Detecting eye movement and muscle artifacts in EEG data using Deep Learning

Leon Ackermann
University of Osnabrück
lackermann@uni-osnabrueck.de

Aaron Maiwald
University of Osnabrück
amaiwald@uni-osnabrueck.de

Abstract

EEG artifacts such as eye or muscular movements are unwanted signals which mislead EEG classification. We present two models to detect EEG artifacts from eye and muscular movements: a CNN with attention and a bidirectional, convolutional LSTM. We train and evaluate our models on the TUH Artifact Corpus. Our CNN achieves an 85 percent accuracy, which is competitive with recent models in the literature, while our best LSTM only achieves 69 percent accuracy. Both of our models perform poorly in terms of recall and precision.

1. Introduction

Electroencephalography (EEG) is a widely used technique for measuring brain activity in a variety of applications. However, processing EEG data can be challenging due to the presence of artifacts, which are unwanted signals that can interfere with the analysis of brain activity. An example would be that muscle artifacts are mistaken for seizures and increase the number of false predictions by the classifier. Artifact detection and removal is therefore an important problem in EEG data processing.

There is substantial literature on EEG artifact detection and removal using traditional machine learning techniques, such as Principal Component Analysis [1] and Independent Component Analysis [2] or Filtering techniques such as Adaptive Filtering [3] and Wiener Filtering [4].

Only very recently have researchers started to apply deep learning to this task. Deep learning is a subfield of machine learning that uses neural networks with multiple layers to automatically learn hierarchical representations of the data. Recent literature concentrated on deploying Convolutional Neural Networks [5] or Attention-based networks [6, 7] or LSTM-based networks [8] for identifying artifacts.

In this paper, we present a deep learning classifier for EEG artifact detection and compare its performance to previous work. The task of our classifier is to recognise segments containing artifacts specifically from muscle or eye movements. To achieve this, we compare various deep learning architectures such as long short-term memory (LSTM) networks and attention-based models. These architectures have shown promising results in other domains and have the potential to improve the accuracy of artifact detection in EEG data. This paper makes two important contributions:
1. While previous models have only classified segments of EEG data as containing artifacts or not, our model also predicts precisely when an artefact occurs within a segment. Insofar as EEG artifact detection is for the purpose of cleaning EEG data, knowing precisely when artifacts occur is very advantageous: researchers can remove EEG data segments in a much more targeted way, meaning that much less data is lost. We demonstrate that, using relatively simple models, it is possible to achieve accuracy and recall that is competetive with state-of-the-art methods.
2. We investigate the performance of a bidirectional, convolutional LSTM on this task. To our knowledge, this has not been tried before in the relevant literature.

In the second section of this paper, we will introduce the dataset that we used and explain our preprocessing steps. The third section will cover the different models we trained: various versions of LSTMs and attention-enhanced CNNs. Section 4 will show our most important results and an analysis of the contributions that various components of our models make to their overall performance. Section 5 will discuss the results and outline the most promising directions for future research.

2. Methods

2.1. Dataset

We train and evaluate our classifier on the TUH EEG Artifact dataset from the Temple University Hospital of Philadelphia (Pennsylvania) [9] which is a subset of the TUH EEG Data Corpus [**?**]. This subset dataset consists of normal EEG signals and EEG signals affected by five types of artifacts: chewing events, eye movements, muscular artifacts, shivering events, and instrumental artifacts (such as electrode pop, electrostatic artifacts, or lead artifacts). In total, the dataset comprises 259 recorded EEG sessions that were recorded over a span of 13 years. TUH recorded 213 patients between 10 and 90 years old. The EEGs were recorded with a sampling frequency of 250 Hz and 16-bit resolution. For the purpose of this project, we limited our analysis to the most common types of artifacts in the positive class: muscular artifacts. Muscle artifacts appear as sharp waves in the dataset such as in Figure 1.

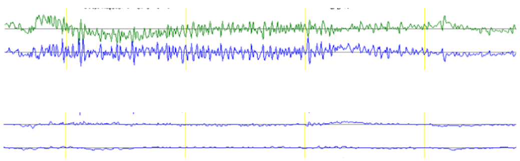

Figure 1. Example of EEG recording with muscle artifact [9] (first) and without any artifact (second). For the purpose of illustration, we selected only two channels.

