Proceedings of the 2nd International Conference on Software Engineering and Machine Learning
DOI: 10.54254/2755-2721/74/20240451

A convolutional transformer for EEG sleep stage classification

Zhuoran Li

Computer Science and Technology (Language Intelligence and Technology), College of Information Science, Beijing Language and Culture University

xk20lzr@163.com

Abstract. Sleep is one of the most critical physiological processes in life, with profound implications for human health and functioning. Sleep stage classification is a fundamental task in sleep research, aiding in the understanding of the various stages of sleep and playing a crucial role in medical and health monitoring. Traditional Convolutional Neural Networks (CNNs) have limitations in handling long-range dependencies, focusing only on information within specific windows. For example, missing transitions between Rapid Eye Movement (REM) sleep stages and Non-Rapid Eye Movement (NREM) sleep stages. RNN may not effectively capture local features. For instance, RNNs might not effectively capture short awakenings or microarousals during sleep, which are crucial for accurate sleep stage classification. Subsequently, one limitation of transformers is their inefficiency in handling sequential data with long-range dependencies. To address these issues, we propose the Conformer model, which combines convolutional neural networks and transformers to efficiently capture both local and global dependencies in an electroencephalogram sequence. Conformer models employ various parameter reduction strategies, making the model more lightweight and suitable for embedded devices and low-resource environments. To validate our approach, we conducted comprehensive experiments using a widely collected single-channel EEG dataset and evaluated the model's performance. When compared to traditional methods and other deep learning approaches, our model demonstrates a significant advantage in capturing subtle features and patterns associated with different sleep stages.

Keywords: Sleep stage classification, Attention, Convolutional neural networks, Transformer

1. Introduction

Sleep is an indispensable facet of human existence, wielding profound influence over our physical and mental well-being [1]. Beyond its scientific fascination, the study of sleep and its various stages holds immense clinical significance. Conventional methods for classifying sleep stages predominantly rely on laborious manual assessments of polysomnography (PSG) records [2], a process encompassing multiple physiological signals, including electroencephalogram (EEG) [3], electrooculogram (EOG) [4], and electromyogram (EMG) [5].

In recent times, the advantage of deep learning has transformed numerous domains, spanning computer vision [6], natural language processing [7], and speech recognition. Demonstrating exceptional prowess in automating intricate tasks through hierarchical data representations, deep learning stands poised to revolutionize the field of sleep stage classification. Of particular note is its

DOI: 10.54254/2755-2721/74/20240451

potential to revolutionize the monitoring of sleep patterns, particularly when leveraging single-channel EEG data [3], thereby rendering it more accessible and efficient.

This work is motivated by the task of introducing the Conformer model as a potent solution for processing EEG signals [8], with the overarching aim of advancing the field of sleep stage classification. The Conformer model, a fusion of self-attention mechanisms and convolutional layers, emerges as an ideal candidate for processing sequential data such as EEG signals. While self-attention endows the model with the capacity to comprehend global interactions, convolutions adeptly capture local correlations contingent upon relative offsets. This combination of architectural elements offers a promising avenue to enhance the affordability and accessibility of sleep monitoring.

In light of existing challenges, traditional Convolutional Neural Networks (CNNs) [9] and Recurrent Neural Networks (RNNs) [10] exhibit limitations in modeling long-range dependencies within EEG data. By contrast, the Conformer model employs a suite of parameter-reduction strategies, thereby conferring a lightweight profile conducive to deployment in embedded devices and resource-constrained environments.

As a pivotal contribution, this work presents the Conformer model, characterized by its unique architecture that addresses the deficiencies of previous methods. This innovative model possesses the potential to redefine the landscape of sleep stage classification, paving the way for more accurate and resource-efficient monitoring of sleep patterns.

In the following sections, we propose a comprehensive statement of our methodology, expound upon the experimental results, and engage in in-depth discussions, ultimately shedding light on the efficacy of our Conformer-based solution in augmenting both accuracy and efficiency in the classification of sleep stages.

2. Conformer Encoder

We propose a novel architecture using conformer for automatic sleep stages classification. Its operational workflow can be summarized as follows. Its key feature lies in the stacking of multiple Conformer encoder layers [8]. Each encoder layer comprises a series of essential steps. Each module within the Conformer encoder plays a distinct role in the sequence modeling process. The Multi-head Attention Module (MHA) captures context and dependencies, normalization and residual connections ensure stable training, the Convolution Module (Conv) extracts local features, and the Feed Forward Networks Module (FFN) enables non-linear transformations. Together, these modules create a powerful framework for understanding and representing sequential data. After processing the single-channel EEG into a three-dimensional matrix and inputting it into the Conformer system, the system categorizes the output into five stages (represented by five moon patterns in the figure 1), namely, wakefulness (W), rapid eye movement (REM) sleep, and four non-rapid eye movement stages (N1, N2 and N3) [11].

