across overall accuracy, MF1 and  $\kappa$ . In terms of overall accuracy, our study performs better than the state-of-the-art algorithms compared. Moreover, for MF1 and  $\kappa$ , our method reaches the highest level, indicating that feedback loss is highly applicable to the current imbalanced dataset. In Sleep-EDF-13,  $\kappa$  reaches 0.80 (between 0.8 and 1), indicating almost complete agreement between the sleep experts and our method, and  $\kappa$  reaches 0.78 in Sleep-EDF-18 (between 0.61 and 0.80), indicating the agreement between the sleep experts and our method are substantial [30].

**Table 2.** Confusion matrix achieved by the proposed method.

True	Predicted (Sleep-EDF-13)					Predicted (Sleep-EDF-18)				
	W	N1	N2	N3	R	W	N1	N2	N3	R
W	<b>4482</b>	437	36	12	112	<b>1455</b>	128	3	1	4
N1	193	<b>1293</b>	278	3	345	39	<b>161</b>	64	0	25
N2	50	542	<b>11056</b>	371	608	2	102	<b>1021</b>	129	28
N3	2	5	225	<b>3216</b>	1	0	0	16	<b>369</b>	0
R	98	558	254	0	<b>4367</b>	20	97	4	0	<b>428</b>

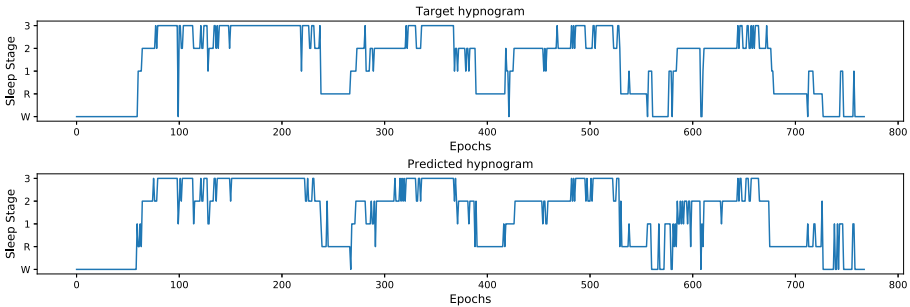


**Table 3.** Classification performance comparison of our method with other methods by evaluation metrics.

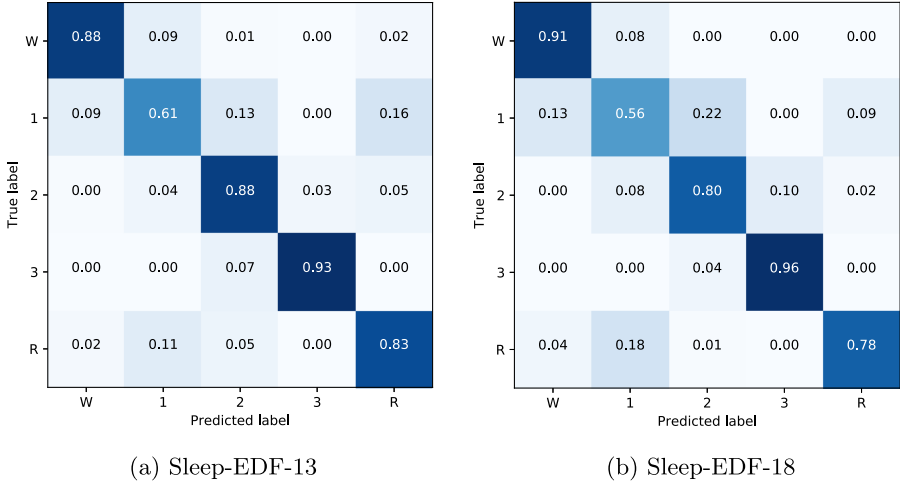
Dataset	Method	ACC (%)	MF1 (%)	$\kappa$
Sleep-EDF-13	SleepEEGNet [14]	84.26	79.66	0.79
	DeepSleepNet [13]	82.0	76.9	0.76
	Tsinalis et al. [11]	78.9	73.7	
	Tsinalis et al. [12]	74.8	69.8	
	<b>This study</b>	<b>85.53</b>	<b>81.18</b>	<b>0.80</b>
Sleep-EDF-18	SleepEEGNet [14]	80.03	73.55	0.73
	<b>This study</b>	<b>83.84</b>	<b>77.36</b>	<b>0.78</b>