2.2. Data Acquisition and Preprocessing

We wrote a custom python script to download the EDF files and convert them to PKL files. The conversion process enabled us to extract the EEG data from the EDF files in a structured and standardized format for further analysis.

The EDF files contained time-stamps for when the artifacts started and when they ended. We decided to create labels for each time-step, indicating with 0s and 1s whether an artifact was present or not.

We created windows of EEG data to segment the continuous time-series into smaller and more manageable parts. We chose a window length of 5 seconds based on previous research by [7], and since not all recordings had the same frequency, we downsampled them to a frequency of 128Hz. Following previous research, we decided to use all and only channels that were present in the entire dataset. The channels we used for analysis included: [list of channels] (see figure X.) In line with common deep learning practice, we normalized the data with a z-transformation.

During the analysis of our dataset, we observed that only 9 percent of time-steps had a positive label. As a result, our models tended to learn to predict only negative labels. To address this issue, we removed all samples (each containing a 5s recording) where no artifacts were present following the cluster-based undersampling approach presented in [10]. The undersampling method reduced our sample size from approximately 50,000 to roughly 11,000 samples in the training set, while increasing the balance of positive (artifacts) to negative (non-artifacts) samples from 9 percent to around 35 percent. We found that this had a positive effect on the performance of our model. We ensured that evaluating our model on the test data and preventing overfitting with the validation set was done on the original, imbalanced data.

Hyperparameters	**Range**	**LSTM**
Bidirectional	True, False	True
LSTM Layers	1,2,4 to 8	1
LSTM Hidden Units	32, 64, 128, 256	128
Convolutional Layers	1,2,3,4	2
Dense Layers	0,1,2,4	1
Hyperparameters	**Range**	**CNN**
Convolutional Layers	1,2,3,4,6	6
Attention	True, False	True
Attention Heads	4,8	8
Dense Layers	1,2,3	1
Optimizer	SGD, Adam, Adagrad	Adam
Learning Rate	0.0001, 0.001, 0.01	0.001

Table 1. Results of hyperparameter optimization

2.3. Models

After we decided on using CNNs and LSTMs based on x,y,z we performed grid search on the hyperparameters in Table 1. The first round focused on architectural parameters such as the number of hidden layers or the number of attention heads. Each configuration was trained for 50 epochs. We then selected the five models with the lowest validation loss and performed a second round of grid-search on non-architectural hyperparameters such as the learning rate and the optimizer. To train our models we used cloud GPUs, specifically NVIDIA RTX 4090 and NVIDIA A100 SXM4 X1. Our loss function was Binary Crossentropy.

2.4. CNN with Attention

Our first model is a CNN with self-attention, see Figure 4. The model receives an input matrix consisting of 19 channels of 5-second windows at a frequency of 128hz, resulting in a 640 x 19 dimensional input matrix. The input matrix is processed by six consecutive 1-D convolutional layers, each with three filters, with the filter size increasing by 8 in each layer. After each convolutional layer, we apply Max Pooling with pool size of two. We then apply a multi-head attention-layer with 8 attention heads, which is connected to a sigmoid layer with 640 units. This layer predicts, for each time step, the probability of a muscle artifact.

2.5. LSTM-based Architectures

There have been several approaches to classify EEG data with LSTMS. We took inspiration from [**?**, 11–14] to design a Convolutional Bidirectional LSTM (see Figure **??**). Our LSTM model receives an input batch with samples consisting of 640 timesteps each containing 19 EEG channels. We then extend the last axis of the training set in order to create the correct dimensions for the Convolutional Block

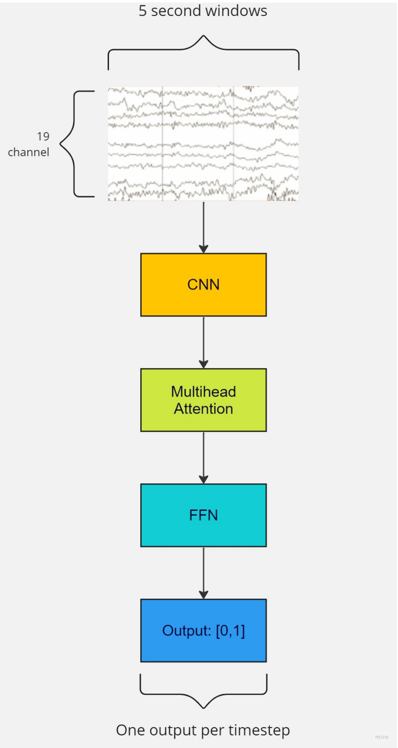

Figure 2. Convolutional Attention-based network

unfolded over time. The convolutional block consists of one 1-d convolutional layer with 8 filters and a kernel size of 3*3 followed by a 1-d Max Pool Layer and Flattening Layer. The output of the convolutional block gets processed by a Bidirectional LSTM with 128 Hidden Units. The Bidirectional LSTM is unfolded over time into 640 timesteps. At each timestep the output of the bidirectional LSTM is passed to the output dense layer.

