Combining Object Detection with EEG for Ubiquitous BMI

ALLAN FREDERICK* and KANISHKA MITRA*, The University of Texas at Austin

With respect to Internet-of-Things devices, object detection can be used to select a device that the user wants to interact with. Motor imagery EEG is a modality measuring electrical brain activity to detect a user's intent to perform a specific movement, such as moving the right arm or left leg. By combining object detection with motor imagery, users can select and control an IoT device using their thoughts. For example, a user could apply object detection to select a computer mouse and then use motor imagery to initiate a click without physically interacting with it. In this project, a deep learning object detection model is used to perform predictions in real-time, where it is then integrated with motor imagery classification to create a more useful, flexible, and ubiquitous brain-machine interface that opens new possibilities for BMI control of assistive devices.

Additional Key Words and Phrases: brain-machine interfaces, object detection, ubiquitous computing, motor imagery

ACM Reference Format:
Allan Frederick and Kanishka Mitra. 2023. Combining Object Detection with EEG for Ubiquitous BMI. 1, 1 (April 2023), 8 pages.

1 INTRODUCTION

Brain-machine interfaces (BMIs) are gaining increasing research interest as they promise to provide seamless interaction between the brain and the surrounding environment. This natural way of augmenting human capabilities opens up a multitude of applications ranging from the field of entertainment to clinical solutions for the aid of disabled patients. Especially of interest is the latter case in which BMIs can translate the patient's intentions into control commands for assistive devices like wheelchairs, spellers, and neuro-prosthetics [10]. The control of such BMIs requires a gateway to the patient's intentions, which could be established non-invasively by analyzing the electrical activity of the brain using Electroencephalography (EEG).

Motor imagery (MI) is a widely used BMI modality [1]. During MI, a motor action is imagined without actually executing it, evoking an event-related desynchronization (ERD) and consequent decrease of spectral power in the μ(8-13 Hz) and β(13-30 Hz) bands [6]. However, it has been shown that MI of the left and right hand can be captured as discrete events characterized by signature activity patterns, as the activity appears contralaterally. If a BMI can detect these discrete neural events, then it will be able to control external objects.

However, due to its non-invasiveness, EEG contains high noise and artifacts. Moreover, EEG is also a non-stationary signal and changes over time in ways that are extremely challenging to figure out due to the complexity of the brain. Therefore, the resolution is rather poor when trying to utilize EEG for more fine, detailed-oriented tasks, such as controlling specific objects over others. There is still a need for finer context in order for EEG to be truly useful in ubiquitous computing. If this can be overcome, then the potential for using EEG data is very promising in dynamic settings, especially for those who are physically incapacitated.

*Both authors contributed equally to this research.

Authors' address: Allan Frederick, allanfrederick1224@utexas.edu; Kanishka Mitra, mitrakanishka@utexas.edu, The University of Texas at Austin.

In order to provide a finer context in BMI control using EEG, we utilize an object detection system. Object recognition using visual data from a camera can provide a BMI system the context it needs for it to work in broader circumstances, where EEG alone is not enough. For example, although it may be possible to determine whether a person imagines moving their right or left arm, it is more difficult to differentiate between a person looking at a chair vs. a table vs. a lamp vs. another person [11]. Fortunately, deep learning and computer vision has advanced significantly, allowing for quick and efficient classification of simple objects; this is much easier than training a user to modulate their neural signals to differentiate between objects. In addition, incorporating object recognition with deep learning models significantly lowers the burden of the EEG classifier and the user who is modulating their neural signals to interface with the BMI. As a result, this provides room for more objects to be interacted with, increasing the usefulness of an assistive BMI. Specifically, the objective of this project is to combine object detection for device selection and EEG MI to interact with the selected object. There has been literature covering the application of using object detection in a BMI system, such as using smart glasses to identify objects and using EEG signals to interact with those objects [5]. However, this paper was published in 2016, and it did not make use of the most recent and cutting-edge deep learning technology for object detection or recent advancements in EEG classification. Moreover, this previous work's EEG hardware was limited to 14 saline-based channels on the Emotiv headset, hindering the quality of the signal. [5]. The approach in our project involves real-time deep learning classifications for object detection and upgraded EEG data quality using 32 channels on a clinical-grade amplifier.

