Proceedings of the
8th Graz Brain-Computer Interface Conference 2019 DOI: 10.3217/978-3-85125-682-6-49

SIMPLE DEEP NEURAL NETWORKS SHOW STATE-OF-THE-ART PERFORMANCE IN ERP-BASED BCI

L. Delobel[1], E. Maby[1] & J. Mattout[1]

[1]Lyon Neuroscience Research Center, CRNL; INSERM, U1028; CNRS, UMR5292;
Brain Dynamics and Cognition Team, Lyon, F-69000, France
[2]University Lyon 1, Lyon, F-69000, France
E-mail: loic.delobel@gmail.com

ABSTRACT: Deep neural networks (DNN) have great success in solving difficult recognition tasks such as speech and image processing. However, performance depends on the amount of data available for training the network. In BCI, very large datasets are still missing and EEG data are particularly complex and noisy. Nevertheless, it is of interest to investigate whether DNN may prove efficient in this context. We tested 6 different deep learning models to accomplish binary classification of single-trial ERPs and compared them with Riemannian Geometry based classifiers. Each model implements a different architecture and uses two different input formats: an image (`2D`) or a video (`3D`). All models were tested on two different datasets, and under three different scenarios: within-subject, cross-subject and cross-experiment classification. Finally, to get insights about the decision process of the most successful DNN, we visualized the learned features using saliency maps. This revealed informative and interpretable differences between the two empirical datasets used for evaluation.

INTRODUCTION

This work focuses on the application of deep learning methods to EEG recordings for Brain-Computer Interfaces (BCIs) [1]. A BCI is a device that allows interaction with a machine through brain signals, bypassing normal neuromuscular outputs. For example, the P300 Speller [2] is a BCI which helps to restore communication with people who cannot control their muscles anymore (e.g. patients with Amyotrophic Lateral Sclerosis). The P300 Speller exploits the P300 event-related potential (ERP) and performs a binary classification task: detecting the presence or not of a P300 wave. To accomplish this task, Riemannian Geometry classifier (RGC) and shrinkage based linear discriminant analysis classifier (sLDA) seem to be, to date, the current state-of-the-art method [3].

Deep learning methods have emerged from the connectionism movement in cognitive science that hopes to explain intellectual abilities using connectionist architectures known as artificial neural networks (ANN [4]). Connectionist architectures have existed for more than 70 years, but new architectures and graphical processing units (GPUs) brought them to the forefront of artificial intelligence. Depending on their architecture, neural networks excel at speech and language recognition (RNN, LSTM [5]) or image and video recognition with convolutional neural networks (CNN [6]). CNN exploit the local connectivity concept that makes them suitable for EEG data. Although still few in number, recent studies have shown that deep learning methods can be effective in classifying ERP components [7–10].

In the current study, we compare the performance of 6 deep learning network architectures for application to a P300 Speller [2] (26 subjects) and a RSVP [2] (11 subjects) paradigm, both exploiting the P300 and N200 components. This makes it possible to observe the behavior of deep learning methods in two different paradigms. We introduce different deep learning network architectures and compare them with state-of-the-art methods that combine XDAWN algorithm [11] and Riemannian geometry [12]. In order to explore the intra-subject and inter-subject generalization abilities, we compare the performance of the classifiers for two different scenarios: Within-Subject testing and Cross-Subject testing. In addition, the best model resulting from these scenarios is tested in a cross-experiment scenario where it is trained with data coming from one experiment and tested on the other.

MATERIALS AND METHODS

Datasets: Two datasets were used for evaluation and model comparisons. The first one is denoted as `P300 Speller` and implements the P300 speller protocol [2]. This protocol consists in displaying a grid of 36 symbols on screen (the 26 letters of the alphabet, the underscore and the 9 digits) and asking the user to focus attention on the desired item. The flashing of the items was similar to that reported in [13], i.e., the splotch stimulus presentation. Twelve pre-defined groups of 6 non-adjacent items are flashed (one at a time) in a pseudo-random order. Each item belongs to two groups only and these two groups have only this particular item in common. A P300 evoked potential is expected when the desired symbol is flashed. A single pass on all groups (all symbols twice) is referred to as a *repetition*. The `P300 Speller` dataset is an electroencephalographic (EEG) collection of 56-channel recorded from

 DOI: 10.3217/978-3-85125-682-6-49

Table 1: Description of the datasets

Name	Rsvp	P300 Speller
Type	RSVP	P300 Speller
Subjects	11	23
Sensor	48	48
Repetitions	10	2 to 4
Sampling Frequency	200	600
Down-sampled to	100	100
Band pass filtered	1Hz-20Hz	1Hz-20Hz
Epoched between	0ms and 600ms	0ms and 600ms

23 healthy subjects during one of our previous studies. We made the spelling challenging by considering very short (2 repetition-long) and short (4 repetition-long) trials [14].