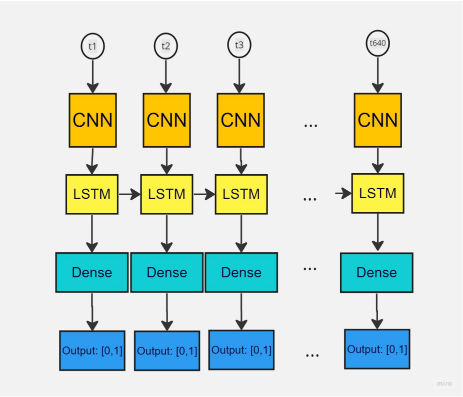

Figure 3. Example picture

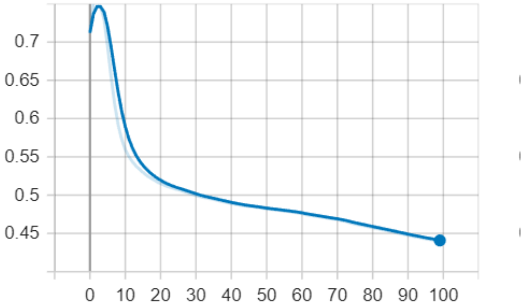

Figure 4. Validation loss of CNN over 100 epochs

3. Results

Our study compared the performance of a convolutional neural network (CNN) and a long short-term memory (LSTM) network on the task of detecting artifacts in EEG data. We summarise the performance of our best models in each category in Table 3. The CNN outperformed the LSTM with an accuracy of 85 percent compared to the LSTM's accuracy of 69 percent. However, both models had poor precision and recall, with the LSTM performing slightly better on both metrics. We note that the test data was heavily imbalanced towards negative labels, and thus a high accuracy on its own is not strong evidence that the model learned its task. In terms of precision and recall, our models performed markedly worse than those from the recent literature, such as Peh et al. (2022) and Cisotto et al. (2022).

3.1. Ablation

4. Discussion

- what models types performed the best?
- out of each model class, which architectual hyperparameter impacted performance the most and why? –¿ assumptions - how does the performance compare to other approaches in literature?
- how do our models compare to other models, in terms of efficiency and model parameters

5. Data and code availability

For the purpose of transparency, we have shared our data and code in this github repository, alongside instructions for how to reproduce our findings.

References

[1] P. Berg and M. Scherg, "Dipole modelling of eye activity and its application to the removal of eye artefacts from the EEG and MEG," *Clinical Physics and Physiological Measurement*, vol. 12, pp. 49–54, Jan. 1991.

Architecture	Accuracy	Precision	Recall
CNN-ATT	0.85	0.3	0.63
LSTM	0.69	0.41	0.40
Peh et al. (CNN-ATT)	0.7	x	0.65
Cisotto et al. (LSTM)	0.83	0.79	0.77

Table 2. Performance of our best models in comparison to similar models from the literature.

[2] T.-P. Jung, S. Makeig, A. J. Bell, and T. J. Sejnowski, "Independent Component Analysis of Electroencephalographic and Event-Related Potential Data," in *Central Auditory Processing and Neural Modeling* (P. W. F. Poon and J. F. Brugge, eds.), pp. 189–197, Boston, MA: Springer US, 1998.

[3] P. He, G. Wilson, and C. Russell, "Removal of ocular artifacts from electro-encephalogram by adaptive filtering," *Medical & Biological Engineering & Computing*, vol. 42, pp. 407–412, May 2004.

[4] B. Somers, T. Francart, and A. Bertrand, "A generic EEG artifact removal algorithm based on the multi-channel Wiener filter," *Journal of Neural Engineering*, vol. 15, p. 036007, June 2018.

