A Hand Gesture Recognition System Using EMG and Reinforcement Learning: A Q-Learning Approach

Juan Pablo Vásconez, Lorena Isabel Barona López, Ángel Leonardo Valdivieso Caraguay, Patricio J. Cruz, Robin Álvarez, and Marco E. Benalcázar(✉)

Artificial Intelligence and Computer Vision Research Lab, Department of Computer Science and Informatics, Escuela Politécnica Nacional, Quito, Ecuador
{juan.vasconez,lorena.barona,angel.valdivieso,patricio.cruz, robin.alvarez,marco.benalcazar}@epn.edu.ec
https://laboratorio-ia.epn.edu.ec/en/

Abstract. Hand gesture recognition (HGR) based on electromyography (EMG) has been a research topic of great interest in recent years. Designing an HGR to be robust enough to the variation of EMGs is a challenging problem and most of the existing studies have explored supervised learning to design HGRs methods. However, reinforcement learning, which allows an agent to learn online while taking EMG samples, has barely been investigated. In this work, we propose a HGR system composed of the following stages: pre-processing, feature extraction, classification and post-processing. For the classification stage, we use Q-learning to train an agent that learns to classify and recognize EMGs from five gestures of interest. At each step of training, the agent interacts with a defined environment, obtaining thus a reward for the action taken in the current state and observing the next state. We performed experiments using a public EMGs dataset, and the results were evaluated for user-specific HGR models by using a method that is robust to the rotations of the EMG bracelet device. The results showed that the classification accuracy reach up to 90.78% and the recognition up to 87.51% for two different test-sets for 612 users in total. The results obtained in this work show that reinforcement learning methods such as Q-learning can learn a policy from online experiences to solve both the hand gesture classification and the recognition problem based on EMGs.

Keywords: Hand gesture recognition · Electromyography · EMG · Reinforcement learning · Q-learning · Experience replay

1 Introduction

Nowadays, the development of hand gesture recognition systems (HGR) that allow humans to better interact and communicate with computers and machines

Supported by Escuela Politécnica Nacional.

I. Farkaš et al. (Eds.): ICANN 2021, LNCS 12894, pp. 580–591, 2021.
https://doi.org/10.1007/978-3-030-86380-7_47

is a challenging research topic [3,7]. The use of HGR models aims to determine which hand gesture was performed, and when it was realized. Currently, the HGR systems are used for the development of intelligent prostheses, sign language recognition, rehabilitation, among others [3,7].

The EMG signals based on surface sensors can be modeled as a stochastic process that depends on two types of contractions: i) static –the muscle is contracted but there is no motion, and ii) dynamic – the muscle fibers and the joints are in motion [10,14]. However, mathematical approaches to model EMG signals are not used in HGR applications since the parameter estimation in non-stationary processes is difficult. Therefore, machine learning (ML) and deep learning (DL) techniques are typically used to classify EMG signals [4,7]. In particular, supervised methods such as support vector machine (SVM), k-nearest neighbors (K-NN), Artificial neural networks (ANN), Long short-term memory networks (LSTMs), and Convolutional neural networks (CNN) have demonstrated promising results as classifiers for HGR systems, which were able to infer hand gesture classes in real time (less than 300 ms) [6,7].

Supervised methods can achieve high classification performances for HGR applications, other methods such as reinforcement learning (RL) approaches can provide different benefits to help to improve HGR systems. In supervised learning, all the samples must be labeled in order to train a model. Meanwhile, reinforcement learning approaches find the optimal policy that allows an agent to take actions in an environment, so the cumulative reward can be maximized from online experience. Thus, this algorithm can learn from experience while using the system, which opens a wide range of new possibilities for HGR applications.

