

PAPER • OPEN ACCESS

A 1D CNN for high accuracy classification and transfer learning in motor imagery EEG-based brain-computer interface

To cite this article: F Mattioli *et al* 2021 *J. Neural Eng.* **18** 066053

View the [article online](#) for updates and enhancements.

You may also like

- [A novel flexible hydrogel electrode with a strong moisturizing ability for long-term EEG recording](#)

Gencai Shen, Kunpeng Gao, Nan Zhao et al.

- [Bidirectional flow of action potentials in axons drives activity dynamics in neuronal cultures](#)

JC Mateus, CDF Lopes, M Aroso et al.

- [Motor imagery practice and feedback effects on functional connectivity](#)

Carlos Alberto Stefano Filho, Romis Attux and Gabriela Castellano

Journal of Neural Engineering



OPEN ACCESS

PAPER

A 1D CNN for high accuracy classification and transfer learning in motor imagery EEG-based brain-computer interface

RECEIVED
29 August 2021REVISED
10 December 2021ACCEPTED FOR PUBLICATION
17 December 2021PUBLISHED
6 January 2022F Mattioli¹ , C Porcaro^{1,2,3,5,*} and G Baldassarre^{1,4,5,*} ¹ Institute of Cognitive Sciences and Technologies (ISTC), National Research Council (CNR), Rome, Italy² Department of Neuroscience and Padova Neuroscience Center (PNC), University of Padova, Padova, Italy³ Centre for Human Brain Health and School of Psychology, University of Birmingham, Birmingham, United Kingdom⁴ Start-up innovativa AI2Life Srl, spin-off company of ISTC-CNR, Roma, Italy⁵ C Porcaro and G Baldassarre have equally contributed to this article as the last authors.

* Authors to whom any correspondence should be addressed.

E-mail: camillo.porcaro@unipd.it and gianluca.baldassarre@istc.cnr.it**Keywords:** brain-computer interface, BCI, electroencephalogram, deep neural networks, data augmentation, transfer learning

Original Content from this work may be used under the terms of the Creative Commons Attribution 4.0 licence.

Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.



Abstract

Objective. Brain-computer interface (BCI) aims to establish communication paths between the brain processes and external devices. Different methods have been used to extract human intentions from electroencephalography (EEG) recordings. Those based on motor imagery (MI) seem to have a great potential for future applications. These approaches rely on the extraction of EEG distinctive patterns during imagined movements. Techniques able to extract patterns from raw signals represent an important target for BCI as they do not need labor-intensive data pre-processing. **Approach.** We propose a new approach based on a 10-layer one-dimensional convolution neural network (1D-CNN) to classify five brain states (four MI classes plus a ‘baseline’ class) using a data augmentation algorithm and a limited number of EEG channels. In addition, we present a transfer learning method used to extract critical features from the EEG group dataset and then to customize the model to the single individual by training its late layers with only 12-min individual-related data. **Main results.** The model tested with the ‘EEG Motor Movement/Imagery Dataset’ outperforms the current state-of-the-art models by achieving a 99.38% accuracy at the group level. In addition, the transfer learning approach we present achieves an average accuracy of 99.46%. **Significance.** The proposed methods could foster the development of future BCI applications relying on few-channel portable recording devices and individual-based training.

1. Introduction

A brain-computer interface (BCI) is a system able to establish a communication route between the brain and an external device [1]. BCI applications can be used for mapping, assisting, augmenting, or treating human cognitive or sensory-motor impairments [2, 3], as well as for recreational purposes [4, 5]. BCI systems are commonly formed by a recording device able to detect brain signals encoding the user’s intentions, a method used to decode the signals, and an actuator to control some devices [1] (e.g. computers, intelligent wheelchairs, or prostheses). In order to record human intentions from brain activity, numerous studies have investigated the use of invasive [2] and non-invasive techniques (e.g. electroencephalography—EEG, functional magnetic

resonance imaging—fMRI [6], and functional near-infrared spectroscopy—fNIRS, [7]). Although invasive techniques have demonstrated good efficacy [8, 9], most of them, such as the implantation of micro-electrode arrays, have proven to be risky [10, 11].

Non-invasive techniques, such as those based on EEG, are easy-to-use alternatives not requiring surgery. In addition, the deployment of commercial and free-scale EEG devices is growing [12], making these cost-effective tools well-suited for future commercial BCI systems [13]. In order to interpret human intentions from EEG signals, a large number of paradigms have been proposed. Among those, the two most popular are visual evoked potentials (VEP) [14, 15] and motor imagery (MI) [2, 3]. A VEP is a stereotyped electric potential recorded after a visual stimulus is

presented to the participant [16]. Although this technique seems to be effective [17], it requires the participant to stare at a screen and can also be strongly influenced by external stimuli [18]. Conversely, MI involves imagining the movement of a specific body part that results in a neuronal desynchronization of contralateral and ipsilateral parts of the sensorimotor cortex [19]. This approach, although requiring training, can be used under numerous conditions [20] such as in daily activities. Among the challenges to be handled by researchers in the EEG-based BCI field, we have the high within- and between-participant trial variability [21], and the low signal-to-noise ratio (SNR); these are mainly due to biological and non-biological artefacts recorded by the EEG system [22]. In order to increase the SNR and reduce the presence of artefacts, a labor-intensive pre-processing is commonly needed before interpreting the EEG signals and this represents a significant obstacle for broad use of MI EEG-based BCI [22]. Effective feature extraction and classification methods have thus become an important research topic [23]. One of the most successful paradigms for data pre-processing is independent component analysis (ICA) [24–27]. Other approaches include the wavelet transform (WT) [28], common spatial patterns (CSP) [29], empirical mode decomposition (EMD) [30], and functional source separation (FSS) [2, 14, 31, 32]. All these procedures require a substantial amount of expert's time to be applied. Once the EEG data are pre-processed and cleaned from artefacts, key features are extracted from the EEG signals and passed to classifiers that evaluate the input instance [33].

Recently, approaches based on *deep learning* (DL) have proven to be highly effective in the development of EEG-based BCI [1]. DL refers to a set of machine learning techniques based on artificial neural networks organized in layers [34]. Each layer uses various non-linear units to perform feature extraction and non-linear signal mapping. In the context of MI EEG-based BCI, DL techniques have been used for both feature extraction [35] and classification [36]. Among different DL paradigms, those based on *convolutional neural networks* (CNNs) have started to be employed with notable success in MI-based EEG classification tasks [1]. CNNs are multilayered neural networks inspired by the organization of the cerebral visual cortex [37]. Each convolutional layer is composed of multiple convolutional filters (or kernels). To perform the activation and training of a the convolutional layer corresponding to a filter, the filter slides over the input layer, which has to be spatially organised, and performs a convolution operation at positions located on the vertexes of a grid. Convolutional layers have two crucial properties: sparse interaction and parameter sharing. These two properties are achieved by using filters that are smaller in size with respect to the input pattern and thus can focus on spatially local features (sparse interaction); moreover, for

each input pattern the connection weights of the filter are used to compute the activation of the convolution layer units and are trained for the multiple visited locations on the pattern (parameter sharing). The result is a drastic decrease of the number of needed parameters and computational efficiency. In addition, and crucially, the non-linear transformations implemented by the multiple layers of the deep neural networks can extract low-level features that make pre-processing unnecessary. This makes this approach a very suitable tool for classification and regression tasks applied to EEG signals [1, 38–41].

Numerous contributions have demonstrated the effectiveness of CNNs in the classification of MI-based EEG signals [1]. Dose *et al* [42] implemented a CNN capable of classifying raw signals related to a 4-task MI dataset with an accuracy of 65.73% using different EEG channel configurations (from 9 to 64). Tang *et al* [30] proposed a CNN and EMD based method to classify a 2-task MI small dataset (360 instances) using 16 EEG channels and reached an 85.83% average accuracy per participant. More recently, Lun *et al* [43] proposed a CNN architecture able to classify a 4-task MI dataset achieving 97.28% average accuracy on multiple participants. Furthermore, the authors proposed an innovative method of channel selection using pairs of symmetrical channels, located near the central brain sulcus, as input to the neural network. Furthermore, the authors proposed an innovative method to feed the network with an input based on signals from electrode pairs rather than a large electrode configuration. The authors also used 1D convolutional layers where the kernels of filters slide only along the temporal dimension of the input. This approach is also used in the system proposed here.

