Proceedings of the
7th Graz Brain-Computer Interface Conference 2017
DOI: 10.3217/978-3-85125-533-1-56

CSP-NN: A CONVOLUTIONAL NEURAL NETWORK IMPLEMENTATION OF COMMON SPATIAL PATTERNS

D. Maryanovsky[1], M. Mousavi[2], N. G. Moreno[3], V. R. de Sa[1]

[1]Cognitive Science, University of California, San Diego
[2]Electrical and Computer Engineering, University of California, San Diego
[3]Computer Science and Engineering, University of California, San Diego, La Jolla, CA, USA

E-mail: maryanovsky@gmail.com

ABSTRACT: In this paper we propose, describe, and evaluate a novel deep learning method for classifying binary motor imagery data. This model is designed to perform CSP-like feature extractions. It can be seen as a neural network with a specifically designed architecture where the latent space corresponds naturally to the features found in CSP methods. Our model allows for easy generalization from spatial filters to spatio-temporal filters. It also allows for the feature extraction and filtering stages to be optimized jointly with the classifier. This allows standard regularization methods to include the filtering stage. In addition the network provides the expressiveness and robustness of deep learning to improve upon the efficiency of CSP filtering methods.

INTRODUCTION

Motor-imagery (MI) brain-computer interfaces (BCIs) work by detecting decreases in power in the mu (7-13 Hz) and beta (13-30 Hz) frequency bands. Decreases in power in those frequencies are known to occur both prior to and during movement, as well as during imagined movement [1]. The relevant decreased power or desynchronization is spatially localized over the motor cortex. Any body part, when imagined to be moving, has a corresponding region of cortex.

Spatially discriminative mu-desynchronization is recognized using a filter that emphasizes the spatial differences between the different motor-imagery classes.

The most commonly used method is the Common Spatial Patterns, or CSP, [2] which finds a set of filters that maximizes the projected variance (power) for one class while minimizing it for the other. Applying CSPs to band-pass filtered signals can greatly emphasize the spatially segregated power differences between the different classes and is common in MI-based BCIs [2].

Let a column vector $x_t \in \mathbb{R}^C$ be the band-passed EEG signal for time t where C is the number of EEG channels on the scalp and $\mathbb{R}$ indicates the set of real numbers. The estimate of the covariance matrix for a two-class experiment can be calculated from the training data using traditional methods. Let the covariance matrix for the two classes 1 and 2 be specified as:

$$\Sigma_y \in \mathbb{R}^{C \times C} \quad y \in \{1, 2\} \tag{1}$$

CSP aims to find a projection $w \in \mathbb{R}^C$ which maximizes the variance of signals for one condition and at the same time minimizes the variance of signals of another condition. One can find w for class 1 by solving the following Rayleigh quotient:

$$R(w) = \frac{w^T \Sigma_1 w}{w^T (\Sigma_1 + \Sigma_2) w} \tag{2}$$

The solution for this problem can be found by solving the generalized eigenvalue problem given in the form:

$$\Sigma_1 w = \lambda(\Sigma_1 + \Sigma_2) w \tag{3}$$

There are C generalized eigenvectors where w_1 corresponding to the largest eigenvalue maximizes the variance for class 1 while minimizing for class 2 and w_c corresponding to the smallest eigenvalue maximizes the variance for class 2 while minimizing for class 1. It is common in the CSP algorithm to select some number (often 3) of the top and bottom eigenvectors as the discriminative spatial filters [3].

Once the CSP filters have been learned, the data are transformed according to the CSP filters, and the class bandpower is computed (via the sum of the squared filtered data, or equivalently, the variance of filtered, zero-mean data). The logarithm of this power output is often taken as the log-power is more normally distributed. These log powered features are then fed into a simple linear classifier such as linear discriminant analysis (LDA) with shrinkage, step-wise LDA, logistic regression, or linear support vector machines. These simple algorithms have been preferred because of their robustness to the large amounts of noise in, and scarcity of, EEG data. Many studies with shallow non-linear algorithms have failed to beat these simple linear algorithms However, recent advances in deep convolutional neural networks (CNNs) have transformed the fields of handwriting recognition, speech recognition, computer vision, and video analysis [4, 5], and are rapidly transforming machine learning more generally. We aim to leverage the advantages central to all of these results for the task of improving MI classification.