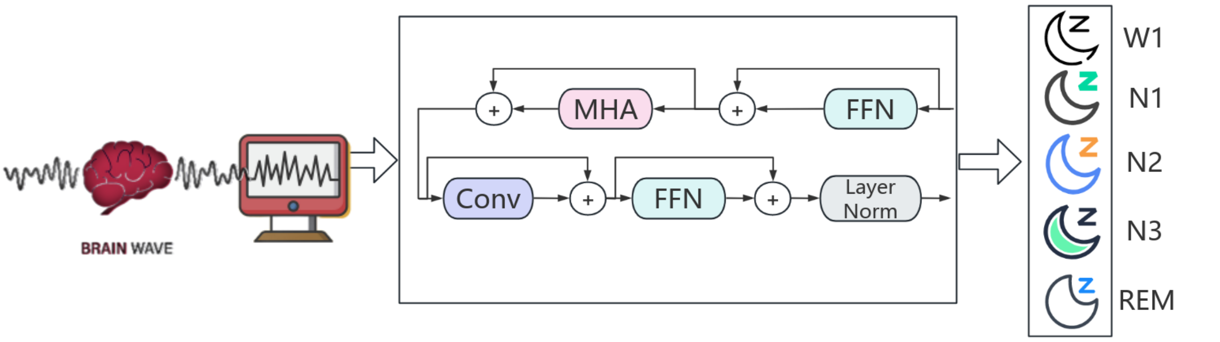

Figure 1. Conformer Encoder Layer. The Conformer Encoder primarily consists of two FFN modules, one MHA module, and one Conv module.

Mathematically, this means, for input X_{input} to a Conformer block i, the output Y_{output} of the block is,

$$\tilde{X}_{input} = X_{input} + \frac{FFN(X_{input})}{2} \tag{1}$$

$$X_{input}' = \tilde{X}_{input} + \text{MHA}\ (\tilde{X}_{input}) \tag{2}$$

$$X_{input}'' = X_{input}' + \text{Conv}\ (X_{input}') \tag{3}$$

$$Y_{output} = \mathscr{L}(X_{input}'' + \frac{FFN(X_{input}'')}{2}) \tag{4}$$

where FFN (·), MHA (·), Conv (·) and $\mathscr{L}$ (·) refer to Feed Forward Module, Multi-head Attention Module, Convolutional Module and Layer Normalization functions respectively, and X_{input} represents the input single-channel $X_i \in R^{B\times T\times S}$.

2.1. *Feed-Forward Neural Network Module (FFN)*

In Conformer, FFN is used to perform position-aware post-processing of self-attention output, enhancing the model's ability to extract information from different positions within the sequence. The first FFN takes the signal as input, and its output will be passed to the MHA module; the second FFN takes the output of Conv and its output will be passed through a residual network into LayerNorm. FFN mainly consists of two linear layers and the Swish activation function. Within the entire FFN, a residual connection is added after normalization.

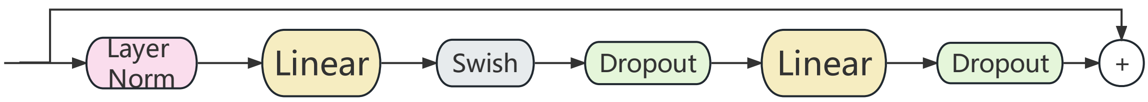

Figure 2. Feed Forward Networks Module. This module undergoes LayerNorm processing, followed by a Linear Layer, and is followed by a Swish Activation function. The Linear Layers connected after it also have Dropout operations both before and after.

For input X_i' to the first three components of the FFN, the output X_{feed_i}' of the block is:

$$X_{feed\_i} = A_{Swish}\ (\mathcal{F}_{linear}\ (\text{LN}\ (X_i'))) \tag{5}$$

where LN (·), $\mathcal{F}_{linear}$ (·), $\acute{R}$ (·) and A_{Swish} (·) refer to Layer Normalization, Linear Layer, Residual Networks and Swish Activation functions respectively.

$$X_{feed\_i}' = \acute{R}\ (\mathcal{D}\ (\mathcal{F}_{linear}\ (\mathcal{D}\ (X_{feed\_i})))) \tag{6}$$

After passing through the Swish activation function, the output X_{feed_i}' is obtained through an additional Linear Layer and Dropout,where $\mathcal{D}$ (·) represents Dropout.

