Convolutional Neural Networks for P300 Signal Detection Applied to Brain Computer Interface

Submitted: 26th June 2019; accepted: 25th March 2020

Mouad Riyad, Mohammed Khalil, Abdellah Adib

DOI: 10.14313/JAMRIS/4-2020/46

Abstract:

A Brain-Computer Interface (BCI) is an instrument capable of commanding machine with brain signal. The multiple types of signals allow designing many applications like the Oddball Paradigms with P300 signal. We propose an EEG classification system applied to BCI using the convolutional neural network (ConvNet) for P300 problem. The system consists of three stages. The first stage is a Spatiotemporal convolutional layer which is a succession of temporal and spatial convolutions. The second stage contains 5 standard convolutional layers. Finally, a logistic regression is applied to classify the input EEG signal. The model includes Batch Normalization, Dropout, and Pooling. Also, It uses Exponential Linear Unit (ELU) function and L1-L2 regularization to improve the learning. For experiments, we use the database Dataset II of the BCI Competition III. As a result, we get an F1-score of 53.26% which is higher than the BN^3 model.

Keywords: *Deep Learning, Convolutional neural network, Brain Computer Interface, P300, Classification*

1. Introduction

A Brain-Computer Interface (BCI) is a mean of communication between the brain and the machine [16]. It consists of translating neural activity to instruction using machine learning algorithms and neuroscience. Such interface can give an improvement in the health field like neurological diseases detection or prosthesis control [2, 15]. Besides, many non-medical applications are possible such as those applied in security and educational fields [10, 21].

Mainly, the Electroencephalography is used to record the brain's waves from the scalp, giving a multichannel signal called Electroencephalogram (EEG) signal [23]. There are many types of EEG signals; each one has its own frequency band, shape, and a related zone in the brain. The most common signals are Motor Imagery and P300 [9]. For the P300 ones, it is an Event-Related Potential (ERP) signal occurred 300ms after a visual or an acoustic stimulus, that is characterized by a low signal-to-noise ratio.

The BCI follows the same steps as those of classic pattern recognition and consists in three important steps. Firstly, it begins with data acquisition using the EEG or other techniques. Secondly, a preprocessing step where the data is filtred and cleaned. Thirdly, the extraction of the most discriminating features. Fourthly, the classification step where a classifier is trained with the data to recognize the pattern. Finally, the translation step where the decision of the classifier is translated into a command. In this work, we are only interested in the feature extraction and the classification steps.

Several techniques have been proposed in the literature. For the feature extraction, the most used are the parametric modeling ones like the Autoregressive model [17], the Time-Frequency domain transformation like the Short-Time Fourier Transform (STFT), Wavelet Transform (WT) and the Filter-Bank Common Space Pattern (FBCSP) [12]. For the classification stage, the Linear Discriminant Analysis (LDA), the Support Vector Machine (SVM) and the Neural Network (NN) are widely used in several schemes [13].

Deep Learning is a new approach mainly used in computer vision and natural language processing, it has also been exploited in BCI [20]. There are many advantages to use the Deep Learning to solve BCI problems ; it allows to merge feature extraction and classification step in a same step. Also, it allows to visualize the learned feature to understand more about brain function [19]. In many studies, we observe that all of them begin with a consecutive convolution that are similar to a spatial filter and a temporal one; this block is called "Space-Time Convolutional Layer" (STCL). [3] used for a P300 problem 4-layers ConvNet, the two first were the STCL and the others are two dense layers, The architecture outperforms the SVM based method of [18]. In the same logic, [11] improved the performance of the previous model by using three dense layers, the rectified linear unit (ReLU) activation was applied. The Batch Normalization was used in STCL and the Dropout between the dense layers. Both architectures used tanh and sigmoid functions which can cause a vanishing gradient problem and slow computation [6]. Also, they are shallow and they do not use more convolutional layers for visualization of the hidden layers in example [19].