Building on these recent contributions, here we present a novel CNN-based approach for MI signal classification of 4 MI tasks and the baseline considered as a fifth class. The system is called '1D-CNN' for ease of reference. The use of the baseline class goes beyond previous works as it allows the network to discriminate between EEG signals related to movement intentions and non-relevant or noisy signals produced when the participants do not intend to issue any command to the controlled device. This is a feature that is necessary for real-life BCI applications. To this purpose, we used a data augmentation procedure (SMOTE; [44]) both to balance the larger dataset of the baseline class with the dataset of the other classes and to have an overall larger dataset supporting a more effective training of the network. We also employed the method to search and find the most suitable EEG channel combinations allowing a parsimonious MI-based BCI favoring future cheap applications. The solution found is based on the use of few channels and achieves an accuracy of 99.38%; this outperforms the current state-of-the-art solution [43] trained on the same dataset (EEG Motor

Movement/Imagery Dataset, [45]) notwithstanding our tests included the additional baseline class. We also propose a transfer learning method that allows fast tailoring of the model for use with new users. To this purpose, we trained the 1D-CNN on the whole EEG dataset to extract general features at the early layers of the neural network; then, we retrained the 1D-CNN late layers by using only the 12-min EEG dataset related to a specific target user. The method achieves 99.46% of accuracy: to our knowledge, this represents a level of performance not previously achieved in this type of task.

2. Materials and methods

The 1D-CNN was implemented using the Python *Tensorflow* framework [46] (version 2.3). The code used for extracting the data from the original dataset, and the code used to implement the 1D-CNN model, is freely available online for download at: <https://github.com/Kubasinska/MI-EEG-1D-CNN>.

2.1. Dataset and ROIs

The data used in this work were obtained from the *EEG Motor Movement/Imagery Dataset V 1.0.0* [45]. This dataset consists of EEG recordings from 109 participants involving 4 tasks and 14 experimental runs. The data relating to participants 38, 88, 89, 92, 100, and 104 were excluded from the sample due to annotation errors, so having a dataset involving 103 participants. The participants performed the MI tasks while a 64-channel EEG signal was recorded with a BCI2000 system using the international 10-10 system (excluding electrodes Nz, F9, F10, FT9, FT10, A1, A2, TP9, TP10, P9, and P10) with a 160 Hz sampling rate and an average reference. The participants performed 14 experimental runs: 2 baseline runs (1 with eyes open, ‘run 1’, and 1 with eyes closed, ‘run 2’, of one minute each), and 3 runs for each of the following 4 task combinations (two minute each):

- (a) Experimental runs 3, 7, 11. A target appears on either the left or the right side of the screen. The participant opens and closes the corresponding fist until the target disappears. Then the participant relaxes.
- (b) Experimental runs 4, 8, 12. A target appears on either the left or the right side of the screen. The participant imagines opening and closing the corresponding fist until the target disappears. Then the participant relaxes.
- (c) Experimental runs 5, 9, 13. A target appears on either the top or the bottom of the screen. The participant either opens and closes both fists (if the target is on top) or moves both feet (if the target is on the bottom) until the target disappears. Then the participant relaxes.
- (d) Experimental runs 6, 10, 14. A target appears on either the top or the bottom of the screen. The

Table 1. Classes used in the tests of the model.

Class label	Description of the class
L	MI of opening and closing the left fist
R	MI of opening and closing the right fist
LR	MI of opening and closing both fists
F	MI of opening and closing both feet
B	Baseline

participant either imagines opening and closing both fists (if the target is on top) or imagines moving both feet (if the target is on the bottom) until the target disappears. Then the participant relaxes.

Here we use only task combinations (b) and (d) involving the imagined movements, that is, experimental runs 4, 6, 8, 10, 12, 14. Each experimental run involves 3 ‘tasks’ (one task involves a 4 s portion of the run dedicated to one mental activity) encoded as follows in the original database: *T0* corresponds to the baseline; *T1* corresponds to the imagined movement of the left fist in the experimental run related to the task combination (b) and of both fists in task combination (d); *T2* corresponds to the imagined movement of the right fist in the experimental runs related to task combination (b) and of both feet in task combination (d). Each run follows a stereotyped repeated timeline that consists of 4 seconds of baseline, 4 seconds of *T1*, 4 seconds of baseline, and 4 seconds of *T2*. In total, each run contains 15 baselines, 8 *T1*, and 7 *T2*. Following the strategy used in [43], we replaced the task nomenclature (*T0*, *T2*, *T3*) used in the original database to distinguish the 5 classes used in our BCI classification as indicated in table 1. This strategy uses the class label ‘L’ for *T1* related to task combination (b) (left fist), and the class label ‘LR’ for *T1* related to task combination (d) (left-right fists); similarly, it uses the class label ‘R’ for *T2* related to task combination (b) (right fist) and the class label ‘F’ for *T2* related to task combination (d) (both feet). In addition, a fifth class ‘B’ corresponding to the baseline was added for the reasons illustrated in section 1.

Since the classification task is related to imagined movements, we examined different subsets of channels located over the sensorimotor cortex. As demonstrated in the meta-analysis by Hétu *et al* [47], many studies show recruitment of frontoparietal and central areas during the execution of the imaginary movement of the lower and upper limbs. Building on field literature and previous works [43, 48], we considered the six regions of interest (ROIs) illustrated in figure 1. Consequently, we created and used six different datasets extracted from the original database, corresponding to the different ROIs. For each ROI, each experimental run was divided into 4 s segments and was assigned a label corresponding to one of the five classes used (L, R, LR, F, B). Each instance (a 4 segment) within the dataset consists of a matrix of size 640×2 : 2 corresponds to a pair of specular channels

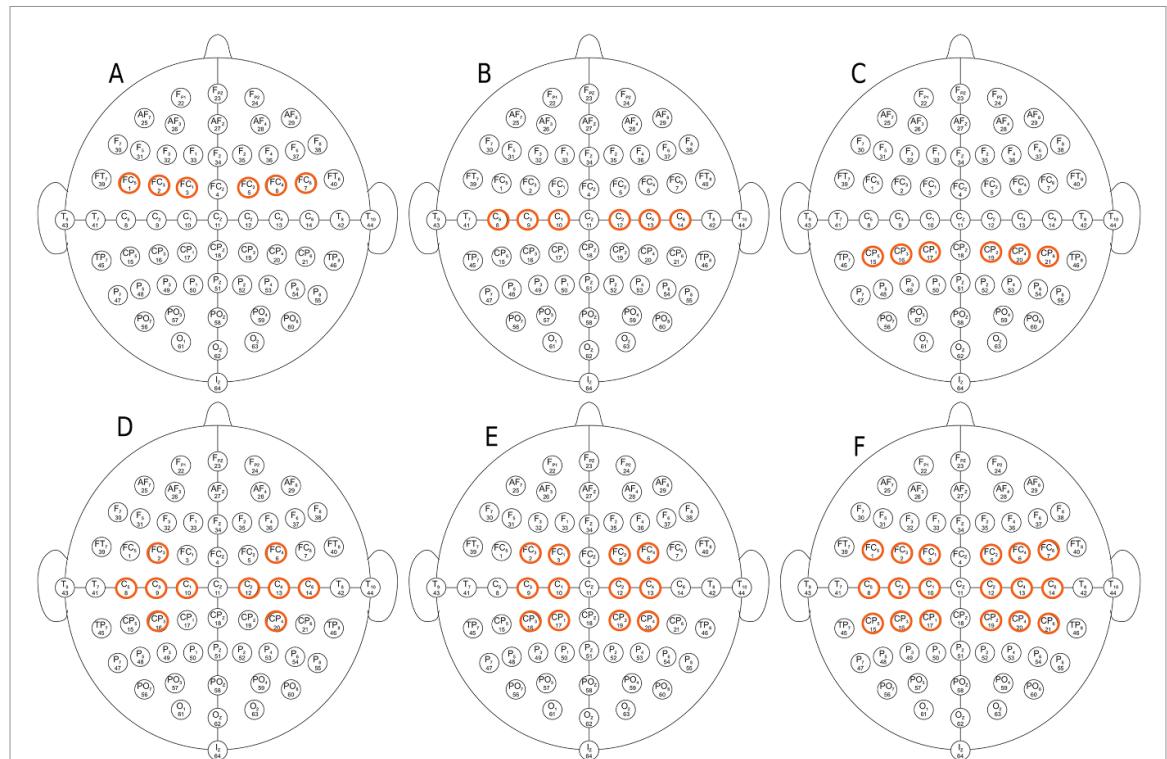


Figure 1. Visualisation of the six regions of interest (ROI) chosen in proximity of the sensorimotor cortex. The six ROIs (letters 'A' to 'F') correspond to the six schemes, each illustrating the position of the 64 EEG electrodes on the scalp (small circles, each reporting the standard designation). The channels forming each ROI are highlighted in orange (see table 2 for details).

Table 2. Description of channel pairs for each ROI.

