International Journal of Innovative
Computing, Information and Control
Volume **18**, Number **5**, October **2022**

 ISSN 1349-4198
pp. 1587–1600

IMPROVING THE CONVOLUTIONAL NEURAL NETWORK PERFORMANCE THROUGH TRANSFER LEARNING FOR BRAIN-MACHINE INTERFACE SYSTEMS

Eneo Petoku[1], Ryota Takahashi[1] and Genci Capi[2]

[1]Graduate School of Science and Engineering
[2]Department of Mechanical Engineering
Hosei University
3-7-2 Kajino-cho, Koganei, Tokyo 184-8584, Japan
{ eneo.petoku.7j; ryota.takahashi.2p }@stu.hosei.ac.jp; capi@hosei.ac.jp

Received April 2022; revised July 2022

Abstract. *In the recent years, deep learning has been widely implemented for robotics applications. However, a main issue that remains to be solved especially for intelligent robot implementations is the limited number of training data. In this paper, we propose a transfer learning-based method to overcome this issue. To verify the performance of the proposed algorithm, we implemented the transfer learning in Convolution Neural Networks (CNNs) that maps the human brain signals into motor movements and its impact on window size is studied. The focus of this work is to investigate the effect of transfer learning in CNNs for subjects performing similar Brain Machine Interface (BMI) tasks. The results are promising in terms of improving the recognition rate of CNNs. The trained CNNs are also implemented to map in real time the brain signals to the humanoid robot arm motion.*

Keywords: Brain Machine Interface (BMI), EEG, Motor execution, Convolutional neural networks, Transfer learning, Classification

1. **Introduction.** The goal of Brain Machine Interface (BMI) systems is to enable humans to interact with computers or machines by using the brain activity [1,2]. BMI systems capture the user brain activity and translate it to a message or command for certain interactive applications. Systems that make use of BMIs paradigms permit the disabled and elderly people to control wheelchairs, home appliances and robots. Another application is to write sentences and move the cursor on the screen by using brain signals, playing video games, creating arts, etc. Brain-computer interfaces are also researched on stroke rehabilitation, and real-time health monitoring [3].

In the recent years, the research literature on BMI systems shifted to implementing deep learning models to map the brain signals into the desired command [1]. In addition, deep learning has shown good performance in robotics applications [4-6]. Eliminating one or more of intermediate processing steps, such as preprocessing, feature extraction and classification is one of the advantages of applying deep learning in BMI/BCI systems. There are two ways to implement CNNs on BMI/BCI systems. In the first method, the raw EEG (electroencephalography) data are directly fed into the CNN [7,8]. The second implementation is to use the CNN only as a classifier, and employ other algorithms for the feature extraction, such as Short-Time Fourier Transform (STFT) [9], or Common Spatial Patterns (CSP) [10] and more advanced versions of it, like Filter Bank Common Spatial Pattern (FBCSP) [11].

DOI: 10.24507/ijicic.18.05.1587

Transfer learning has been used in non deep learning classification methods for motor imagery. In [12] an integration of kernel common spartial patterns and transfer kernel learning for motor imagery classification was proposed. Transfer kernel learning is first used to compute a domain-invariant kernel, and then the kernel common special patterns to find the components with the largest energy difference. Other examples of transfer leaning in non deep learning approach include [13-16]. Transfer learning in deep learning architectures has recently been studied in several studies. In [17] transfer learning based on convolutional neural networks was implemented on the BCI competition IV dataset 2b. The Kappa coefficient was increased by 0.06 due to transfer learning. Transfer learning has also been used on P300 system [18]. In [19] a time frequency spectrogram was fed on the deep learning architecture.

Motor imagery and motor execution are two of the methods that are used to map the EEG signals into external robotic or computer application. Motor imagery is a dynamic state during which the subjects imagine the performing of an action [20]. In motor execution applications, the subject performs actual movements (for example, moving arms and legs), or other daily life activities. The motor cortex is the part of the brain where the brain signals for limb motions originate [2]. Several studies have compared motor imagery with motor execution [21]. In both paradigms, the brain signals have patterns which can be used for classification.

Attempts to associate the movements of the limbs to brain signals began in 1960s by studying the cerebral cortex of the monkeys [22]. It was shown that certain tasks, such as arm and wrist movements, caused a particular change on the brain signals of the monkeys. Classification of human wrist movements was studies in [23]. In [24], it was shown that it is possible to calculate the direction of the arm movement of monkeys in a three-dimensional environment by using a mathematical formulation of EEG data. Similar studies involving humans have also been published. For example, in [25,26] a robotic arm was controlled by brain signals generated by microelectrodes inserted in the motor cortex area corresponding to hand movements. These studies on animals and humas, clearly show a correlation between the brain activity and arm movements.

Previous research works are focused on mapping the brain signals to a particular hand movement. Three main hand movements categories are movements that do not involve any object (intransitives); movements in which the action is directed towards the use of a single object (transitives); and movements in which an intermediate object is used to move the final object (tool-mediated) [27-30]. Deep learning has also been utilized to classify arm movements [31,32]. In [33], authors study the decoding of Activities of Daily Life (ADL), through a BCI (Brain-Computer Interface) system. The study showed the feasibility of using a task-based approach to control an external robotic application. In [34], the authors investigated the use of delta oscillations for decoding directional information of a single limb joint.

In difference from previous works [35-38], we investigate the impact of CNN architecture based transfer learning on different window sizes. This is done in order to investigate the possibility of shortening the latency to improve the performance of BMI systems. In BMI implementations, the dataset is small. Therefore, it will be interesting to investigate the possibility to utilize transfer learning to train faster new CNNs for BMI implementations. In this work, we trained CNNs and implemented transfer learning for arm motion execution tasks. In addition, the trained CNNs are utilized to map in real time the users brain signals into the humanoid robot arm motion.

The rest of the paper is organized as follows. Section 2 discusses the proposed transfer learning method. We explain the structure of CNN, the transfer learning paradigm, and the method of data augmentation. In Section 3, we describe the device used for recording,

methodology and other technical aspects in extracting the EEG data. Section 4 presents the results of the CNNs and the real robot implementation. Section 5 concludes the paper and gives the future work directions.

2. **Proposed Method.** The CNN architecture for the transfer learning implementation is shown in Figure 1. The convolutional neural network is used both as feature extractor and classifier. The basic idea is to initially train a CNN1 for a specific task, and then, to train a new CNN2 for another similar task by utilizing what has been learned in CNN1.