2 METHODS

2.1 System Overview

Our system utilizes a deep learning object detection model to make real-time classifications of objects and a decoder to classify motor imagery intentions. These intentions, left or right, binarily control a device selected by the object detection model.

In the system pipeline illustrated by Fig. 1, the object is first detected with a Google Coral Camera connected to the Google Coral Dev board. The GStreamer Python framework was used to obtain camera data and feed it into the object detection model running on the board. The Google Coral Dev board runs the MobileNet SSD V2 object detection model in real-time. Example code was obtained by the google-coral repository on GitHub [8]. The Coral Dev board was chosen because it uses an ASIC specifically designed for running deep learning models in real-time, the Edge TPU. Out of the box, the model outputs a list of top predictions per inference made, and it does so extremely fast. There is too much information provided by the model that is being output too fast for it to be useful to control devices. In order for the system to be practical, it needs to be stable in its predictions. Therefore, our object detection Python script implements a prediction buffer, storing 250 predictions. This was done in order to obtain a "final prediction" based on evidence accumulation, providing a stable output of predictions in a timely manner.

We predefined a list of objects to be classified in order to use for device interaction: laptop/keyboard, computer mouse, remote, and person. Once a final detection is made by the model, there is a check to see if that prediction is in the defined list of devices to control. Once the model predicts an object that is in the list of devices to control, this information is transmitted to an ESP32 microcontroller over wireless UDP. The check to determine if the detected object is in the predefined list ensures that the ESP32 is receiving only the predictions we want.

After the ESP32 receives the proper object classification, it then waits to receive the appropriate motor imagery command, illustrated by system 2 in Fig 2. The motor imagery classifier runs on a separate laptop, providing a visual

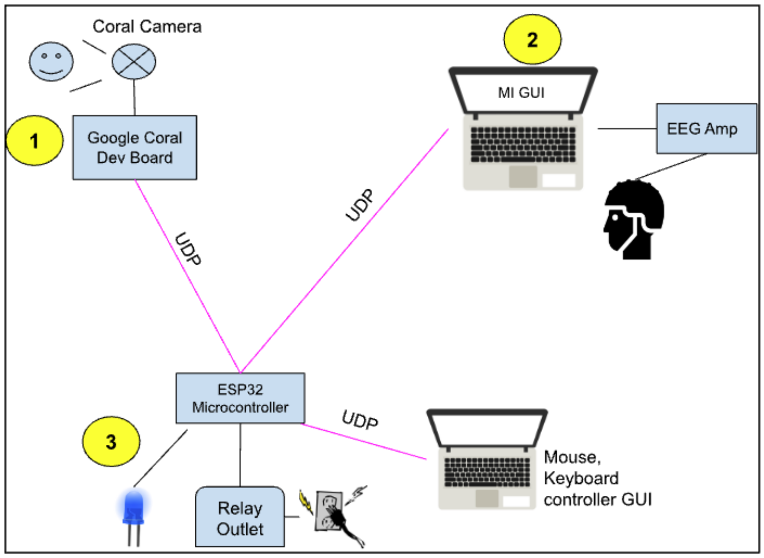

Fig. 1. Full system set-up and integration. System 1 combines the camera module and the Google Coral Dev board. System 2 is the EEG cap, amplifier, recording software, and visual interface for the subject. System 3 is the ESP32 microcontroller which controls various peripheral objects.

interface for the user (see below). A '0' or '1' is sent over UDP to the ESP32, depending on the output of the MI decoder. '1' initiates "on" or the action item, while '0' turns "off" or is the absence of an action item for the device.

The ESP32 microcontroller acts as a server, receiving information from both the Google Coral Dev board and the motor imagery classifier (on a laptop). The firmware on the microcontroller initially expects input from object detection. Once the object is successfully detected and sent to the microcontroller, a unique output is assigned to be implemented: a laptop/keyboard would enable typing "hello world" on the computer, the computer mouse would enable a left click on the computer, and a person would enable turning on an LED. The system was also extended to detect a remote control that enables activation of a digital pin that could be connected to an outlet relay to power anything plugged in. A simple GUI was developed for the purpose of displaying mouse and keyboard control. After a single instance of controlling the device, the ESP32 program returns to expecting an object detection input. This process is repeated indefinitely, providing a way for the user to select and control multiple objects, one at a time.