In this paper, we suggest to create a ConvNet architecture, fully data-driven, capable of understanding the P300 signal used in the P300 Speller. The choice is motivated by its ability to overcome the main problems of the standard methods like the overfitting or the curse of dimensionality. Moreover, ConvNets are compatible with the nature of the EEG signals. Also, it needs a minimum of preprocessing and it does not need a handcrafted feature extraction step because of its high complexity (cost of processing, choose of the method,...). Hence, we create a ConvNet with 7 layers unlike the existing architectures which are shallow [3, 11]. The first stage is dedicated to a space-time con-

Fig. 1. The P300 Speller matrix [3]

volution, we choose to follow the suggestion of [9] for the design of the STCL which is simulating the FBCSP. The second stage is a 5 standard convolutional layers instead of the dense layers. The third stage is a logistic regression. Obviously, Dropout and Batch Normalization are used. Furthermore, our model is designed to avoid overfitting by the use of the ELU [4].

The paper is organized as the following: In section 2, we provide details about the P300 speller and the ConvNet. In section 3, we introduce the proposed model and justify its hyperparameters. The obtained results are discussed in section 4. The section 5 contains the conclusion.

2. Background

2.1. P300 Speller

The P300 Speller, based on P300 waves, is the neural response for an event that manifests itself in the form of a positive peak of a voltage appearing 300 ms after the event essentially in the occipital and parietal lobes.

The speller is based on the Oddball Paradigm, where a row or a column of a 6X6 character matrix as illustrated in Fig. 1, is randomly flashed. When the subject is aiming a character, a P300 wave is detected 300 ms after the flashing of the column and the row corresponding to the selected character. Hence, the problem will be transposed to a binary one : detecting a P300 wave or not. In the binary case, the speller paradigm generates an unbalanced size of P300/non-P300 signal.

2.2. ConvNet

The ConvNets are the hierarchical neural networks inspired by the architecture of the visual field to process the matrix-like data such signal but essentially image and video [7]. It can learn from raw data the most imminent features automatically with multiple levels of abstraction. Also, it is characterized by the sparse interactions, parameter sharing, and equivariant representations. It is based on many layers such :

- Convolution layer: Applies the convolution to extract the essential features.
- Pooling layer: Down-samples the data to reduce the hyper-parameters in the network.

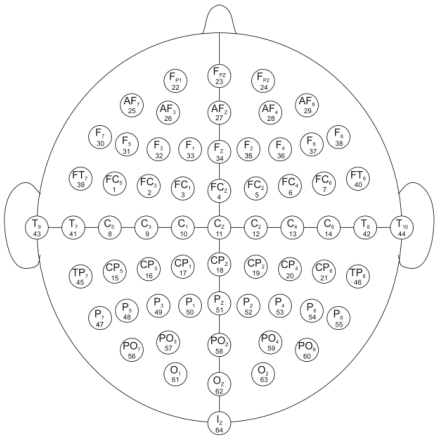

Fig. 2. The EEG montage for the P300 data set [3]

- Activation layer: Increases the non-linearity of the network.
- Regularization layer: Penalizes any non-pertinent information, prevents the network from overfitting.
- Fully connected layer: Classifies the extracted feature in the previous stages.

3. Methods

3.1. Dataset and Preprocessing

For the dataset, we choose the "BCI Competition III Dataset II". The speller follows the paradigm described previously for 2 subject. The signal is composed of 64 channels schematized in Fig. 2, initially sampled at 240 Hz and filtered with a bandpass filter between 0.1-60 Hz.

We adopt the same preprocessing protocol as in [11]. So, we extract the segment 0-667 ms after the end of the flashing and as result we get an EEG signal with a dimension of 64×160 where the fist dimension represents space and the second one the time ($S \times T$). Then, we apply a 0.1-20 Hz 8th-order Bandpass Butterworth filter to the obtained EEG signals. To overcome the unbalanced size between the P300 and non-P300 signals, we replicate the signal with an offset of {-2;-1;0;1;2} in the training dataset. We choose to use only the subject 'B'. For the validation dataset, we use a validation dataset with a proportion of 6.66% from the train dataset after the balancing.

3.2. Architecture

The proposed ConvNet will be composed of three parts: STCL, multiple convolutional layers and regression.

The first stage is STCL, which is composed of two convolution supposed to simulate a temporal filter (convolution across time axis) by using a convolution in the temporal axis ($1 \times n_t$) and a spatial filter by using a convolution across the spatial axis ($C \times 1$) where C is the number of the channels. This approach is inspired by the Filter Bank Common Spatial Pattern (FBCSP) and used in many studies [3, 9, 19]. For [19], splitting the two convolutions is better than merge them into one convolution (e.g. $n_s \times n_t$) like proved [22]. Also,

the Batch Normalization will be used after the convolutions. The linear activation is used for the both convolutions because non-linear activation does not give an improvement. The non-linear activation function is used after the third Batch Normalization [9]. The L_1 and L_2 regularizations are used for the convolutions with Dropout at the end of the block.