[5] P. Nejedly, J. Cimbalnik, P. Klimes, F. Plesinger, J. Halamek, V. Kremen, I. Viscor, B. H. Brinkmann, M. Pail, M. Brazdil, G. Worrell, and P. Jurak, "Intracerebral EEG Artifact Identification Using Convolutional Neural Networks," *Neuroinformatics*, vol. 17, pp. 225–234, Apr. 2019.

[6] G. Cisotto, A. Zanga, J. Chlebus, I. Zoppis, S. Manzoni, and U. Markowska-Kaczmar, "Comparison of Attention-based Deep Learning Models for EEG Classification," Dec. 2020. arXiv:2012.01074 [cs, eess, q-bio].

[7] W. Y. Peh, Y. Yao, and J. Dauwels, "Transformer Convolutional Neural Networks for Automated Artifact Detection in Scalp EEG," Aug. 2022. arXiv:2208.02405 [eess].

[8] Hasib-Al-Rashid, N. K. Manjunath, H. Paneliya, M. Hosseini, W. D. Hairston, and T. Mohsenin, "A Low-Power LSTM Processor for Multi-Channel Brain EEG Artifact Detection," in *2020 21st International Symposium on Quality Electronic Design (ISQED)*, pp. 105–110, Mar. 2020. ISSN: 1948-3287.

[9] A. Hamid, K. Gagliano, S. Rahman, N. Tulin, V. Tchiong, I. Obeid, and J. Picone, "The Temple University Artifact Corpus: An Annotated Corpus of EEG Artifacts," in *2020 IEEE Signal Processing in Medicine and Biology Symposium (SPMB)*, pp. 1–4, Dec. 2020. ISSN: 2473-716X.

[10] S.-J. Yen and Y.-S. Lee, "Cluster-based under-sampling approaches for imbalanced data distributions," *Expert Systems with Applications*, vol. 36, pp. 5718–5727, Apr. 2009.

[11] P. Nagabushanam, S. Thomas George, and S. Radha, "EEG signal classification using LSTM and improved neural network algorithms," *Soft Computing*, vol. 24, pp. 9981–10003, July 2020.

[12] K. Singh and J. Malhotra, "Two-layer LSTM network-based prediction of epileptic seizures using EEG spectral features," *Complex & Intelligent Systems*, vol. 8, pp. 2405–2418, June 2022.

[13] Z. Ni, A. C. Yuksel, X. Ni, M. I. Mandel, and L. Xie, "Confused or not Confused? Disentangling Brain Activity from EEG Data Using Bidirectional LSTM Recurrent Neural Networks," in *Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology,and Health Informatics*, ACM-BCB '17, (New York, NY, USA), pp. 241–246, Association for Computing Machinery, Aug. 2017.

[14] E. Tuncer and E. Doğru Bolat, "Classification of epileptic seizures from electroencephalogram (EEG) data using bidirectional short-term memory (Bi-LSTM) network architecture," *Biomedical Signal Processing and Control*, vol. 73, p. 103462, Mar. 2022.

6. Appendix: Implementation

6.1. Logging

6.1.1 Metrics

We logged the previousy described metrics for each model architecture type through a keras callback. Our metrics can be found on our tensorboard.dev (here the link should be), moreover our logs are in our github repository.

6.1.2 Checkpoints

After each epoch, we saved our model in case that our server crashes. This was done through a keras callback.

6.1.3 Early Stopping

Early stopping is a technique commonly used to prevent overfitting and save time during the training of deep learning models. To implement early stopping in our project, we created a TensorFlow callback that monitored the loss function of our model during training. If the loss did not improve for three consecutive epochs, training was stopped early. This approach helped us to prevent overfitting and avoid unnecessary computational expenses. This was done through a keras callback.

6.1.4 Model saving

After our models were trained for the set amount of epochs, we saved it for later purposes. If we stopped the training early due to Early Stopping as explained above, we saved the model with the lowest loss on the training data set. Our models can be found in our github repository.

6.2. Data transfer from server

In order of transferring data from the server to our local machines we set up a SSH connection between the two parties. To achieve this, we generated a pair of public and private keys. After creating the keys, we stored the public key on the Vast AI platform, allowing for secure authentication without requiring a password. Finally, we utilized the 'scp' (secure copy) command to transfer the log files from the remote server to the local machine, ensuring a safe and efficient data transfer while maintaining the confidentiality of our research data. A documentation of this process can be found in our github repository.