2.2 BMI Experimental Setup and Protocol

In order to capture stable and reliable left-hand and right-hand MI, we developed a visual interface to prompt and record the subject's motor intentions. The starting position involved the subject sitting and facing a computer screen which will prompt left or right-hand motor imagery for a set duration. Each trial started with a 3-second fixation period depicted by a white cross in the middle of the screen, which indicates to the user that a MI trial is about to begin. At the end of the fixation period, the visual interface will randomly indicate a left or right arrow which signals the type of MI. If a right arrow is shown, then the user should perform right-hand MI and vice-versa. There is also a continuous countdown bar that indicates the duration of time the subject has to hold MI (around 5 seconds), after which the bar clears, concluding one trial. The fixation period was introduced so that subjects could time the MI as precisely as possible based on visual feedback. Also, subjects were explicitly instructed to keep their arms relaxed so there were no

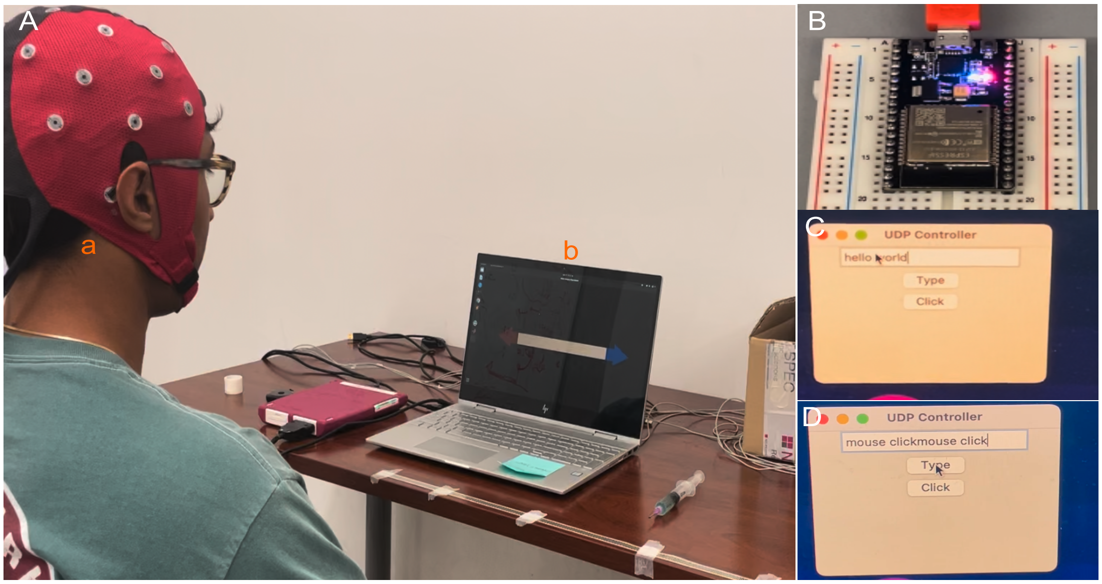

Fig. 2. Experimental setup and task paradigm pipeline. Panel A shows a representative subject performing MI (a. EEG Cap, b. visual interface giving real-time feedback), while panel B shows the LED, panel C depicts a UI where text is typed, and panel D is the UI where the mouse is clicked (see text for details).

collateral peripheral-level movements to impede or support their MI performance. This design allowed for comparing *i)* left-hand MI and right-hand MI, where features were extracted and classified between these periods. One measurement run consisted of 20 trials, and each participant completed four runs with ~2-3 minutes of rest in between. A recording session lasted for about 30 minutes. During the online session, we trained a decoder using the offline data and had the participant control their MI in real-time. The visual interface would prompt the user to either perform left or right-hand MI, and then the participant will attempt to control a bar (decoder's probability of the user's intention) till a certain threshold is achieved for a successful 'hit' (the controlled bar was able to reach the preset threshold). If the subject misses the threshold or hits the opposite intention, then that trial will be considered a 'miss.' If a 'hit' is achieved, then a '1' is sent to the ESP32; if the trial is a 'miss,' then a '0' will be sent via UDP. Each online run consisted of 20 trials, with equal amounts of left and right-hand MI.