There have been other variants on the basic CSP algorithm, some of which are reviewed in [6]. Common spatio-spectral patterns (CSSP) [7] uses the temporal structure information to improve CSP. Spectrally weighted common spatial patterns (Spec-CSP) [8] learns the spectral weights as well as the spatial weights in an iterative way. Invariant CSP (iCSP) [9] minimizes variations in the EEG signal caused by various artifacts using a pre-calculated covariance matrix characterizing these modulations. Stationary CSP (sCSP) [10] regularizes CSP filters into stationary subspaces. Local temporal common spatial patterns (LTCSP) [11, 12] uses temporally local variances to compute the spatial filters. Canonical correlation approach to common spatial patterns (CCACSP) incorporates the temporal structure of the data to extract discriminative and uncorrelated sources [13].

A neural network implementation of CSP filtering easily allows for the development of new spatio-temporal extensions to CSP. We note that these extensions may be implementable in a standard filtering pipeline, but that implementing them as part of a neural network allows for easy, quick extensions to well-studied variations of CSP, and, moreover, allows for testing novel architectures that may not be intuitive in the framework that CSPs are typically studied and used in. By considering CSP filtering as a special case of convolutional neural networks, one can quickly run through entirely novel CSP extensions, and optimize them in tandem with the classifier, simply by modifying a few canonical parameters.

Also because the CSP filters are typically trained with non-iterative algorithms, and without validation, they are prone to overfitting. Implementing both filtering and classification in one framework allows for joint monitoring and regularizing, to combat overfitting.

A major advantage of CNNs over traditional neural networks is that they are a special case of the latter. Convolutional networks are motivated by, and based on, the structure of the visual system [14–17]. Convolutional neural networks have a shared weight structure – local receptive fields are learned and the learned structure is shared throughout the input [17]. This means that each training sample provides many windows of training data for the same (shared) sets of weights which greatly increases the effective training data.

CNNs can be seen as an architectural constraint on neural networks in general, specifically, they are constrained such that they perform operations that have traditionally been used, in a non-machine learning setting, to great effect on the same task. This approach allows us to optimize the filters and classifier together, and to transparently leverage a validation set.

We hypothesize four explicit advantages to employing the CSP filtering as part of the classification network. First, early stopping or other, more advanced, deep learning regularization techniques can be used to prevent the spatial filters from overfitting. Second, the filtering and classification being optimized together may result in improved performance than performing these separately, as our filters are going to differ from CSP in that they are sensitive to classification accuracy as opposed to just class variance. Third, the CSP algorithm finds linear spatial filters whereas the neural network extension would be able to find smooth non-linear extensions. Finally, once implemented in a neural network, the network may be modified, in the common ways described in the deep learning literature, to improve on the CSP algorithm. The natural extensions to both can be leveraged to extend and strengthen the basic model described here. For example, the network's architecture can be easily extended, via weight-sharing, to allow for natural structurally-restricted spatial filters.

MATERIALS AND METHODS

EEG data were recorded from 6 healthy participants recruited from the UC San Diego student population. Participants were naive to BCI and signed a consent form approved by the University Institutional Review Board before participating in the experiment. Participants were instructed to perform kinesthetic motor imagery of their right or left hand to control a cursor to hit a target on a monitor in front of them. The cursor and the target were each represented by a circle having 2 cm diameter and colors blue and white respectively. The cursor moved discretely, one second at a time. Each trial began with the cursor at the center of the screen and the target at either end - the center was three cursor steps away from each end. After 1.5 seconds the target disappeared to minimize distraction for the participant and the cursor began moving towards or away from the target. Participants were lead to believe that they were in control of the cursor; however, in order to provide consistent cursor movements between participants, the cursor was moved based on a pre-programmed sequence. There were a total of 10 blocks and each block consisted of 20 trials [18].

Data were collected with a 64-channel EEG system (Brain Products GmbH). The electrodes were arranged based on the 10-20 international system. Data were collected at 5000 Hz sampling rate and were down-sampled to 500 Hz. Pre-processing of the data was done in MATLAB [19] and EEGLAB [20] where the data were first bandpass filtered with an FIR filter of order 500 in 1 to 200 Hz. Then clean-line [21] which is an EEGLAB plugin, was applied to remove the line noise. Then up to five channels with high power in frequencies above 60 Hz - indicating muscle artifacts - were removed. Next, the EEG on each channel was re-referenced to the common average over the remaining channels. Data were visually inspected for large muscle artifacts and less than 10% of the trials were removed. Independent component analysis (ICA) was applied and ICA components regarding muscle and eye were removed. The pre-processed data were bandpass filtered with FIR filters with 500 taps in the following eleven frequency bands: 1-3, 2-5, 4-7, 6-10, 7-12, 10-15, 12-19, 18-25, 19-30, 25-35, and 30-40

Hz and epoched from 150 to 950 ms after each cursor movement.