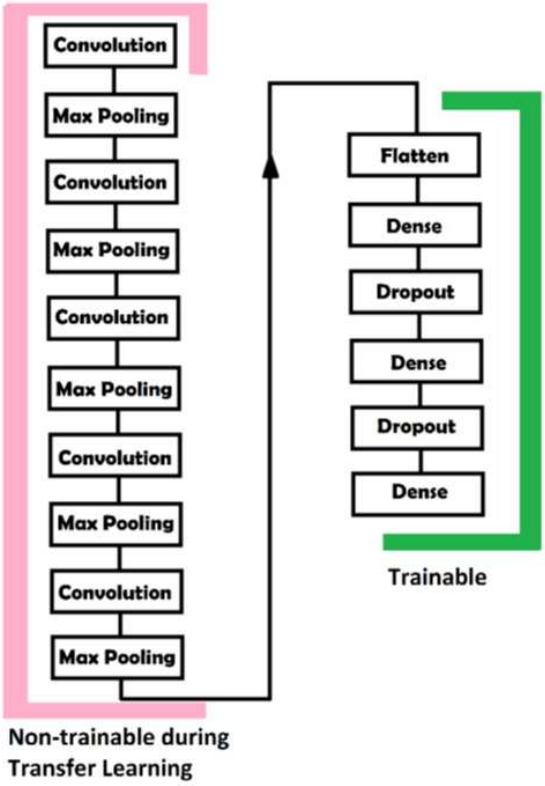


FIGURE 1. CNN architecture: (left) the non-trainable layers, (right) trainable layers

In our implementation, the architectures of CNN1 and CNN2 (Figure 1) are the same. They consist of two parts: the convolutional part (left part of Figure 1), and the dense part (right part of Figure 1). In the case of CNN1, both parts are trainable. When the transfer learning is applied on the CNN2, the right part of the neural network, which consists of successive convolution and max pooling layers, is set as non-trainable. The reason that left part is non-trainable is because the convolutional part of the network, can find edges on data, which correspond to certain brain activities. The goal is to retain the calculating method of these patterns, to utilize what has been learnt on the other hand. On the other hand, the right part remains trainable.

In [39], it is shown a correlation between the brain signals of motor cortex and the limb motions. Through the transfer learning, we can further investigate this issue. Therefore, to verify the performance, we considered implementation of CNNs for classifying the brain signals to their respective arm motion classes. We train the CNN1 for the right arm motion (task 1) and then implement transfer learning to train the CNN2 for left arm motion. The reverse is also done, by implementing transfer learning of the right arm utilizing what has been learnt for the left arm. Through transfer learning, we want to improve the classification accuracy of the CNN2.

In our implementation, the CNNs map the EEG signals to three classes corresponding to three arm motions. The recorded EEG data are augmented, by the method shown in Figure 2. The EEG data have two dimensions that correspond to the number of electrodes and the data points. To augment the data a window is applied. The window includes the data from all the electrodes, while it moves in the time frame of the data. Each time, the window moves a certain number of data points forward, determined by the hop size. The movement of the window stops when the end of time frame is reached.

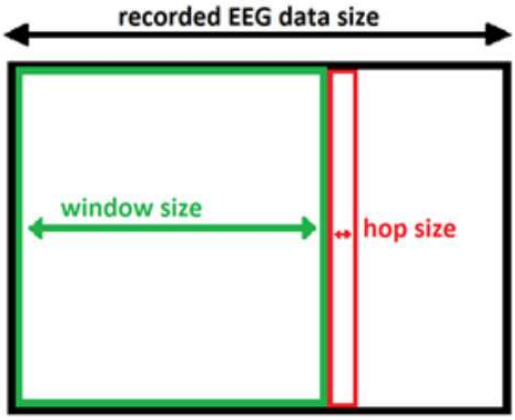


FIGURE 2. Data augmentation method

Therefore, a larger window size results in a small number of window movements for a certain size of the hop. The reason to implement this augmentation method is because we are interested in real-time control of the humanoid robot arm using the brain signals. Therefore, we must determine an optimal time frame to be able to reduce the response time of the robot motion. Knowing that the accuracy gets lower as the window size gets smaller, it is important to find an optimal solution for this problem. The number of window movements is calculated as follows:

$$\text{Number of window movements} = \frac{\text{EEG data size} - \text{Window size}}{\text{Hop size}} \tag{1}$$

where EEG data size is the non-augmented recorded data frame, window size is the number of selected datapoints to be augmented, and hop size is the number of datapoints that window jumps each time. If the result of Equation (1) is not an integer, it is rounded to the nearest smaller integer, so the window does not exceed the EEG data frame. The EEG data frames for each arm are fed to the CNN for classification and the accuracy, the percentage that the CNN makes the right prediction over the total number of predictions, is calculated for both limbs.

The proposed method consists on investigating the effect of window size and hop size on the accuracy of transfer learning. The reason is that in brain-machine interface systems these parameters are strongly related to the time delay between the brain signals and the robot response. Window size corresponds to latency of commands since it gives the time length to record brain signals and the hop size is the density of these data frames that are used as input of the deep learning architecture.

3. **Data Acquisition.** To record the user's brain activity, we use a Mitsar-EEG-201 system [40], shown in Figure 3. An electro-cap connected to the device is used to record the EEG signals. The recording frequency of the device is 500 Hz. We record the brain activity using seven electrodes (F3, F4, C3, C4, Cz, P3, P4), shown in different colors in Figure 4. In our implementation, we measured the difference in voltage between the