ROI \ channels	Channels
A	[FC1, FC2], [FC3, FC4], [FC5, FC6]
B	[C5, C6], [C3, C4], [C1, C2]
C	[CP1, CP2], [CP3, CP4], [CP5, CP6]
D	[FC3, FC4], [C5, C6], [C3, C4], [C1, C2], [CP3, CP4]
E	[FC1, FC2], [FC3, FC4], [C3, C4], [C1, C2], [CP1, CP2], [CP3, CP4]
F	[FC1, FC2], [FC3, FC4], [FC5, FC6], [C5, C6], [C3, C4], [C1, C2], [CP1, CP2], [CP3, CP4], [CP5, CP6]

in the sagittal plane, and 640 corresponds to the time points considered (i.e. 160×4 , corresponding to 160 Hz and 4 s). Importantly, as in Lun *et al* [43], the dataset of each ROI is formed by the data related to each channel couple composing it considered as independent from the other couples and hence fed to the network as a separate input pattern. For example, for ROI E the data related to couple C3–C4, couple FC3–FC4, etc., are fed to the network as distinct input patterns during the training process. The channel pairs for each ROI are described in table 2. Henceforth, when we say 'dataset', we refer to a tensor having three dimensions: *instances, time points, channels*.

Importantly, for each ROI, the complete dataset is heavily unbalanced between the baseline class (B) and the other classes. Indeed, within each run, there are 15 baselines and 8 + 7 MI tasks (either L, R, LR, or F) as indicated above. These differences in the total

number of instances implies that data related to class B are about five times more than those related to the other classes. In addition, pilot tests showed that an imbalance between classes within the dataset creates a bias during network training. To solve the unbalance problem, we used the SMOTE data augmentation procedure for each ROI (see section 2.2). This solution leverages the possibility of using SMOTE (or analogous data augmentation procedures) to produce as many additional examples as needed to increase the number of instances of the small-size data classes.

The whole dataset of each ROI was randomly split into three sub-datasets where 80% of the data formed the training set, 10% the validation set, and 10% the test set. The splitting was stratified for the classes in a balanced manner. The split ensures that the neural network has no information about the test set both during training (based on the training set) and searching of the meta-parameters by the experimenter (based on the validation set). The three datasets for each ROI were scaled separately using a min-max normalization.

2.2. Data augmentation

The data augmentation procedure used is the *synthetic minority over-sampling technique* (SMOTE; [44]). SMOTE solves the imbalance problem between classes by creating synthetic data for the classes having fewer data, the so called 'minority classes'. The data augmentation procedure is not applied to the class with the most significant number of instances,

Table 3. Number of instances for each channel combination (ROIs) before and after the application of the data augmentation technique SMOTE.

ROI \ SMOTE	Before SMOTE	After SMOTE
A	43 994	110 160
B	43 994	110 160
C	43 994	110 160
D	73 324	183 600
E	87 988	220 320
F	131 983	330 480

the baseline (B), since it represents the ‘majority class’. This allows one to increase the number of instances of the minority classes to have the same number as the majority class. The creation of synthetic data is based on a k -nearest neighbors algorithm and linear interpolation. In particular, SMOTE chooses at random an instance x_i from the minority class, and then a neighbour instance x_{zi} from the k nearest neighbors of x_i (here, as typically done, $k = 5$). A new synthetic instance x_{new} is generated by linearly interpolating between x_i and x_{zi} by drawing a random number λ in the range $[0, 1]$:

$$x_{\text{new}} = x_i + \lambda(x_{zi} - x_i). \quad (1)$$

New synthetic instances are created in this way for each minority class until all classes achieve a balanced numerosity equal to the one of the majority class (baseline). The validation and test datasets are not involved in this procedure so that testing and validation are based only on real instances rather than synthetic ones. Table 3 shows the number of instances that each ROI dataset had before and after the application of SMOTE.

2.3. One-dimensional convolutional neural network (1D-CNN)

The MI-EEG BCI system proposed here is based on a one-dimensional convolutional neural network (1D-CNN; [49]) characterised by the fact that during convolution the CNN kernels slide only over the elements of 1 dimension of the input pattern, here time. In particular, the 1D-CNN takes as input a matrix with dimensions $M \times N$ where M is the length of the time window considered and N is the number of EEG channels (in our case, $M = 640$, i.e. 160×4 steps, and $N = 2$, which corresponds to two symmetrical channels). Each 1D convolutional layer uses a kernel of variable dimension $Q \times N$ where Q is the temporal window covered by the filter and $N = 2$ since the kernel does not slide across the channels. The mathematical notation of a 1D convolutional layer is:

$$y_r = f \left(\sum_{q=1}^Q \sum_{n=1}^N w_{qn} x_{r+q, r+n} + b \right) \quad (2)$$

where y_r is the output of the unit r of the filter feature map of size R ($R = M$ in the case stride = 1 and

‘padding’ is used, see below); x is the two-dimensional input portion overlapping to the filter; w is the connection weight of the convolutional filter; b is the bias term and f the activation function of the filter. To calculate the dimension of the filter feature map after the convolution operation (R), we can use the following formula:

$$R = \left[\frac{M - (K - 1) + 2 \times P}{S} \right] \quad (3)$$

where P is the padding size (padding involves the addition of 0-value pixels to the image sides to preserve an input-output proportion per dimension equal to the stride); and S is the stride (number of positions skipped by each shift of the filter during convolution).

The overall features of the CNN architecture are reported in table 4. The first convolutional layer (L1) uses 32 filters of size 20 with a stride of 1 and padding preserving the exact size of the input and output (referred to as ‘SAME’ in table 4, as in the Tensorflow framework). After the first convolutional layer, *batch normalization* (BN) is applied [50]. This involves normalization of the input to the next layer, which usually leads to substantially increased learning speed and has notable regularization effects improving the network generalization. BN works differently during training and testing. BN normalizes and zero-centers the input during training based on the entire batch (set of instances used to compute the loss and the gradient used by the learning algorithm). This allows the model to learn the optimal scaling of the input. In order to normalize and zero-center, the input BN estimates the parameter-dependent mean μ and variance σ^2 computed over the batch:

$$\mu = \frac{1}{b} \sum_{i=1}^b \mathbf{X}^{(i)} \quad (4)$$

$$\sigma^2 = \frac{1}{b} \sum_{i=1}^b (\mathbf{X}^{(i)} - \mu)^2 \quad (5)$$

where b is the number of instances in the batch and $\mathbf{X}^{(i)}$ an instance. Then the zero-centered normalized value $\hat{\mathbf{X}}^{(i)}$ for each instance is computed ($\xi = 10^{-5}$ avoids zero divisions):

$$\hat{\mathbf{X}}^{(i)} = \frac{\mathbf{X}^{(i)} - \mu}{\sqrt{\sigma^2 + \xi}}. \quad (6)$$

The normalisation might not be good for a given task, and so BN adds a further step during training, using trained parameters, that further scales and offsets the values as needed:

$$\mathbf{z}^i = \gamma \otimes \hat{\mathbf{X}}^{(i)} + \beta \quad (7)$$

Table 4. Network architecture.

Layer \ Features	Type	Input size	Filters	Kernel size	Stride	Padding	Activation
L1	1DConvolution	(640, 2)	32	20	1	SAME	relu
	Batch Normalization	(640, 32)					
L2	1DConvolution	(640, 32)	32	20	1	VALID	relu
	Batch Normalization	(621, 32)					
L3	Spatial Dropout	(621, 32)	32	6	1	VALID	relu
	1DConvolution	(621, 32)					
L4	Average Pooling 1D	(616, 32)	32	6	1	VALID	relu
L5	1DConvolution	(308, 32)					
L6	Spatial Dropout	(303, 32)	32	6	1	VALID	relu
	Flatten	(1, 9696)					
L7	Fully-connected	(1, 296)	32	6	1	VALID	relu
	Dropout	(1, 296)					
L8	Fully-connected	(1, 148)	32	6	1	VALID	relu
	Dropout	(1, 148)					
L9	Fully-connected	(1, 74)	32	6	1	VALID	relu
	Dropout	(1, 74)					
L10	Fully-connected	(1, 74)	32	6	1	VALID	softmax

where \otimes is the element-wise multiplication between each input value and the corresponding scaling parameter, γ are the scale parameters, and β are the offset parameters (both trained through backpropagation). During testing, the mean μ and variance σ^2 parameters cannot be computed based on the batch. So the algorithm uses for them the values computed with a moving average during training. Layer 2 (L2) is a second convolutional layer with the same parameters as the first convolutional layer (L1) with the difference that padding in this layer is not applied (referred to as ‘VALID’ in table 4 as in the Tensorflow framework). BN was applied to L2 as done for L1. In addition, *spatial dropout* [51] was applied for additional regularisation. Standard dropout works during the training phase by excluding units, with probability p at each training step, from spreading and learning. In the case of spatial dropout, entire feature maps are discarded in order to improve generalisation across maps. Here the spatial dropout probability was set to $p = 50\%$. Afterward, an additional convolutional layer (L3) was applied, having a smaller kernel (6) and ‘VALID’ padding. This was followed by a 1D average pooling layer (L4). Pooling is essential for CNNs to reduce the input size and decrease the needed computation and number of network parameters. Furthermore, this size reduction tends to make the representation space invariant concerning small translations of the input. This allows the network to recognize specific patterns at different locations within the feature map. This operation is performed on each feature map independently. The size of the pooling operation was 2×1 applied with a stride of 2, thus reducing the size of each dimension by a factor of 2. Next, an additional convolution layer (L5) and spatial dropout was applied. This is followed by a *flatten layer* (L6) that reshapes the matrix input to a vector to support the processing of the following non-spatial layers. The following three layers are a

series of fully connected layers with dropout, formed by respectively 296 units (L7), 148 units (L8), and 74 units (L9). In each of these layers, all units are fully connected to all units in the previous layer so that each unit activation $y_j(l)$ is computed as follows:

$$y_j^{(l)} = f \left(\sum_{i=1}^I w_{ji}^{(l)} \cdot x_i^{(l-1)} + b_j^{(l)} \right) \quad (8)$$

where I is the number of units in the previous layer, l is the current layer, $w_{ji}^{(l)}$ is the weight of the connection between unit j of this layer and unit i of the previous layer, $b_j^{(l)}$ is the bias term of unit j , and f is the unit transfer function. With the exception of the output layer, the transfer function f of the units of all layers is a *rectified linear unit* (ReLU) function:

