Convolutional neural networks for decoding of covert attention focus and saliency maps for EEG feature visualization

Amr Farahat, Christoph Reichert, Catherine M Sweeney-Reed and Hermann Hinrichs

1 Neurocybernetics and Rehabilitation Research Group, Department of Neurology, Otto-von-Guericke University Hospital, Leipziger Str. 44, 39120 Magdeburg, Germany
2 Department of Behavioral Neurology, Leibniz Institute for Neurobiology, Brenneckestr. 6, 39118 Magdeburg, Germany
3 Forschungscampus STIMULATE, Sandtorstr. 23, 39106 Magdeburg, Germany
4 Centre for Behavioral Brain Sciences (CBBS), Universitätsplatz 2, 39106 Magdeburg, Germany

E-mail: amr.f.farahat@gmail.com

Received 19 April 2019, revised 23 July 2019
Accepted for publication 15 August 2019
Published 23 October 2019

Abstract

Objective. Convolutional neural networks (CNNs) have proven successful as function approximators and have therefore been used for classification problems including electroencephalography (EEG) signal decoding for brain–computer interfaces (BCI). Artificial neural networks, however, are considered black boxes, because they usually have thousands of parameters, making interpretation of their internal processes challenging. Here we systematically evaluate the use of CNNs for EEG signal decoding and investigate a method for visualizing the CNN model decision process. Approach. We developed a CNN model to decode the covert focus of attention from EEG event-related potentials during object selection. We compared the CNN and the commonly used linear discriminant analysis (LDA) classifier performance, applied to datasets with different dimensionality, and analyzed transfer learning capacity. Moreover, we validated the impact of single model components by systematically altering the model. Furthermore, we investigated the use of saliency maps as a tool for visualizing the spatial and temporal features driving the model output. Main results. The CNN model and the LDA classifier achieved comparable accuracy on the lower-dimensional dataset, but CNN exceeded LDA performance significantly on the higher-dimensional dataset (without hypothesis-driven preprocessing), achieving an average decoding accuracy of 90.7% (chance level = 8.3%). Parallel convolutions, tanh or ELU activation functions, and dropout regularization proved valuable for model performance, whereas the sequential convolutions, ReLU activation function, and batch normalization components reduced accuracy or yielded no significant difference. Saliency maps revealed meaningful features, displaying the typical spatial distribution and latency of the P300 component expected during this task. Significance. Following systematic evaluation, we provide recommendations for when and how to use CNN models in EEG decoding. Moreover, we propose a new approach for investigating the neural correlates of a cognitive task by training CNN models on raw high-dimensional EEG data and utilizing saliency maps for relevant feature extraction.

5 Author to whom any correspondence should be addressed.
Machine learning techniques are effective in learning discriminative signal patterns from brain data. Most conventional machine learning techniques, however, require an initial step before classification, namely feature extraction. Feature extraction includes preprocessing like noise removal, dimensionality reduction and transformations like principal component analysis (PCA) or frequency decomposition, and feature selection like channel and time interval selection [2, 55, 56]. Feature extraction is hypothesis-driven and requires domain expertise and thus using a machine learning algorithm that autonomously determines relevant features could save time and effort and potentially make use of features that would have been overlooked or removed.

Deep learning—a subarea of machine learning—has emerged in recent years as a technique that could be applied to extract features automatically from the data that maximize between-class discriminability [34]. Deep learning has proven to be effective, especially on image and audio data, in a variety of tasks [29, 47]. Most of these successes are attributed to a specific type of deep learning models called convolutional neural networks (CNNs) [33]. CNN refers to a special neural network architecture, which is inspired by the hierarchical structure of the ventral stream of the visual system [23]. Stacking neurons with local receptive fields on top of each other leads to the development of simple and complex cells that extract features at different levels of abstraction [33].

The first attempt to use CNNs for P300 classification involved developing a 5-layer CNN [8], which was trained and tested on public dataset II of the third BCI competition [5]. The method achieved the best results after 10 stimulus repetitions, but the performance was not as high as that of an ensemble of support vector machines (SVMs) after 15 repetitions. The advantage of the CNN was that all available electrodes were used, instead of a priori electrode selection. A general CNN, EEGNet, was developed that can classify EEG signals for different BCI tasks [32]. It follows the all-convolutional approach without fully connected layers before the output layer [50]. It was trained and tested on a P300 dataset and yielded a significant improvement over classification using a shallow approach. Furthermore, a 3D CNN was developed to decode the P300 component in an auditory BCI [7]. A 3D CNN is characterized by requiring the input as a 3D array: 2-dimensions for space and one time dimension. Instead of keeping the values of all electrodes at one time point in a single, one-dimensional array, the spatial structure of the data is further preserved. Comparison of the performance to that of a 2D version of the network and also to SVM and Fisher’s discriminant analysis (FDA) revealed significant improvement in decoding accuracy using the 3D CNN. Batch normalization is an algorithm that was developed to help facilitate
faster training of neural networks with convergence on lower minima [25]. A CNN was developed to examine the impact of including batch normalization on P300 detection accuracy [36]. It was shown that removing more than one batch normalization layer markedly decreased the classification accuracy. CNNs trained on EEG data showed also promising potential in decoding motor imagery tasks relative to the commonly used filter bank common spatial patterns (FBCSP) algorithm [32, 48].