As baseline, we used CSP in combination with regularized linear discriminant analysis (LDA). CSP is trained on data from each filter and the top 3 filters for each imagery class is selected to be passed into an LDA. Fig. 1 shows the structure of CSP+LDA classifier. We ran 3 instances of 5-fold cross-validation

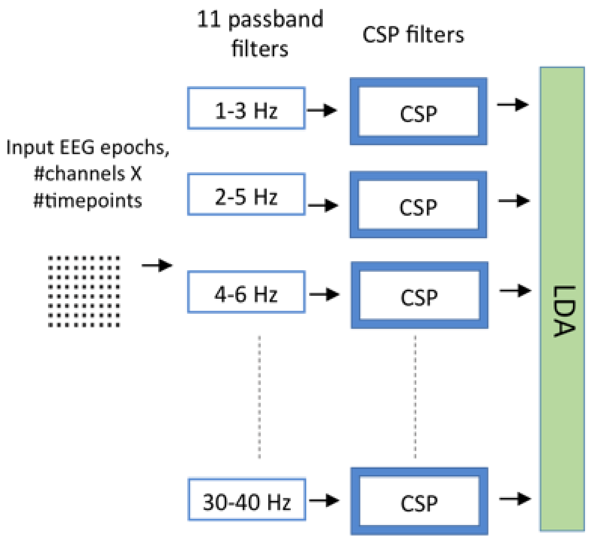

Figure 1: Conventional CSP+LDA method.

We have created two methods inspired by the success of CSPs. First we introduce a hybrid CSP/Deep Net model, which will serve as a control or reference, that learns and feeds CSPs of our eleven pass bands into a fully connected neural network with 500 hidden units. Second, we introduce the proposed model, a fully deep net with CSP-like architectures. This network's architecture has been structured such that its latent space naturally encodes for the same variance-optimizing spatial transformations described above. Similar to the hybrid net, our CSP like Neural Network, or CSP-NN, is trained on all eleven passbands of a signal.

The convolutional network computes the equivalent of a convolution of a "receptive field" with the input. It is implemented by having local connectivity between a neuron in the convolutional layer and the lower input layer. This neuron is then replicated with shifted input connectivity. The key feature is that the weights are shared between all of the neurons within a map in the layer, so that they are all forced to learn the same receptive field/kernel. At the same time many maps may be learned in parallel. These convolutional layers are commonly followed by pooling layers that combine the input from several nearby neurons within the same map. Common pooling operations are max-pooling where the maximum value of the combined inputs are output, average-pooling, where the average value of the combined inputs are output. There is also norm-pooling where the L_p norm of the combined inputs is output [22].

Hybrid CSP Net

Our hybrid model performs a standard D-dimensional CSP on each passband. treating each epoch as a data point. D is necessarily an even number, as we choose the first D/2 and last D/2 vectors of the CSP solution.The CSP filtered signals are then fed as inputs to a densely connected neural network which performs binary classification. This densely connected neural network is composed of 500 neurons fitted with hyperbolic tangent activation. We used a dropout mask with a probability of .5 in this dense layer as decided for the proposed networks discussed next. We selected binary cross-entropy as our objective function due to the binary classification nature of our network. The activations are fed into a single sigmoid output unit. The dense layer was implemented and trained in Keras [23], using Theano [24] as a backend. Fig. 2 shows the architecture of this network.

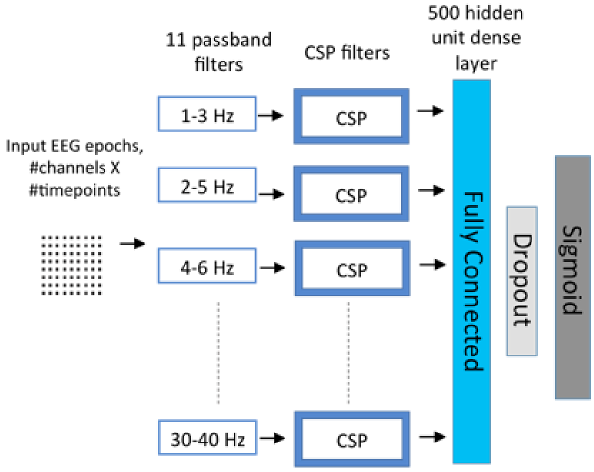

Figure 2: Structure of the hybrid CSP network.