The first Linear layer and its components introduce a non-linear mapping, allowing the model to learn a mapping from the input X_i' to a hidden representation, capturing features and patterns in the data. The second Linear Layer and its components further introduce non-linear mapping, improving feature representation. This helps the model better capture abstract features and patterns in the data. Typically, there is a non-linear activation function called Swish [12], between these two Linear Layer parts. This activation function introduces non-linearity, enabling the model to learn complex non-linear relationships. The entire FFN, through these two linear transformations and the non-linear activation function, maps the input X_i' to a higher-level feature representation for further processing and learning.

DOI: 10.54254/2755-2721/74/20240451

2.2. Multi-head Attention Module (MHA)

The MHA calculates the attention weights to capture relationships between different positions in the input sequence.

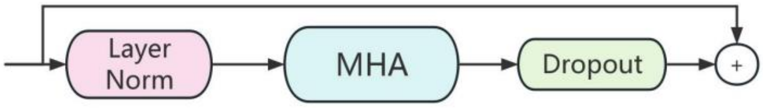

Figure 3. Multi-head Attention Module. We employ multi-headed self-attention with relative positional embeddings within a pre-norm residual unit.

$$Q_i = Ƒ_{linear}\ (X_{feed\_i}')\quad (K_i、V_i\ are\ the\ same) \tag{7}$$

where Q_i represents queries, K_i represents keys, V_i represents values and $Ƒ_{linear}$ $(\cdot)$ represents Linear Layer. These three matrices are used to calculate attention weights to capture relationships between different positions in the input sequence [13].

$$Scores_i = Q_i \cdot K_i / \sqrt{d_k} \tag{8}$$

$Scores_i$ means scaled attention score, which refers to the attention scores that have been scaled. These scores are used to calculate attention weights, quantifying the relationship between one position and other positions in the input sequence. Scaling helps control the magnitude of attention scores, making them more stable and easier to train.

$$W_i = Ś\ (Scores_i + A_m) \tag{9}$$

W_i and Ś $(\cdot)$ refers to Attention weights and Softmax transformation [14], representing the importance or relevance of different positions in the input sequence with respect to a specific position or query. These weights are calculated through the attention mechanism and determine how much each position contributes to the output. Subsequently, "A_m" represents “Attention Mask”, a binary mask or a weight matrix used in the attention mechanism. It is applied to control which positions in the input sequence are considered or ignored during the attention calculation.

$$W_{va}^{h} = W_i \cdot V_i \tag{10}$$

"W_{va}^{h} " refers to the results of summing the values (V_i) based on the attention weights. In self-attention mechanisms, the similarity between queries (Q_i) and keys (K_i) determines the weight for each value, and these values are then linearly weighted according to these weights to produce weighted values. This is a core operation in self-attention mechanisms, used to combine information from the input when generating the output.

$$MHA_o = Ɗ\ (\parallel (W_{va}^{h})) \tag{11}$$

where “$||$ $(\cdot)$” refers to Concatenate, multiplying vectors from W_{va}^{h}, followed by a Dropout ($Ɗ$ $(\cdot)$)operation to obtain the output MHA_o.

2.3. Convolution Module (Conv)

A convolution module is introduced to effectively capture local features [9]. This module consists of components such as one-dimensional convolution, Gated Linear Unit (GLU) activation [15], normalization, activation functions, and linear layers. It equips the model with the ability to represent features at multiple scales.

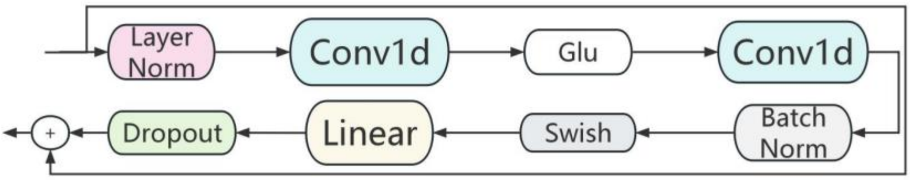

Figure 4. Convolutional Module. This module mainly consists of two 1D Convolutional layers, a Linear Layer, and several auxiliary operations, including two activation functions (Swish and Glu), Batch Normalization, Layer Normalization, and Dropout.

$$X_{con\_i} = \text{Conv1d}\ (\text{LN}\ (MHA_o)) \tag{12}$$

MHA_o comes from the output of the previous module MHA and serves as the input to the Convolutional module. The input tensor undergoes one-dimensional Convolution (Conv1d $(\cdot)$) and Layer Normalization (LN $(\cdot)$) processing, effectively filtering and normalizing the data to extract and balance features.