In order to make CNNs interpretable, several feature visualization methods have been developed to explain the performance of the CNN models and the relevant features that drive their outputs [3, 40, 44]. Some of these techniques were adopted and other novel techniques were developed for CNNs applied to EEG data [32, 48, 52]. Here we adopt the gradient-based visualization technique commonly used in computer vision research called saliency maps [49]. A saliency map is generated as the gradient of the CNN output with respect to its input example. It is computed using backpropagation, after fixing the weights of the trained network, by propagating the gradient with respect to the layer’s inputs back to the first layer that receives the input data. We can interpret the saliency map as representing the relevant features that drive the network’s decision.

Here we investigated the potential of a CNN model in decoding the focus of covert attention using EEG data, systematically tested the impact of different components of the model on its performance, and examined the model’s capacity for transfer learning. We explored the use of saliency maps for visualizing what the CNN model has learned from the data and investigated the possibility of using the features extracted by the CNN to gain insights into the electrophysiological signals underlying the task. Moreover, we compared the decoding performance of the CNN model to those obtained with a linear discriminant analysis (LDA) classifier, which is reported to be one of the best performing classifiers for ERP BCI applications [4, 30, 39], and also to the EEGNet model [32] to investigate the potential of CNNs in general in decoding the focus of covert attention.

Methods

Dataset

The EEG data classified in the current study were recorded in a previous study, combined with simultaneous MEG recordings to compare both modalities for BCI use [46]. In brief, 19 participants (10 females) performed a covert attention task in multiple runs (10.7 runs per participant on average), each comprising 12 trials. In each trial, the participants covertly attended to one of 12 stimuli of their choice in an unpredictable order. The stimuli were presented as colored and numbered spherical objects at equidistant spaces over the circumference of an invisible circle with the fixation point located at its center. All the objects flashed five times in a pseudorandom order, leading to 60 flashes with a stimulus onset asynchrony (SOA) of 167 ms. The pseudo-random order ensured that the same object would not flash twice in a row, but with at least two other objects flashing in between. The EEG data were recorded at 29 standard electrode locations (Fp1, Fp2, Fz, F3, F4, F7, F8, FC1, FC2, Cz, C3, C4, T7, T8, CP1, CP3, Pz, P3, P4, P7, P8, PO3, PO4, PO7, PO8, Oz, O9, O10, Iz), sampled at 508.63 Hz, and referenced against the right mastoid. Vertical and horizontal electrooculograms (EOG) were also recorded to ensure fixation of the gaze and exclude classifications based on eye movements.

Preprocessing

In order to test the potential of CNNs in dealing with high-dimensional EEG data, we created two datasets: one to represent high-dimensional, minimally preprocessed data and one to represent low-dimensional data preprocessed with a feature extraction method. For the first dataset, the data were only downsampled to half of the original sampling frequency (254.3 Hz) after applying an antialiasing FIR lowpass filter, and we call it dataset $f_{250}$. For the second dataset, the data were decimated by a factor of 10 by averaging over blocks of 10 time points, which essentially is equivalent to downsampling after applying a moving average filter. These averages were then used as the temporal features [31]. This process led to a sampling frequency of 50.8 Hz, and hence we call it dataset $f_{50, avg}$. This method is justified based on the knowledge that the P300 response—the signal of interest—is of low frequency relative to the sampling frequency [24, 31]. Each trial, comprising 60 stimuli flashes, was then segmented into 60 segments from 100 ms pre-stimulus to 700 ms post-stimulus. Each segment represents an input example, $x$, whose corresponding output, $y$, is either 1 if the participant attended this location or 0 otherwise. Each segment was then baseline-corrected by subtracting the average of the pre-stimulus interval. Since the dataset was imbalanced, we oversampled the minority class in the training set by randomly replicating examples until the fractions of both classes were equal before training the neural network models [20]. The input data matrix was then normalized by subtracting the mean and dividing by the standard deviation, both determined from the training set.