CSP-NN

Our CSP-NN takes the 11 chosen passbands, and feeds each into one of 11 parallel 1-D convolution layers. We use D convolution kernels, where D is analogous to the number of CSP dimensions described above. Thus, we generate $11 \times D$ feature maps per data point. Each of these feature maps are element-wise squared and globally summed, analogous to the process of extracting variance in CSP, creating $11 \times D$ positive scalars. Note this operation corresponds to using norm 2 pooling instead of the commonly used max pooling, and it simulates the operation of extracting variance from CSP projections.

Interestingly, max pooling was originally motivated in the classical deep learning architectures [15, 16] motivated by complex cells and their translation invariance. However, the energy model of complex cells [25], "the de facto standard description of complex cells in primary visual cortex (Adelson and Bergen 1985)" actually models them as computing the sum of squared simple cells, and this has been found to produce a better fit [26]. These pooled output scalars are fed into a fully connected network, identical to the net used in the hybrid-model, which generates our predictions. This process is detailed below. By building a net of this architecture, and by differentiating through a global sum of squares (norm 2 pooling), we create a model that learns convolution kernels that are analogous to the CSP filter solutions found in traditional approaches. Initially, we use convolutional kernels of length one. These convolutions apply the same set of spatial weights across all time points as in the standard CSP spatial filters. Details of our models can be found in Fig. 3. We also tested kernels with some temporal sensitivity, replacing the length-1 filter with a length-T or T-degree convolution through time. These later convolutions can be said to apply a spatio-temporal filter with a T-point temporal resolution, which is analogous to common spatial temporal patterns, or CSTP filtering methods. CSTP-NN is based on sensitivity to temporal dependencies between windows of successive points. CSTP-NN is an example of a simple extension, a single parameter change in fact, to the CSP-NN that corresponds to a major class of CSP extensions in the traditional BCI literature.

The parameters learned by our proposed method code for similar features to the traditional CSP and CSTP methods. However they are not trained to maximize variance/power for one class and minimize it for the other, but aim to discover the set of kernels that provide optimal classification performance for the network as a whole. This latter advantage increases the possibility of overfitting, which we handle with a very strict early-stopping schedule as described above and detailed below. Within our method, an epoch is bandpassed into 11 separate band signals, which are concatenated into an 11 by C by T array, where C and T are number of channels and timepoints, respectively. Each level of this array is fed into one of eleven separate 1-D convolutional layers, each containing D unique feature kernels, where D is chosen to always be even, and is 6, 10, or 16 in our analysis. The length of each kernel is 1, and the width of each is equal to the number of EEG channels in our data, which is 64. These kernels are convolved through time, and are functionally similar to applying a CSP-like spatial filter to each time-point in our 11 signal bands. This yields $11 \times D$ feature maps, or filtered signals, each of which is element-wise squared and globally summed, or norm-2 pooled. This operation yields $11 \times D$ feature scalars, and is functionally similar to extracting variance of the first and last $D/2$ spatial filters in CSP. These features are concatenated and fed as an $11 \times D$ input vector into dense neural network layer, which always has 500 hidden units with hyperbolic tangent activation. This layer is fed into a single sigmoid unit. The network is trained end-to-end using stochastic gradient descent, with Nesterov momentum of .9 and training rate of .0001. We use dropout in the 500 hidden units, setting a random half of them to 0 at each training epoch. The CSP-NN, and by extension, the CSTP-NN, are both implemented in Theano [24], using the Lasagne library [27]. Again, CSTP-NN differs from CSP-NN by only a single parameter in our implementation.

The training duration was set to a maximum of 5000 epochs with early stopping. If 50 of the last 100 epochs resulted in worse validation accuracy, then early stopping was triggered, and the weights were set to those that produced the recorded best validation results. We searched over the following hyperparameters for participant 1 in the CSP-NN and kept the parameters the same for every other participant: Dropout (p = .5) vs non-dropout, learning rate, .001 vs .0001, number of kernels, 6 vs 10 vs 16 and kernel lengths, 1, 5,10, and 15. We tested hyperbolic tangent vs. rectified linear (ReLU) as activation function in the hidden layer and found that hyperbolic tangent performed better. We selected binary cross-entropy as our objective function due to the binary classification nature of our network.