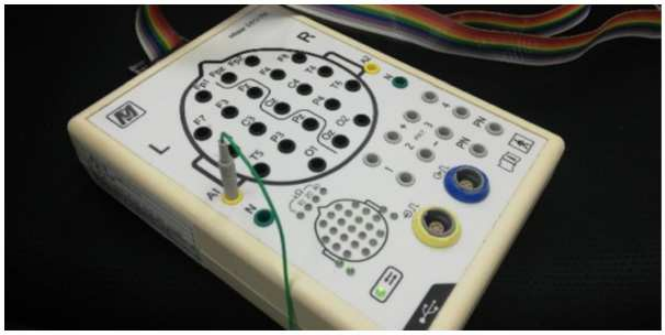


FIGURE 3. Mitsar-EEG-201 system

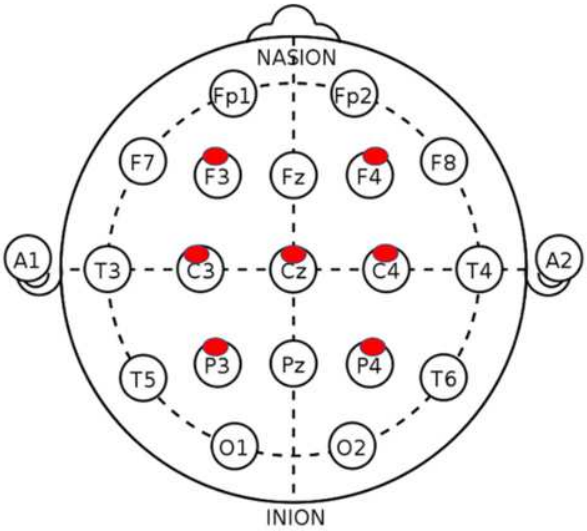


FIGURE 4. Transverse configuration of electrodes

selected and reference electrodes (A1 and A2). The recording time for each arm motion is 3 seconds. Therefore, a matrix of 1500×7 data points is generated for each trial. The high pass filter is applied at 0.16 Hz, and the low pass filter is applied at 70 Hz. A notch filter is also added at $50+/-5$ Hz to minimize the electric supply disturbances. During the experiments, the subject sits in front of the monitor showing a moving bar that correspond to the arm moving direction (Figure 5). The moving bar directions are randomly generated. The preparation time before starting the motion lasts 3 seconds (green color moving bar).

The duration of the arm motion also lasts 3 seconds, which is shown in the monitor by the red color moving bar. The arm moving directions are considered as two classes. The data collected during the motion preparation are classified as another class called "no motion". The recorded data of the brain activity of each class, is assumed to have some features or patterns that make it distinguishable from each other.

Common Average Reference (CAR) [31], is used to diminish any noise generated by user's face or arm movements. The common average reference is given as

$$X_{\mathrm{CAR},i}(t) = X_i(t) - \frac{1}{N}\sum_{k=1}^{N} X_k(t) \tag{2}$$

where i is the time frame, N is the number of electrodes, and $X(t)$ is the electrode value for the t time step. Each value of electrode is subtracted by the average of all electrodes

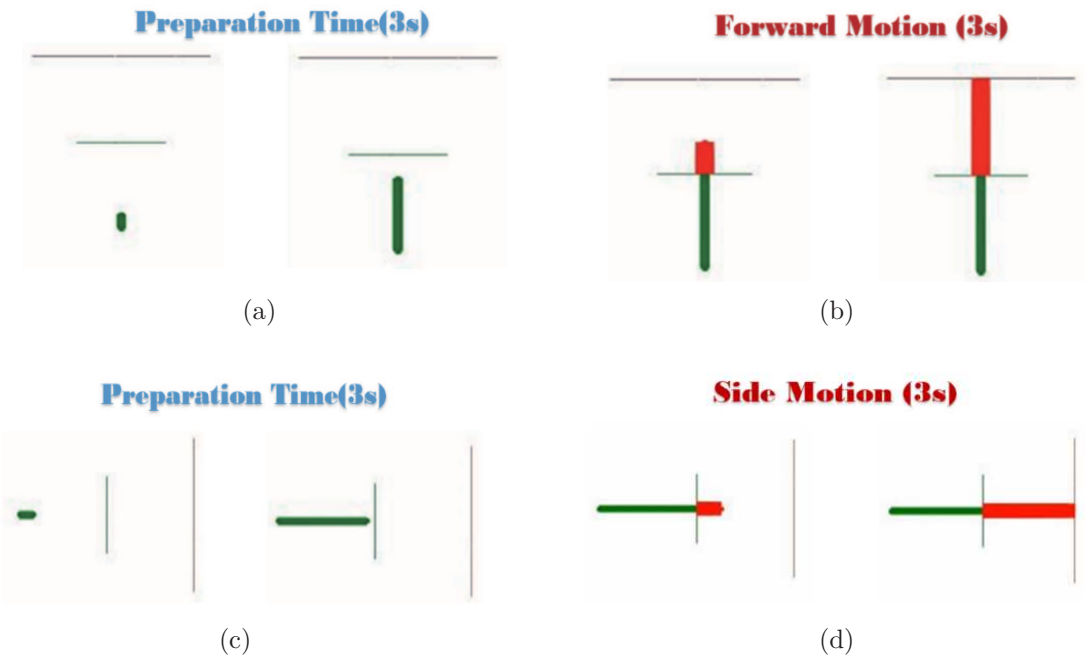


FIGURE 5. (color online) Screenshots of the animation played in the monitor: (a) Forward motion preparation time; (b) forward motion; (c) side motion preparation time; (d) side motion

FIGURE 6. Video capture during data collection experiments: (a) No motion; (b) left arm forward motion; (c) left arm side motion

on that time step. In this way, noises are significantly reduced. During the training phase the aim is to learn the features in the brain signals for each class.

Four healthy male subjects, aged 20 to 50 years old, participated in the experiments. They all were right-handed, and without any hearing or vision abnormalities. For each subject, five recording sessions are held in two different days at Assistive Robotics Laboratory of Hosei University. Figure 6 shows the video capture during experiments for data collection. In each session, 16 trials were recorded, 8 for each class (forward and side motions). A third class is generated by randomly picking 8 trials from no-motion data. Therefore, in total there are 120 trials for each subject, or 480 trials for all subjects.