Convolutional neural networks (CNNs)

A convolutional layer of a CNN is formed from multiple feature maps. Each feature map comprises artificial neurons that share the same set of weights, but they apply their weights to a different patch of the input. These weights are called a filter, a kernel, or a feature detector, and their input patches are the equivalent of receptive fields, in the analogy between CNNs and the visual system. Each feature map thus represents an input example, $x$, whose corresponding output, $y$, is either 1 if the participant attended this location or 0 otherwise. Each segment was then baseline-corrected by subtracting the average of the pre-stimulus interval. Since the dataset was imbalanced, we oversampled the minority class in the training set by randomly replicating examples until the fractions of both classes were equal before training the neural network models [20]. The input data matrix was then normalized by subtracting the mean and dividing by the standard deviation, both determined from the training set.
A Farahat et al

A pooling operation is defined by a pooling size and a pooling stride. The pooling size controls the input patch size, and the pooling stride controls how to move to the next input patch. The pooling layer serves to reduce the dimensionality of the network and therefore the complexity of the model. It also causes the model to be spatially invariant.

The branched model. We designed a model consisting of five convolutional layers plus the input and output layers (figure 1). The first convolutional layer is purely spatial, i.e., the filter extends through all the channels at each time point separately. On top of the output of the spatial filters layer, we implemented two parallel temporal convolutional layers: one with a small filter size that is equal to 100 ms and the other one with a large filter size that is equal to roughly 400 ms. EEG data are thought to be formed from overlapping harmonics of different frequencies. The small and large filters should be able to detect high frequency and low frequency patterns in the data, respectively [53]. We then added two more deep convolutional layers after the small temporal filters branch so that the model was formed only from the two parallel convolutional layers on top of the spatial convolutional layer. We used \( \text{tanh} \) as the non-linear activation function for each layer, except for the output layer, for which we used a sigmoid function. The reason behind choosing \( \text{tanh} \) as an activation function is that its output range includes positive and negative values with a mean of zero. This is more suitable for representing features of EEG data, which include positive and negative values, than the commonly used \( \text{ReLU} \) function [17], whose output range has only positive values. Additionally, we used batch normalization [25] and dropout [51] of 0.5 after each layer to accelerate training and regularize the network. Moreover, we used L2 regularization with regularization strength 0.01 for each layer in the network.

We validated the impact of different model components by omitting single components, retraining the model and comparing the results. We therefore trained five additional models:

- **No_BN**: batch normalization was omitted.
- **No_Dropout**: dropout was omitted.
- **Not_Branched**: the branch with the large temporal filter was omitted to validate the impact of the parallel convolutions.
- **Not_Deep**: the last two layers after the small temporal filters branch were omitted so that the model was formed only from the two parallel convolutional layers on top of the spatial convolutional layer.
- **ReLU**: we replaced the \( \text{tanh} \) activation function with the \( \text{ReLU} \) function [17].
- **ELU**: we replaced the \( \text{tanh} \) activation function with the \( \text{ELU} \) function [11].

We compared the performances of the branched model with a general CNN model for EEG-based BCIs, the EEGNet model [32], in order to test the potential of CNNs in general, and also with a traditional machine learning technique frequently found in the BCI literature, linear discriminant analysis (LDA) [26] and LDA with shrinkage [4].

Cross-validation schemes

To evaluate the performance of the models, we validated within-subject models as well as cross-subject models. For each participant, a model was trained only using the participant’s individual data, rendering the model subject-specific. Since the data are limited per participant, we implemented a 5-fold sequential cross-validation scheme. For LDA, five models were trained per participant, each trained with data from four folds as the training set, and the models were then tested on the data of the held-out fold. For CNN models, we further divided the data from the four folds in a stratified random manner into 80% training data and 20% validation data.

To investigate the model’s transfer learning ability, we implemented a leave-one-subject-out cross-validation scheme in which one participant’s data were held back as a test set, and the data from the remaining 18 participants were used as training and validation sets. We then fine-tuned each cross-subject model with a subset from the test participant’s data in order to evaluate whether we could achieve better performance than with random initializations.
Saliency maps

We calculated a within-subject saliency map for each participant based on the participant’s own data and one cross-subject saliency map based on the data of all the participants. In order to calculate the within-subject saliency maps, we trained one model per participant by dividing the participant’s data in a random stratified manner into a 90% training set and a 10% validation set. For the cross-subject saliency map, we trained one cross-subject classifier by also randomly dividing the data of all the participants in a random stratified manner into a 90% training set and a 10% validation set. The saliency maps were then calculated by averaging over all the saliency maps of the target examples in the dataset.

Implementations and training settings

LDA models were implemented using the Scikit-learn library [41]. CNN models were implemented using Keras API [9] with Tensorflow backend [1]. The training experiments were conducted on our university’s Medusa CPU cluster.