We have found in the literature a few works that have used reinforcement learning for hand gesture and arm movement recognition. Most of them use different approaches of RL for a particular experiment configuration for a hand or arm recognition application. Moreover, only a few of those approaches try to solve the HGR problem by using EMG signals. For example, in [12] the Myo armband sensor is used to obtain only raw accelerometer data from arm movement. The experiment consists of 3 arm movement, and each class has 30 samples for training and 20 for testing. The authors used Q-learning based on CNN and LSTM, and they defined the rewards of the experiments considering the human feedback. A reinforcement learning classifier for elbow, finger, and hand movement was presented in [9]. In this work, a PowerLabb 26TSystem was used to obtain EMG signals with 3 electrodes, and statistical features (variance, waveform-length, mean, and zero crossing) were extracted. Then, a Q-learning based on a neural network classifier was used to infer six elbow positions and 4 finger movement classes. For this purpose, the authors used 10 subjects, and 144 samples were taken for training and 95 for testing. Furthermore, in [13], a CNN was used to extract EMG signal features, and a dueling deep Q-learning technique was applied to learn a classification policy. Firstly, this approach was trained to learn 6 different hand gestures based on a dataset of 2700 EMG signal samples. Then, data augmentation was used to enlarge the dataset by adding Gaussian noise to the data. A gesture recognition and trajectory calculation

system was developed [16]. This work proposes an interactive learning system that uses EMG and inertial information to teach a robot manipulator. The learning process is composed for a gesture recognition stage –EMG and inertial signals processed and trained off-line–; a statistical encoding stage –Gaussian mixture model–; and a reproduction and reinforcement learning stage –Gaussian Q learning model to correct the grab attitude and position of the robot–.

The current literature has some limitations and issues such us i) the small amount of data (few users and few samples per user), ii) the use of a single reward parameter to evaluate a single evaluation metric (classification accuracy), and iii) the lack of use of experience replay which can improves Q-learning performance. Considering these issues, the main contributions of this work are presented below:

- We use a public large dataset EMG-EPN-612 composed of 612 users. We used 306 users to train, validate, and test a user-specific model to find the best hyper-parameters. Then, we use them to train and test user-specific models for the other 306 users to test our model with different data and evaluate over-fitting.
- We successfully combine the EMG signals with a reinforcement learning method –Q-learning– considering two different rewards: the first reward to classify, and the second to recognize the hand gestures.
- We propose the use of experience replay that helps the Q-learning method to improve the model performance.

This work is organized as follows. In Sect. 2, the proposed architecture for an HGR system based on EMG and Q-learning is presented. Moreover, each stage of such architecture is reviewed in detail. The classification and recognition results are presented in Sect. 3. Finally, the conclusions are presented in Sect. 4.

2 Hand Gesture Recognition Architecture

In this section, we propose an architecture to solve the HGR problem based on EMG signals and Q-leaning, which is illustrated in Fig. 1. As can be observed, such architecture is conformed by a data acquisition, pre-processing, feature extraction, classification based on Q-learning, and post-processing. Following, each stage is explained in detail.

2.1 Data Acquisition

In this work, we used the public dataset EMG-EPN-612 collected by using the Myo armband device –8 channels 200 Hz– to obtain the EMG signals from 5 different gestures –wave in, wave out, fist, open, and pinch– [5]. If the gesture is not one of the above mentioned, then is considered as the no gesture (relax gesture). Such dataset is composed of 612 users, where 306 of them –training set A– are used to train, validate and test models to obtain the best possible hyper-parameter configurations, and the other 306 users –training set B– are

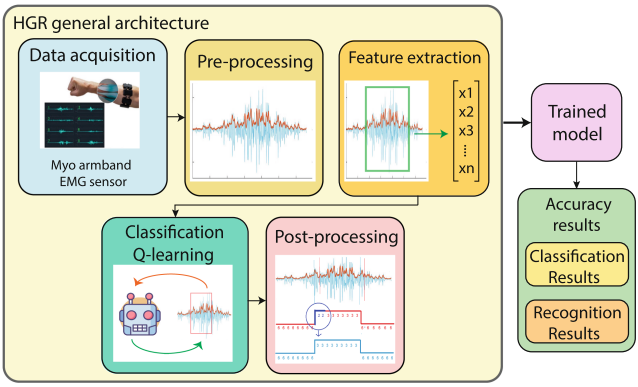

Fig. 1. Hand gesture recognition architecture based on Q-learning to learn to classify and recognize EMG signals.

used to perform an over-fitting evaluation. Each user of the training set A and B have 300 hand gesture repetitions. However, only the first 150 repetitions of training set B have the class gesture labeled and the muscular activity segmented –ground-truth–, which is key to run our tests and give rewards at each step. To access the full labeling information of the training set B it is necessary to test our model in our online web page as stated in [1]. Further details related to the dataset information, acquisition protocol, and the web page information for testing the final model can be found in [1,5]. The dataset distribution for both training set A and B that we used in this work can be observed in Table 1. It is worth mentioning that during training, samples with the no gesture (relax gesture) were not considered since most of the other signals have already several samples of such gesture. However, no gesture signal samples were considered for the final test results.