Table 1. CCN parameters

Layer	Filters	Size	Parameters	Output
Conv2D	16	(3, 1)	64	(1398, 7, 16)
MaxPooling2D	–	(2, 1)	0	(699, 7, 16)
Conv2D	64	(3, 1)	3136	(697, 7, 64)
MaxPooling2D	–	(2, 1)	0	(348, 7, 64)
Conv2D	64	(3, 1)	12352	(346, 7, 64)
MaxPooling2D	–	(2, 1)	0	(173, 7, 64)
Conv2D	128	(3, 1)	24704	(171, 7, 128)
MaxPooling2D	–	(2, 1)	0	(85, 7, 128)
Conv2D	128	(3, 1)	49280	(83, 7, 128)
MaxPooling2D	–	(2, 1)	0	(41, 7, 128)
Flatten	–	–	0	(36736)
Dense	–	512	18808832	(512)
Dropout	–	0.3	0	(512)
Dense	–	512	262144	(512)
Dropout	–	0.3	0	(512)
Dense	–	3	1536	(3)

* in color non-trainable layers during transfer learning

4. **Results and Discussion.** The CNN parameters for the window size 1400 are shown in Table 1. As explained in Section 2, the first layers of the architecture consist of convolutional layers to find edges on the data that can be utilized by transfer learning. We tested the CNN performance with different parameters. The parameters shown in Table 1 gave the best performance. The convolutional layers have a kernel size of $(3, 1)$, while the dense layers consist of 512 neurons. We implement the CNN on Tensorflow 2.8 using the Keras API. The graphic card used for training is Nvidia Geforce RTX 3070 Super.

First, we trained the CNN for the right and left arm motion classification without implementing transfer learning. The trained weights are utilized in the next step for transfer learning. The dataset is split randomly into 60% as training data, 20% as validation data, and the remaining 20% as test data. After the dataset is randomly splitted, the augmentation phase occurs for each of the subsets, training, validation and test data. The network is trained for 100 epochs using the Adam optimizer. The learning rate is set to 0.000001.

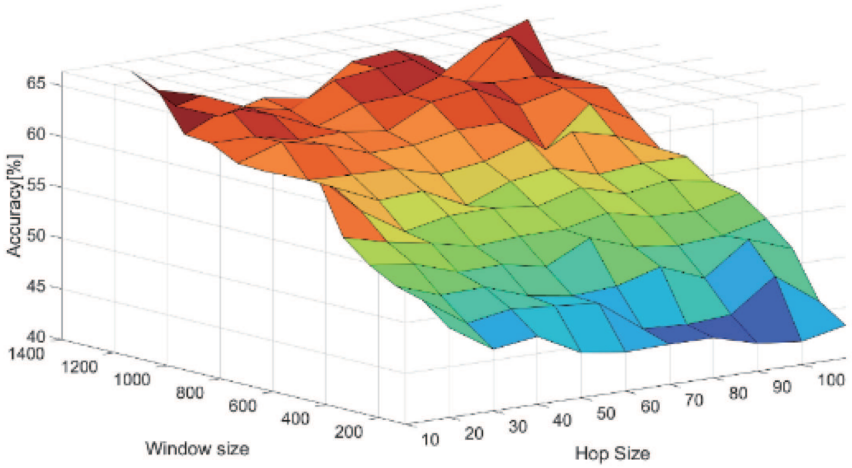
The window size varies from 200 to 1400 data points, each step increasing by 100 data points. The hop size varies from 10 to 100 data points, increasing by 10 data points each time.

A better way to illustrate the improvement in accuracy is by calculating the Kappa value as follows:

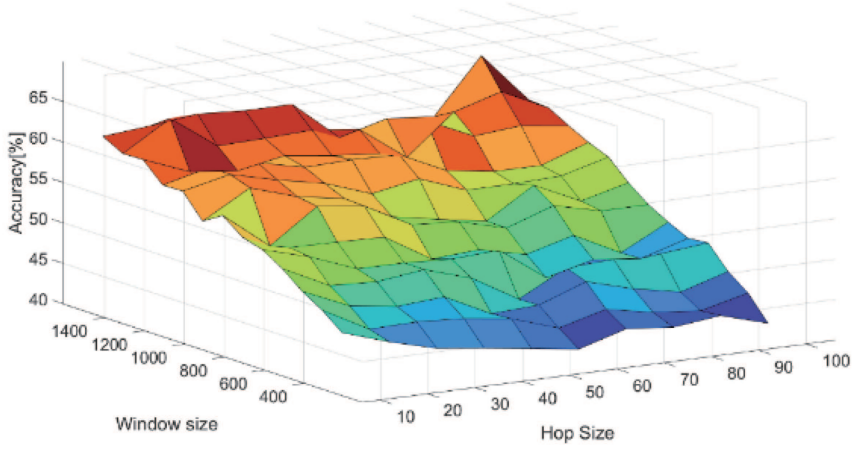
$$\text{Kappa} = \frac{\text{Po} - \text{Pe}}{1 - \text{Pe}} \tag{3}$$

where Po is the proportion of observed agreement, and Pe is the probability that agreement is due to chance [7,41]. Kappa relates the achieved accuracy with the random agreement rate. In our implementation, Pe is set to 0.33 since we are working on a 3-class system.

The results of the CNN for the right and left arm motion classification are shown in Figure 7. Initially, the holdout cross-validation results show that there is no overfitting. The graph shows that as the window size decreases, the accuracy also decreases. To better illustrate this, the data for all the hop sizes and both arms is averaged. The relationship between the window size and recognition rate is shown in Figure 8. This figure shows that as the window size gets larger, the recognition rate gets higher. The reason is that more



(a)



(b)

FIGURE 7. (color online) Accuracy of data according to window size and hop (for 4 subjects): (a) Left arm; (b) right arm

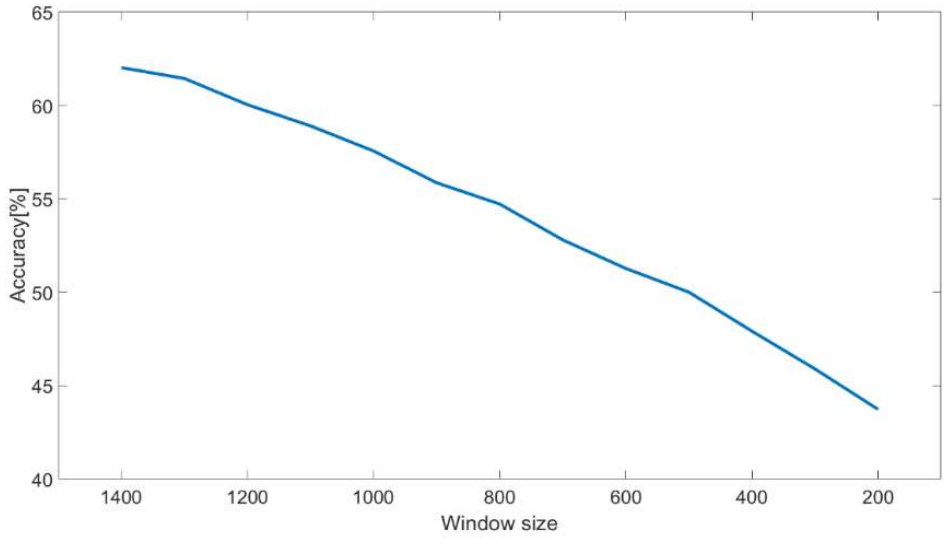


FIGURE 8. Graph of average accuracy of both arms as the window size decreases

TABLE 2. Accuracy and Kappa (Average for all hops)