To optimize the models, we used the Adam optimizer with the recommended settings (learning rate $= 10^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$) [27]. For fine-tuning, we used stochastic gradient descent with learning rate $= 10^{-4}$ and momentum $= 0.9$. We used a decaying learning rate scheme that halves the learning rate if there was no decrease in the validation loss for 10 epochs. The early stopping technique was applied to avoid overfitting and to reduce the computational time by stopping the training when the validation loss did not decrease for 20 epochs. The batch size for the mini-batch gradient descent was set to 64 examples. The weights of the networks were initialized with the Glorot Uniform method [16].

Results

Model performance

With our branched model, we achieved average decoding accuracies of 90.6% (SD: 10%) and 90.7% (SD: 8.6%) on the lower-dimensional and higher-dimensional datasets, respectively (figure 2(a)). The diagrams show the decoding accuracies averaged over the participants. Decoding accuracy indicates the prediction of the target stimulus location as obtained from the binary classification of each stimulus, i.e., the chance level
is at 8.3%. Since each stimulus location was flashed five times, we averaged the model outputs of the five repetitions to obtain more robust and accurate predictions. Asterisks above the bars represent the p-value of a Wilcoxon signed-rank test between this particular bar and the original branched model configuration (figure 1). (*: $p < 0.05$, **: $p < 0.001$ corrected for multiple comparisons) (a) Within-subject models. (b) Cross-subject models.

Impact of model components

We investigated the impact of different components of the branched model by comparing the decoding accuracies for different alternative within-subject models, in which we omitted or changed one of the model components (figure 3(a)). Again, we used Wilcoxon signed-rank tests with Bonferroni correction to test the significance of the differences. Omitting the dropout and changing the activation function to ReLU caused the most significant negative effect on both datasets. Removing (figure 2(b)). There was no significant difference between the branched model and any of the other three models on the $f_{50\text{avg}}$ dataset, but it still yielded significantly higher accuracy than both LDA classifiers on the $f_{250}$ dataset.

Moreover, the effect of shrinkage on the LDA classifier is not significant, as in the case of within-subject classifiers on both datasets. This is because shrinkage works in situations where the number of training examples is small compared with the number of features. In this case, when the data were pooled from 18 participants, this ratio was high and therefore, LDA was less prone to overfitting, and the regularization was not advantageous.

Figure 3. Effect of different model components on the decoding accuracies of within-subject (a) and cross-subject (b) models applied to $f_{50\text{avg}}$ (left) and $f_{250}$ (right) datasets and averaged over the 19 participants. Asterisks denote the statistical significance of the difference between this particular bar and the original branched model configuration (figure 1). (*: $p < 0.05$, **: $p < 0.001$ corrected for multiple comparisons) (a) Within-subject models. (b) Cross-subject models.
the large temporal filter branch induced a significant negative effect on the f50_avg dataset. The effect of batch normalization was mainly negligible and differed across the datasets. Using the ELU activation function led to a non-significant improvement in the performance across both datasets. The same effect was observed when removing the deep convolutional layers (Not_deep).

We also tested the impact of the model components for the cross-subject models. We observed a similar trend to that seen with the within-subject models: mainly the performance decrease of the ReLU activation function, No_dropout, and Not_branched and the performance increase of Not_deep and the ELU activation function (figure 3(b)).

Because the addition of a 400 ms large filter branch is comparable to applying a low pass filter, which is of relevance in ERP identification, we additionally evaluated the performance using only this extra large branch in the CNN model. As shown in figure 3(a), removing the large filter branch (the Not_branched model) led to a 3.1% and 0.8% decrease in the decoding accuracy of the within-subject models applied to the lower- and higher-dimensional datasets, respectively, while removing the deep layers of the small filter branch (the Not_deep model) yielded a 0.2% and 0.4% increase in the decoding accuracy, respectively. Additionally, training within-subject models on the lower- and higher-dimensional datasets after removing the whole small filter branch leaving only the large filter led to a decrease in decoding accuracy of 13.2% and 2.4%, respectively.

**Fine-tuning and transfer learning**

Training cross-subject branched models yielded average decoding accuracies of 79.9% and 81.2% on the f50_avg and f250 datasets, respectively (figure 2(b)). Starting with these models, we continued to fine-tune them with the test participants’ data. We used 10, 20, 30, 50, 60 and 70 trials to further train the models and re-evaluated them using the remaining data (figure 4). This approach simulates the situation when a new participant uses the model out-of-the-box and tunes it with subject-specific training trials. Incrementally adding subject-specific trials, a significant improvement in accuracy was observed for both datasets from 30 additional trials. After using 70 trials, the models achieved decoding accuracies of 91.8% (SD: 9.2%), and 91.1% (SD: 9.6%) on the f50_avg and f250 datasets, respectively.