### 3.5 Method Analysis

In order to see the difference between the predictions of our method and the labels, we select one file of test dataset named SC4001E0, and draw the predicted hypnogram and target hypnogram. We can find our method’s judgments are the same as the labels in most epochs (see Fig. 3). For understanding the confusion matrix in Table 2 better, we visualize them (see Fig. 4). The value in the cell is the recall, the ratio of the number in corresponding cell of the confusion matrix to the number of 30-s epochs for corresponding sleep stage. For the recall values in the cells, there is a no identification error in many pairs including W-N3, N1-N3 and N3-R. In addition, the confusion matrix is almost symmetric across the diagonal proving class-imbalanced problem is eliminated to some extent.



**Fig. 3.** Comparison of the target hypnogram (top) with the predicted hypnogram (bottom) in SC4001E0. The overall accuracy, MF1,  $\kappa$ , and the recall of N1 stage reach 87.63%, 83.32%, 0.84, and 60.38% respectively.



**Fig. 4.** Confusion matrix visualization based on recall. For each square, the larger the number, the darker the color. In both datasets, it can be seen the worst performance is noted for N1 stage, and the N3 stage reaches the best performance.

## 4 Discussion

As for the recognition difficulty of N1 stage, after analyzing the confusion matrix and consulting relevant information, we find that the reason for this result lies in the data itself. The first reason is the number of N1 stage is small compared with the other sleep stages. For example, the proportion of N1 stage in Sleep-EDF-13 and Sleep-EDF-18 are 6.63% and 11.01% respectively. The second reason is N1 stage is easy to confuse with REM. Because  $\theta$  wave (frequency ranges from 4 hz to 8 hz) occurs only during REM stage and N1 stage [31], making the characteristics of N1 stage and REM stage are similar, leading to the failure of classifier in distinguishing the two stages.

In general, the larger the dataset, the better the performance of the classifier. However, we test our method on Sleep-EDF-13 and Sleep-EDF-18, finding this is not the case. Same problem appears in other study [32]. Therefore, the labels of Sleep-EDF-18 dataset may not be accurate enough due to the heavy workload of expert manual annotation. For  $\kappa$ , [33] find it between 0.48 and 0.89 by studying the agreement between two experts. Similarly, [34] find it between 0.72 and 0.85. These studies prove that manual classification is defective. As the era progresses, the manual classification may lag behind the automatic classification.

The class-imbalance data is particularly common in many fields, and the same situation appears in this study. We use two methods to solve this problem. The first is to simply copy the samples of minority classes to make them reach the same number as the majority classes. The second is mainly through the design of feedback loss. Except for these methods, we have tried other methods, such as random under-sampling and SMOTE over-sampling, but the performance of

our method does not improve. There are many studies on the improvement of loss algorithm, including focal loss [35], dice loss [36], etc. We can combine their characteristics with feedback loss. For the current feedback loss, we directly use the F1-score as feedback information to adjust the weight. However, PID control is widely used in the feedback regulation. If we regard the error between the current F1-score and the expected F1-score as feedback information, put it into the PID controller [37] and use the output as penalty weight for feedback loss, our method may improve unexpectedly.