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ 0.01 \cdot x, & \text{otherwise.} \end{cases} \quad (9)$$

The five units of the last output layer use a *softmax* function to encode the probabilities \hat{y}^i of the five categories of the MI classification task:

$$\hat{y}_i = \operatorname{argmax} \left(\frac{e^{y_i}}{\sum_{i=1}^5 e^{y_i}} \right). \quad (10)$$

2.4. Network training

The 1D-CNN was trained separately for each of the six different ROIs discussed in section 2.1. The optimisation of the neural network parameters was based on the categorical cross-entropy loss function [34]:

$$\text{loss} = - \sum_{i=1}^5 (y_i^* \cdot \log \hat{y}_i) \quad (11)$$

where \hat{y}_i is the i the output prediction and y_i^* is the corresponding target value (1 for the correct class and

0 otherwise). The Adam optimisation algorithm [52] was used to minimize the categorical cross-entropy loss and update the CNN parameters. Choosing the optimization algorithm is still an open research question in the DL literature [1]. However, in their systematic review, Alzahab and colleagues [1] reported many studies that use Adam optimisation. For this reason, we have chosen to use this same optimisation algorithm instead of testing different ones. Adam optimisation is a stochastic gradient descent (SGD) algorithm based on a different learning rate for each parameter that adapts based on the first-order and second-order moments of the gradient. The hyperparameters of the algorithm were set as follows: $\alpha = 0.0001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$. The SGD used to train the 6 networks used a batch with size 10 (i.e. the error gradient was calculated for blocks of 10 instances).

Training used a non-fixed number of epochs (one epoch corresponds to one learning iteration using all the instances once in a randomised order), in particular it used the *validation early stopping* technique to avoid overfitting [53]. Based on this technique, training is stopped when the loss on the validation set does not improve for at least a threshold value (1^{-3}) for four consecutive epochs. In addition, the *checkpointing* technique [54] was also used to improve the efficiency of the training process. In particular, at every epoch e_t all parameters of the 1D-CNN were saved. If in an epoch e_t the validation loss did not improve, the parameters from the previous epoch e_{t-1} were used for the next epoch e_{t+1} , thus discarding epochs lowering performance. Training and testing were performed on a machine using the GPU NVIDIA GeForce RTX 2060 s. The training of a 1D-CNN for a given ROI took about 60 min.

2.5. 1D-CNN transfer learning

Transfer learning [55] can be used to exploit the general regularities and features that a neural network can gather from a large dataset (e.g. from MI-EGG data from a large number of individuals) to become able to predict data on a new similar dataset (e.g. related to a new individual) with minimum additional training. In our approach, we applied transfer learning to one individual considered as a new *target individual*. The procedure was repeated for seven randomly selected individuals considered as targets: 34, 10, 65, 90, 101, 53, 4. For each target individual, we used this procedure: (a) a dataset was built with instances of all individuals of the original dataset but the target individual (so 101 individuals); (b) the resulting dataset was split between a training set (90% of data) and a validation set (10% of data); (c) the training set was augmented with the SMOTE approach illustrated in section 2.2; (d) the resulting dataset was used to train the 1D-DNN as described in section 2.4; (e) the dataset related to the target individual, which

at this step of the procedure was new to the network, was extracted from the original dataset; (f) the resulting dataset was divided into a training set (80% of data) and a test set (20% of data; note that in this phase of transfer learning, a validation set is not required as the network hyper-parameters are already optimised in early phases through the ‘population’ validation set and then left unchanged); (g) the target-individual training set was augmented with SMOTE; (h) the weights of the entire network up to, and including, layer L3 were ‘frozen’, that is, excluded from the following training (and also from the batch normalisation and spatial dropout); (i) the rest of the network (from Layer 4) was re-trained with the target-individual training set using a fixed number of epochs (30); (j) finally, the model was tested with the target-individual test set.

2.6. Performance metrics

The 1D-CNN performance on the 6 different ROIs and for each task were measured through four indexes: precision, recall, accuracy, and F1-score. These indexes are defined on the basis of ratios between the frequencies of the cells of the *confusion matrix* reporting the instance frequencies of the true classes along the rows, and the instance frequencies of the predicted classes along the columns. In the case of multi-class classification, for a given class the cells, or set of cells, considered by the ratios are as follows: *true positives* (TP) indicate the number of instances of the class that are correctly predicted by the algorithm; *false positives* (FP) indicate the number of instances not belonging to the class that are mistakenly classified as belonging to the class; *true negatives* (TN) indicate the number of instances not belonging to the class (thus involving all other classes) that are correctly predicted as non belonging to the class; *false negatives* (FN) indicate the number of instances of the class that are mistakenly classified as non belonging to it. On this basis, the four indexes are computed as follows.

Precision for a given class:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \quad (12)$$

Recall for a given class:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \quad (13)$$

Accuracy for a given class:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \quad (14)$$

F1-score for a given class (the index is the harmonic average of Recall and Precision):

$$\text{F1-score} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}. \quad (15)$$

Table 5. 1D-CNN training epochs, final test-set loss, and performance indexes of the different ROIs (A–F) averaged over all classes and computed with the test set.

Metrics \ ROI	A	B	C	D	E	F
Epochs number	56	57	45	46	54	33
Test-set loss	0.12	0.15	0.13	0.04	0.02	0.06
Avg recall (%)	95.55	94.31	95.36	98.52	99.21	98.29
Avg precision (%)	96.81	96.05	96.52	99.12	99.46	98.71
Avg F1-score(%)	96.14	95.15	95.92	98.82	99.33	98.50
Avg accuracy(%)	96.66	95.65	96.44	98.89	99.38	98.59

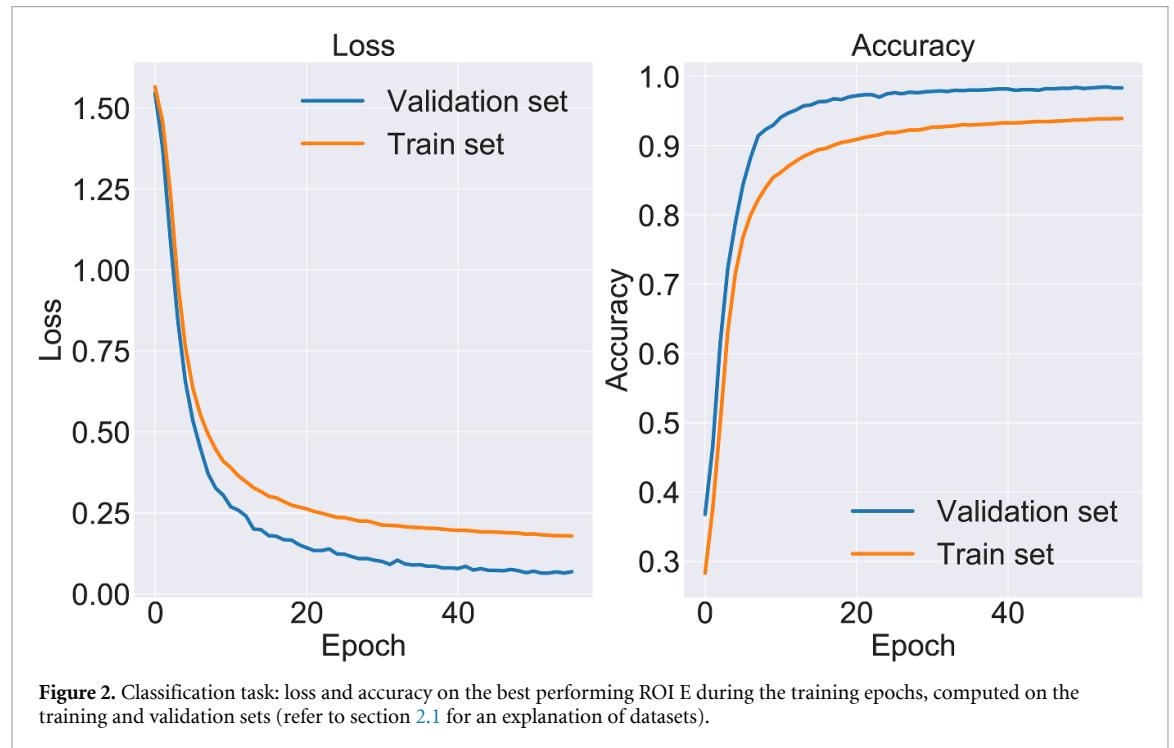


Figure 2. Classification task: loss and accuracy on the best performing ROI E during the training epochs, computed on the training and validation sets (refer to section 2.1 for an explanation of datasets).