Window size	Without transfer learning				With transfer learning			
	Accuracy right arm	Accuracy left arm	Kappa right arm	Kappa left arm	Accuracy right arm	Accuracy left arm	Kappa right arm	Kappa left arm
1400	60.83%	63.20%	0.4125	0.4481	61.18%	63.35%	0.4178	0.4503
1300	59.60%	63.27%	0.3941	0.4491	60.44%	63.58%	0.4067	0.4536
1200	58.06%	62.01%	0.3709	0.4301	59.76%	62.09%	0.3964	0.4313
1100	56.75%	61.04%	0.3513	0.4157	58.10%	61.96%	0.3714	0.4294
1000	55.56%	59.54%	0.3335	0.3932	56.08%	61.54%	0.3412	0.4230
900	53.62%	58.08%	0.3044	0.3712	55.24%	60.09%	0.3285	0.4013
800	52.71%	56.71%	0.2908	0.3507	54.14%	58.07%	0.3121	0.3710
700	50.21%	55.37%	0.2533	0.3306	51.41%	55.48%	0.2711	0.3321
600	49.24%	53.29%	0.2387	0.2994	50.18%	53.53%	0.2527	0.3030
500	48.51%	51.48%	0.2277	0.2723	49.42%	52.24%	0.2413	0.2836
400	46.27%	49.55%	0.1941	0.2433	47.34%	51.49%	0.2101	0.2723
300	44.62%	47.15%	0.1694	0.2073	44.93%	49.49%	0.1738	0.2423
200	42.82%	44.62%	0.1424	0.1693	42.58%	46.75%	0.1388	0.2011

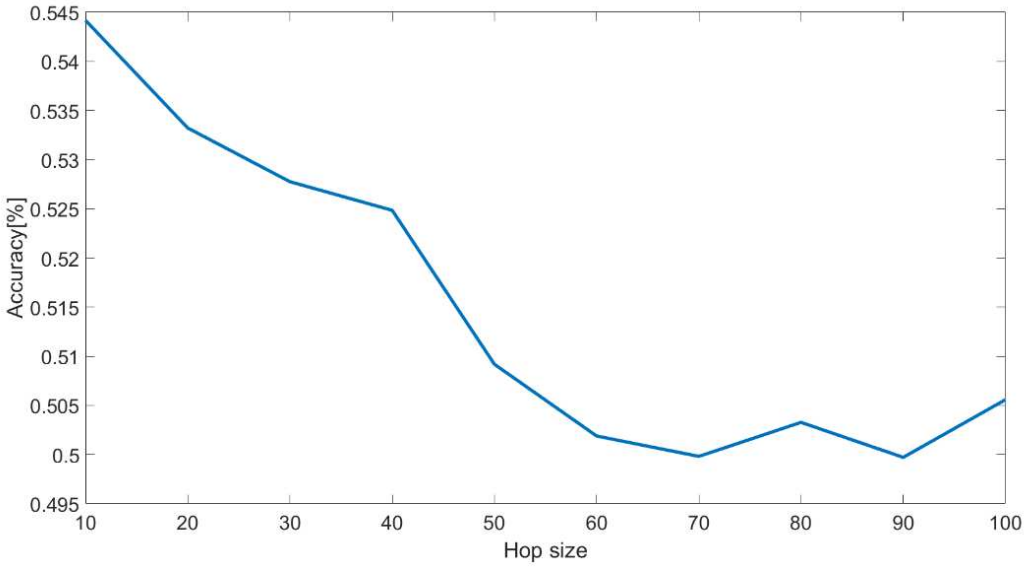


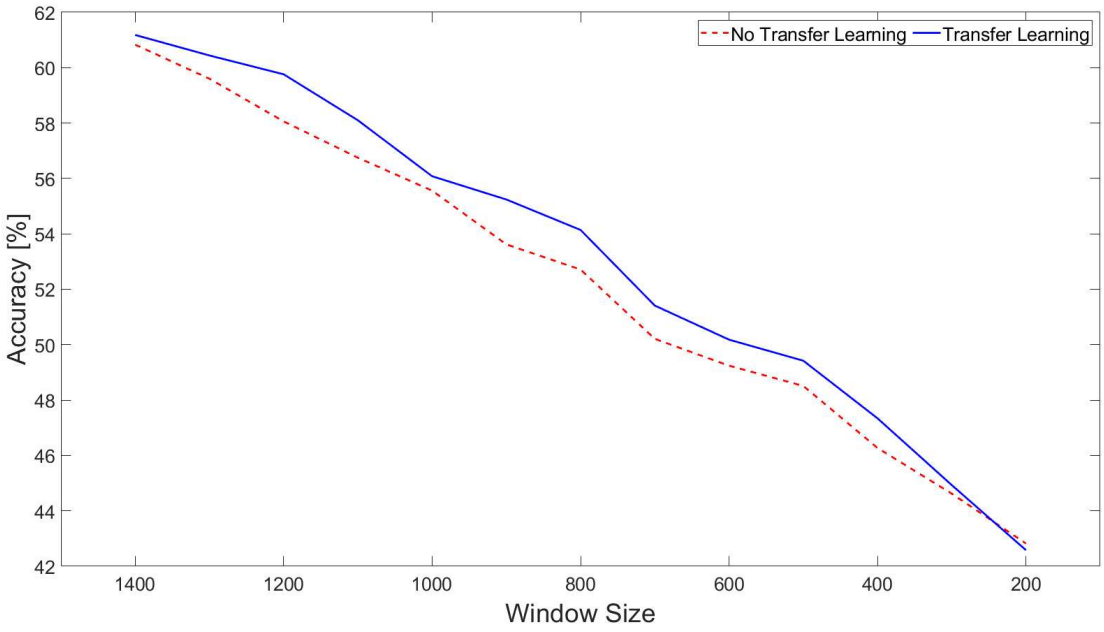
FIGURE 9. Graph of average accuracy of both arms as the hop size increases

information is fed to the CNN. For the largest value of window size (1400 data points), the averaged accuracies for all hop sizes are 60.83% and 63.20% for the right and left arms, respectively (Table 2).

