УДК 501; 57.085.1
doi:10.18720/SPBPU/2/id23-83

Vyacheslav V. Potekhin [1],
Candidate of Technical Sciences, Associate Professor;
Lara Assalama [2] (Syria),
M. Sc., Intern of Higher School of Cyber-Physical Systems and Control

HAND GESTURE RECOGNITION USING SEMG WITH XGBOOST, AND AVERAGING FOR AUGMENTATION

Peter the Great St. Petersburg Polytechnic University, St. Petersburg, Russia;
[1] slava.potekhin@gmail.com, [2] larams.lisl@gmail.com

Abstract. In the recent years, XGBoost has proved to be a powerful, quick machine-learning algorithm, and many researcher used it to solve different types of machine learning problems. In this work, we invested it to recognize hand gesture using sEMG signals, we used with it averaging data samples as an augmentation method to overcome the lack of big dataset size, which sEMG datasets suffer from due to difficulties of gathering samples. The presented augmentation method proved to be faster than many other methods, moreover it with XGBoost gave state-of-the-art results on the used dataset.

Keywords: sEMG, XGBoost, augmentation, averaging, Myo armband, NinaPro DB5.

Потехин Вячеслав Витальевич [1],
доцент, канд. техн. наук, доцент;
Ассалама Лара (Сирия) [2],
магистр, стажёр

РАСПОЗНАВАНИЕ ЖЕСТОВ РУК С ИСПОЛЬЗОВАНИЕМ SEMG С XGBOOST И УСРЕДНЕНИЕ ДЛЯ УВЕЛИЧЕНИЯ

[1, 2] Россия, Санкт-Петербург Санкт-Петербургский политехнический университет Петра Великого,
[1] slava.potekhin@gmail.com, [2] larams.lisl@gmail.com

Аннотация. В последние годы XGBoost зарекомендовал себя как мощный и быстрый алгоритм машинного обучения, и многие исследователи использовали его для решения различных типов задач глубокого обучения. В этой работе мы использовали его для распознавания жестов рук с использованием сигналов sEMG, мы использовали с ним усреднение выборок данных в качестве метода увеличения, чтобы преодолеть недостаток большого размера набора данных, от которого страдают наборы данных sEMG из-за трудностей сбора выборок. Представленный метод увеличения оказался быстрее многих других методов, более того, он с XGBoost дал самые современные результаты по используемому набору данных.

Ключевые слова: sEMG, XGBoost, увеличение, усреднение, повязка Myo, Nina ProD B 5.

Introduction

The handicapped face serious problems trying to merge and have an effective roll in the society because of the lack of proper means and abilities. Thus, prostheses (artificial limbs) became very necessary to help handicapped merge. With the evolution of prostheses from mere rigid extremities to moving and reactive systems, an efficient control system became essential; here comes the role of AI. Therefore, machine learning is expanding in the field of control systems. Moreover, when talking about sEMG signals and machine learning methods it is indeed a promising field of study; because of the quick advances in industry that allow getting reliable equipment yet for affordable prices, and because of the big amount of datasets available on the Internet.

One of the machine learning algorithms that is widely used in the last few years is XGBoost, it has gained its reputation after being recommended by many machine learning competitions winners, for its state-of-the-art results and quick performance compared to other methods.

The same goes for EMG signals, which are getting more and more useful in the health field through machine learning techniques. From limb gesture recognition to nerve and muscles illnesses diagnosis [1], EMG are a promising handy tool.

Alas, this promising tool still have a serious drawback; it is the small datasets, to solve this issue many augmentation algorithms have been presented so far, some with state-of-the-art results, but slow. In this work, we present a quick algorithm without compromising the quality of the results.

This paper is organized as follows: the second part will be about sEMG, its definition, and a comparison with EMG signals, justifying our choice of sEMG signals. In the third part we summaries the main features of the Myo armband. Then in the fourth part, we present the main categories of the used dataset. The next part will be about the main properties of XGBoost, and the mathematical modeling of the problem it solves. We introduce the augmentation method in the sixth part, then the preprocessing stage before we discuse the result and sum up everything in the conclusion.