For all indexes, a higher value indicates a higher performance of the model. Note that cross validation, commonly useful for small datasets, was not needed here as the original dataset is relatively large and was further enlarged with the data augmentation procedure. The good final performance achieved by our approach supports this choice.

3. Results

3.1. 1D-CNN overall classification performance

Table 5 shows the results of the classification for the 6 channel combinations (ROIs) considered. In particular, the table reports the number of epochs, the test-set final loss, and the performance indexes computed on the test set and averaged over all classes. Lower performance was obtained for ROIs A (96.66%), B (95.65%), and C (96.44%) and higher performance for ROIs D (98.89%), E (99.38%), F (98.59%). Overall, the best performance was obtained by ROI E (99.38%).

Figure 2 shows the progression of loss and accuracy during the training of the best ROI E, for the

training and validation sets. The curves, in particular those of the validation set, reach a plateau thus indicating that the training did not incur in overfitting.

Figure 3 shows the confusion matrix for ROI E. In the matrix, the number of test instances in each class is 5508 (B), 1366 (R), 1389 (RL), 1370 (L), and 1366 (F), respectively. The matrix shows the system incurs in few errors. Figure 4 shows the ROC curve for ROI E. The curve shows that the algorithm found class B slightly harder to learn, likely because the baseline class is highly variable between individuals and across time. Table 6 reports the performance metrics for each class of ROI E. These results show the capacity of the 1D-CNN to classify the test set instances with a similar final accuracy (>99%) for all classes, including the baseline.

3.2. Classification performance: contribution of different techniques

We run three tests to investigate the contribution of the different elements of the 1D-CNN and training procedures to the system's high performance. In all these tests, we focused on the best performing ROI E.

B	5490	6	5	1	6
R	4	1361	0	0	1
RL	10	4	1374	0	1
L	18	0	1	1350	1
F	9	1	0	0	1356

Figure 3. Classification task: confusion matrix for ROI E on the test set. The letters represent the classes: B: baseline; R: imagined movement of the right fist; RL: imagined movement of both fists; L: imagined movement of the left fist; F: imagined movement of both feet.

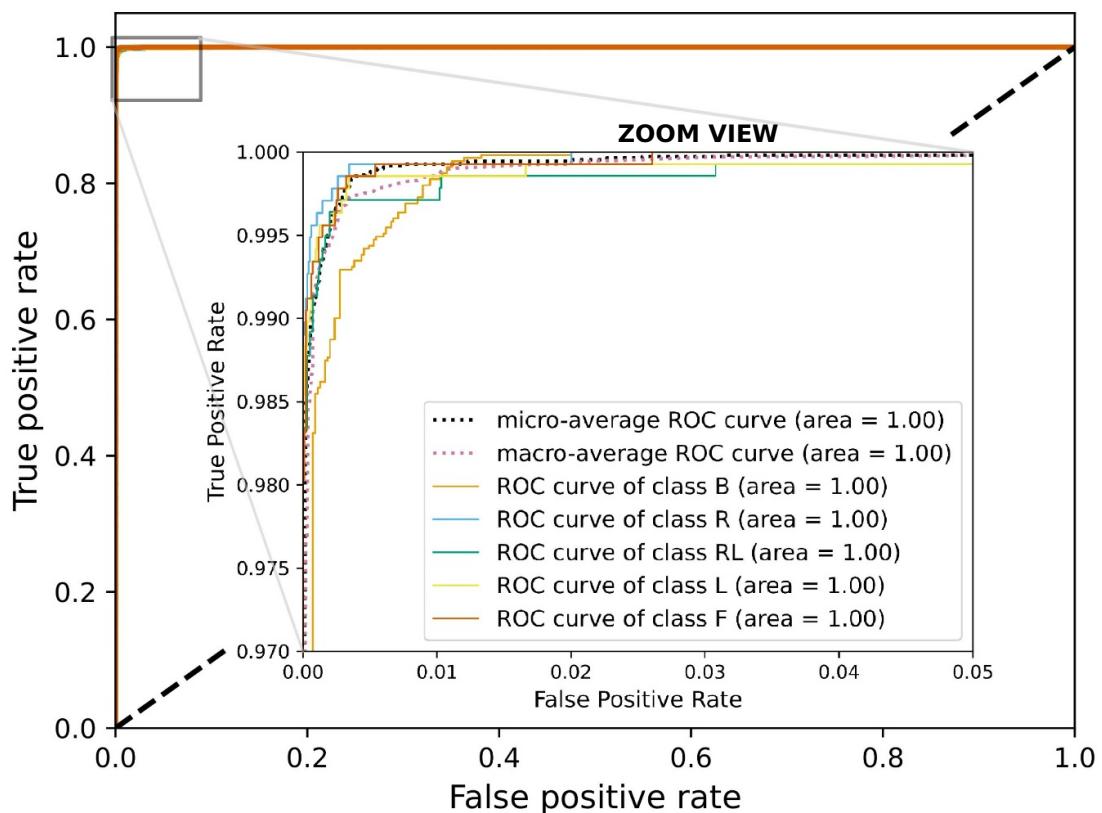


Figure 4. ROC curve for ROI E (the small plot is a magnification of the critical part of the curves).

Table 6. Classification task: metrics of ROI E for each class.

Metrics \ class	B	R	LR	L	F
Recall (%)	99.67	99.63	98.92	98.54	99.27
Precision (%)	99.26	99.20	99.57	99.93	99.34
F1-score (%)	99.47	99.42	99.24	99.23	99.30
Accuracy (%)	99.67	99.63	98.92	98.54	99.26

Table 7. Classification task: performance of the 1D-CNN when some techniques employed for training are omitted.

Metrics \ technique	No BN	No early stopping	No data augmentation
Epochs number	37	100	15
Test-set loss	0.17	0.07	10.61
Avg recall (%)	96.24	96.85	24.13
Avg precision (%)	95.32	97.20	21.01
Avg F1-score (%)	96.65	97.97	20.16
Test-set accuracy (%)	96.75	97.99	33.38

Table 8. Transfer learning with seven target individuals (S34, s10, etc.): performance metrics when we trained the 1D-CNN with all data but those of the target individual, and then tested it with the latter (marked with *), or when we used transfer learning (all others), involving the retrain with the target individual.

Metrics \ participants	S34	S10	S65	S90	S101	S53	S4	Average
Loss *	1.92	2.0	2.27	1.94	1.86	2.21	2.12	2.04
Loss	0.00	0.00	0.00	0.00	0.05	0.00	0.04	0.09
Precision (%)	100.00	98.57	100.00	100.00	98.52	100.00	98.90	99.40
Recall (%)	100.00	99.63	100.00	100.00	98.39	100.00	97.75	99.35
F1-score (%)	100.00	99.07	100.00	100.00	98.65	100.00	98.28	99.42
Accuracy (%) *	53.41	45.13	52.67	50.98	47.16	50.56	51.58	50.21
Accuracy (%)	100.00	99.07	100.00	100.00	98.60	100.00	98.61	99.46

Table 9. Comparison of the performance of the system presented here with other state-of-the-art systems using the same dataset and a CNN.

Work \ metrics	Classes no	Avg accuracy (%)	Transfer learning (% avg 7 partic.)
Dose <i>et al</i> [42]	4	65.73	68.51
Karácsány <i>et al</i> [56]	4	76.37	—
Lun <i>et al</i> [43]	4	97.28	—
This work	4 + 1	99.38	99.46

3.2.1. 1D-CNN classification performance with no batch normalization

To test the contribution of BN, we performed the training of the system again while not using it and then measured the performance indexes again. Table 7 shows the results. On average, the indexes have a drop of about $\approx 3\%$ points, from 99.38% to 96.75%, thus showing the importance of the BN for EEG-based MI-BCI.

3.2.2. 1D-CNN classification performance with no early stopping

In order to evaluate the contribution of the early stopping and checkpointing techniques, we set the training epochs number to a fixed value, 100, and did not use checkpointing. This could cause overfitting and learning drift, alongside a longer training time. Table 7 shows the resulting performance metrics on the test set. The results show that the performance drops of about $\approx 1.5\%$ points, from 99.38% to 97.99%, notwithstanding the doubled training time (from 54 to 100 epochs), thus showing the importance of *early stopping* and *checkpointing* for the task at hand.