2.3 Data Collection and Pre-processing

EEG data were collected with an eego system (ANT Neuro, The Netherlands) at a sampling rate of 512 Hz from 22 cortical locations according to the international 10-20 system, covering mostly the sensorimotor cortex (locations F7, F3, Fz, F4, F8, FC5, FC1, FC2, FC6, C3, Cz, C4, CP5, CP1, CP2, CP6, P7, P3, Pz, P4, P8, POz). The reference electrode was placed over CPz. Signals were first band-pass filtered using a 4^{th} order zero-phase Butterworth filter with cutoff frequencies of 0.1 to 45 Hz.EEG data were re-referenced to the common average reference.

For motor intent classification, 5-second left-hand MI and right-hand MI samples were obtained as the corresponding time periods (see Figure 2A). For the left vs. right-hand MI comparison, left and right-MI data was selected as the 5 seconds preceding their respective cue. From these 5-second-long epochs, spectral power was computed in a sliding window fashion, utilizing a window size of 0.5 seconds and a step size of 1/16 second in the 8-30 Hz range at a one Hz resolution. In each window, the power spectrum was estimated using Welch's method. This procedure yielded

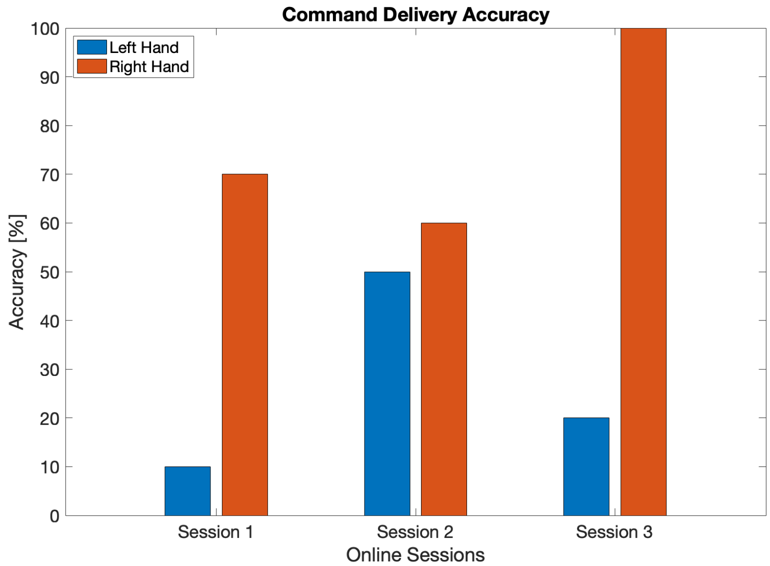

Fig. 3. Command delivery accuracy over three online sessions.

$22\ channels * 23\ frequencies = 506$ raw power features for each window. As each epoch provided 41 windows, this resulted in $41\ windows * 20\ trials * 4\ runs = 3,280$ samples per condition for each subject.

2.4 Classification Approach

Previous studies indicated that Riemannian geometry-based classifiers (RGBCs) could achieve superior performance in classifying MI-related neural patterns [2, 4]. In addition, RGBCs allow for online distribution matching of training and test samples, promoting robust and stable performance over multiple BMI sessions [7]. From these considerations, we decided to use a Riemannian geometry-based, minimum distance to the mean (MDM) classifier — which utilizes covariance matrices estimated from epoched EEG data as features — to detect the transitions from left to right-hand MI in an online pipeline. Decoders were trained on the same 5-second data segments, but the performance was also evaluated continuously and in real-time over the new online runs. We employed a LORO-CV (Leave-one-run-out cross-validation) scheme. In each run, we used only the top 10 features with the highest Fisher scores (as obtained only from training data) for training and evaluation [9]. Chance level was estimated as 52.7% at significance level $\alpha = 0.05$ assuming a binomial distribution of misclassifications as proposed by [3].