Table 1. Data set distribution to evaluate user-specific models [5].

	User-specific model (one model for each of the 306 users)			
	Number of models	Training	Validation	Test
Training set A	306 models trained (to find the best hyper-parameters)	100 samples per user	13 samples per user	12 samples per user
Training set B	306 models trained (to use the best founded hyper-parameters)	150 samples per user	–	150 samples per user

2.2 Pre-processing

The EMG signals need to be split into multiple windows to be analyzed, which is known as the segmentation procedure. Thus, each gesture sample was processed through the use of sequential sliding windows that slide over the entire EMG signal [4,7]. In this work, each window was tested for several points of separation –stride– and windows size during the validation process, but only the configurations that obtained the highest performance were selected for the testing process –stride = 40 and window size =300 –. As part of the pre-processing stage, the EMG energy criterion was used to identify if a current analyzed window is able to be classified or not. For this purpose, each EMG window has to surpass an energy threshold –17%– to be computed for the next stage, which is feature extraction. It is worth mentioning that our method is robust against the rotations of the bracelet device, as is explained in detail in [2].

2.3 Feature Extraction

Feature extraction methods are used to convert the EMG signal into a set of relevant features by extracting them in different domains, such as time, frequency, or time-frequency. In this work, five feature extraction methods in the time domain were used over every sliding window only when it surpassed the threshold of energy. The feature extraction methods that were used are: Standard deviation (SD), Absolute envelope (AE), Mean absolute value (MAV), Energy (E), and Root mean square (RMS) [2].

2.4 Classification Using Q-Learning

Q-learning is an off-policy algorithm of reinforcement learning. For any finite Markov Decision Process (MDP), Q-learning finds an optimal policy by maximizing the expected value of the total reward (i.e., return function) given both an initial state and an action [15]. For this work, we assume we have access only to observations instead of the full state of the environment. This occurs because it is not guaranteed a one-to-one mapping between the set of EMG window observations and the set of feature vectors used in this work. Therefore, it is possible a feature vector characterizes two different window observations of a EMG. Thus, this problem can be treated as a Partially Observable Markov Decision Process (POMDP), in which we apply Q-learning to the set of observations instead of the set of states. The Q-learning function is presented in the following equation:

$$Q^{(new)}\left(O_t, A_t\right) \leftarrow Q^{(old)}\left(O_t, A_t\right) + \alpha\left(R_{t+1} + \gamma \cdot \underset{a'}{max}[Q\left(O_{t+1}, a'\right)] - Q^{(old)}\left(O_t, A_t\right)\right) \quad (1)$$

where $Q^{(new)}\left(O_t, A_t\right)$ and $Q^{(old)}\left(O_t, A_t\right)$ are the updated and the old Q value, respectively, for the observation O_t and the action A_t, α is the learning rate, γ is the discount factor, $\underset{a'}{max}[Q\left(O_{t+1}, a'\right)]$ is the estimated optimal future

Q value, and R_{t+1} is the reward received when moving from the observation O_t to the observation O_{t+1} when taking the action A_t in O_t.

Q-learning can be implemented using lookup tables to store the Q values for each state and action of a given environment [15]. However, this approach is not suitable for environments with a large state-action spaces. For this case, Q-learning can be combined with function approximation to facilitate its application for large finite or infinite state-action spaces. In this work, we approximated the Q function using a feed-forward artificial neural network (ANNs) [9,15], which will be referred as QNN. To update the QNN weights, we used the following equation:

$$W \leftarrow W + \alpha \left(R_{t+1} + \gamma \cdot \underset{a'}{max}[Q\left(O_{t+1}, a', W\right)] - Q\left(O_t, A_t, W\right) \right) \cdot \nabla Q\left(O_t, A_t, W\right) \quad (2)$$

where $\nabla Q\left(O_t, A_t, W\right)$ denotes the gradient of the outputs of the QNN with respect to W, which is a vector with the weights and biases of the QNN.