## 5 Conclusion and Future Work

We propose a deep learning model which can extract time invariant features and classify sleep stage under retaining information, widening the network and increasing the receptive field. We also implement a two-step training algorithm: pre-training model on oversampled datasets with model hyperparameters adjusted by Bayesian Optimization, and proposing feedback loss in fine-tuning to alleviate class-imbalanced problem. Experimental results show that our method outperforms the state-of-the-art methods on the sleep stage classification task. Since our model automatically learns from the original EEG, we believe that our method is a better way to implement sleep stage classification than the hand-engineering methods. When developing an automated system, imbalanced dataset often occurs, such as the arrhythmia detection by ECG and the epilepsy detection by EEG, and feedback loss can make a contribution in these areas.

## References

1. Panossian, L., Avidan, A.: Review of sleep disorders. *Med. Clin. N. Am.* **93**(2), 407–425 (2009)
2. Wulff, K., Gatti, S., Wettstein, J., Foster, R.: Sleep and circadian rhythm disruption in psychiatric and neurodegenerative disease perspectives. *Nat. Rev. Neurosci.* **11**(8), 589–99 (2010)
3. Alickovic, E., Subasi, A.: Ensemble SVM method for automatic sleep stage classification. *IEEE Trans. Instrum. Measure.* **67**(6), 1258–1265 (2018)
4. Berry, R., et al.: AASM scoring manual updates for 2017 (version 2.4). *Journal of clinical sleep medicine : JCSM : official publication of the American Academy of Sleep Medicine* 13 (04 2017)
5. Hobson, J.A.: A manual of standardized terminology, techniques and scoring system for sleep stages of human subjects. *Electroencephalogr. Clin. Neurophysiol.* **26**(6), 644 (1969)
6. Chen, J., Valehi, A., Razi, A.: Predictive modeling of biomedical signals using controlled spatial transformation. [arXiv: Signal Processing](#) (2018)
7. Liu, Z., Sun, J., Zhang, Y., Rolfe, P.: Sleep staging from the eeg signal using multi-domain feature extraction. *Biomed. Sig. Process. Control* **30**, 86–97 (2016)
8. Maity, A., et al.: Multifractal detrended fluctuation analysis of alpha and theta EEG rhythms with musical stimuli. *Chaos Solitons Fractals* **81**, 52–67 (2015)

9. Shayegh, F., Sadri, S., Amirfattahi, R., Ansari-Asl, K.: A model-based method for computation of correlation dimension, lyapunov exponents and synchronization from depth-EEG signals. *Comput. Meth. Programs Biomed.* **113**, 323–337 (2013)
10. Zaeri-Amirani, M., Afghah, F., Mousavi, S.: A feature selection method based on shapley value to false alarm reduction in ICUS a genetic-algorithm approach. In: 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 319–323 (2018)
11. Tsinalis, O., Matthews, P.M., Guo, Y.: Automatic sleep stage scoring using time-frequency analysis and stacked sparse autoencoders. *Ann. Biomed. Eng.* **44**(5), 1587–1597 (2016)
12. Tsinalis, O., Matthews, P.M., Guo, Y., Zafeiriou, S.: Automatic sleep stage scoring with single-channel EEG using convolutional neural networks. *ArXiv* (2016)
13. Supratak, A., Dong, H., Wu, C., Guo, Y.: Deepsleepnet: a model for automatic sleep stage scoring based on raw single-channel EEG. *IEEE Trans. Neural Syst. Rehabil. Eng.* **25**(11), 1998–2008 (2017)
14. Mousavi, S., Afghah, F., Acharya, U.R.: SleepEEGNet: automated sleep stage scoring with sequence to sequence deep learning approach. *PLOS ONE* **14**, e0216456 (2019)
15. Snoek, J., Larochelle, H., Adams, R.P.: Practical Bayesian optimization of machine learning algorithms. *Adv. Neural Inf. Process. Syst.* **4**, 2951–2959 (2012)
16. Al-Zuhairi, M., Pradhan, B., Lee, S.: Application of convolutional neural networks featuring Bayesian optimization for landslide susceptibility assessment. *CATENA* **86**, 104249 (2019)
17. Frean, M., Boyle, P.: Using Gaussian processes to optimize expensive functions. In: Wobcke, W., Zhang, M. (eds.) *AI 2008. LNCS (LNAI)*, vol. 5360, pp. 258–267. Springer, Heidelberg (2008). [https://doi.org/10.1007/978-3-540-89378-3\\_25](https://doi.org/10.1007/978-3-540-89378-3_25)
18. Liang, X.: Image-based post-disaster inspection of reinforced concrete bridge systems using deep learning with Bayesian optimization. *Comput. Aid. Civ. Infrastruct. Eng.* **34**(5), 415–430 (2019)
19. Martinez-Cantin, R., Freitas, N., Brochu, E., Castellanos, J., Doucet, A.: A Bayesian exploration-exploitation approach for optimal online sensing and planning with a visually guided mobile robot. *Auton. Robots* **27**, 93–103 (2009)
20. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. *arXiv 1409.1556* (2014)
21. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: *Proceedings of the 27th International Conference on Machine Learning*, pp. 807–814 (2010)
22. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift (2015). [arXiv: Learning](https://arxiv.org/abs/1502.03197)
23. Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.: Striving for simplicity: the all convolutional net (2014). [arXiv: Learning](https://arxiv.org/abs/1411.2524)
24. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**, 1929–1958 (2014)
25. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2014). [arXiv: Learning](https://arxiv.org/abs/1412.0441)
26. Goldberger, A.L., et al.: Physiobank, physiokit, and physionet components of a new research resource for complex physiologic signals. *Circulation* **101**(23), 215–220 (2000)