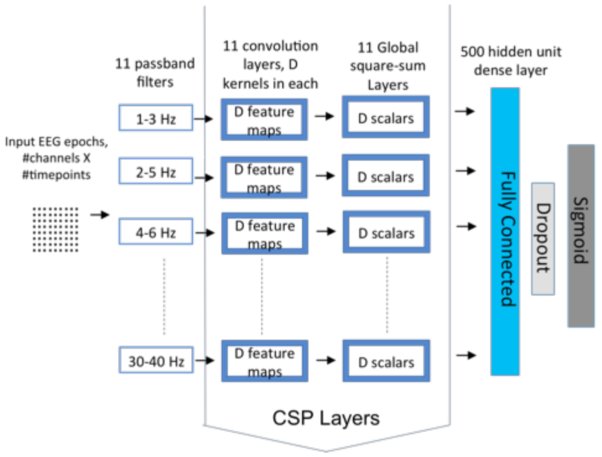

Figure 3: Structure of the CSP-NN or CSTP-NN, depending on kernel length (1 vs. > 1).

RESULTS

Results are reported for 5-fold cross-validation in Tab. 1. The rate for the baseline method, i.e. CSP+LDA, and the hybrid CSP method are reported as the average of 3 instances of 5-fold cross-validation. In each instance, the number of trials in right and left classes were balanced by dropping trials from the class with more trials.

The proposed methods, i.e. CSP-NN and CSTP-NN, were also tested in a 5-fold cross-validation scheme. We applied this cross-validation scheme to each participant individually, and we report the mean accuracy across all five folds per participant. Again, we made sure that the right and left classes were balanced. Our proposed methods outperformed on certain participants (P-2, P-3 and P-6), but underperformed on P-4. We also found that the increasing the length of the 1-D convolution also improves performance, but also increases the error bounds significantly. This is consistent with the small size of our data. We found that 16 kernels per convolution layer, as well as a kernel length, (or temporal sensitivity) of 5 produced the strongest results from among the values tried in Participant 1 (and was used for all subjects). Considering only length-1 kernel instances, the best results were found with the same parameters in Participant 1, including 16 kernels per convolution layer.

Table 1: Results - Classification accuracy. Note that * is used for CSP-NN and CSTP-NN for P-1 as results for this participant may be overfit due to parameter tuning.

ID	CSP+LDA	Hybrid NN	CSP-NN	CSTP-NN
P-1	0.8629	0.8307	0.8297*	0.8459*
P-2	0.6482	0.6381	0.7311	0.7041
P-3	0.6485	0.6553	0.6351	0.7243
P-4	0.6526	0.6571	0.6069	0.5250
P-5	0.7173	0.7104	0.7125	0.6750
P-6	0.7105	0.6970	0.7338	0.7730

The standard error among 5-fold for the results reported in Tab. 1 is at most 0.025.

DISCUSSION

The CSP-NN and CSTP-NN methods perform comparably to standard methods. It is possible that the proposed methods would have even stronger performance, with tighter error bounds, if given a larger corpus of data. Grand averages across all participants show that CSTP-NN and CSP-NN are the highest performing methods that we attempted however there is large variability between participants.

CONCLUSION

CSP-NN style approaches perform similarly to standard non end-to-end models, and can be easily extended when implemented. They can also be regularized more transparently. We find that this class of model suffers from the same dependence on large corpus of data as all CNN models, but offers a stronger method when that data is available, and we will further study the advantages and disadvantages of this class of model in the future.

ACKNOWLEDGMENTS

This work was supported by the NSF grants IIS 1219200, SMA 1041755, IIS 1528214, and UCSD FISP G2171.

References

[1] Wolpaw, Jonathan R., and Dennis J. McFarland. ”Control of a two-dimensional movement signal by a noninvasive brain-computer interface in humans.” Proceedings of the National Academy of Sciences of the United States of America 101, no. 51 (2004): 17849-17854.

[2] Müller-Gerking, Johannes, Gert Pfurtscheller, and Henrik Flyvbjerg. ”Designing optimal spatial filters for single-trial EEG classification in a movement task.” Clinical neurophysiology 110, no. 5 (1999): 787-798.

[3] Blankertz, Benjamin, Ryota Tomioka, Steven Lemm, Motoaki Kawanabe, and K-R. Muller. ”Optimizing spatial filters for robust EEG single-trial analysis.” IEEE Signal processing magazine 25, no. 1 (2008): 41-56.

[4] Karpathy, Andrej, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. ”Large-scale video classification with convolutional neural networks.” In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 1725-1732. 2014.

[5] Donahue, Jeffrey, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. ”Long-term recurrent convolutional networks for visual recognition and description.” In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2625-2634. 2015.