To accelerate the training of the QNN, we used experience replay [8,11]. For this purpose, we stored in a dataset $\mathcal{D} = \{E_1, E_2, \cdots, E_T\}$ the tuple $E_t = (O_t, A_t, R_t, S_{t+1})$, which contains the agent's experience at time t. During learning, we updated the parameters of the QNN using Eq. 2 computed on mini-batches of experience drawn uniformly at random from $\mathcal{D}$ [8,11].

Finally, the proposed POMDP solves the sequential decision-making problem that represents the EMG sliding window classification. The scheme of interaction between the QNN agent and the environment is shown in Fig. 2.

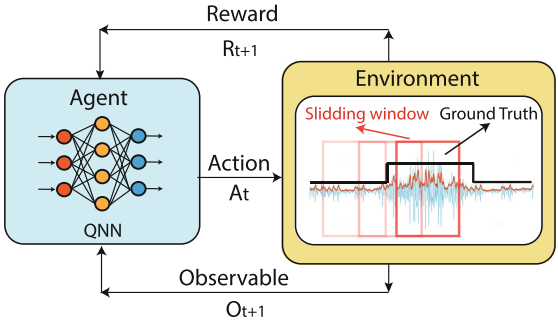

Fig. 2. Scheme of the interaction between the QNN agent and the environment. The EMG classification problem is modeled as a partially observable markov decision process (POMDP).

Below we define the elements of the proposed classification stage modeled as a Q-Learning problem:

Agent: The agent is the decision-maker entity, based on the QNN network, whose objective is to take actions, at any state of the environment, to maximize the total reward. In this work, the goal of the agent is to predict correctly as

many labels as possible in the sequence of labels obtained by classifying each window observation of a given EMG. Here, the interaction between the agent and the environment is episodic.

Observable: An observable contains some information of the actual state of the environment with which the agent is interacting. In this work, we define the observable O_t for the unknown state S_t as the result of applying the feature extraction functions (SD,AE, MAV, E, and RMS) to each EMG channel observed through a window, with $t = 1, 2, \cdots, N$. In this case, N denotes the number of window observations that we can extract from an EMG using a given window length and stride. The end of an episode of interaction between the agent and the environment occurs when the last observable O_N is reached. It is worth mentioning that there is not necessarily a bijection between the set of observables and the set of states of the environment.

Action: An action is the choice or behavior A_t that the agent makes in the current observable O_t. In this work, the set of actions that the agent can take in each observable includes the classes: wave in, wave out, fist, open, pinch, and no gesture.

Environment: The environment is defined in code as the sliding window information –feature vectors and labels– extracted from each EMG signal.

Reward: The reward is the feedback information that the agent receives from its interaction with the environment. In this work, we define two rewards: one for classification and the other for recognition. If the agent predicts correctly the label for an EMG window observation, then it receives a reward $R_t = +1$; otherwise, the reward the agent gets is $R_t = -1$. Once an episode is over, the ground truth (vector of known labels) is compared with the vector of labels predicted by the agent, and if the overlapping factor of the predicted gestures different from the no-gesture is more than 70%, then the recognition is considered successful and the agent receives an additional reward of $R_t = +10$; otherwise $R_t = -10$. A key condition for the recognition procedure is that if one or more predicted labels are different from the ground truth labels, then the recognition is considered to be wrong. A detailed illustration of the reward the agent gets for classification and recognition is presented in Fig. 3.

2.5 Post-processing

To improve the accuracy of the proposed HGR system, the post-processing stage adapts the output of the classification to the final application. Most of the works found in the literature use the majority voting technique, elimination of consecutive repetitions, threshold method, the gesture mode, and velocity ramps [2,7]. In this work, we calculate the mode on the predicted vector of classes that are different from the label no-gesture. Then, all the labels in such vectors that are different from the mode gesture are replaced with such gesture for that EMG sample.