The second dataset denoted as `Rsvp` and available online [15], was acquired during a Rapid serial visual presentation (RSVP) task. The RSVP paradigm is a variant of the P300 speller where all the symbols are presented one-by-one in a serial manner and in the center of the screen (on fovea). This dataset is made of data from 12 healthy subjects but we discarded the data of subject `VPgce` because of the use of a different set of channels. In this experiment, item selection relied on 10 repetitions (see [15] for a thorough description of the paradigm).

Data formats and pre-processing: Table 1 summarizes the main characteristics and pre-processing steps for the two studied datasets. They were preprocessed in the same manner: for each subject, raw signals were band-pass filtered between 1Hz and 20Hz and downsampled to 100Hz. Epochs were extracted from 0ms to 600ms after stimulus onset, resulting in two big matrices (containing the Target/Non-Target samples) $X \in R^{N \times C \times T}$ with the number of EEG sensors denoted by C, time samples denoted by T and the number of epochs denoted by N. We selected 48 electrodes common to both datasets (F1-F8-Fz, FC1-FC6-FCz, T7-T8, C1-C6-Cz, TP7-TP8, CP1-CP6-CPz, P1-P8-Pz, PO7-PO8-POz, O1-O2).

We considered two different data formats for subsequent classification with deep learning models. The first one is the standard method where each sample is a matrix $X \in R^{C \times T}$ with the number of EEG sensors denoted by C and time samples denoted by T. This is more convenient for an online application where no additional transformations are needed. But the disadvantage is that the classifier is dependent on the number of electrodes on which it has been trained. This format is denoted by `2D` from now on. The second format aims at alleviating the disadvantages of the first format. Considering the X matrix presented above, it consists of interpolating the space vector (sensors) into a square matrix (16×16) for each time sample. It results in a matrix $X \in R^{H \times W \times T}$ with $H = 16$ and $W = 16$ which correspond to a topographic scalp map evolving over time (a video). Therefore we used a 2D clough tocher (cubic) interpolation [16]. Having a spatial representation (as a pixelized image) allows better processing of datasets with different EEG electrode montages. It also makes perfect sense to apply such a spatial smoothing, because of the blurring effect of head tissues. Because of their excellent performance on natural images, we expected convolutional neural networks to better handle this kind of format [17, 18]. However, we must bear in mind that such a transformation has a cost (computation time) and can complicate the real-time pipeline. It also increases the number of degrees of freedom, hence the complexity and training time. This format is denoted by `3D` from now on.

Models and training procedures: Riemannian geometry based classifiers – We compared our deep learning models with two classifiers based on Riemannian geometry which outperformed other approaches in BCI competitions. This method is considered as a strong reference in the field [12]. We used a Riemannian method based on tangent space mapping which shows overall better performance compared with the minimum distance to mean algorithm. A recommended pipeline is to estimate XDAWN covariances and project them into the tangent space, then classify with logistic regression [12]. We used 2 components to estimate the XDAWN covariances corresponding to the 2 classes (Target/Non-Target). XDAWN reduces the dimensionality of data, which facilitates and accelerates training. This pipeline is denoted as `XDAWN + Riemann`. We also used a variant with no XDAWN filters as XDAWN is not very effective in transfer learning situations. This pipeline is denoted as `Riemann`.