1. sEMG signals

EMG stands for electromyography. It is the study of muscle electrical signals. EMG is sometimes referred to as myoelectric activity. The EMG signal is a biomedical signal that measures electrical currents generated in muscles during its contraction representing neuromuscular activities [2].

There are two types of EMG sensors: invasive which needs injecting a measuring needle into the muscle, as shown in Figure 1, and noninvasive which are usually referred to as sEMG since the sensor is placed on the surface of the skin, as shown in Figure 2.

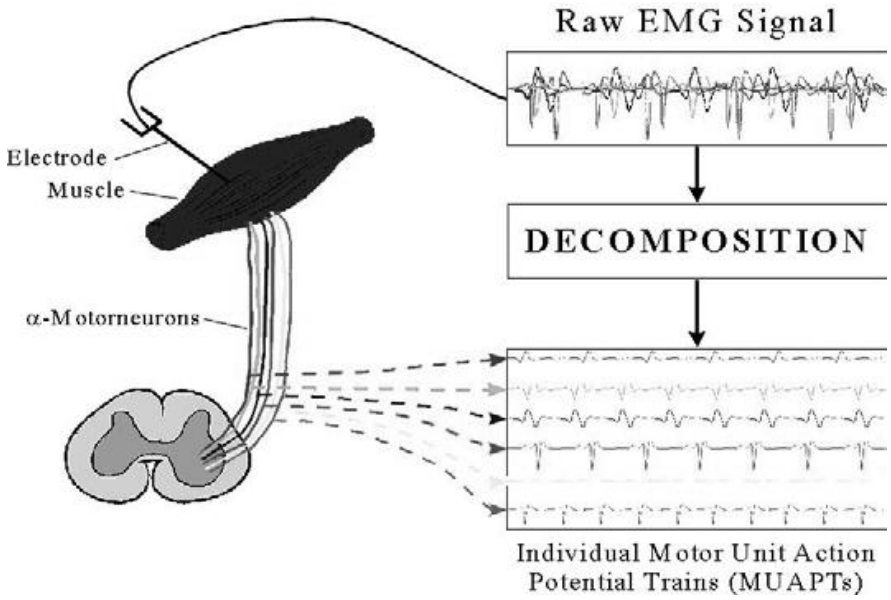

Fig. 1. Invasive EMG sensors

Invasive EMG sensors offer higher resolution signal, because the signal is measured within the muscle itself rather than on its surface.

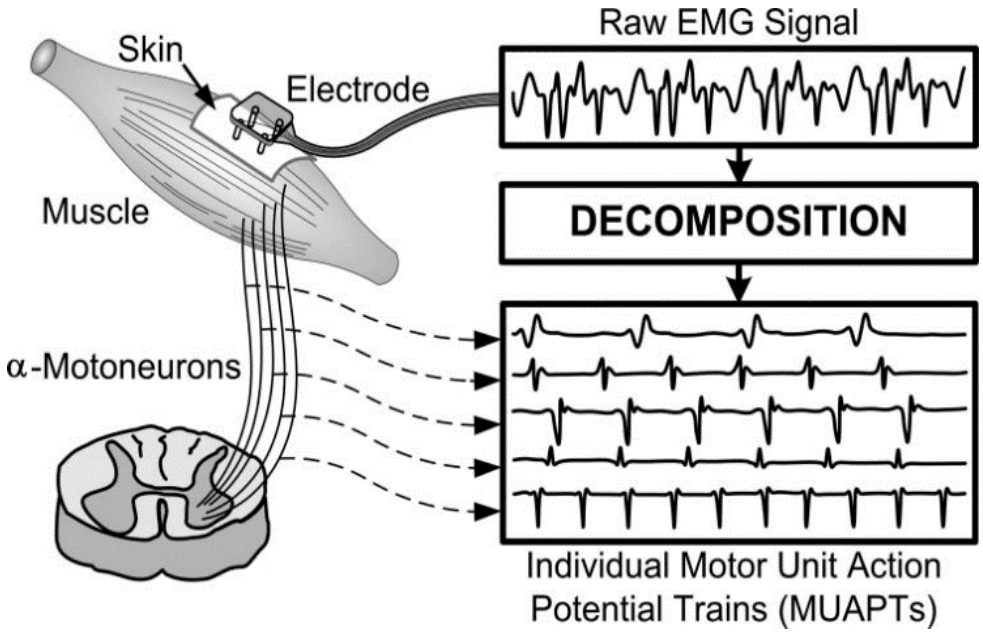

Fig. 2. Noninvasive EMG sensor (sEMG)