3.2.3. 1D-CNN classification performance with no data augmentation

To show the importance of the data augmentation procedure, we retrained the system with the original

dataset ROI E, involving 87 988 vs. 220 320 training instances (with 10 999 validation instances, and 10 999 test instances). Training the 1D-CNN with this reduced and unbalanced dataset leads the *early stopping* process to terminate the training after a few epochs since the validation-set loss stops improving. Table 7 shows that this results in a very low accuracy (33.38%).

3.3. 1D-CNN transfer learning performance

We then evaluated the performance of the 1D-CNN in transfer learning (section 2.5). Table 8 shows the performance indexes of the 1D-CNN for all the seven participants considered for transfer learning as target individuals. For comparison, the table also shows the loss and accuracy when the network is trained with the whole dataset but the target individual and then directly tested with the latter without performing the additional training with his/her data. The results show the high effectiveness of the used transfer learning procedure, reaching an average accuracy for the seven target individuals of 99.46% vs. 50.21% obtained in the case of lack of additional training with their specific data.

3.4. Comparison with other systems

Table 9 presents a comparison of the performance of our system with the performance of other state-of-the-art systems [42, 43, 56] that used the same dataset

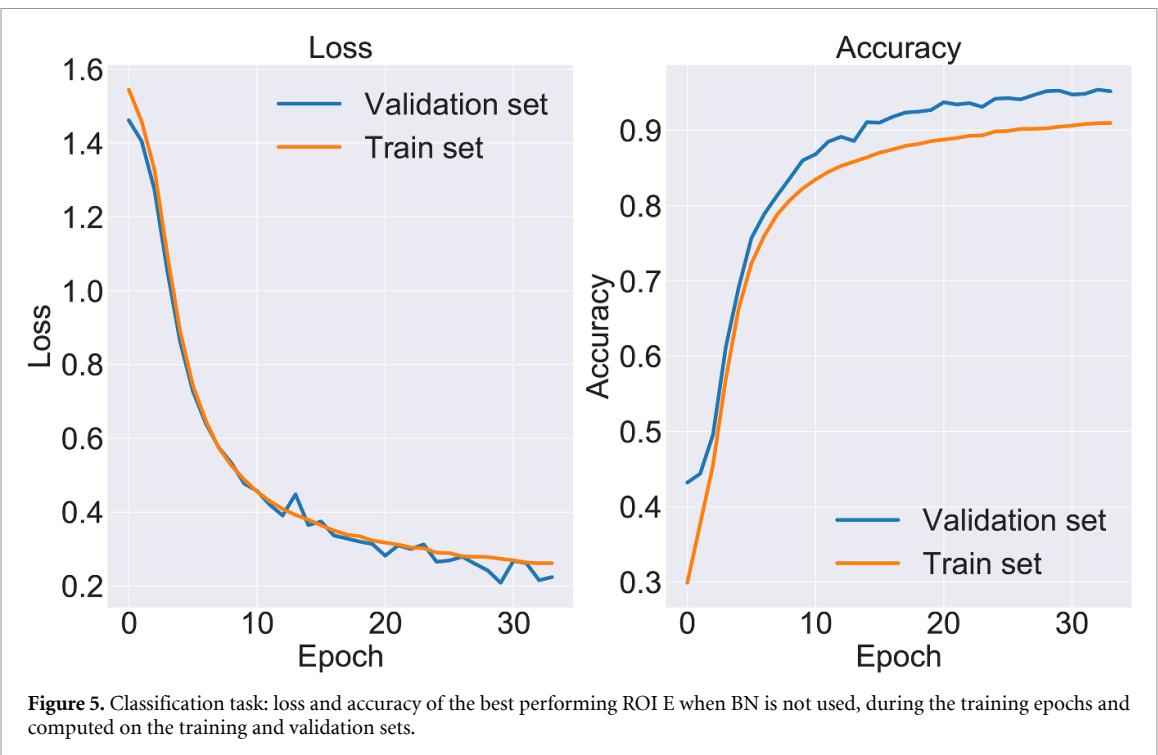


Figure 5. Classification task: loss and accuracy of the best performing ROI E when BN is not used, during the training epochs and computed on the training and validation sets.

used here [45] and an approach based on a CNN. Our system achieved higher performance than the other systems even if we considered the additional challenging baseline class and the other MI-related classes. The table also reports the performance of our system in the transfer learning task.

4. Discussion

The results on classification (section 3.1) demonstrate the high performance of the 1D-CNN proposed here in predicting five different MI-induced mental states also involving a baseline class. Our analysis allowed the identification of the most effective channel combination (ROI E, formed by six couples of channels), leading to the high classification accuracy of 99.38% (table 5). The system finds the baseline class slightly more challenging to learn than the other classes (figure 4). Finally, it achieves a similar high performance for all five classes (table 6) in a relatively short learning time (59 experiences of the dataset).

An important aspect of the BCI system proposed here is that the 1D-CNN is trained with data from various channel pairs close to the motor cortex, but once trained the network requires only an EEG input signal coming from a *single pair* of specular channels to be able to classify the brain state. This feature opens up the possibility of developing BCI systems based on simple EEG devices using a few electrodes. Table 5 shows that the length of training varies between the different ROIs. In part, this is due to the random initialization of the convolutional filters. However, it is also due to the different size of the datasets of the

ROIs, caused by the fact that each ROI encompassed a different number of electrode couples and hence had a training dataset of different sizes. For example, ROI F had the fastest convergence time and also had the most extensive dataset.

The use of *early stopping* and *checkpointing* techniques were very effective to reduce the number of epochs needed for training the system, and to reduce the risk of overfitting. The results indeed showed that the number of epochs needed for convergence is lower than in other works; for instance, our 1D-CNN using the best ROI E took 54 training epochs whereas Lun *et al* [43], the previous state-of-the-art system (also based on a CNN), took 2000 epochs to converge with the same dataset. Another relevant element of the approach proposed here is *batch normalization*, used after the first two convolutional layers. Figure 5 shows how, without batch normalization, accuracy and loss increase more slowly and are less stable during training in comparison to when it is used. Recent works have investigated the use of different techniques for EEG data augmentation [57–59].

Among different data augmentation procedures, SMOTE has been successfully applied to augment magnetic resonance imaging data [60, 61]. Here we used the SMOTE algorithm for two purposes. First, we aimed to balance the dataset between the different classes, especially after introducing the important baseline class. Second, we aimed to have more data in general, particularly as SMOTE is known to contribute to more significant within-class variability of instances, which improves generalisation. Table 7 shows that with ROI E and SMOTE, the 1D-CNN achieves an accuracy of 99.38% while without

SMOTE, the network achieves only an accuracy of 33.38%, close to the chance level (20% with five classes).

The approach we proposed for transfer learning showed to be very effective. The approach trained the 1D-CNN with the whole dataset with the exclusion of a ‘target individual’, and then ‘froze’ the early network layers and re-trained the last layers with the data from the target individual. Table 8 shows how the network has an average accuracy of 50.21% (average over seven target individuals with which the experiment was repeated) when it is used after training on the whole population but without the additional transfer-learning re-training of the late layers. Instead, with the re-training of the late layers based on the data from the target individual, it achieves an accuracy of 99.46% (average over the seven target individuals). This shows the potential of the approach for BCI applications because such high accuracy is obtained with only 12 min of EEG recording from the target individual. BCI systems that require minimal training for new users are essential for future applications [62]. Other recent systems have developed CNN-based methods that allow for participant-independent classification [63, 64], but the results tend to use long training times and to achieve a lower performance than our system, for example, an accuracy ranging from 50% to 80% [63, 65]. Instead, our approach achieves a high accuracy (99.46%) with additional short training (12 min EEG recording) and often with a simpler architecture [60].

Table 9 compares the overall performance (accuracy) of our system with the best performing previous systems and shows its highest performance. The reasons for this higher performance are explained by these differences in the approaches used. The system proposed in [42], achieving an average accuracy of 65.73% with four classes (no baseline class), used raw input data from all the 64 channels. Moreover, it used a rather simple neural network and a 2D CNN approach rather than the 1D approach applied here and in [43]. In addition, it did not use any form of regularization and data augmentation technique. These differences could explain the lower performance with respect to the other systems reported in the table. The work also used transfer learning (based on additional training of the whole network) and so increased the average performance with the data of new participants to 68.51%. The system proposed in [56], achieving an average accuracy of 76.37% with four classes (no baseline class), used different channel configurations, and achieved a maximum performance with input data from all the 64 channels. The system used pre-processed data, rather than raw data as here, so possibly losing relevant information. Moreover, it used a 2D CNN approach, rather than the 1D approach used here and in [43], it did not use the spatial drop-out, and it did not use any data augmentation technique, thus missing the

opportunities given by these strategies. Finally, the system proposed in [43] reports the second-best performance after our system (97.28%) but it used only four classes with no baseline, making the task simpler than the one faced here. The lower performance with respect to our system (99.38%) could be explained by the fact that they did not use any form of data augmentation. In addition, the work in [43] did not systematically explore different ROIs as here: in particular, it focused on the ROI F that was shown here to have an accuracy of 98.59% versus the best ROI E achieving 99.38%.