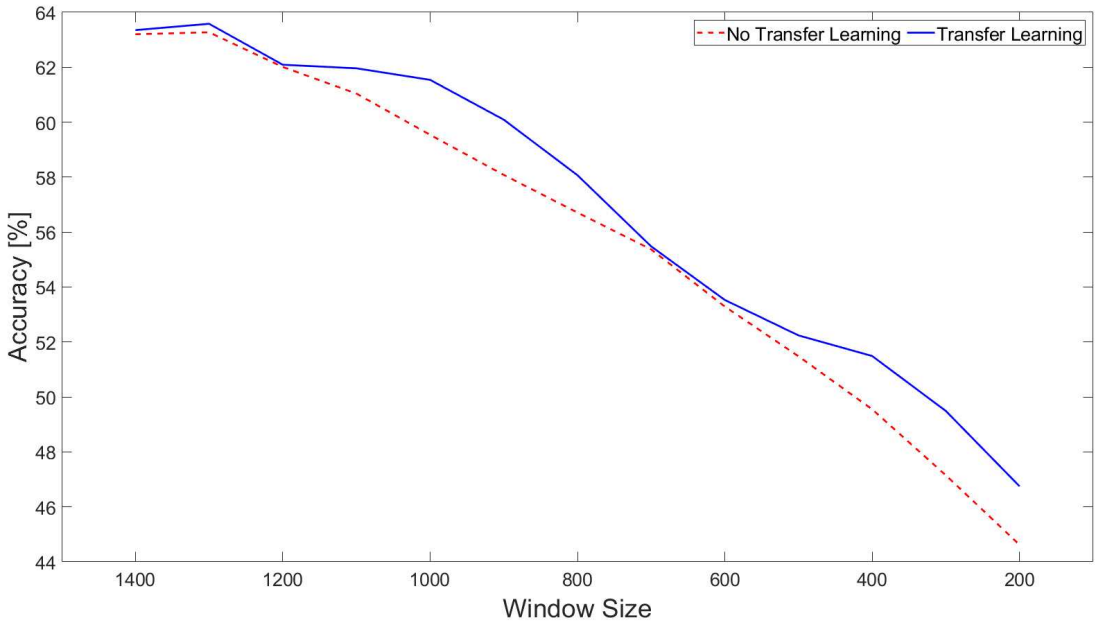
The standard deviation, on the maximum window size according to all hops was 2.29% and 1.89% for right and left arm, respectively. For the smallest window size (200 data points) the average accuracy drops to 42.82% and 44.62% for the right and left arms, respectively. The relationship between the hop size and recognition rate averaged for all window sizes, is shown in Figure 9. The results show that a small hop size leads to a higher accuracy. For the smallest hop size, the accuracy is 54.42%, while for the largest one it falls to 50.56%. In addition, we compared the performance of CNN with Support

Vector Machine (SVM) method. The results show that accuracy of CNN is over 10% higher than that of SVM.

The results for both transfer learning implementations averaged for all hop sizes are shown in Table 2. The averaged CNN accuracies with and without transfer learning are shown in Figure 10(a) and Figure 10(b) for the right and left, respectively. For the largest window size, the recognition rate of right arm increases from 60.83% to 61.18%. The best improvement is seen for the window size 1200 (an increase of 1.7%).



(a)



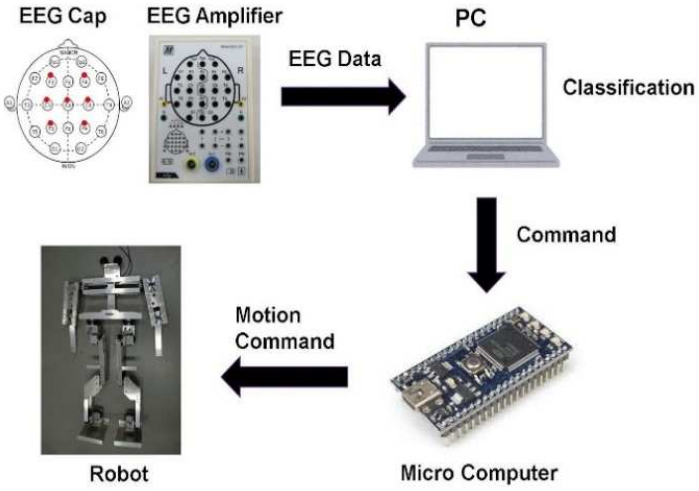
(b)

FIGURE 10. Relation between the window size and the recognition rate: (a) Right arm; (b) left arm

Similar results are seen in implementing transfer learning for the left arm motion. The biggest improvement is seen for window size 300, with an improvement of 2.34%. A 0.24% decrease of the recognition rate resulted in only one case (right arm, window size 200). In all other window sizes, the accuracies improved after transfer learning.

In our implementation, we achieved a Kappa value of 0.4125 in the case of right arm data and 0.4481 in case of left arm. When transfer learning is applied on the right arm data, the Kappa value increased to 0.4178, while for the left arm Kappa increased to 0.4503. The respective Kappa values of each accuracy are shown in Table 2. As in the case of accuracy rate, the Kappa value decreases as the window size gets smaller. Due to transfer learning, we get an improvement of the Kappa value in 25 out of 26 of the cases.

Figure 11(a) shows the system for mapping the brain signals to the humanoid robot arm motion. Figure 11(b) shows a video capture of the trained CNN mapping the brain sig-nals to the robot motion. The developed humanoid robot has 18 degree of freedom. As stated above there is a tradeoff between window size and robot command accuracy. A small window size leads to faster response of the robotic arm, but the accuracy is not optimal. As the window size gets bigger, the robotic arm responses with higher accuracy, but the motion takes longer to start.