In addition, the signal has minimum amount of noise, unlike that sEMG suffers because of the effects of other muscles signals and the skin and other tissues around the muscle. However, invasive EMG sensors may cause infection, and can only be used by medical experts to decide the right place to inject the sensing needle. Therefore, sEMG sensors are a better choice for the case of research and lab work, while the noise can be minimized with proper preprocessing, and the lack of high resolution can be compensated with the developing machine learning methods. One example of sEMG sensors is Myo armband.

2. Myo armband

Myo armband is a compact device with integrated sensors, processing and transmitting unit that allows the correct detection of the needed signals simply by wearing it on the forearm, without using cables because the transmission is performed through BLE technology [3].

The Myo armband presents a good choice compared to other sensors because of its features:

– Low cost

– Availability

– Safety: low operating voltage that cannot harm the user, moreover the sensor is placed on the surface of the skin, no needle needed

– Ease of use: no needles, no wires, and the manufacturers offer free operating software

– 2 types of sensors: EMG and IMU

– Preprocessed signals: no filtering needed.

3. Dataset

The dataset chosen for this work is NinaPro dataset DB5. It is available free online on their site:
http://ninaweb.hevs.ch/#:~:text=Ninapro%20is%20a%20publicly%20available,machine%20learning%20based%20control%20systems.

The samples are systematically obtained and well described in the corresponding papers for each group. Many researchers used these datasets; which means the results of our work can be compared to many others.

We chose DB5 because the samples are obtained using two Myo armbands, and the results of our work are to be applied using a Myo armband. The dataset as described in the work of (Stefano Pizzolato, 2017) [5]. it consists of several Matlab files (.mat) representing the sEMG signal of the different gestures done by each subject, 3 files per candidate, a file for each exercise of the NinaPro 3 exercises set, shown in Figure 3.

Fig. 3. NinaPro exercises set: a) Exercise 1: 12 basic flexions and extensions of figures. b), c) Exercise 2: 8 isometric and isotonic hand configuration, and 9 basic wrist movements. d) Exercise 3: 23 grasp and functional mevements

4. XGBoost algorithm

4.1. Why XGBoost?

XGBoost stands for e**X**treme **G**radient **Boost**ing. It was developed by Tianqi Chen and is laser focused on computational speed and model performance. Gradient boosting is one of the most powerful techniques for building predictive models. It is a scalable machine learning system for tree boosting; the system is available as open source package [6].

The Gradient Boosting algorithm involves three elements:

1. A loss function to be optimized, such as cross-entropy for classification or mean squared error for regression problems.

2. A weak learner to make predictions, such as a greedily constructed decision tree.

3. An additive model, used to add weak learners to minimize the loss function.

The real interest is the speed provided by the careful engineering of the implementation, including:

- **Parallelization** of tree construction using all of your CPU cores during training.
- **Distributed Computing** for training very large models using a cluster of machines.
- **Out-of-Core Computing** for very large datasets that do not fit into memory.
- **Cache Optimization** of data structures and algorithms to make the best use of hardware.

4.2. XGBoost mathematics

For a given dataset $\mathcal{D} = \{(x_i, y_i)\}(i = 1, \ldots, n, x_i \in \mathbb{R}^m, y_i \in \mathbb{R})$, XGBoost uses a tree ensemble with K additive functions to predict the output as follows:

$$\hat{y} = \phi(x_i) = \sum_{k=1}^{K} f_k(x_i) \ , f_k \in \mathcal{F}$$

where:

$$\mathcal{F} = \{f(x) = w_{q(x)}\}(q : \mathbb{R}^m \longrightarrow T, w \in \mathbb{R}^T),$$

- T: the number of leaves;
- q: the structure of each tree that maps an example to the corresponding leaf index.