Deep learning classifiers – With the use of the two data formats, we developed 6 deep learning models that exploit the temporal and spatial dimensions of the data. Also, EEG signals are known to have inherent temporal and spatial smoothness properties. In other words, close features in space and time dimension are dependent. Thus, we wanted to test different architectures with different assumptions about local dependencies over space and time. CNN and GRU (Gated Recurrent Unit) layers are both strong candidates to accomplish this task. CNN layers are known to be very effective for the classification of image-like data with spatial pixels. As for the GRU layers, they enable the learning of temporal dependencies. GRU layers are a less resource-intensive alternative to LSTMs, while maintaining good performance. We first developed three models that take the `2D` format as input. `MLP` is a multilayer perceptron (MLP) composed of 1 hidden layer. This is the simplest architecture: it does not take into account the temporal and spatial dependencies in EEG data. The `CNN1D_T2` is composed of a convolutional layer. It is inspired by the proposed EEGNet one [9] but only one convolution operation was done to learn both spatial and temporal features in one go. We found empirically that splitting the operation into two steps does not improve performance. The learned filters are also more easily interpretable. This is a 1-dimension convolution because the convolutional filters are moved in only one direction. The 10 filters of size 48×2 represent a

 DOI: 10.3217/978-3-85125-682-6-49

time window of $\simeq 10ms$ (2 timesteps at 100Hz). The feature maps are passed directly to the final `LogSoftMax` layer with no additional dense layers in-between (non-linear activations). This has been proven to work effectively while reducing the number of parameters [19]. The `CNN1D_T2_GRU` is the `CNN1D_T2` with an additional output GRU layer. A GRU layer is a recurrent layer that allows the learning of temporal dependencies [20]. We then developed three other models that take the `3D` format as an input: The `CNN2D_S3` exploits the 3-dimensional input format by learning spatial features with a 2-dimensional convolutional layer applied to the spatial dimension. The weights of the filters are shared across the spatial dimension but not across the temporal dimension. The `CNN2D_S3_T2` is a variant of `CNN2D_S3` that shares the weights of the convolutional filters over the temporal dimension. A 3-dimensional convolutional layer is needed to perform this operation. The `CNN2D_S3_T2_GRU` is the `CNN2D_S3_T2` with an additional output GRU layer.

All models also include a batch normalization layer that influences the training by normalizing the data before they enter the hidden layer and the LogSoftmax layer at the end, before predicting the output probabilities pertaining to the two classes.

All deep learning models were trained with the `RMSProp` optimization algorithm that allows adapting the learning rate during training. We used an early stopping strategy during training, meaning that the training was interrupted as soon as the valid loss started to increase. Precisely, we train for N epochs (big enough not to fall into an under-fitting situation) and we go select the epoch with the smallest valid loss. The learning rate was set to 10^{-4} or 10^{-5} depending on the experiment. We set the batch size to 128 and the optimizer momentum to 0.9 as recommended in [21]. We performed L2-norm regularization with a weigh decay of 5^{-4}. All deep learning models terminate with a `LogSoftMax` layer. The negative log-likelihood loss was used for our two-class classification problem (Target/Non-Target). No manual rescaling weight or over/undersampling methods were used to balance the samples between the two classes. The unbalanced dataset is not problematic because of the final averaging (in a Bayesian fashion, see below).

Evaluation metrics and statistics: In such a BCI protocol, there are two levels of classification, hence two levels of accuracy. The first one is the accuracy at the level of the binary classifier (number of correctly classified Target/Non-Target trials). The second one is the accuracy corresponding to the number of correctly classified (or spelled) symbols in a given sentence. We only report the latter that corresponds to the one of interest for BCI use. Initially, all items are assumed to be equiprobable targets. At each new observation (after computing the probability for the current sample to be a target), this belief is updated following Bayes rule, by optimally combining the data likelihood and prior and by considering the posterior belief as the prior for the next observation. In this way, all items of the sentence are predicted. At the end of the sentence, we compute the final spelling accuracy (number of correct symbols) with different repetitions. Obviously, better accuracy is expected as repetitions increase. For statistical significance, we used a one-way ANOVA (III) and post-hoc Tukey Test on the spelling accuracy.

Evaluation procedures: We evaluated the spelling accuracy of the classifiers with three different scenarios: within-subject, cross-subjects and cross-experiments.

Within-subject testing – We first explored the intra-subject generalization. In both datasets, each subject completed a calibration phase and a test phase. In order to simulate online classification, we trained our classifiers using calibration data only and tested on the test phase data. To avoid overfitting, we perform cross-validation in a way that the data from the subject's calibration phase is split to form the training (80%) and validation (20%) sets. The dataset's test phase simply formed the test set. Thus, N tested subjects lead to N fits composed by a training, validation and test set.