These results cannot be compared directly to the results of the cross-validation scheme of the within-subject models trained from random initializations because of the difference in the test schemes. We therefore trained within-subject models with the same cross-validation scheme but starting from the cross-subject model as a starting point instead of random initializations. These new within-subject models achieved decoding accuracies of 91.9% (SD: 8.5%), and 93% (SD: 6.7%) on the f50_avg and f250 datasets, respectively. They showed improvements over the 90.6% and 90.7% reported for cross-subject models (figure 2(b)). These improvements were significant for both datasets (f50_avg: p < 0.05; f250: p < 0.001).

**Saliency maps**

To create the within-subject saliency maps, the single-example saliency maps (see figure 5) for all the target examples in the participant’s dataset were averaged. We examined the within-subject saliency maps of two participants for the branched model, who had decoding accuracies of 98.4% and 73.6% respectively, and were chosen to illustrate saliency maps for high- and low-performing participants (figure 6). The features expected to drive the classification using this paradigm are identifiable on inspection of the saliency map of the high-performing participant: central and parietal electrode locations (FC2, CP1, CP2, and P7) show salient features 300–500 ms post-stimulus. The saliency map of the low-performing participant, on the other hand, has a more scattered appearance across electrodes and over time, which could be due to noisy data leading to low decoding accuracy and poor noisy saliency maps.

The cross-subject saliency map (figure 7) is more suitable for visualizing more general task-relevant features, as the cross-subject model is less likely to pick up subject-specific,
task-irrelevant features (see the difference for a single input example in figure 5). These subject-specific features could help improve the decoding accuracy of within-subject models but at the expense of their generalizability to other participants. Examining the grand average ERP at the most discriminative electrode, we observed some resemblance between the ERP waves and the gradients shown on the saliency map (figure 7). It is important to note that the resemblance is not exact as the models were trained on a set of single trials which differed from the average wave. We observe that the grand ERP for the target stimuli shows higher amplitude than the grand ERP of the standard stimuli around 200 ms and after 300 ms and lower amplitude around 100 ms and 300 ms. These differences are reflected on the saliency maps by positive and negative gradients. Note that the negative difference at around 100 ms is not represented on the saliency map which could be because the difference is small and not relevant for the classification task.

**Spatial and temporal features extraction**

CNN models were found to be especially powerful in dealing with high-dimensional data (J250), as they are capable of automatically extracting task-relevant features from the data. Therefore, besides gaining insights from the saliency maps by visual inspection, it would be also useful to quantify them in order to learn the relevant spatial and temporal features that drive the network decision. To this end, we quantified the saliency maps by computing the average of their absolute values across space and time.

Spatial features are defined as the most discriminative electrodes across the whole trial time period. The eight most discriminative electrodes for each participant separately computed on the within-subject saliency maps are illustrated in figure 8(a). We can observe that they were variable across participants, but were mainly located around central and parietal electrodes, with some participants having temporal, occipital, or frontal extensions. The eight most discriminative electrodes extracted from the cross-subject saliency map also showed central and parietal distribution (figure 8(b)).

Temporal features are defined as the most discriminative time periods for the detection of the target stimuli. The most discriminative 100 ms time window across all the electrodes for within-subject models and for the cross-subject model fell in the interval between around 300 and 500 ms post-stimulus (figure 9).

These spatial and temporal features correspond with the well-established spatial distribution and time period of the P300 component [43]. The distribution over time of the normalized gradients, averaged across all the electrodes for each time point for both the within-subject and cross-subject saliency maps, resembled a trimodal distribution, with the highest
contribution coming from the time period between 300 and 500 ms, followed by contributions from the time period between 100 and 200 ms and around 600 ms (figure 10). Each peak in the distribution could be interpreted as reflecting a specific neural process in the brain that is modulated by covert attention at a specific time period.

Validation of the extracted features

We validated the interpretation of the absolute values of the gradients in the saliency maps as the discriminative power of the features by comparing the decoding accuracy using all electrodes with that obtained using only the eight most and the eight least discriminative electrodes, setting all other electrodes to zero in the latter cases (figure 11(a)). Inside the cross-validation scheme of evaluating the performance of the within-subject branched models, we used the trained models to generate saliency maps, from which we computed the rank of the electrodes (as mentioned previously and shown in figure 8(a)). Visual inspection clearly reveals a difference in the decoding accuracies in the case of using the eight most discriminative versus the eight least discriminative electrodes (figure 11(b)). Using only eight out of the 29 electrodes means using only 27.5% of the features. Using the eight most discriminative electrodes, we retained 72% of the decoding accuracy versus only 16% in the case of using the eight least discriminative electrodes.