We define the problem to solve as:

$$Hypothesis: \quad \hat{y} = \phi(x_i)$$

regulirized cost function:

$$\mathcal{L}(\phi) = \sum_i l(\hat{y}, y) + \sum_k \Omega(f_k)$$

where:

$$\Omega(f_k) = \gamma T + \frac{1}{2}\lambda \|w\|^2$$

$$objective: \qquad \min_{\omega} imize\, \mathcal{L}(\phi)$$

As the model is trained in an additive way, the cost function at the *t*-th iteration is written as:

$$\mathcal{L}^t(\phi) = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \qquad (1)$$

Using second order approximation, the Eq (1) is rewritten into:

$$\mathcal{L}^t(\phi) = \sum_{i=1}^{n} [l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t) \qquad (2)$$

where:

- g_i : first order gradient of cost function

$$g_i = \frac{\partial}{\partial f_t} l\left(y_i, \hat{y}_i^{(t-1)}\right)$$

- h_i : second order gradient of cost function

$$h_i = \frac{\partial^2}{\partial {f_t}^2} l\left(y_i, \hat{y}_i^{(t-1)}\right)$$

If we change the point of view and look at the previous term of the leaves rather than the input angle, in other words: define $I_j = \{i | q(x_i) = j\}$ as the instance set of leaf j, we can rewrite Eq (2) as follows:

$$\mathcal{L}^t(\phi) = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)}\right) + \sum_{j=1}^{T} [(\sum_{i \in I_j} g_i) w_j + \frac{1}{2} (\sum_{i \in I_j} h_i) w_j^2] + \Omega(f_t) =$$

$$= \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)}\right) + \sum_{j=1}^{T}[(\sum_{i\in I_j} g_i)\, w_j + \frac{1}{2}(\sum_{i\in I_j} h_i) w_j^2] + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 ;$$

$$\mathcal{L}^t(\phi) = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)}\right) + \sum_{j=1}^{T}[(\sum_{i\in I_j} g_i)\, w_j + \frac{1}{2}(\sum_{i\in I_j} h_i + \lambda) w_j^2] + \gamma T \qquad (2)$$

The previous term (Eq. (3)) is at its minimum when:

$$w_j = w_j^* = -\frac{\sum_{i\in I_j} g_i}{\sum_{i\in I_j} h_i + \lambda}.$$

In addition, the term:

$$\tilde{\mathcal{L}}^{(t)}(q) = -\frac{1}{2}\sum_{j=1}^{T} \frac{(\sum_{i\in I_j} g_i)^2}{\sum_{i\in I_j} h_i + \lambda} + \gamma T.$$

is used as a scoring function to evaluate a tree structure q [6].

While enumerating all possible tree structures is impossible, a greedy algorithm, which starts from one leaf and iteratively add new branches to the tree, is used instead, and the loss reduction after the split is given by:

$$\mathcal{L}_{split} = \frac{1}{2}\left[\frac{(\sum_{i\in I_L} g_i)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{(\sum_{i\in I_R} g_i)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{(\sum_{i\in I} g_i)^2}{\sum_{i\in I} h_i + \lambda}\right] + \gamma T$$

However, to have good results using XGBoost the number of samples should be bigger than the number of features, as the results of this work will demonstrate. Moreover, the ratio: number of samples to number of features should be at least > 0.5.

5. Augmentation

One of the problems one will face when working on sEMG signals acquisition is the time consuming process and the few numbers of subjects willing to do the experiments to get the measures needed. Therefore, sEMG datasets suffer of the small number of samples, which is a serious problem in machine learning.

The solution is to find a way to augment the number of samples without the need for new experiments, some ways were presented so far; such as: jittering [7, 8], rotation [8], scaling [8, 9], magnitude warping [8], permutation [8], slicing, window warping [10], SPAWNER [11], wDBA [12], RGW, and DGW [13].

EMG is a time series data, B.Kenji Iwana et el in their work [14] classified augmentation methods used with time series datasets into four categories:

- Random transformation (Jittering, Rotation, Time Warping, Frequency Masking, etc.)
- Pattern Mixing (DFM, Averaging, EMDA, etc.)
- Generative models (LGT, Gaussian Trees, etc.)
- Decomposition (STL, ICA, EMD, etc.).