Cross-subjects testing – Many studies attempt to reduce the time of the calibration phase. In fact, the calibration stage is exhausting and demotivating for the subject. The ideal situation would be to completely free oneself from it. To assess the ability of the classifiers to generalize from a pool of subject to a new subject, we built a scenario where each subject is tested with a classifier which has been trained with the calibration plus test data of all other subjects. More specifically, the data from each subject's calibration and test phase minus the tested one were merged to form the training and validation sets (80%/20%). The data from the test phase of the tested subject simply form the test set. Thus, N tested subjects lead to N fits composed by a training, validation and test set.

Cross-experiment testing – To further assess the ability of the models to generalize across different experiments involving different subjects but also different protocols, we chose the model that performed best in the two previous scenarios and tested it in a cross-experiment scenario. We trained the chosen model on the calibration data of the `P300 Speller` dataset plus the full `Rsvp` dataset except tested subject.

Exploring the learned features:

Deep learning models are able to automatically extract the relevant features for classification, in an unsupervised manner. Apart from the choice of design and hyperparameters, no a priori knowledge was injected into the model to accomplish the classification task. Although interesting and useful, this property makes it difficult to apprehend the rationale of the decision process that the model implemented. Thus, the purpose of this experiment was to reveal the features that the neural networks identified as most relevant in order to classify a sample as a target or non-target response. Also, understanding what neural networks have learned may point to interest-

 DOI: 10.3217/978-3-85125-682-6-49

Table 2: Number of model parameters and training time (approximate ratio) on Rsvp dataset.

Classifiers	#Params	Within	Cross
Riemann XDWAN	-	1x	1x
Riemann	-	12x	150x
MLP	5862	6x	11x
CNN1D_T2	1582	11x	28x
CNN1D_T2_GRU	1662	52x	85x
CNN2D_S3	4242	44x	192x
CNN3D_S3_T2	3287	83x	450x
CNN3D_S3_T2_GRU	2317	152x	780x

ing and unknown neurophysiological phenomena. In image classification, a common approach to understand the decisions of a classifier is to find regions of an image that were particularly influential (saliency maps). We chose a simple technique which allows elucidating the relevant pixels in the input image [22]. We chose the CNN1D_T2 model to test this method. It takes the 2D format as an input.

RESULTS

We used PyTorch to build the models. The code to reproduce the results and a more detailed description of the models are available online.[1] Figure 1 shows the accuracy performance at 2 repetitions for all classifiers applied to the two datasets in Within-subject testing and Cross-subject testing. Accuracies can be interpreted as online accuracies because only calibration data were used for training. Table 2 shows the training time for each classifiers on Rsvp dataset. The number of model parameters is also reported. The training times are approximated and have been calculated by multiplying the time of an iteration by the number of iterations (for deep learning models). Note that the training time of a model has been chosen to ensure that the model has been fully trained. Therefore, the optimal training time of a model may be less than the one indicated.

Within-Subject testing:

Result 1 – Deep learning classifiers that use the 2D format as input do as well as Riemann's classifiers on both datasets.

Result 2 – Deep learning models that use the 3D input format are statistically less effective than others on both datasets (p-value < 0.05).

Result 3 – Training deep learning classifiers took substantially longer than Riemann ones, at least for the most complex ones.

Cross-Subject testing:

Result 4 – XDAWN + Riemann is significantly worse than Riemann (p-value < 0.05).

Result 5 – Overal Cross-Subject testing performance is significantly worse than the Within-Subject testing (p-value < 0.05).

[1] https://git.io/fj317

Result 6 – All classifiers (except CNN3D_S3_T2_GRU) do as well as Riemann on both datasets (p-value < 0.05).

Cross-Experiment testing: Figures 2 shows the accuracy performance for the Cross-Experiment testing.

Result 7 – Cross-Experiment performance is no better than Cross-Subject performance.

Exploring the learned features: – Figure 3 shows, for CNN1D_T2 model, the absolute value of the difference between the mean gradient of the Target samples and the mean gradient of the Non-Target samples. In other words, pixels with relatively high gradient values are useful for distinguishing target from non-target samples.

Result 8 – CNN1D_T2 has learned well known features on both datasets (N200 around 200ms on P300 Speller dataset and P300 around 300ms on Rsvp dataset).

Result 9 – CNN1D_T2 focuses on earlier features (around 200ms) when trained on P300 Speller dataset compared to when trained on Rsvp dataset (around 300ms).