The stage follows the following equations :

$$\mathbf{a}^{(0)} = BN(\mathbf{X}) \tag{1}$$

$$\mathbf{a}_{N_t}^{(t)} = BN(pad_{same}(\mathbf{a}^0) * W_{N_t}^{(t)}) \tag{2}$$

$$\mathbf{a}_{N_s}^{(s)} = g(BN((\mathbf{a}^s) * W_{N_s}^{(s)})) \tag{3}$$

$$\mathbf{x}_{N_1}^{(1)} = \mathbf{r} * \mathbf{a}_{N_s}^{(s)} \tag{4}$$

where $x_{N_i}^i$ is the output of the layer i with N_i feature map, $g(x)$ the activation function, $pad_{same}(x)$ a function that applies padding to get the same dimension in the output of a convolution, $BN(x)$ the Batch normalization function, and $w_{N_i}^{(i)}$ the weight of layer i with N_i feature map. As input of the layer, we give a matrix with $C \times T$. The first convolutional kernel has a size of $1 \times \frac{f_s}{2}$ where f_s is the sampling frequency as in [9]. The second convolutional kernel size has a shape of $C \times 1$ where C is the number of electrodes. As output, we get a matrix of $F \times T$ where F is the number of feature map that we unify for the convolutional.

The second stage is composed of multiple convolutional layers based on EEGNet and DeepConvNet. Unlike the BN^3, our configuration contains 5 layers which is deeper than the others. The convolutional kernels follow the same paradigm of DeepConvNet which gave good performance in our tests. Also, we follow the same disposition of the layers of the EEGNet.

So, the actual stage begins with a convolutional layer where the kernel size follows the pattern like in Tab. 1 with no padding and we use a high number of feature map for better performance. Then, the convolutional layers are followed by Batch Normalization and an activation with a non-linear function. For reducing the number of parameters, a pooling is performed with the maxpooling which gave better scores. Finally, the layer is regularized by only a Dropout.

The equations bellow describes a single layer where k represent it the number :

$$\mathbf{a}_{N_k}^{(k)} = g(BN(pad_{same}(\mathbf{x}^{k-1}) * W_{N_t}^{(k)})) \tag{5}$$

$$\mathbf{x}_{N_k}^{(k)} = \mathbf{r} * pool(\mathbf{a}_{N_k}^{(k)}) \tag{6}$$

The third stage is a layer of regression where the features are classified into their corresponding classes. According to the nature of the problem, the chosen activation function is a sigmoid. We omit the fully connected layer to reduce any risk of overfitting unlike BN^3 which can lead to overfitting and increase the complexity of the network.

To be more specific, the goal of the Batch Normalization is to reduce the internal covariate shift in the neural network. The data shift to the saturation of some activation function like tanh or sigmoid causing a decrease of the gradient. Such a phenomenon slows the training when the network is deep. The solution is to Normalize every Batch to avoid the saturation. The formula is as below :

$$\hat{y}_i(j) = \frac{y_i(j) - \mu(j)}{\sigma^2(j) + \epsilon} \tag{7}$$

where μ is the mean of the batch, σ the variance and ϵ a small constant for numerical stability.

Also, the dropout [1] is a technique allowing to decrease the complex co-adaptation between neurons on the training data. It adds a noise by giving to every neuron a probability of $1 - p$ to be set to 0 in the forward propagation. That means it gives an average neural network from many possibilities.

3.3. Training Setting

For the implementation, we use the framework Keras and Tensorflow as backend and a NVIDIA K80 GPU. We choose to use the Adam optimizer with default setting, and a binary cross-entropy loss function. We choose the hyperparameters as follows:

- We use Adam optimizer because it is the most used, also it is faster and need low computational power compared to the other techniques, we use the default value for the parameters [8].
- For the parameter F introduced previously, we use a cross validation with the proportion described above, we compare between 4, 16, 32 and 64.
- We use the ELU function rather than the sigmoid or tanh function because it has proven its superiority in terms of accuracy and speed.
- For the dropout, we choose the value 0.5 as in [1].
- From our test, the epoch around 100 are the best choice, so we choose it.
- Batch size should be small and we choose 32 following the recommendation in [14].
- Glorot Uniform method is used to initialize the parameters [5].
- We omit the use of the biais like in [9] but just in convolutional layers.