Although random transformation methods might be tempting because of the very small time they need to generate a new sample compared to other methods, but they also affect the training accuracy negatively, according to B. Kenji Iwana et el in their work [14]. It is expected because they do not preserve the original signals distribution like pattern mixing methods and decomposition methods. Pattern mixing methods are time-consuming methods but they proved to enhance the accuracy. In this work, we propose averaging the signals as an augmentation method. Its computation complexity is $O(T)$; while other pattern mixing methods are $O(T^2)$, where T is the length of the sample. Moreover, Averaging helps overcome the differences in EMG signals due to the differences between subjects in gender, age, height, weight, health and muscle fatigue; since it preserves the common changes of the signal.

6. Preprocessing

NinaPro DB5 sEMG samples are stored in 30 different files, 3 for each subject; each file of those contains the samples of one group of NinaPro's exercises.

The samples are stored in one array with one dimension, so the first step was to read and separate them using the labels array. The second step was relabeling, because each file started labeling from zero not taking into consideration other exercises (files) labels, the first exercise consists of 12 moves, the second consists of 17 moves, and the third consists of 22 moves, with the Rest sample we have 53 classes. The next step was to make all the samples of the same length by cutting the extra points according to a chosen signal length. To choose a proper signal length we studied the distribution of signals lengths. As shown in Figure 4, most of the movement sEMG signals lengths are in the range [600, 900] points, just the Rest (class "0") sEMG signals lengths are in the range [1000,1500] points. Taking into consideration the percentage of signals lengths around one length, the best signal length is 600points. Moreover, this choice will not have a massive effect on the Rest (class "0") signals, since their changes around zero are very similar along the whole signal.

Once the length was chosen, the new sample was gained by cutting of the extra from both sides, assuming that such signals tend to have the main data around the middle.

Then comes the augmentation step, every two samples with the same label were averaged, and the resulting sample was added to the data set, thus we got n(n-1)/2 new samples for each class; where n is the number of the class samples. Figure 5 shows an example of averaging two sample of the second class, of the first sensor, for the same subject.

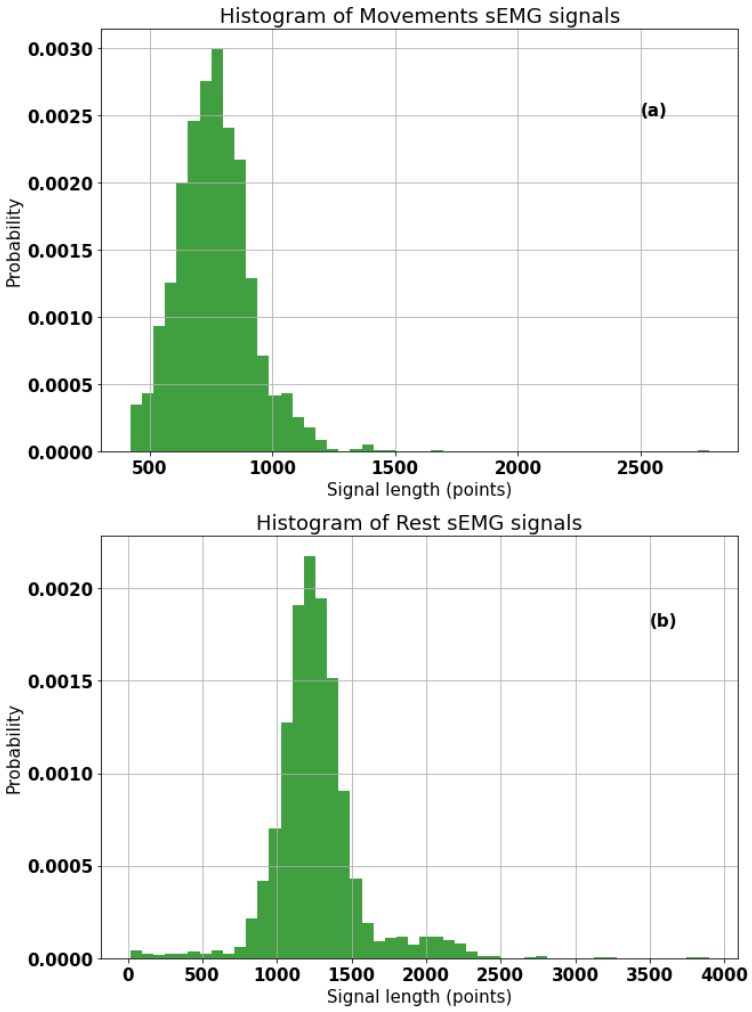

Fig. 4. NinaPro DB5 dataset sEMG signals lengths histograms: (a) Movements sEMG signals Histogram; (b) Rest sEMG signals histogram