[6] Lotte, Fabien, and Cuntai Guan. ”Regularizing common spatial patterns to improve BCI designs: unified theory and new algorithms.” IEEE Transactions on biomedical Engineering 58, no. 2 (2011): 355-362.

[7] Lemm, Steven, Benjamin Blankertz, Gabriel Curio, and K-R. Muller. ”Spatio-spectral filters for improving the classification of single trial EEG.” IEEE transactions on biomedical engineering 52, no. 9 (2005): 1541-1548.

[8] Tomioka, Ryota, Guido Dornhege, Guido Nolte, Benjamin Blankertz, Kazuyuki Aihara, and Klaus-Robert Müller. ”Spectrally weighted common spatial pattern algorithm for single trial EEG classification.” Dept. Math. Eng., Univ. Tokyo, Tokyo, Japan, Tech. Rep 40 (2006).

[9] Blankertz, Benjamin, Motoaki Kawanabe, Ryota Tomioka, Friederike U. Hohlefeld, Vadim V. Nikulin, and Klaus-Robert Müller. ”Invariant Common Spatial Patterns: Alleviating Nonstationarities in Brain-Computer Interfacing.” In NIPS, pp. 113-120. 2007.

[10] Samek, Wojciech, Carmen Vidaurre, Klaus-Robert Müller, and Motoaki Kawanabe. ”Stationary common spatial patterns for brain–computer interfacing.” Journal of neural engineering 9, no. 2 (2012): 026013.

[11] Wang, Haixian, and Wenming Zheng. “Local temporal common spatial patterns for robust single-trial EEG classification.” IEEE Transactions on Neural Systems and Rehabilitation Engineering 16, no. 2 (2008): 131-139.

[12] Wang, Haixian. ”Discriminant and adaptive extensions to local temporal common spatial patterns.” Pattern Recognition Letters 34, no. 10 (2013): 1125-1129.

[13] Noh, Eunho, and Virginia R. de Sa. ”Canonical correlation approach to common spatial patterns.” In Neural Engineering (NER), 2013 6th International IEEE/EMBS Conference on, pp. 669-672. IEEE, 2013.

[14] Hubel, D., and Wiesel, T. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. Journal of Physiology, 160:106, 1962.

[15] Fukushima, Kunihiko. ”Cognitron: A self-organizing multilayered neural network.” Biological cybernetics 20, no. 3-4 (1975): 121-136.

[16] Fukushima, Kunihiko, and Sei Miyake. ”Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition.” In Competition and cooperation in neural nets, pp. 267-285. Springer Berlin Heidelberg, 1982.

[17] LeCun, Y., Boser, B, Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., and Jackel, L.D. Handwritten digit recognition with a back-propagation network. In Advances in Neural Information Processing Systems 2, pages 396-404. Morgan Kaufmann, 1989.

[18] Mousavi, Mahta, Adam S. Koerner, Qiong Zhang, Eunho Noh, and Virginia R. de Sa. ”Improving motor imagery BCI with user response to feedback.” Brain-Computer Interfaces (2017): 1-13.

[19] MATLAB and Statistics Toolbox Release 2012b, The MathWorks Inc., Natick, Massachusetts, United States (2012).

[20] Delorme, Arnaud, and Scott Makeig. “EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis.” Journal of neuroscience methods 134.1 (2004): 9-21.

[21] Available here: `http://www.nitrc.org/projects/cleanlin`

[22] Gulcehre, Caglar, Kyunghyun Cho, Razvan Pascanu, and Yoshua Bengio. “Learned-norm pooling for deep feedforward and recurrent neural networks.” In Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 530-546. Springer Berlin Heidelberg, 2014.

[23] François Chollet. Keras. 2015. Available from: `https://github.com/fchollet/keras`

[24] Theano Development Team (2016). Theano: A Python frame-work for fast computation of mathematical expressions. arXiv e-prints, abs/1605.02688.

[25] Adelson, Edward H., and James R. Bergen. ”Spatiotemporal energy models for the perception of motion.” JOSA A 2, no. 2 (1985): 284-299.

[26] Vintch, Brett, J. Anthony Movshon, and Eero P. Simoncelli. ”A convolutional subunit model for neuronal responses in macaque V1.” Journal of Neuroscience 35, no. 44 (2015): 14829-14841.

[27] Sander Dieleman, Jan Schlüter, Colin Raffel, Eben Olson, Søren Kaae Sønderby, Daniel Nouri, et al. (2015). Lasagne: First release. [Data set]. Zenodo. `http://doi.org/10.5281/zenodo.27878`