4. Result and Discussion

The goal of our work is to build a P300 speller system, which is capable of translating the P300 signals in a small amount of time, this is why we design our model first by its architecture based on convolutional neural network. To evaluate the performance of our method. We use Recognition, Precision, Recall and F1-score as metrics:

$$Recognition = \frac{TP + TN}{TP + TN + FP + FN} \tag{8}$$

Tab. 1. The details of the proposed model in Keras codification

Block	Layers	# filters	size	# params	Output	Activation	Option
0	input				(T,C)		
	Reshape						
	permute						
	BatchNormalization			4			
1	Conv2D	F	$(1, 120)$	$120 * F$		Linear	$mode = same, l1 = l2 = 0.001$
	BatchNormalization			$4 * F$			
	Conv2D	F	$(64, 1)$	$64 * F^2$		Linear	$mode = valid, l1 = l2 = 0.001$
	BatchNormalization						
	Activation					ELU	
	Dropout						$p = 0.5$
	Permute						$(2, 1, 3)$
2	Conv2D	F	$(F, 12)$	$12 * F^2$		Linear	$mode = valid$
	BatchNormalization			$4 * F$			
	Activation					ELU	
	MaxPooling2D		$(2, 2)$				
	Dropout						$p = 0.5$
	Permute						$(2, 1, 3)$
3	Conv2D	$2 * F$	$(\frac{F}{2}, 6)$	$6 * F^2$		Linear	$mode = valid$
	BatchNormalization			$8 * F$			
	Activation					ELU	
	MaxPooling2D		$(2, 2)$				
	Dropout						$p = 0.5$
	Permute						$(2, 1, 3)$
4	Conv2D	$4 * F$	$(F, 3)$	$12 * F^2$		Linear	$mode = valid$
	BatchNormalization			$16 * F$			
	Activation					ELU	
	MaxPooling2D		$(2, 2)$				
	Dropout						$p = 0.5$
	Permute						$(2, 1, 3)$
5	Conv2D	$8 * F$	$(2 * F, 3)$	$48 * F^2$		Linear	$mode = valid$
	BatchNormalization			$32 * F$			
	Activation					ELU	
	MaxPooling2D		$(2, 2)$				
	Dropout						$p = 0.5$
	Permute						$(2, 1, 3)$
6	Conv2D	$16 * F$	$(4 * F, 3)$	$192 * F^2$		Linear	$mode = valid$
	BatchNormalization			$32 * F$			
	Activation					ELU	
	MaxPooling2D		$(2, 2)$				
	Dropout						$p = 0.5$
	Permute						$(2, 1, 3)$
7	Dense	1				Sigmoid	$(2, 1, 3)$

$$Recall = \frac{TP}{TP + TN} \tag{9}$$

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

$$F1\text{-}score = 2.\frac{Recall.Precision}{Recall + Precision} \tag{11}$$

with true positive (TP), false positive (FP), true negative (TN) and false negative (FN).

We focus only on the F1-score because [11] results suggest that it is the most convenient to be an indicator of performance. We use cross validation to choose the right value of F. The obtained results are presented in Tab. 2. We observe that the best recognition rate, recall and F1-score are obtained by the value 32. Also, the best precision is for value 4. Moreover, the other values did not get important result. So, the value 32 is

Tab. 2. Cross validation result

F	Recognition	Precision	Recall	F1-score
64	0.9029	0.8751	0.94	0.9064
32	0.9035	0.8744	0.9423	**0.9071**
16	0.7882	0.9209	0.6305	0.7486
4	0.5211	0.9500	0.044	0.085

Tab. 3. Compared metrics for multiple methods

Architecture	TP	TN	FP	FN	Recognition	Precision	Recall	F1-score
MCNN-3 [3]	2077	11997	3003	923	0.7819	0.409	0.692	0.514
BN^3 [11]	2084	12139	2861	916	0.7902	0.4214	0.6947	0.5246
Proposed with Max pooling	1996	12501	2499	1004	0.7992	0.4440	0.6653	**0.5326**
Proposed with Average pooling	1952	12343	2657	1048	0.7941	0.4235	0.6506	0.5130