3 RESULTS

We primarily looked at online performance and its ability to control various objects. But we also observed the metrics to evaluate the offline decoder. Left-MI could be distinguished from right-MI with better-than-chance (> 52.7%) accuracy for all runs, with an average run accuracy of $67.8 \pm 5.6\%$. To evaluate the online performance, we looked at the command delivery accuracy or the number of times the subject was able to successfully reach the threshold of the prompted MI (hits over total trials). Fig. 3 depicts the delivery accuracies across three online runs. We can observe that the subject performed much better for right-hand MI compared to the left. This could imply two conclusions: (1) the decoder is

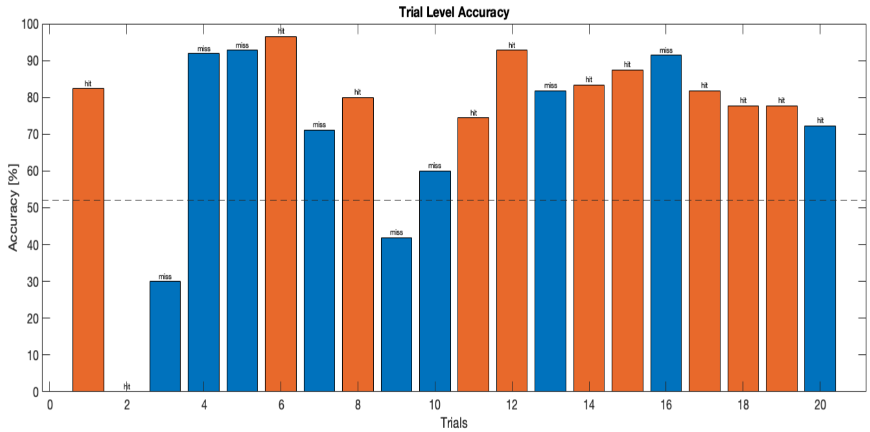

Fig. 4. Sample accuracies over 20 trials. Blue bars indicate a left-hand MI cue, and red indicates a right-hand MI cue. The black dashed line is the estimated chance level (52.7%). The top label indicates the results of the trial: hit or miss.

quite biased and classifies most features as right-MI, or (2) since the subject was right-handed, it was a lot easier for them to perform right-MI.

We also observed the accuracy on a trial-by-trial level depicted in Fig. 4, which indicated how well the decoder predicted the subject's motor intentions regardless if they were the correct ones. These values were computed by this equation: $SP = (P_{cueMI} - (0.95)P_{noncueMI})/0.05$. This metric can tell us the accuracy of our decoder per sample and how close the subject was to their target goal. Based on Fig. 4, we can see that the classifier performed more reliably and with more confidence on the right-hand MI. While decoding for left-hand MI, the classifier was equally confident for most trials, which can indicate that the subject tends to perform poorly on left MI rather than explaining the poor performance on the basis of the model. However, more subjects and training data will be required to reach a more conclusive result to truly evaluate the decoder's performance. With the integrated system, the subject was able to control all three objects: LED, mouse, and keyboard, with <3 second latency (demonstrations are shown in the presentation). This shows that fast and reliable BMI control is possible with object detection.

Regarding object detection, the MobileNet SSD V2 model performed as intended. The model was quite easily able to detect the necessary objects. We did not find it necessary to include any evaluation of the MobileNet SSD V2 model as this is beyond the focus of our project; the model is already well established, and we sought to apply it in our solution.

4 DISCUSSION

We learned how to implement a deep-learning object detection model in real-time and wirelessly communicate the information to a microcontroller. This pipeline is not only extremely useful in the space of BMIs but also for a wide variety of applications that involve real-time object detection. We also learned how to integrate different modalities into a system for control, which involved adjusting the temporal flow to synchronize object detection and MI. For instance, object detection needed to be output in a steady and consistent manner before MI is performed. MI needed to be performed only after an object detection was made and sent over to the ESP32 microcontroller. In this way, the appropriate UDP calls are sent to the microcontroller at the right time, ensuring proper functionality.

It is important to note that a critical factor in the performance of the system relies on the user's motor imagery performance, which varies from person to person. Upon initial design of the system, we decided to use right-MI to initiate the one desired action item of a device and left-MI to initiate another desired action item of the device. For example, we originally wanted right-MI to turn on an LED and left-MI to turn it off. Or right-MI to perform the right click of a mouse and left-MI to perform the left click of a mouse. After observing the online decoder performance, the challenge of performing both left-MI and right-MI with equal accuracy and consistency became apparent. Consequently, we decided to simplify the functionality where any correct MI delivers a single action item to the device, like activating the LED or initiating a left mouse click. Conversely, an incorrect classification or "miss" on the MI decoder would turn off the device (in the case of the mouse and keyboard, perform no action at all).