27. Kemp, B., Zwinderman, A.H., Tuk, B., Kamphuisen, H.A.C., Obery, J.J.: Analysis of a sleep-dependent neuronal feedback loop: the slow-wave microcontinuity of the EEG. *IEEE Trans. Biomed. Eng.* **47**(9), 1185–1194 (2000)
28. Ounpraseuth, S.: Gaussian processes for machine learning. *J. Am. Stat. Assoc.* **103**, 429–429 (2008)
29. Morales, J., Nocedal, J.: Remark on “algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound constrained optimization”. *ACM Trans. Math. Softw.* **38**(7), 1–4 (2011)
30. Hassan, A.R., Subasi, A.: A decision support system for automated identification of sleep stages from single-channel EEG signals. *Knowl. Based Syst.* **128**, 115–124 (2017)
31. Cheng, J.: Research of Sleep Staging Based on EEG Signals. Ph.D. thesis, Beijing Institute of Technology (2015)
32. Sors, A., Bonnet, S., Mirek, S., Vercueil, L., Payen, J.: A convolutional neural network for sleep stage scoring from raw single-channel EEG. *Biomed. Sig. Process. Control* **42**, 107–114 (2018)
33. Stepnowsky, C., Levendowski, D., Popovic, D., Ayappa, I., Rapoport, D.: Scoring accuracy of automated sleep staging from a bipolar electroocular recording compared to manual scoring by multiple raters. *Sleep Med.* **14**(11), 1199–1207 (2013)
34. Wang, Y., Loparo, K., Kelly, M., Kalpan, R.: Evaluation of an automated single-channel sleep staging algorithm. *Nat. Sci. Sleep* **7**, 101–111 (2015)
35. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollar, P.: Focal loss for dense object detection. In: *The IEEE International Conference on Computer Vision (ICCV)*, pp. 2999–3007, October 2017
36. Li, X., Sun, X., Meng, Y., Liang, J., Wu, F., Li, J.: Dice loss for data-imbalanced NLP tasks. *ArXiv* (2019)
37. Qin, Y., Sun, L., Hua, Q., Liu, P.: A fuzzy adaptive PID controller design for fuel cell power plant. *Sustainability* **10**(7), 2438 (2018)