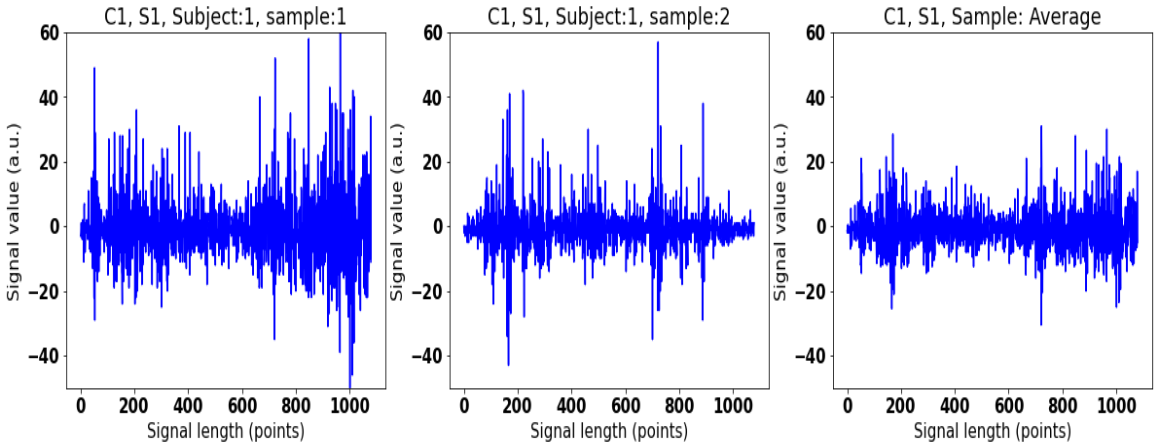

Fig. 5. Averaging example, two signals of the first class of the first sensor, for the first subject

Figure 6 shows another example of the 40th class, of the first sensor but for the 1st subject and the 7th subject. Averaging smoothens those spike-like changes, which are in electrical terms are noise rather than real data.

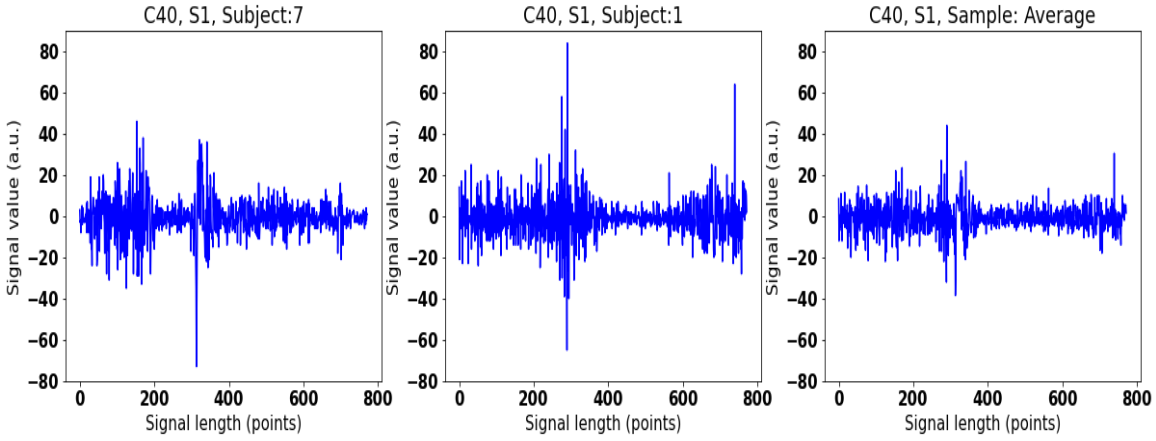

Fig. 6. Averaging example, two signals of the 40th class, of the first sensor, for the first and seventh subjects

Averaging two samples took less than 1ns, even for the signals of 900points length, with the whole 16 sensors of the two armbands taken into consideration.