Tab. 4. Compared metrics for different deep of the second part of the proposed method

Number of Layers	TP	TN	FP	FN	Recognition	Precision	Recall	F1-score
0	1960	12273	2727	1040	0.7907	0.4181	0.6533	0.5099
1	1797	12674	2326	1203	0.8039	0.4358	0.599	0.5045
2	1911	12398	2602	1089	0.7949	0.4234	0.637	0.5087
3	1840	12622	2378	1160	0.8034	0.4362	0.6133	0.5098
4	1595	13257	1743	1405	0.8251	0.4778	0.5316	0.5033
5	1996	12501	2499	1004	0.7992	0.4440	0.6653	**0.5326**

the validated.

Tab. 3 shows the results for the test dataset and we compare with the BN^3 one [11] and MCNN-3 of [3]. Also, we train two versions of our model one with Max-pooling, and the other with AveragePooling. We observe that our model with MaxPooling outperforms the BN^3 model. Our model has higher recognition, the precision and the F1-score. Beside, the main metrics is higher with 0.008 for our model that make our model more satisfying than the BN^3.

This paper is based on the argument that a deeper network is better than a shallow one. It takes origin from computer vision improvement on the last years. To demonstrate the effect of the deep on the BCI case, we train multiple versions of our model with different number of layer in the second part. We add progressively one layer following the architecture of the proposed model and we test all the generated model on the test dataset. The result are in Tab. 4, as expected the deeper model get better performance for the F1-score and the recall, the 4 layers model get the best recognition and precision.

There are many factor behind those result. First, our model is deeper than the others by using more layers which implies a great number of parameters. Secondly, we omit the use of fully connected layer and replace them by multiple convolutional layers allowing them to learn more accurate features and decrease the risk of overfitting and complexity of the computation. For the pooling, the MaxPooling seems to be the appropriate method for our model compared with AveragePooling. But, our model contains more parameters than the other making the learning time longer.Also, it has a high loss at the end. Furthermore, the architecture is created for offline classification.

5. Conclusion

We presented a new ConvNet Architecture for a P300 speller application by the translation of the P300 signal. The proposed model is based on EEGNet and Deep ConvNet. The model contains 7 convolutional layers with batch normalization and dropout that improved the performance. The model is composed of an STCL simulating the FBCSP which is one of the main traditional technique. Then, five standard convolutional layers follow the same logic of DeepConvNet. Finally, a logistic regression is applied to get final decision.The model is capable of outperforming the existing models which lead to a new important architecture. Also, the design allows the visualizing of the learned features. Moreover, we tried to justify our hyperparameter based on previews work. Furthermore, we experimentally justify the necessity of a deep neural network rather than a shallow neural network. Further works will focus on designing lighter architecture with better performance. Also, we will introduce a hybrid neural network based on ConvNet and recurrent neural network.

AUTHORS

Mouad Riyad* – Hassan II University Of Casablanca, LIM@II-FSTM, B.P. 146, Mohammedia 20650, Morocco, e-mail: riyadmouad1@gmail.com.

Mohammed Khalil – Hassan II University Of Casablanca, LIM@II-FSTM, B.P. 146, Mohammedia 20650, Morocco, e-mail: mohammed.khalil@univh2c.ma.

Abdellah Adib – Hassan II University Of Casablanca, LIM@II-FSTM, B.P. 146, Mohammedia 20650, Morocco, e-mail: abdellah.adib@fstm.ac.ma.

*Corresponding author

REFERENCES

[1] P. Baldi and P. Sadowski, "Understanding dropout". In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, Red Hook, NY, USA, 2013, 2814–2822.

[2] S. Biswal, J. Kulas, H. Sun, B. Goparaju, M. B. Westover, M. T. Bianchi, and J. Sun, "SLEEPNET: Automated Sleep Staging System via Deep Learning", *arXiv:1707.08262 [cs]*, 2017.

[3] H. Cecotti and A. Graser, "Convolutional Neural Networks for P300 Detection with Application to Brain-Computer Interfaces", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, 2011, 433–445, 10.1109/TPAMI.2010.125.

[4] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)", *arXiv:1511.07289 [cs]*, 2016.

[5] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, 249–256.