DISCUSSION

This study explored the performance of deep neural networks for feature extraction and classification in the context of ERP-based BCIs, namely the P300-Speller and RSVP paradigms. Overall, we showed that deep learning methods with an appropriate architecture can perform as well as Riemannian geometry for intra-subject and cross-subject generalization, on both paradigms.

A surprising result is that the MLP model, a very simple one, proves able to perform as well as the best models, even in a transfer learning situation. Keeping in mind that the raw data are band-pass filtered in the low frequencies and downsampled, this certainly eases the learning in the time domain. Besides, we noticed that the Batch Normalization step plays a major role in the convergence of the models. Hence a MLP could be a strong candidate for BCIs, since its simplicity implies short training time.

In this study, We also compared two different input formats for EEG data: an image (2D) versus a topographic video of the scalp (3D). From a practical point of view, the first one is easy to use and does not require much computing power but requires that the model is compatible with the montage (the number and position of sensors). Conversely, the latter is convenient for mixing datasets from different experiments with potentially different setups. However, transforming EEG signals into a topographic video is time-consuming and can become challenging in real-time. In terms of performance, we found that the suggested models that use the 3D format are no better than those that use the 2D format. Although it might be that other model architectures might be able to extract more information for the video format, our results suggest that despite its flexibility with regard to the number of channels, it did not yield higher performance in the tasks we studied.