Furthermore, we compared the eight most discriminative electrodes determined with the saliency map approach to an approach that is based on considering the weights (spatial filters) of the first hidden layer [8, 36]. We trained within-subject LDA classifiers with shrinkage using only the eight most discriminative electrodes determined by both approaches. There was no significant difference between the two approaches, as using saliency map and spatial filter approaches yielded decoding accuracies of 80.70% and 81.19%, respectively.
Moreover, by comparing the eight most discrimina-
tive electrode sites for the cross-subject model, there was only
a difference in one electrode (replacing electrode C3 from
figure 8(b) by electrode CP2).

**Discussion**

We have demonstrated that CNNs can be employed to decode
electrophysiological signals for P300 BCI applications and
can furthermore provide insights into the underlying brain
processing when combined with visualisation using saliency
maps. CNNs performed at least on a par with commonly used
decoding methods, and their performance was significantly
superior when applied to high-dimensional, minimally-
processed data on both within-subject and cross-subject
decoding tasks. CNNs also achieved higher accuracies on
the high-dimensional data than on the low-dimensional data
with hypothesis-driven feature extraction. This superior per-
formance can be attributed to their ability to extract optimal
relevant features automatically from the data that leverage
separability of the data. This applicability to high-dimensional
data has particular potential in novel decoding problems, in
which we have no *a priori* information regarding the most
discriminative signal. For example, in the covert attention-
decoding problem, the use of downsampling after applying a
moving average filter is permissible, because it is known that
the P300 signal is of low frequency in relation to the EEG
acquisition sampling frequency [24, 31]. For cases in which the discriminative signals are represented in high frequency bands, this method would cause attenuation of the discriminative information and therefore reduction of the decoding accuracy. CNNs also allowed transfer learning through the process of network fine-tuning. Fine-tuning the cross-subject branched model led to a significant increase in the decoding accuracy of the within-subject models in comparison to training from random initialization. Such transfer learning approach would potentially reduce the time needed for acquisition of training data, which in turn would make BCI systems more accessible for patients who need them.

A limitation of the approach is that optimizing CNN models is a time-consuming and computationally expensive process in comparison to training a simple linear classifier like the LDA. The models usually contain tens or hundreds of hyper-parameters that need to be tuned to achieve reasonable performance. These hyper-parameters are data-dependent and are tuned mainly through intensive initial parameter adjustment, intuition, and experience with the data. There is, however, a growing research interest in automated neural architecture search [35, 42], but it is still computationally expensive.

Furthermore, the results showed that not all the standard choices of these hyperparameters led to the best results. For example, applying the commonly used ReLU activation function led consistently to poorer results in comparison to tanh or ELU activation functions. This is probably because the ELU and tanh activation functions have similar properties, such that they both have positive and negative output values in contrast to the ReLU activation function, which has only positive output values. Using the dropout regularization technique, however, caused significant improvement of the model’s performance, which was particularity observable when the dataset was small (within-subject) and high-dimensional (f250). Moreover, the finding that removing the large filter branch reduced decoding accuracy, while decoding accuracy actually improved slightly on removal of the deep layers of the small filter branch suggests that parallel convolutions are more suitable for EEG data decoding than the deep ones. Moreover, we note that to achieve the best EEG decoding results in our case, both large and small filters were required to run in parallel on the time domain data. We also observe that the effect of the filter sizes is smaller on the high-dimensional dataset, which shows the versatility of CNNs in discovering the optimal features for the classification problem at hand when they are applied to high-dimensional minimally-processed data. Furthermore, batch normalization did not provide a significant improvement in contrast to what was reported in [36], which fits with the notion that designing the best network architecture is data-dependent.

The fact that CNNs can work directly on raw, high-dimensional data and automatically extract relevant features poses an opportunity to discover information about the cognitive task at hand, if we can visualize these relevant features. We showed how saliency maps can be used for feature visualization in EEG data. It was possible to visualize the discriminative power of the features on a single example level, which is suitable for single-trial EEG analysis, on a within-subject level, which is relevant for studying between-subject variability, and on a cross-subject level, which is necessary for studying general task-relevant features. Moreover, quantifying the discriminability of the features across electrodes or time provided insight into the task-relevant electrodes and time periods. The cross-subject saliency map showed that the eight most discriminative electrodes were PO8, P4, P7, Pz, Cz, C3, P3, and C4 (figure 8(b)). This finding is in agreement with the 8-channel montage optimized by [31] and [30] for P300 spellers in 5 electrodes (PO8, Cz, Pz, P3, and P4). Additionally, their montage included Oz, P07, and Fz, which with our approach were ranked at positions 18, 13, and 16 respectively. The saliency map approach determined, on the other hand, the electrodes P7, C3, and C4 to be more predictive, in agreement with the