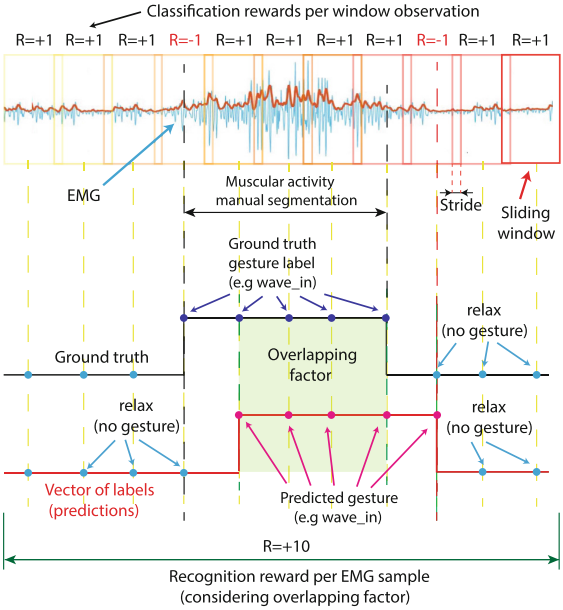

Fig. 3. Reward the agent gets for classification and recognition illustration. The classification is considered successful if each predicted window label matches the ground truth label for that point. On the other hand, each recognition reward is assigned if the recognition procedure that considers the overlapping factor between the vector of predicted labels per sample and the ground truth if higher than a threshold –70%–.

3 Results

In this section, we first present the results of training and validating the HGR user-specific proposed models on the training set A to find the best possible hyper-parameters. We illustrate in Fig. 4 a Q-learning training sample where the Average of wins vs the number of epochs can be observed. It is to be noticed that the average of wins per episode converge to their optimal value after 1000 epochs. To obtain this results, we repeat the experiences of the 100 training samples of each user 10 times through the neural network that represents the agent. The procedure of using 10 times the 100 training samples was necessary since the learning procedure is online and its performance depends on the number of experiences that the agent can use to learn.

The hyper-parameter configurations for our experiments is summarized in Table 2. It is worth mentioning that we tested 72 different hyper-parameter configurations (different number of neurons, learning rate α, momentum, and mini-batch size) to find the best fit for our models. Moreover, we tested different window sizes –150, 200, 300, and 400 points– for each of those 72 configurations during the validation process. However, only the best hyper-parameters configurations and window size are presented in Table 2.

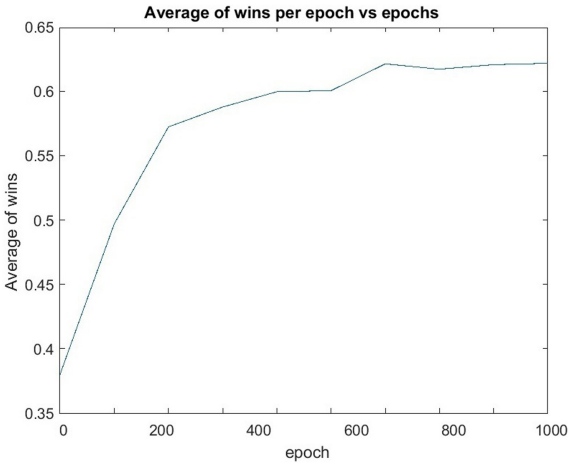

Fig. 4. Q-learning training results. Average of wins vs. number of epochs during training. The agent try to find a policy that maximize the total reward, which means to correctly predict as many gesture categories as possible for the sliding windows in the EMGs. If the agent is able to recognize an EMG signal correctly, that episode is considered as won.

In addition, by using the hyper-parameters from Table 2, we tested 160 users (since this procedure is heavily time-consuming) of the validation set from training set A. For this procedure, we changed the learning rate α as well as the window stride, and the results are presented in Fig. 5. This procedure helps us to find the best possible learning rate and stride configuration for our experiments. It can be observed that $\alpha = 0.03$ and stride $= 40$ achieve the best accuracy results with high classification and recognition as well as low variance.