Note that obviously, these results do not reflect a fully

 DOI: 10.3217/978-3-85125-682-6-49

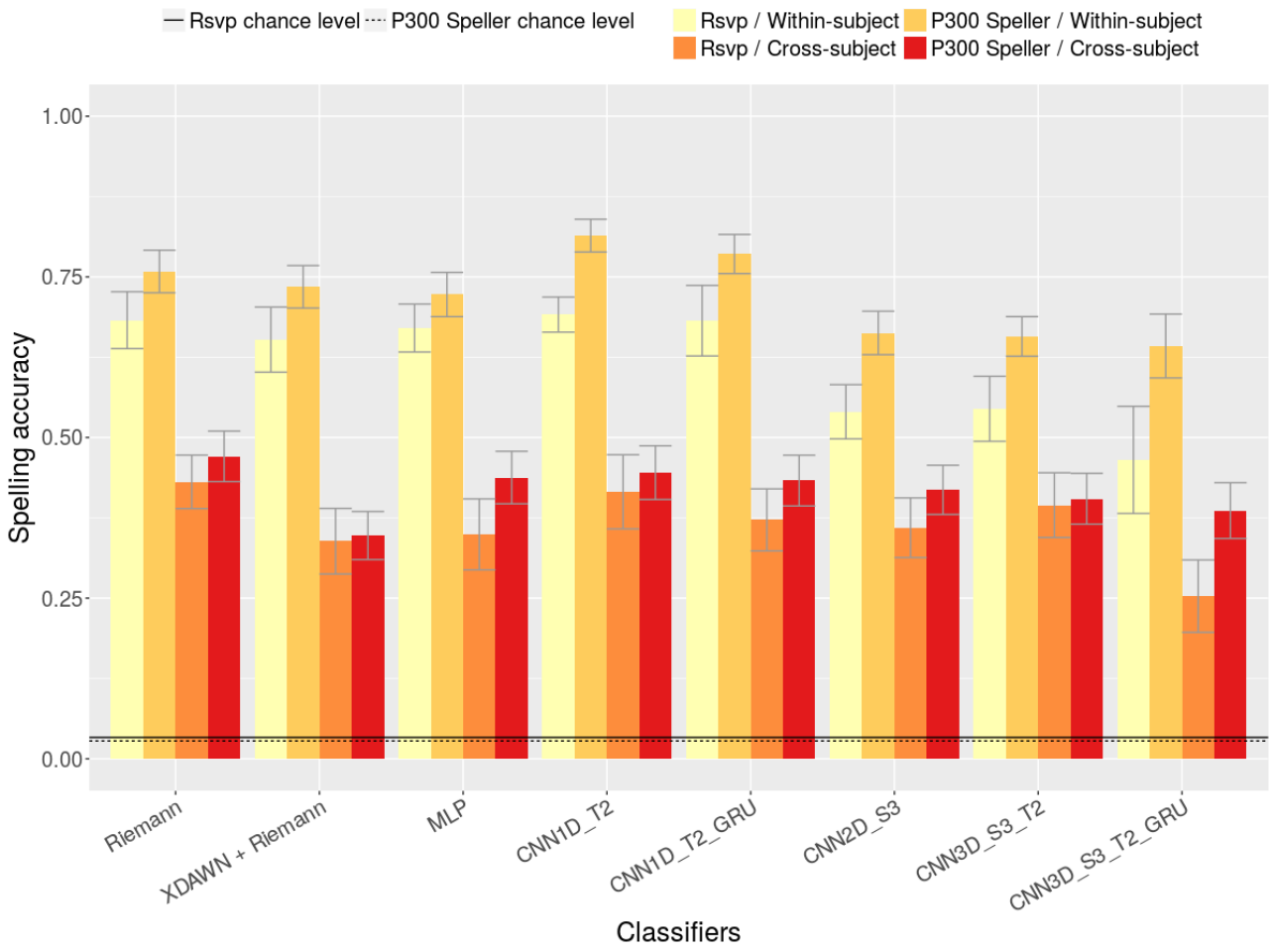

Figure 1: Spelling accuracy performance of all classifiers for Within-subject testing and Cross-subject testing on the two datasets, at 2 repetitions.

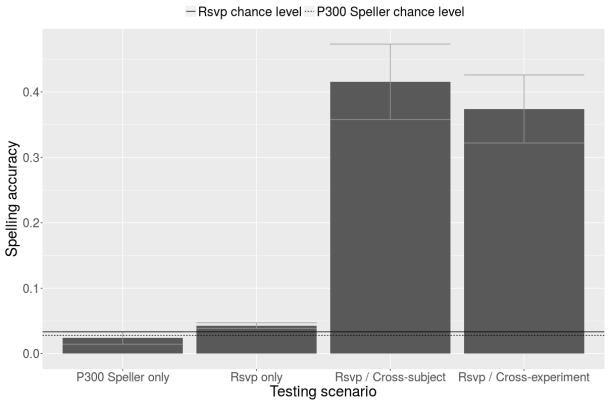

Figure 2: Accuracy performance of model CNN1D_T2 for cross-experiment testing. `RSVP only` (resp. `P300 Speller only`) is the performance of the model when trained on the `Rsvp` data (resp. `P300 Speller` data) and tested on the `P300 Speller` (resp. `Rsvp`) data. `Rsvp / Cross-experiment` refers to the performance of the model when trained on data samples from the two datasets and tested on the `Rsvp` dataset. For comparison `Rsvp / Cross-subject` is the one obtained when trained and tested on the `Rsvp` data only.

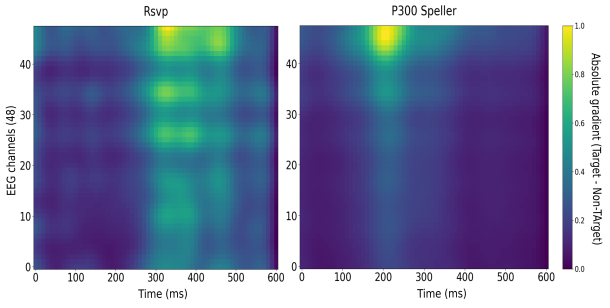

Figure 3: Absolute value of the difference between the mean gradient of the Target samples and the mean gradient of the Non-Target samples (normalized between 0 and 1).

realistic simulation of an online experiment. These deep learning models have a well-known limit that could affect their real-time use: the training time. Although the proposed 2D models (and the MLP) seem applicable using a laptop computer with few resources, this will probably not be the case for 3D models. But this will have to be carefully evaluated in practice, online.

Explaining the decisions of the neural networks is very interesting, notably to have more confidence in the model but also to discover new relevant features in the data. Vi-

 DOI: 10.3217/978-3-85125-682-6-49

sualization techniques can help to tackle this difficulty. In this aim, we applied a well-known approach coming from image classification in deep learning to visualize the inner space of trained neural networks. Our results show that, when trained on P300-Speller and Rsvp data, deep learning models learn well-known discriminating features such as the N200 and P300 components. In addition, the models trained on the `P300 Speller` dataset seem to focus on the N200 ERP while the models trained on the `Rsvp` dataset rather exploit the P300 ERP. This actually fits quite well with the specificity of these two paradigms. Indeed, in Rsvp, the target and non-target stimuli are all displayed at the same location on the screen, and at the center of the fovea, with the same intensity. This means that only attention related components will contribute to the classification, hence mostly the P300. Conversely, in P300-Speller, only the target is flashed at the center of the fovea, which implies that early visual components will also contribute to the classification.