(a)

(b)

FIGURE 11. Real robot implementation using brain signals: (a) Developed system; (b) video capture of real-time robot arm motion

5. **Conclusion.** In this paper, we proposed a transfer learning-based technique to improve the CNN accuracy. The transfer learning was implemented to train CNNs that predicts the arm motion based on the brain signals. The results showed that transfer learning improved the CNN recognition rates. In addition, the window size had a significant impact on the prediction accuracy of CNNs. As the window size decreased, the accuracy of the system also decreased. The hop size also showed a great effect on the CNN performance. A smaller hop size improved the recognition rate. We implemented the trained CNNs to control in real time the robot arm motion using the brain signals. The robot replicated the subject arm motion with a short response time.

Future research will include 1) Implementing transfer learning to train CNNs of new subjects based on the previously trained CNNs; 2) Investigating the effect of transfer learning on recognition rate, the effect and training time.

REFERENCES

[1] L. F. Nicolas-Alonso and J. Gomez-Gil, Brain computer interfaces, a review, *Sensors*, vol.12, no.2, pp.1211-1279, DOI: 10.3390/s120201211, 2012.

[2] C. S. Nam, A. Nijholt and F. Lotte, *Brain-Computer Interfaces Handbook: Technological and Theoretical Advances*, CRC Press, 2018.

[3] C. Guger, M. Tangermann and B. Z. Allison, Brain-computer interface research: A state-of-the-art summary 9, in *Brain-Computer Interface Research. Springer Briefs in Electrical and Computer Engineering*, C. Guger, B. Z. Allison and M. Tangermann (eds.), Cham, Springer, DOI: 10.1007/978-3-030-60460-8_1, 2021.

[4] D. Hossain and C. Capi, Genetic algorithm based deep learning parameters tuning for robot object recognition and grasping, *International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, vol.11, pp.591-595, 2017.

[5] D. Hossain and G. Capi, Multiobjective evolution for deep learning and its robotic applications, *2017 8th International Conference on Information, Intelligence, Systems & Applications (IISA)*, pp.1-6, DOI: 10.1109/IISA.2017.8316404, 2017.

[6] S. Nilwong, D. Hossain, S. I. Kaneko and G. Capi, Deep learning-based landmark detection for mobile robot outdoor localization, *Machines*, vol.7, no.2, pp.1-14, 2019.

[7] X. Zhao, H. Zhang, G. Zhu, F. You, S. Kuang and L. Sun, A multi-branch 3D convolutional neural network for EEG-based motor imagery classification, *IEEE Trans. Neural Systems and Rehabilitation Engineering*, vol.27, no.10, pp.2164-2177, DOI: 10.1109/TNSRE.2019.2938295, 2019.

[8] R. T. Schirrmeister, J. T. Springenberg, L. D. J. Fiederer et al., Deep learning with convolutional neural networks for brain mapping and decoding of movement-related information from the human EEG, *arXiv Preprint*, arXiv: 1703.05051, 2017.

[9] J. Zhang, C. Yan and X. Gong, Deep convolutional neural network for decoding motor imagery based brain computer interface, *2017 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, Xiamen, pp.1-5, DOI: 10.1109/ICSPCC.2017.8242581, 2017.

[10] S. Kumar, A. Sharma, K. Mamun and T. Tsunoda, A deep learning approach for motor imagery EEG signal classification, *2016 3rd Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE)*, Nadi, pp.34-39, DOI: 10.1109/APWC-on-CSE.2016.017, 2016.

[11] K. K. Ang, Z. Y. Chin, H. Zhang and C. Guan, Filter bank common spatial pattern (FBCSP) in brain-computer interface, *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pp.2390-2397, DOI: 10.1109/IJCNN.2008.4634130, 2008.

[12] A. Krizhevsky, I. Sutskever and G. E. Hinton, ImageNet classification with deep convolutional neural networks, *Proc. of Advances in Neural Information Processing Systems*, pp.1090-1098, 2012.

[13] V. Jayaram, M. Alamgir, Y. Altun, B. Scholkopf and M. G. Wentrup, Transfer learning in brain-computer interfaces, *IEEE Computational Intelligence Magazine*, vol.11, no.1, pp.20-31, 2016.

[14] A. M. Azab, L. Mihaylova, K. K. Ang and M. Arvaneh, Weighted transfer learning for improving motor imagery-based brain-computer interface, *IEEE Trans. Neural Systems and Rehabilitation Engineering*, vol.27, no.7, pp.1352-1359, 2019.