Table 2. Best hyper-parameters found during tuning procedure on Training set A.

	Best hyper-parameters
Number of neurons	40, 50, 50 and 6 for the input layer, hidden layer 1, hidden layer 2, and output layer respectively
Sliding window size	300 points
Initial momentum	0.3
Mini batch size	25
ϵ-greedy value	0.2
Discount factor γ	1
Training set replay per user	10 times
Learning rate (α) decay factor	α*exp(-5*CurrentEpoch/NumEpochs)

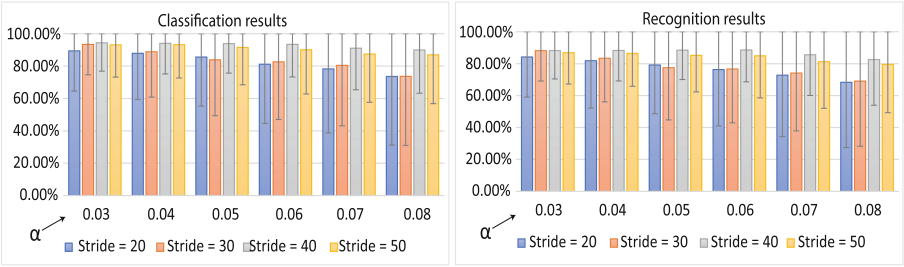

Fig. 5. User-specific HGR model classification and recognition accuracy results for 160 users for validation of training set A with different values of learning rate α and stride. It can be observed that $\alpha = 0.03$ and stride $= 40$ achieve the best results.

Based on the best hyper-parameters described in Table 2, and the values of $\alpha = 0.03$, and stride $= 40$, we used the test-set of the training set A to evaluate such models. The test accuracy results for 306 users of the training set A were $90.78\% \pm 20\%$ and $84.43\% \pm 21\%$ for classification and recognition respectively. Then, we use the best hyper-parameters that we found to train the 306 users of the training set B, which will help us to evaluate our models with different data and analyze over-fitting. The test results for 306 users of the training set B were $90.47\% \pm 20\%$ and $87.51\% \pm 21\%$ for classification and recognition respectively. We summarized the test results for 306 users for both training set A and B with the best found hyper-parameters in Table 3. Since there is barely a difference between the test results of training set A and B, we can infer that our model is robust against over-fitting. It can be seen that the standard deviation is moderate, which could be due to the variability of the data set, as the model works well for most samples and fails for a few. Finally, we also present the confusion matrix that represents the classification results on the test set of training set B in Fig. 6, which allow us to observe in detail the results for each hand gesture. It is worth mentioning that the processing time of each window observation is in average 19 ms.

Table 3. User-specific test accuracy results of the HGR best model for training set A and B (306 users for training set A and B respectively - 612 users in total).

Test results - Training set A		Test results - Training set B	
Classification	Recognition	Classification	Recognition
$90.78\% \pm 20\%$	$84.43\% \pm 21\%$	$90.47\% \pm 14.24\%$	$87.51\% \pm 14.1\%$

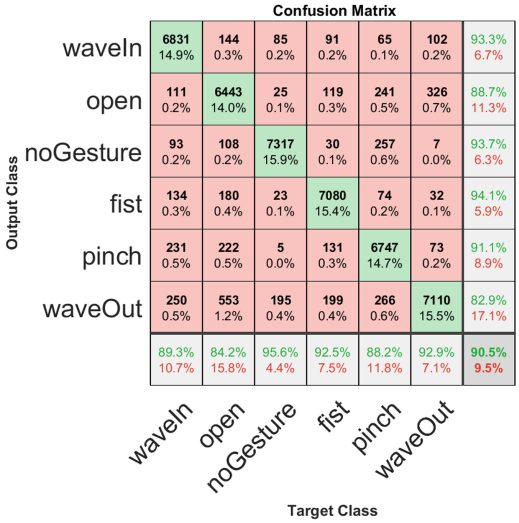

Fig. 6. User-specific HGR model confusion matrix for 306 users of test set B with the best hyper-parameter configuration. Each hand gesture class results can be observed in detail.