CONCLUSION

To conclude, we showed that fairly simple Deep learning models are serious candidates for ERP-based BCI, showing similar performance as state-of-the-art methods. In addition, their reasonable calibration times make them suitable for real-time application. On the other hand, we have shown that visualization techniques are promising for explaining the decisions of neural networks and for identifying possible new electrophysiological markers.

REFERENCES

[1] Wolpaw JR, Birbaumer N, McFarland DJ, Pfurtscheller G, Vaughan TM. Brain-computer interfaces for communication and control. Clinical Neurophysiology: Official Journal of the International Federation of Clinical Neurophysiology. 2002;113(6):767–791.

[2] Farwell LA, Donchin E. Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials. Electroencephalography and Clinical Neurophysiology. 1988;70(6):510–523.

[3] Lotte F et al. A review of classification algorithms for EEG-based brain-computer interfaces: a 10 year update. Journal of Neural Engineering. 2018;15(3):031005.

[4] Haykin S. Neural Networks: A Comprehensive Foundation. 2nd. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1998.

[5] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation. 1997;9(8):1735–1780.

[6] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521(7553):436–444.

[7] Cecotti H, Graeser A. Convolutional Neural Networks for P300 Detection with Application to Brain-Computer Interfaces. IEEE transactions on pattern analysis and machine intelligence. 2011;33:433–45.

[8] Manor R, Geva AB. Convolutional Neural Network for Multi-Category Rapid Serial Visual Presentation BCI. Frontiers in Computational Neuroscience. 2015;9.

[9] Lawhern VJ, Solon AJ, Waytowich NR, Gordon SM, Hung CP, Lance BJ. EEGNet: A Compact Convolutional Network for EEG-based Brain-Computer Interfaces. arXiv:1611.08024 [cs, q-bio, stat]. 2016.

[10] Tal O, Friedman D. Using recurrent neural networks for p300-based brain-computer interfaces. 6.

[11] Rivet* B, Souloumiac A, Attina V, Gibert G. xDAWN Algorithm to Enhance Evoked Potentials: Application to Brain–Computer Interface. IEEE Transactions on Biomedical Engineering. 2009;56(8):2035–2043.

[12] Congedo M, Barachant A, Bhatia R. Riemannian geometry for EEG-based brain-computer interfaces; a primer and a review. Brain-Computer Interfaces. 2017;4(3):155–174.

[13] Townsend G et al. A novel P300-based brain-computer interface stimulus presentation paradigm: moving beyond rows and columns. Clinical Neurophysiology: Official Journal of the International Federation of Clinical Neurophysiology. 2010;121(7):1109–1120.

[14] Perrin M, Maby E, Daligault S, Bertrand O, Mattout J. Objective and Subjective Evaluation of Online Error Correction During P300-based Spelling. Adv. in Hum.-Comp. Int. 2012;2012:4:4–4:4.

[15] Acqualagna L, Blankertz B. Gaze-independent BCI-spelling using rapid serial visual presentation (RSVP). Clinical Neurophysiology: Official Journal of the International Federation of Clinical Neurophysiology. 2013;124(5):901–908.

[16] Alfeld P. A trivariate clough—tocher scheme for tetrahedral data. Computer Aided Geometric Design. 1984;1(2):169–181.

[17] Bashivan P, Yeasin M, Rish I, Codella N. Learning representations from eeg with deep recurrent-convolutional neural networks. 2016;15.

[18] Maddula RK, Stivers JT, Mousavi M, Ravindran S, Sa VRd. Deep Recurrent Convolutional Neural Networks for Classifying P 300 Bci Signals. In: 2017.

[19] Springenberg JT, Dosovitskiy A, Brox T, Riedmiller M. Striving for Simplicity: The All Convolutional Net. arXiv:1412.6806 [cs]. 2014.

[20] Cho K et al. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. en. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Doha, Qatar: Association for Computational Linguistics, 2014, 1724–1734.

[21] Tieleman T, Hinton G. *Lecture 6.5 - RmsProp: Divide the gradient by a running average of its recent magnitude*. COURSERA: Neural Networks for Machine Learning. 2012.

[22] Simonyan K, Vedaldi A, Zisserman A. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. arXiv:1312.6034 [cs]. 2013.