![Figure 10](image-url). Each time point represents the normalized average of the absolute value of the gradients across space. It shows how discriminative each time point is and therefore how the discriminability of the EEG features develops over time. The within-subject curve is averaged across the 19 participants with the shade denoting the standard deviation.
findings of Brunner et al [6], who reported that this 8-channel montage is suboptimal when the participants do not fixate their gaze on the stimulus. The branched model identified central and parietal electrodes as being more discriminative than occipital and parieto-occipital ones. These locations are in line with the parietal and central distribution of the P300 component, which is known to be elicited in covert attention paradigms, in contrast to the occipital and parieto-occipital distribution of VEPs, known to be modulated in overt attention paradigms [54]. The difference between covert (as performed here) and overt attention paradigms is that participants foveate the target stimulus when attention is overt. Furthermore, quantifying the cross-subject saliency map across time (figures 9 and 10) showed that the most discriminative time period started from 300 ms post-stimulus. There are still contributions from the time period between 100 and 200 ms post-stimulus, which corresponds to the attentional modulations of VEPs, but they are less prominent. It has been reported that for overt attention tasks, the time period from 180 to 250 ms post-stimulus is the most informative, when P2 (second positive component) and N1 (first negative component) occur. On the contrary, for covert attention tasks the post-300 ms time period is the most informative, in which the P300 component occurs [54].

Within-subject saliency maps could also be used to investigate the variability between participants. Examination of the variability of the eight most discriminative electrodes across participants showed that most participants have a central and parietal distribution of the relevant electrodes with occipital extension and sometimes a frontal one (figure 8(a)). The discriminative time periods are mostly concentrated in the time period between 300 and 500 ms post-stimulus across participants (figures 9 and 10). Determining between-subject variability and within-subject variability (by comparing the saliency maps of early and late trials) could have a potential application in the development of biomarkers to diagnose and monitor the prognosis of certain neurological or psychiatric disorders, e.g. prolonged P300 latency in dementia patients [43].

Another approach for determining the most discriminative electrodes is to consider the weights of the first hidden layer [8, 36] and interpret them as spatial filters. Taking the average of the absolute values of the layer’s spatial filters gives an estimate of the discriminative power of the electrodes [8]. This approach, however, has disadvantages. First, it is architecture-dependent, because it can only be applied to the first hidden layer in cases in which this layer is a spatial convolution layer. Secondly, it does not account for the transformation that will be applied to the output of this layer in the subsequent layers. Thirdly, it can only extract spatial features but not temporal or spatiotemporal features, so for example, it can not provide information about the most discriminative time periods. Comparing the saliency map approach for relevant feature extraction with this approach showed that the saliency map approach is at least as effective in extracting spatial features. Additionally, it has the advantages of being architecture-independent and able to extract temporal features as well, which can potentially help explore the brain dynamics associated with a cognitive task.

It is also worth noting that these saliency maps can potentially offer an objective heuristic for doing ERP research. An objective approach could be achieved through training a CNN model to classify the different conditions under investigation, and by visualizing the saliency maps for each class, we could gain an indication of where and when to look for different neural correlates for the conditions in a data-driven way.

Conclusion

Convolutional neural networks performed significantly better than traditional decoding techniques on the high-dimensional data. However, this superior performance comes at the expense of time and computational costs. Using the standard recommended model components does not guarantee the best performance and therefore manual optimization is still needed. More studies that validate and potentially replicate our design
recommendations are also still needed. Thus, we believe that for cases, in which we have domain knowledge of the classification problem, like in the case of P300 classification, using a simple linear classifier on top of a feature extraction method that is based on this domain knowledge, offers a more practical solution. On the other hand, in novel cases, in which we do not have sufficient prior knowledge of the task-relevant effects on the EEG signal, using CNN models is justifiable and recommended. Training CNN models in these cases is not only more likely to produce performances competitive with other methods, but also to provide insight into the task-relevant features through visualization techniques like saliency maps.

Future research

For such models to be applicable in real world situations, they need to be tested on other datasets. Further investigation is required to establish whether they could be used ‘out of the box’ with reasonable performance, needing only some fine-tuning with a small amount of data, or whether retraining from scratch is necessary.

Moreover, there is still room for improvement, through exploring different neural network architectures, such as recurrent neural networks (RNN) [18], long short term memory networks (LSTM) [21], hybrid networks, and automatic neural architecture search techniques [35, 42].

Using CNN models for EEG decoding still needs further development before such models could become the standard approach and replace other techniques. The research in this area needs to go beyond intensive initial parameter adjustment and start to investigate the underpinnings of the models when they are applied to EEG data.

Acknowledgments

This work has been partly funded by the Federal Ministry of Education and Research, grant number 13GW0095/D.