4 Conclusions

In this work, we proposed a HGR system based on Q-learning to classify and recognize five gestures using EMGs from a public dataset. The results were evaluated for user-specific HGR models. The system was tested for two different test sets for 612 users, where the classification accuracy reached up to 90.78% and the recognition accuracy reached up to 87.51%. The results obtained are encouraging, and they show that Q-learning can learn a policy from online experience to classify and recognize gestures based on EMGs. It is worth mentioning that using Q-learning for HGR allows an agent to learn from online experience. Therefore, the model can be improved at each iteration if the ground truth is defined or if we use an automatic labeling procedure. Future work includes testing other RL algorithms to create a state estimator from observations to evaluate the proposed POMDP model with belief states, as well as testing on more data-sets.

Acknowledgment. The authors gratefully acknowledge the financial support provided by the Escuela Politécnica Nacional (EPN) for the development of the research project "PIGR-19-07 Reconocimiento de gestos de la mano usando señales electromiográficas e inteligencia artificial y su aplicación para la implementación de interfaces humano—máquina y humano—humano".

References

1. EMG Gesture Recognition Evaluator. https://aplicaciones-ia.epn.edu.ec/webapps/home/session.html?app=EMGGestureRecognitionEvaluator
2. Barona López, L.I., Valdivieso Caraguay, Á.L., Vimos, V.H., Zea, J.A., Vásconez, J.P., Álvarez, M., Benalcázar, M.E.: An energy-based method for orientation correction of emg bracelet sensors in hand gesture recognition systems. Sensors **20**(21), 6327 (2020)
3. Benalcázar, M.E., Jaramillo, A.G., Zea, A., Páez, A., Andaluz, V.H., et al.: Hand gesture recognition using machine learning and the myo armband. In: 2017 25th European Signal Processing Conference (EUSIPCO), pp. 1040–1044. IEEE (2017)
4. Benalcázar, M.E., et al.: Real-time hand gesture recognition using the myo armband and muscle activity detection. In: 2017 IEEE Second Ecuador Technical Chapters Meeting (ETCM), pp. 1–6. IEEE (2017)
5. Benalcázar, M., Barona, L., Valdivieso, L., Aguas, X., Zea, J.: Emg-epn-612 dataset (2020). https://doi.org/10.5281/zenodo.4027874
6. Englehart, K., Hudgins, B.: A robust, real-time control scheme for multifunction myoelectric control. IEEE Trans. Biomed. Eng. **50**(7), 848–854 (2003)
7. Jaramillo-Yánez, A., Benalcázar, M.E., Mena-Maldonado, E.: Real-time hand gesture recognition using surface electromyography and machine learning: a systematic literature review. Sensors **20**(9), 2467 (2020)
8. Kapturowski, S., Ostrovski, G., Quan, J., Munos, R., Dabney, W.: Recurrent experience replay in distributed reinforcement learning. In: International Conference on Learning Representations (2018)
9. Kukker, A., Sharma, R.: Neural reinforcement learning classifier for elbow, finger and hand movements. J. Intell. Fuzzy Syst. **35**(5), 5111–5121 (2018)
10. McGill, K.: Surface electromyogram signal modelling. Med. Biol. Eng. Comput. **42**(4), 446–454 (2004)
11. Mnih, V., et al.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529–533 (2015)
12. Seok, W., Kim, Y., Park, C.: Pattern recognition of human arm movement using deep reinforcement learning. In: 2018 International Conference on Information Networking (ICOIN), pp. 917–919. IEEE (2018)
13. Song, C., Chen, C., Li, Y., Wu, X.: Deep reinforcement learning apply in electromyography data classification. In: 2018 IEEE International Conference on Cyborg and Bionic Systems (CBS), pp. 505–510. IEEE (2018)
14. Sugiyama, M., Kawanabe, M.: Machine Learning in Non-Stationary Environments: Introduction to Covariate Shift Adaptation. MIT Press, Cambridge (2012)
15. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (2018)
16. Wang, F., et al.: Robot learning by demonstration interaction system based on multiple information. In: 2018 IEEE 8th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER), pp. 138–143. IEEE (2018)