This difference might be due to the nature of the used 1D CNN strategy. The core of the idea of using the 1D-CNN applied to EEG is to use data from different channel-couples while not informing the network about the spatial localisation of the channels’ electrodes on the scalp. This has two important effects: (a) it forces the network to capture time regularities of the EEG signals while abstracting over the differences due to the different spacial localisation of the electrodes (which, within a certain ROI, are not so important as the EEG has a low spatial resolution); (b) it multiplies the dataset by the number of channel couples forming the ROI, so representing a ‘natural’ form of data augmentation: this is very important as deep neural networks can dramatically increase their performance when the training dataset is substantially enlarged. This data augmentation advantage, however, can encounter the limitation due to the fact that if the ROI is excessively large (e.g. ROI F), it involves channels that are increasingly distant from the relevant cortical signal source, and so might introduce space-related differences that confound the network. Instead, the smaller ROI E is formed by spatially contiguous channels that allow data augmentation and at the same time do not introduce too impairing distortions due to their spacial localisation of the involved electrodes.

5. Conclusions

We have proposed a new BCI method for classifying motor-imagery-based EEG signals into five brain-state classes also including a challenging baseline class. The method is based on a 1D-CNN and integrates *batch-normalization* to improve generalization, *data augmentation* (SMOTE) for increasing and class-balancing the dataset, and *early stopping* and *checkpointing* to avoid overfitting and learning drift. The system learns on the basis of data collected from couples of channels located in the proximity of the motor cortex, and once trained it is able to classify the target brain states based on the signals from only one channel couple. The system achieves a high level of accuracy (99.38%) and outperforms the current state-of-the-art model (97.28%, [43]) even if it classifies the additional baseline class.

In addition, we have proposed a transfer learning approach directed to support an EEG BCI that is applicable to the single individual with minimal additional training. The approach allows the 1D-CNN to learn general features from a large population of individuals and then can be applied to a target individual with only additional training involving data from a 12-min EEG recording. The approach achieves a high accuracy of 99.46% that, to our knowledge, has not been achieved by previous BCI approaches.

Future work will aim to reduce the 4 s time window used by this and other CNN-based BCI approaches as such window implies a too long delay for future online BCI applications. Moreover, we will also address the continuous incoming flow of data from the EEG recordings. This does not allow the alignment of the data window fed to the CNN with the onset of the imagination events. This makes the detection of motor imagery events much harder. Notwithstanding these remaining challenges, this work shows how DL techniques, particularly those based on CNNs, are a valuable tool to release BCI systems based on EEG signals produced with motor imagery. In particular, the presented results, alongside the use of the signals from only two EEG channels, show the potential of the proposed approach for the future development of cheap portable EEG-based BCI systems.

Data availability statement

The data that support the findings of this study are openly available at the following URL/DOI: <https://physionet.org/content/eegmmidb/1.0.0/>.

Acknowledgment

We thank Adriano Capirchio, CEO of AI2Life Srl, for his help on the use of the data augmentation method SMOTE.

Funding

This project has received funding from the European Union's Horizon 2020 Research and Innovation Program under Grant Agreement No. 713 010, Project 'GOAL-Robots-Goal-based Open-ended Autonomous Learning Robots'. This research has also received support from the the 'Advanced School of AI', AS-AI 2020-2021, <https://as-ai.org/>.

ORCID iDs

F Mattioli  <https://orcid.org/0000-0001-8298-4143>
C Porcaro  <https://orcid.org/0000-0003-4847-163X>
G Baldassarre  <https://orcid.org/0000-0002-1277-4447>

References

- [1] Alzahab N A, Apollonio L, Di Iorio A, Alshalak M, Iarlori S, Ferracuti F, Monterù A and Porcaro C 2021 Hybrid deep learning (HDL)-based brain-computer interface (BCI) systems: a systematic review *Brain Sci.* **11** 75
- [2] Di Pino G, Porcaro C, Tombini M, Assenza G, Pellegrino G, Tecchio F and Rossini P M 2012 A neurally-interfaced hand prosthesis tuned inter-hemispheric communication *Restorative Neurol. Neurosci.* **30** 407–18
- [3] Tombini M, Rigosa J, Zappasodi F, Porcaro C, Citi L, Carpaneto J, Rossini P M and Micera S 2012 Combined analysis of cortical (EEG) and nerve stump signals improves robotic hand control *Neurorehabil. Neural Repair* **26** 275–81
- [4] Vasiljevic G A M and de Miranda L C 2020 Brain–computer interface games based on consumer-grade EEG devices: a systematic literature review *Int. J. Hum.–Comput. Interact.* **36** 105–42
- [5] Marshall D, Coyle D, Wilson S and Callaghan M 2013 Games, gameplay and BCI: the state of the art *IEEE Trans. Computat. Intell. AI Games* **5** 82–99
- [6] Weiskopf N, Mathiak K, Bock S W, Scharnowski F, Veit R, Grödd W, Goebel R and Birbaumer N 2004 Principles of a brain-computer interface (BCI) based on real-time functional magnetic resonance imaging (fMRI) *IEEE Trans. Biomed. Eng.* **51** 966–70
- [7] Hong K S, Naseer N and Kim Y H 2015 Classification of prefrontal and motor cortex signals for three-class fNIRS-BCI *Neurosci. Lett.* **587** 87–92
- [8] Collinger J L et al 2013 High-performance neuroprosthetic control by an individual with tetraplegia *Lancet* **381** 557–64
- [9] Hochberg L R et al 2012 Reach and grasp by people with tetraplegia using a neurally controlled robotic arm *Nature* **485** 372–5
- [10] Biran R, Martin D C and Tresco P A 2005 Neuronal cell loss accompanies the brain tissue response to chronically implanted silicon microelectrode arrays *Exp. Neurol.* **195** 115–26
- [11] Vetter R J, Williams J C, Hetke J F, Nunamaker E A and Kipke D R 2004 Chronic neural recording using silicon-substrate microelectrode arrays implanted in cerebral cortex *IEEE Trans. Biomed. Eng.* **51** 896–904
- [12] Ranky G and Adamovich S 2010 Analysis of a commercial EEG device for the control of a robot arm *Proc. 2010 IEEE 36th Annual Northeast Conf. (NEBEC)* (IEEE) pp 1–2
- [13] Carrino F, Dumoulin J, Mugellini E, Abou Khaled O and Ingold R 2012 A self-paced BCI system to control an electric wheelchair: evaluation of a commercial, low-cost EEG device *2012 ISSNIP Biosignals and Conf.: Biosignals and Robotics for Better and Safer Living (BRC)* (IEEE) pp 1–6
- [14] Ferracuti F, Casadei V, Marcantonio I, Iarlori S, Burattini L, Monterù A and Porcaro C 2020 A functional source separation algorithm to enhance error-related potentials monitoring in noninvasive brain-computer interface *Comput. Methods Programs Biomed.* **191** 105419
- [15] Ferracuti F, Freddi A, Iarlori S, Longhi S, Monterù A and Porcaro C 2021 Augmenting robot intelligence via EEG signals to avoid trajectory planning mistakes of a smart wheelchair *J. Ambient Intell. Humanized Comput.* **1**–13
- [16] Vialatte F B, Maurice M, Dauwels J and Cichocki A 2010 Steady-state visually evoked potentials: focus on essential paradigms and future perspectives *Prog. Neurobiol.* **90** 418–38
- [17] Combaz A and Van Hulle M M 2015 Simultaneous detection of p300 and steady-state visually evoked potentials for hybrid brain-computer interface *PLoS One* **10** e0121481
- [18] Salelkar S and Ray S 2020 Interaction between steady-state visually evoked potentials at nearby flicker frequencies *Sci. Rep.* **10** 5344
- [19] Pfurtscheller G and Neuper C 2001 Motor imagery and direct brain-computer communication *Proc. IEEE* **89** 1123–34