ORCID IDs

Amr Farahat https://orcid.org/0000-0002-2141-8217
Christopher Reichert https://orcid.org/0000-0002-8649-9791
Catherine M Sweeney-Reed https://orcid.org/0000-0002-3684-1245

References

[1] Abadi M et al 2016 Tensorflow: a system for large-scale machine learning 12th USENIX Symp. on Operating Systems Design and Implementation (Savannah, GA: USENIX Association) pp 265–83
[2] Amini Z, Abootalebi V and Sadeghi M T 2010 A comparative study of feature extraction methods in P300 detection 17th Iranian Conf. Biomed. Eng. ICBME pp 1–4
[3] Ancona M, Ceolini E, Öztireli C and Gross M 2018 Towards better understanding of gradient-based attribution methods for deep neural networks Int. Conf. Learn. Representations (arXiv: 1711.06104)
[4] Blankertz B, Lemm S, Tredter M, Haufe S and Müller K R 2011 Single-trial analysis and classification of ERP components—a tutorial Neuroimage 56 814–25
[5] Blankertz B, Müller K R, Krusienski D, Schalk G, Wolpaw J, Schlögl A, Pfurtscheller G, Millán J, Schröder M and Birbaumer N 2006 The BCI competition III: validating alternative approaches to actual BCI problems IEEE Trans. Neural Syst. Rehabil. Eng. 14 153–9
[6] Brunner P, Joshi S, Briskin S, Wolpaw J R, Bischof H and Schalk G 2010 Does the ‘P300’ speller depend on eye gaze? J. Neural Eng. 7 056013
[7] Carabzé E, Sugi M, Nambo I and Wada Y 2017 Convolutional neural networks with 3D input for P300 identification in auditory brain–computer interfaces Comput. Intell. Neurosci. 2017 8163949
[8] Cecotti H and Graser A 2011 Convolutional neural networks for P300 detection with application to brain–computer interfaces IEEE Trans. Pattern Anal. Mach. Intell. 33 433–45
[9] Chollet F et al 2015 Keras (https://keras.io)
[10] Cinotti F, Pichiorri F, Aricò P, Aloise F, Leotta F, de Vico Fallani F, Millán J del R, Molinari M and Mattia D 2012 EEG-based brain–computer interface to support post-stroke motor rehabilitation of the upper limb 2012 Annual Int. Conf. of the IEEE Eng. Med. Biol. Soc. (IEEE) pp 4112–5
[11] Clevert D-A, Unterthiner T and Hochreiter S 2016 Fast and accurate deep learning network by exponential linear units (ELUs) Int. Conf. on Learning Representations (arXiv: 1511.07289v5)
[12] Courville I G, Bengio Y and Aaron C 2016 Deep Learning (Cambridge, MA: MIT Press)
[13] Demšar J 2006 Statistical comparisons of classifiers over multiple data sets J. Mach. Learn. Res. 7 1–30
[14] Di Russo F, Martínez A and Hillyard S A 2003 Source analysis of event-related cortical activity during visuo-spatial attention Cerebral Cortex 13 486–99
[15] Farwell L and Donchin E 1988 Talking off the top of your head: toward prosthesis utilizing event-related brain potentials Electroencephalogr. Clin. Neurophysiol. 70 510–23
[16] Glorot X, Bordes A and Bengio Y 2011 Deep Learning Proc. of the 13th Int. Conf. Artif. Intell. Stat. vol 9, ed Y W Teh and M Titterington (PMLR) pp 249–56
[17] Glorot X, Bordes A and Bengio Y 2011 Deep sparse rectifier neural networks Proc. of the 14th Int. Conf. on Artificial Intelligence and Statistics vol 15, ed Gordon G et al (PMLR) pp 315–23
[18] Graves A, Mohamed A R and Hinton G E 2013 Speech recognition with deep recurrent neural networks 2013 IEEE Int. Conf. Acoust. Speech Signal Process. pp 6645–9
[19] Guy V, Soriani M-H, Bruno M, Papadopoulo T, Desnuelle C and Clerc M 2018 Brain computer interface with the P300 speller: usability for disabled people with amyotrophic lateral sclerosis Ann. Phys. Rehabil. Med. 61 5–11
[20] Haibo He H and Garcia E 2009 Learning from imbalanced data IEEE Trans. Knowl. Data Eng. 21 1263–84
[21] Hochreiter S and Schmidhuber J 1997 Long short-term memory Neural Comput. 9 1735–80
[22] Hoffmann U, Vesin J M, Ebrahimi T and Diserens K 2008 An efficient P300-based brain–computer interface for disabled subjects J. Neurosci. Methods 167 115–25
[23] Hubel D H and Wiesel T N 1962 Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex J. Physiol. 160 106–54
[24] Intriglione J and Polich J 1994 On the relationship between background EEG and the P300 event-related potential Biol. Psychol. 37 207–18