[6] X. Glorot, A. Bordes, and Y. Bengio, "Deep Sparse Rectifier Neural Networks". In: *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011, 315–323.

[7] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, The MIT Press: Cambridge, Massachusetts, 2016.

[8] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization", *arXiv:1412.6980 [cs]*, 2017.

[9] V. J. Lawhern, A. J. Solon, N. R. Waytowich, S. M. Gordon, C. P. Hung, and B. J. Lance, "EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces", *Journal of Neural Engineering*, vol. 15, no. 5, 2018, 10.1088/1741-2552/aace8c.

[10] L.-D. Liao, C.-Y. Chen, I.-J. Wang, S.-F. Chen, S.-Y. Li, B.-W. Chen, J.-Y. Chang, and C.-T. Lin, "Gaming control using a wearable and wireless EEG-based brain-computer interface device with novel dry foam-based sensors", *Journal of NeuroEngineering and Rehabilitation*, vol. 9, no. 1, 2012, 10.1186/1743-0003-9-5.

[11] M. Liu, W. Wu, Z. Gu, Z. Yu, F. Qi, and Y. Li, "Deep learning based on Batch Normalization for P300 signal detection", *Neurocomputing*, vol. 275, 2018, 288–297, 10.1016/j.neucom.2017.08.039.

[12] F. Lotte, L. Bougrain, A. Cichocki, M. Clerc, M. Congedo, A. Rakotomamonjy, and F. Yger, "A review of classification algorithms for EEG-based brain–computer interfaces: a 10 year update", *Journal of Neural Engineering*, vol. 15, no. 3, 2018, 10.1088/1741-2552/aab2f2.

[13] F. Lotte, M. Congedo, A. Lécuyer, F. Lamarche, and B. Arnaldi, "A review of classification algorithms for EEG-based brain–computer interfaces", *Journal of Neural Engineering*, vol. 4, no. 2, 2007, R1–R13, 10.1088/1741-2560/4/2/R01.

[14] D. Masters and C. Luschi, "Revisiting Small Batch Training for Deep Neural Networks", *arXiv:1804.07612 [cs]*, 2018.

[15] P. W. Mirowski, Y. LeCun, D. Madhavan, and R. Kuzniecky, "Comparing SVM and convolutional networks for epileptic seizure prediction from intracranial EEG". In: *2008 IEEE Workshop on Machine Learning for Signal Processing*, 2008, 244–249, 10.1109/MLSP.2008.4685487.

[16] L. F. Nicolas-Alonso and J. Gomez-Gil, "Brain Computer Interfaces, a Review", *Sensors*, vol. 12, no. 2, 2012, 1211–1279, 10.3390/s120201211.

[17] J. Pardey, S. Roberts, and L. Tarassenko, "A review of parametric modelling techniques for EEG analysis", *Medical Engineering & Physics*, vol. 18, no. 1, 1996, 2–11, 10.1016/1350-4533(95)00024-0.

[18] A. Rakotomamonjy and V. Guigue, "BCI competition III: dataset II- ensemble of SVMs for BCI P300 speller", *IEEE transactions on bio-medical engineering*, vol. 55, no. 3, 2008, 1147–1154, 10.1109/TBME.2008.915728.

[19] R. T. Schirrmeister, J. T. Springenberg, L. D. J. Fiederer, M. Glasstetter, K. Eggensperger, M. Tangermann, F. Hutter, W. Burgard, and T. Ball, "Deep learning with convolutional neural networks for EEG decoding and visualization", *Human Brain Mapping*, vol. 38, no. 11, 2017, 5391–5420, https://doi.org/10.1002/hbm.23730.

[20] J. Schmidhuber, "Deep learning in neural networks: An overview", *Neural Networks*, vol. 61, 2015, 85–117, 10.1016/j.neunet.2014.09.003.

[21] J. Sohankar, K. Sadeghi, A. Banerjee, and S. K. Gupta, "E-BIAS: A Pervasive EEG-Based Identification and Authentication System". In: *Proceedings of the 11th ACM Symposium on QoS and Security for Wireless and Mobile Networks*, New York, NY, USA, 2015, 165–172, 10.1145/2815317.2815341.

[22] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision", *arXiv:1512.00567 [cs]*, 2015.

[23] M. Teplan, "Fundamentals of EEG measurement", *Measurement science review*, vol. 2, no. 2, 2002, 1–11.