$$X_{con\_i}' = \text{BN}\ (\text{Conv1d}\ (A_{Glu}\ (X_{con\_i}))) \tag{13}$$

The processing involves applying the GLU ($A_{Glu}(\cdot)$) activation function [15], one-dimensional convolution (Conv1d $(\cdot)$), and Batch Normalization (BN $(\cdot)$). This combination effectively leverages A_{Glu} $(\cdot)$ to control information flow, followed by Conv1d for feature extraction and batch normalization for improved training stability.

$$X_{con\_i}'' = \mathcal{D}\ (\mathcal{F}_{linear}\ (A_{Swish}\ (X_{con\_i}'))) \tag{14}$$

This process involves the application of the Swish activation function (A_{Swish} $(\cdot)$), followed by a Linear Layer ($\mathcal{F}_{linear}$ $(\cdot)$) and Dropout ($\mathcal{D}$ $(\cdot)$), allowing the model to capture non-linear relationships in the data while enabling the extraction of features and patterns.

3. Conformer Encoder

3.1. Data

In the experiments, two publicly available datasets were used, namely Sleep-EDF-20 and Sleep-EDF-78, as shown in Table 1 [16]. For each dataset, a single EEG channel was employed in various experiments.

Table 1. Two Public Datasets. (Each sample is a 30-second epoch)

Datasets	EEG Channel	Sampling Rate
Sleep-EDF-20	Fpz-Cz	100 Hz
Sleep-EDF-78	Fpz-Cz	100 Hz

3.2. Performance of Conformer

Table 2 display the comparison for the application of the proposed model on the Fpz-Cz channel for the Sleep-EDF dataset. The comparison are calculated by summing the scores of all test data. Performance of the Conformer model is evaluated by comparing the results obtained using the AttnSleep method [11]. The comparison is conducted in terms of accuracy (test set accuracy) and F1 score. From the table, it can be observed that for the EDF-78 dataset, both accuracy and F1 score show better performance compared to AttnSleep. However, for the EDF-20 dataset, while F1 score is better, accuracy is slightly lower than AttnSleep. The formulas for "accuracy" and "val_loss" are as follows,

$$accuracy = Correctly\ predicted\ number\ of\ samples/total\ number\ of\ samples \tag{15}$$

$$val\_loss = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(p_{ij}) \tag{16}$$

Where N is the total number of samples in the validation set, and C is the number of classes (in this case, C is 5); y_{ij} is an indicator function representing whether sample i belongs to class j, if true, it is 1, otherwise 0; p_{ij} is the model's predicted probability for sample i belonging to class j.

Table 2. The comparison for the proposed model applied to the Fpz-Cz channel in EDF-20 and EDF-78 dataset. The best data has been bolded.

Dataset	Method	Overall Metrics Accuracy	F1 score
Sleep-EDF-20	AttnSleep [11]	**84.4**	80.9
	Conformer *(ours)*	83.1	**82.4**
Sleep-EDF-78	AttnSleep [11]	81.3	78.1
	Conformer *(ours)*	**82.31**	**78.4**

To prevent over-fitting and ensure efficient training, the experiment incorporated an early stopping strategy. If the validation loss failed to improve for a specified number of consecutive epochs (determined by the patience parameter), training was halted. During early stopping, saving the model parameters at their best performance.

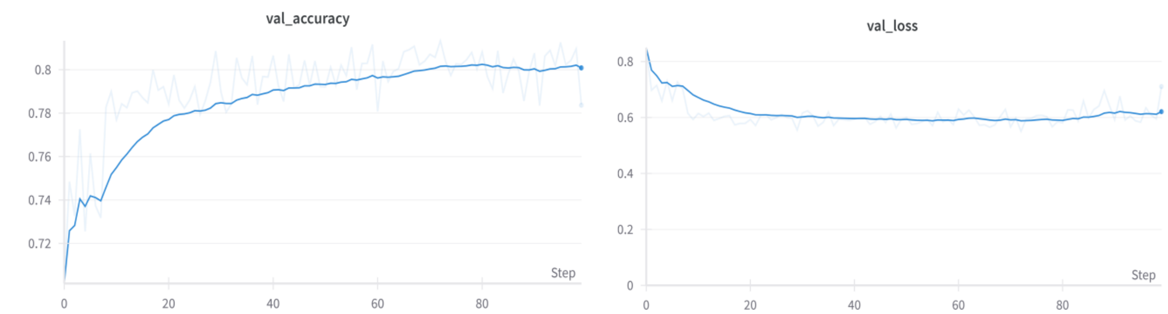

Figure 5. A random fold of the validation set accuracy and loss (data sourced from EDF-20).