4.1 Contributions

The original elements of this work include using AI-specific hardware to run onboard deep learning models for real-time object detection for the purpose of creating an assistive BMI based on motor imagery. The ESP32 microcontroller firmware to receive inputs from the object detector and motor imagery and thus control devices is also original in the sense that we built the code from the ground up.

5 CONCLUSION AND FUTURE WORK

Here we proposed a novel interaction of a non-invasive BMI system with auxiliary objects detected by deep learning techniques. Both offline performance and online integration of MI produced promising performances, indicating that the approach might be viable for multiple object control. Our method thus could provide a useful tool for future assistive applications combining BMI technology and environmental control for ubiquitous computing.

5.1 Future Work

There are a few important limitations regarding object detection that we would like to improve upon. First, the object detection model itself is limited in which labels it was trained on. For example, labels do not exist for LEDs or lamps or for most devices that would be useful to interact with. This is why we initially relied on symbolic object control for interacting with certain devices (i.e., detecting a person to select LED and detecting a remote to select the outlet relay). Person and remote were labels the object detection model could easily identify. In the future, we would like to tune the model to be able to recognize custom labels such as LED or lamps. Fortunately, the flexibility to train on custom images does exist – we can apply transfer learning on the base model. Second, the physical camera connection to the board was limiting. The cable was only a couple of inches long, severely limiting the mobility of the camera. Mobility is extremely important for ubiquity and detecting objects throughout the environment. We'd like to implement the camera and Coral Dev board to be wearable. This could be accomplished by using a USB-connected camera mounted on the head with a longer cable. Further mobility could be accomplished by using the Google Coral Dev board micro, which has a much smaller form factor and can be worn more easily.

In regards to motor imagery, we would like to achieve a more balanced performance between left-MI and right-MI. This would require further subject training as a user learns how to modulate neural signals with practice.

REFERENCES

[1] Minkyu Ahn and Sung Chan Jun. 2015. Performance variation in motor imagery brain–computer interface: a brief review. *Journal of neuroscience methods* 243 (2015), 103–110.

[2] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten. 2010. Riemannian geometry applied to BCI classification. In *9th International Conference LVA/ICA*. 629–636.

[3] Etienne Combrisson and Karim Jerbi. 2015. Exceeding chance level by chance: The caveat of theoretical chance levels in brain signal classification and statistical assessment of decoding accuracy. *Journal of Neuroscience Methods* 250 (2015), 126–136.

[4] Marco Congedo, Alexandre Barachant, and Rajendra Bhatia. 2017. Riemannian geometry for EEG-based brain-computer interfaces; a primer and a review. *Brain-Computer Interfaces* 4, 3 (2017), 155–174.

[5] Eline De Buyser, E. Coninck, Bart Dhoedt, and Paul Simoens. 2016. Exploring the Potential of Combining Smart Glasses and Consumer-grade EEG/EMG Headsets for Controlling IoT Appliances in the Smart Home. 4 (6 .)–4 (6 .). https://doi.org/10.1049/ic.2016.0053

[6] Jean Decety. 1996. The neurophysiological basis of motor imagery. *Behavioural Brain Research* 77, 1 (1996), 45–52. https://doi.org/10.1016/0166-4328(95)00225-1

[7] Satyam Kumar, Florian Yger, and Fabien Lotte. 2019. Towards adaptive classification using Riemannian geometry approaches in brain-computer interfaces. In *7th International Winter Conference on Brain-Computer Interface*. 1–6.

[8] manderfeld, mtyka, enmasse, scottamain, hjonnala, mbrooksx, Namburger, ArnoutDevos, davidp94, and fjp. 2022. *examples-camera*. https://github.com/google-coral/examples-camera

[9] Serafeim Perdikis, Luca Tonin, Sareh Saeedi, Christoph Schneider, and José del R Millán. 2018. The Cybathlon BCI race: Successful longitudinal mutual learning with two tetraplegic users. *PLoS Biology* 16, 5 (2018), e2003787.

[10] Luca Tonin and José del R Millán. 2021. Noninvasive brain–machine interfaces for robotic devices. *Annual Review of Control, Robotics, and Autonomous Systems* 4 (2021), 191–214.

[11] Mojtaba Yavandhasani and Foad Ghaderi. 2021. Visual Object Recognition From Single-Trial EEG Signals Using Machine Learning Wrapper Techniques. *IEEE Transactions on Biomedical Engineering* 69, 7 (2021), 2176–2183.