- [20] Schuster C, Hilfiker R, Amft O, Scheidhauer A, Andrews B, Butler J, Kischka U and Ettlin T 2011 Best practice for motor imagery: a systematic literature review on motor imagery training elements in five different disciplines *BMC Med.* **9** 75
- [21] Hall C 1985 Individual differences in the mental practice and imagery of motor skill performance *Can. J. Appl. Sport Sci.* **10** 17S–21S
- [22] Lotte F, Jeunet C, Mladenović J, N'Kaoua B and Pillette L 2018 A BCI challenge for the signal processing community: considering the user in the loop
- [23] Huang Q, Zhang Z, Yu T, He S and Li Y 2019 An EEG-/EOG-based hybrid brain-computer interface: application on controlling an integrated wheelchair robotic arm system *Frontiers Neurosci.* **13** 1243
- [24] Ruan J, Wu X, Zhou B, Guo X and Lv Z 2018 An automatic channel selection approach for ICA-based motor imagery brain computer interface *J. Med. Syst.* **42** 253
- [25] Barbati G, Porcaro C, Zappasodi F, Rossini P M and Tecchio F 2004 Optimization of an independent component analysis approach for artifact identification and removal in magnetoencephalographic signals *Clin. Neurophysiol.* **115** 1220–32
- [26] Porcaro C, Medaglia M T and Krott A 2015 Removing speech artifacts from electroencephalographic recordings during overt picture naming *NeuroImage* **105** 171–80
- [27] Delorme A and Makeig S 2004 EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis *J. Neurosci. Methods* **134** 9–21
- [28] Xu B, Zhang L, Song A, Wu C, Li W, Zhang D, Xu G, Li H and Zeng H 2018 Wavelet transform time-frequency image and convolutional network-based motor imagery EEG classification *IEEE Access* **7** 6084–93
- [29] Gaur P, Gupta H, Chowdhury A, McCreadie K, Pachori R B and Wang H 2021 A sliding window common spatial pattern for enhancing motor imagery classification in EEG-BCI *IEEE Trans. Instrum. Meas.* **70** 4002709
- [30] Tang X, Li W, Li X, Ma W and Dang X 2020 Motor imagery EEG recognition based on conditional optimization empirical mode decomposition and multi-scale convolutional neural network *Expert Syst. Appl.* **149** 113285
- [31] Porcaro C, Coppola G, Di Lorenzo G, Zappasodi F, Siracusano A, Pierelli F, Rossini P M, Tecchio F and Seri S 2009 Hand somatosensory subcortical and cortical sources assessed by functional source separation: an eeg study *Hum. Brain Mapp.* **30** 660–74
- [32] Porcaro C and Tecchio F 2014 Semi-blind functional source separation algorithm from non-invasive electrophysiology to neuroimaging *Blind Source Separation* (Berlin: Springer) pp 521–51
- [33] Vaid S, Singh P and Kaur C 2015 EEG signal analysis for BCI interface: a review *2015 Fifth Int. Conf. on Advanced Computing & Communication Technologies* (IEEE) pp 143–7
- [34] Goodfellow I, Bengio Y and Courville A 2016 *Deep Learning* (Cambridge, MA: MIT Press)
- [35] Dai G, Zhou J, Huang J and Wang N 2020 HS-CNN: a CNN with hybrid convolution scale for EEG motor imagery classification *J. Neural Eng.* **17** 016025
- [36] Sun B, Zhao X, Zhang H, Bai R and Li T 2020 EEG motor imagery classification with sparse spatiotemporal decomposition and deep learning *IEEE Trans. Automat. Sci. Eng.* **18** 541–51
- [37] Alzubaidi L et al 2021 Review of deep learning: concepts, CNN architectures, challenges, applications, future directions *J. Big Data* **8** 53
- [38] Raurale S A, Boylan G B, Mathieson S R, Marnane W P, Lightbody G and O'Toole J M 2021 Grading hypoxic-ischemic encephalopathy in neonatal EEG with convolutional neural networks and quadratic time-frequency distributions *J. Neural Eng.* **18** 046007
- [39] Abbasi H, Gunn A J, Unsworth C P and Bennett L 2021 Advanced deep learning spectroscopy of scalogram infused CNN classifiers for robust identification of post-hypoxic epileptiform EEG spikes *Adv. Intell. Syst.* **3** 2000198
- [40] Schirrmeister R T, Springenberg J T, Fiederer L D J, Glasstetter M, Eggensperger K, Tangermann M, Hutter F, Burgard W and Ball T 2017 Deep learning with convolutional neural networks for EEG decoding and visualization *Hum. Brain Mapp.* **38** 5391–420
- [41] Craik A, He Y and Contreras-Vidal J L 2019 Deep learning for electroencephalogram (EEG) classification tasks: a review *J. Neural Eng.* **16** 031001
- [42] Dose H, Moller J S, Iversen H K and Puthusserpady S 2018 An end-to-end deep learning approach to MI-EEG signal classification for BCIs *Expert Syst. Appl.* **114** 532–42
- [43] Lun X, Yu Z, Chen T, Wang F and Hou Y 2020 A simplified CNN classification method for MI-EEG via the electrode pairs signals *Front. Hum. Neurosci.* **14** 338
- [44] Chawla N V, Bowyer K W, Hall L O and Kegelmeyer W P 2002 Smote: synthetic minority over-sampling technique *J. Artif. Intell. Res.* **16** 321–57
- [45] Schalk G, McFarland D J, Hinterberger T, Birbaumer N and Wolpaw J R 2004 Bci2000: a general-purpose brain-computer interface (BCI) system *IEEE Trans. Biomed. Eng.* **51** 1034–43
- [46] Abadi M et al 2016 Tensorflow: large-scale machine learning on heterogeneous distributed systems (arXiv:[1603.04467](https://arxiv.org/abs/1603.04467))
- [47] Hétu S, Grégoire M, Saimpont A, Coll M P, Eugène F, Michon P E and Jackson P L 2013 The neural network of motor imagery: an ALE meta-analysis *Neurosci. Biobehav. Rev.* **37** 930–49
- [48] Hermosilla D M, Codorníu R T, Baracaldo R L, Zamora R S, Rodriguez D D, Albuerne Y L and Alvarez J R N 2021 Shallow convolutional network excel for classifying motor imagery EEG in BCI applications *IEEE Access* **9** 98275–86
- [49] Kiranyaz S, Avci O, Abdeljaber O, Ince T, Gabbouj M and Inman D J 2021 1D convolutional neural networks and applications: a survey *Mech. Syst. Signal Process.* **151** 107398
- [50] Ioffe S and Szegedy C 2015 Batch normalization: accelerating deep network training by reducing internal covariate shift *Int. Conf. on Machine Learning (PMLR)* pp 448–56
- [51] Park S and Kwak N 2016 Analysis on the dropout effect in convolutional neural networks *Asian Conf. on Computer Vision* (Springer) pp 189–204
- [52] Kingma D P and Ba J 2014 Adam: A method for stochastic optimization (arXiv:[1412.6980](https://arxiv.org/abs/1412.6980))
- [53] Prechelt L 1998 Early stopping-but when? *Neural Networks: Tricks of the Trade* (Berlin: Springer) pp 55–69
- [54] Siskind J M and Pearlmutter B A 2018 Divide-and-conquer checkpointing for arbitrary programs with no user annotation *Optim. Methods Software* **33** 1288–330
- [55] Pan S J and Yang Q 2009 A survey on transfer learning *IEEE Trans. Knowl. Data Eng.* **22** 1345–59
- [56] Karácsony T, Hansen J P, Iversen H K and Puthusserpady S 2019 Brain computer interface for neuro-rehabilitation with deep learning classification and virtual reality feedback *10th Augmented Human International Conf. 2019* pp 1–8
- [57] Zhang K, Xu G, Han Z, Ma K, Zheng X, Chen L, Duan N and Zhang S 2020 Data augmentation for motor imagery signal classification based on a hybrid neural network *Sensors* **20** 4485
- [58] Freer D and Yang G Z 2020 Data augmentation for self-paced motor imagery classification with C-LSTM *J. Neural Eng.* **17** 016041
- [59] Zhang Z, Duan F, Sole-Casals J, Dinares-Ferran J, Cichocki A, Yang Z and Sun Z 2019 A novel deep learning approach with data augmentation to classify motor imagery signals *IEEE Access* **7** 15945–54
- [60] Zhang Y D, Zhang Y, Phillips P, Dong Z and Wang S 2017 Synthetic minority oversampling technique and fractal dimension for identifying multiple sclerosis *Fractals* **25** 1740010
- [61] Zhang Y D, Zhao G, Sun J, Wu X, Wang Z H, Liu H M, Govindaraj V V, Zhan T and Li J 2018 Smart pathological

- brain detection by synthetic minority oversampling technique, extreme learning machine and Jaya algorithm *Multimedia Tools Appl.* **77** 22629–48
- [62] Wan Z, Yang R, Huang M, Zeng N and Liu X 2021 A review on transfer learning in EEG signal analysis *Neurocomputing* **421** 1–14
- [63] Kwon O Y, Lee M H, Guan C and Lee S W 2019 Subject-independent brain–computer interfaces based on deep convolutional neural networks *IEEE Trans. Neural Netw. Learn. Syst.* **31** 3839–52
- [64] Xu L, Xu M, Ke Y, An X, Liu S and Ming D 2020 Cross-dataset variability problem in EEG decoding with deep learning *Front. Hum. Neurosci.* **14** 103
- [65] Zhang K, Robinson N, Lee S W and Guan C 2021 Adaptive transfer learning for EEG motor imagery classification with deep convolutional neural network *Neural Netw.* **136** 1–10