[15] I. Hossain, A. Khosravi, I. Hettiarachchi and S. Nahavandi, Multiclass informative instance transfer learning framework for motor imagery based brain-computer interface, *Computational Intelligence and Neuroscience*, vol.2018, 2018.
[16] P. Zanini, M. Congedo, C. Jutten, S. Said and Y. Berthoumieu, Transfer learning: A Riemannian geometry framework with applications to brain-computer interfaces, *IEEE Trans. Biomedical Engineering*, vol.65, no.5, pp.1107-1116, 2018.
[17] M. Parvan, A. R. Ghiasi, T. Y. Rezaii and A. Farzamnia, Transfer learning based motor imagery classification using convolutional neural networks, *2019 27th Iranian Conference on Electrical Engineering (ICEE)*, pp.1825-1828, DOI: 10.1109/IranianCEE.2019.8786636, 2019.
[18] O. Akinari, Convolutional neural network transfer learning applied to the affective auditory P300-based BCI, *Journal of Robotics and Mechatronics*, vol.32, no.4, pp.731-737, DOI: 10.20965/jrm.2020.p0731, 2020.
[19] S. Roy, A. Chowdhury, K. McCreadie and G. Prasad, Deep learning based inter-subject continuous decoding of motor imagery for practical brain-computer interfaces, *Frontiers in Neuroscience*, DOI: 10.3389/fnins.2020.00918, 2020.
[20] L. Li, J. Wang, G. Xu, M. Li and J. Xie, The study of object-oriented motor imagery based on EEG suppression, *PLoS ONE*, DOI: 10.1371/journal.pone.0144256, 2015.
[21] A. Batula, J. Mark, Y. Kim and H. Ayaz, Comparison of brain activation during motor imagery and motor movement using fNIRS, *Computational Intelligence and Neuroscience*, pp.1-12, DOI: 10.1155/2017/5491296, 2017.
[22] D.-H. Kim, H. G. Yeom, M. Kim, S. Kim, T.-W. Yang, O.-Y. Kwon and Y.-S. Kim, Introduction of brain computer interface to neurologists, *Annals of Clinical Neurophysiology*, vol.23, pp.92-98, DOI: 10.14253/acn.2021.23.2.92, 2021.
[23] G. Barrett, H. Shibasaki and R. Neshige, Cortical potentials preceding voluntary movement: Evidence for three periods of preparation in man, *Electroencephalography and Clinical Neurophysiology*, vol.63, pp.327-339, DOI: 10.1016/0013-4694(86)90017-9, 1986.
[24] A. P. Georgopoulos, R. E. Kettner and A. B. Schwartz, Primate motor cortex and free arm movements to visual targets in three-dimensional space, *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience*, vol.8, no.8, pp.2913-2927, 1988.
[25] L. R. Hochberg, D. Bacher, B. Jarosiewicz, N. Y. Masse, J. D. Simeral, J. Vogel et al., Reach and grasp by people with tetraplegia using a neurally controlled robotic arm, *Nature*, vol.485, pp.372-375, 2012.
[26] J. L. Collinger, B. Wodlinger, J. E. Downey, W. Wang, E. C. Tyler-Kabara, D. J. Weber et al., High-performance neuro-prosthetic control by an individual with tetraplegia, *Lancet*, vol.381, pp.557-564, 2013.
[27] V. Catrambone, G. Averta, M. Bianchi et al., Toward brain-heartcomputer interfaces: A study on the classification of upper-limb movements using multisystem directional estimates, *J. Neural. Eng.*, DOI: 10.1088/1741-2552/abe7b9, 2021.
[28] A. Bartolo et al., Cognitive approach to the assessment of limb apraxia clin, *Neuropsychol.*, vol.22, pp.27-45, 2008.
[29] B. Petreska et al., Apraxia: A review, *Prog. Brain. Res.*, vol.164, vol.61-83, 2007.
[30] L. Canzano et al., The representation of objects in apraxia: From action execution to error awareness, *Front. Hum. Neurosci.*, vol.10, DOI: 10.3389/duhum.2016.00039, 2016.
[31] S. Lim, D. P. Jang and H. Choi, Deep learning classification features visualization for arm movement brain-computer interface, *2020 8th International Winter Conference on Brain-Computer Interface (BCI)*, pp.1-3, DOI: 10.1109/BCI48061.2020.9061651, 2020.
[32] E. Quiles et al., Low-cost robotic guide based on a motor imagery brain-computer interface for arm assisted re-habilitation, *International Journal of Environmental Research and Public Health*, vol.17, DOI: 10.3390/ijerph17030699, 2020.
[33] S. Bandara, J. Arata and K. Kiguchi, A non-invasive BCI approach for predicting motion intention of ADL tasks for an upper-limb wearable robot, *International Journal of Advanced Robotic Systems*, DOI: 10.1177/1729881418767310, 2018.
[34] A. Korik, R. Sosnik, N. Siddique and D. Coyle, Decoding imagined 3D hand movement trajectories from EEG: Evidence to support the use of mu, beta, and low gamma oscillations, *Frontiers in Neuroscience*, DOI: 10.3389/fnins.2018.00130, 2018.
[35] A. Sakai, Y. Minoda and K. Morikawa, Data augmentation methodsfor machine-learning-based classification of bio-signals, *Proc. of the 10th Biomed. Eng. Int. Conf. (BMEiCON)*, pp.1-4, 2017.

[36] S. Prasomphan, Towards cultural heritage content retrieval by convolution neural network, *ICIC Express Letters*, vol.16, no.2, pp.137-144, DOI: 10.24507/icicel.16.02.137, 2022.
[37] M. Dai, D. Zheng, S. Liu and P. Zhang, Transfer kernel common spatial patterns for motor imagery brain-computer interface classification, *Computational and Mathematical Methods in Medicine*, vol.2018, 2018.
[38] M. R. V. Sanchez, S. Mishima, M. Fujiwara, G. Ai, M. Jouaiti, Y. Kobryn, S. Rimbert, L. Bougrain, P. Hénaff and H. Wagatsuma, Methodological design for integration of human EEG data with behavioral analyses into human-human/robot interactions in a real-world context, *ICIC Express Letters*, vol.14, no.7, pp.693-701, DOI: 10.24507/icicel.14.07.693, 2020.
[39] J. P. Gallivan, D. A. McLean, J. R. Flanagan and J. C. Culham, Where one hand meets the other: Limb-specific and action-dependent movement plans decoded from preparatory signals in single human frontoparietal brain areas, *Journal of Neuroscience*, vol.33, no.5, pp.1991-2008, DOI: 10.1523/JNEUROSCI.0541-12, 2013.
[40] *https://www.braininstitute.com.au/equipment*, Accessed on 20 February 2022.
[41] J. L. Fleiss and J. Cohen, The equivalence of weighted Kappa and the intraclass correlation coefficient as measures of reliability, *Educational and Psychological Measurement*, vol.33, no.3, pp.613-619, 1973.
[42] E. Petoku and G. Capi, Object movement motor imagery for EEG based BCI system using convolutional neural networks, *2021 9th International Winter Conference on Brain-Computer Interface (BCI)*, Gangwon, Korea, pp.1-5, DOI: 10.1109/BCI51272, 2021.

Author Biography

Eneo Petoku received the Bachelor of Science and Master of Science in Electronics Engineering degrees from Polytechnic University of Tirana. He is currently pursuing his Ph.D. degree at Graduate School of Science and Engineering, Hosei University. His research interests include brain-computer interfaces and artificial intelligence.

Ryota Takahashi received the Bachelor of Science and Engineering degree from Hosei University, Japan. He is currently a Master student at the Assistive Robotics Laboratory, Graduate School of Science and Engineering, Hosei University. His research interests include the BMI systems and deep learning.

Genci Capi received the B.E. degree from Polytechnic University of Tirana, in 1993 and the Ph.D. degree from Yamagata University, in 2002. He was a Researcher at the Department of Computational Neurobiology, ATR Institute from 2002 to 2004. In 2004, he joined at the Department of System Management, Fukuoka Institute of Technology, as an Assistant Professor, and in 2006, he was promoted to Associate Professor. In 2007, he joined the Department of Electrical and Electronic Systems Engineering, University of Toyama, Toyama, Japan. From 2016, he is a Professor at the Department of Mechanical Engineering, Hosei University, Tokyo, Japan. His research interests include intelligent robots, BMI, multi-robot systems, humanoid robots, learning, and evolution.