4. Conclusion

In conclusion, this study introduced a novel approach, the Convolutional Transformer, for Sleep Stage Classification. The fusion of convolutional and transformer architectures demonstrated promising results in accurately categorizing sleep stages. Through experimentation on diverse datasets, including EDF-20 and EDF-78, our model exhibited superior performance in comparison to existing methods, particularly showcasing notable improvements in both validation accuracy and loss metrics. The unique combination of convolutional operations for capturing local features and transformer mechanisms for global context integration proved to be an effective strategy for extracting meaningful representations from sleep-related data. As the field of sleep stage classification continues to evolve, the Conformer presents itself as a compelling framework, offering enhanced interpretability and performance, thereby contributing to the advancement of sleep analysis and related healthcare applications. Future work may involve further fine-tuning of the model architecture, exploration of additional datasets, and potential applications in real-world clinical scenarios.

References

[1] Luyster, F.S., Strollo Jr, P.J., Zee, P.C. and Walsh, J.K., 2012. Sleep: a health imperative. Sleep, 35(6), pp.727-734.

DOI: 10.54254/2755-2721/74/20240451

[2] Yildirim, O., Baloglu, U.B. and Acharya, U.R., 2019. A deep learning model for automated sleep stages classification using PSG signals. International journal of environmental research and public health, 16(4), p.599.

[3] Cooper, R., Osselton, J.W. and Shaw, J.C., 2014. EEG technology. Butterworth-Heinemann.

[4] Wang, G., Teng, C., Li, K., Zhang, Z. and Yan, X., 2015. The removal of EOG artifacts from EEG signals using independent component analysis and multivariate empirical mode decomposition. IEEE journal of biomedical and health informatics, 20(5), pp.1301-1308.

[5] Tankisi, H., Burke, D., Cui, L., de Carvalho, M., Kuwabara, S., Nandedkar, S.D., Rutkove, S., Stålberg, E., van Putten, M.J. and Fuglsang-Frederiksen, A., 2020. Standards of instrumentation of EMG. Clinical neurophysiology, 131(1), pp.243-258.

[6] Gururaj, N., Vinod, V. and Vijayakumar, K., 2023. Deep grading of mangoes using convolutional neural network and computer vision. Multimedia Tools and Applications, 82(25), pp.39525-39550.

[7] Otter, D.W., Medina, J.R. and Kalita, J.K., 2020. A survey of the usages of deep learning for natural language processing. IEEE transactions on neural networks and learning systems, 32(2), pp.604-624.

[8] Gulati, A., Qin, J., Chiu, C.C., Parmar, N., Zhang, Y., Yu, J., Han, W., Wang, S., Zhang, Z., Wu, Y. and Pang, R., 2020. Conformer: Convolution-augmented transformer for speech recognition. arXiv preprint arXiv:2005.08100.

[9] Li, Z., Liu, F., Yang, W., Peng, S. and Zhou, J., 2021. A survey of convolutional neural networks: analysis, applications, and prospects. IEEE transactions on neural networks and learning systems.

[10] Lipton, Z.C., Berkowitz, J. and Elkan, C., 2015. A critical review of recurrent neural networks for sequence learning. arXiv preprint arXiv:1506.00019.

[11] Eldele, E., Chen, Z., Liu, C., Wu, M., Kwoh, C.K., Li, X. and Guan, C., 2021. An attention-based deep learning approach for sleep stage classification with single-channel EEG. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 29, pp.809-818.

[12] Eger, S., Youssef, P. and Gurevych, I., 2019. Is it time to swish? Comparing deep learning activation functions across NLP tasks. arXiv preprint arXiv:1901.02671.

[13] Koizumi, Y., Yatabe, K., Delcroix, M., Masuyama, Y. and Takeuchi, D., 2020, May. Speech enhancement using self-adaptation and multi-head self-attention. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 181-185). IEEE.

[14] Qin, Z., Sun, W., Deng, H., Li, D., Wei, Y., Lv, B., Yan, J., Kong, L. and Zhong, Y., 2022. cosformer: Rethinking softmax in attention. arXiv preprint arXiv:2202.08791.

[15] Veness, J., Lattimore, T., Budden, D., Bhoopchand, A., Mattern, C., Grabska-Barwinska, A., Sezener, E., Wang, J., Toth, P., Schmitt, S. and Hutter, M., 2021, May. Gated linear networks. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 35, No. 11, pp. 10015-10023).

[16] PhysioBank, P., 2000. Physionet: components of a new research resource for complex physiologic signals. Circulation, 101(23), pp.e215-e220.