Table 1 shows the time needed to get new samples using averaging, and the number of new samples obtained, it took less than 18 minutes to generate 73436 Samples of the length 600point for 16 sensors.

Table 1

Samples augmentation time. *: only the signals of one armband (8 sensors). **: The signals of both armbands (16 sensors)

Signal length (points)	Samples number	New samples number	Augmentation Time
900 * 16	2	1	<1ns
200 * 8	3180	93810	3m 57s
400 * 8	3180	93810	8m 13s
600 *8	2789	73436	8m 16s
200 * 16	3180	93810	8m 10s
400 * 16	3180	93810	17m 57s
600 * 16	2789	73436	17m 17s

The time needed is dependent linearly on the signal length, in another ward the computation complexity is O(T), where T is the signal length.

Figure 7 shows that clearly.

The augmentation was done using an Intel(R) Core(TM) i7-10870H CPU @ 2.20GHz 2.21GHz processor with 16G Ram.

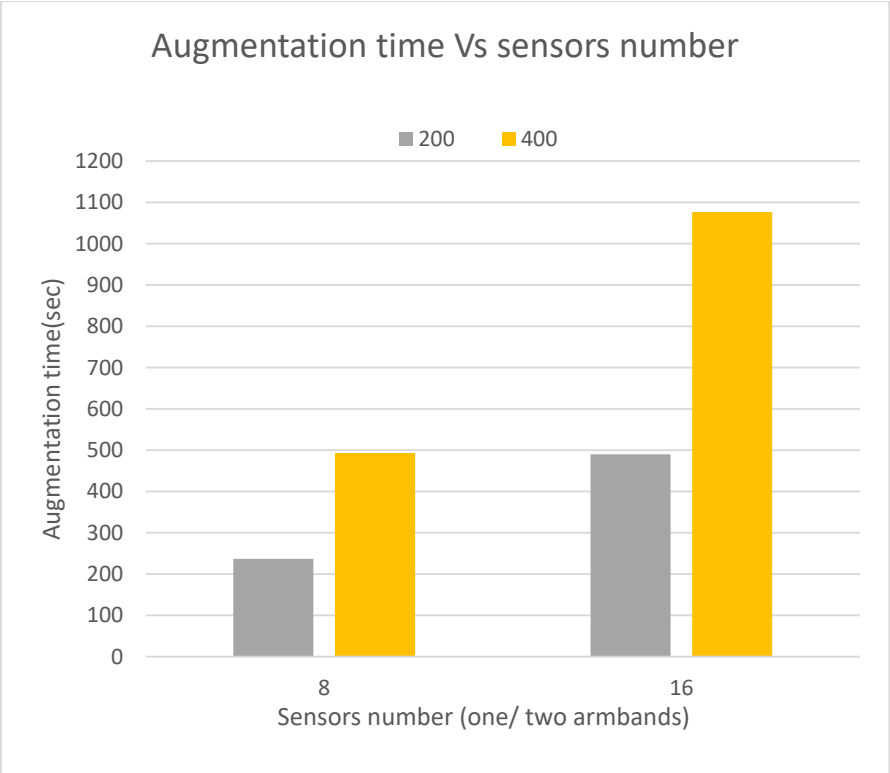

Fig. 7. Time required to generate new samples averaging every two samples with the same label, when signal length is 200 or 400 points

7. Training

We Trained the model using the GPU with the XGBoost (an option in the Python package), and the gbtree option for the Booster. Different Experiments were done using different number of features (signal length * number of sensors).

We used four evaluation metrics:

- Accuracy: number of right predictions to the total number of the test samples
- Precision, Recall and F_score.

8. Results

Table 2 shows the test results of three trained models, the first was trained using the original signals of one Myo armband, the second using sEMG signals of the same armband with the generated samples, and the third using the signals of both Myo armbands (original and augmented).

The results of the first model shows that we cannot get good results when the number of samples is less than half the number of features. The results of the second and the third are so close, the proposed method scored 98.6 %.

Table 2

Test results

Features number	Samples number	Samples to features	Test (20 % of dataset)	
600 (<900) *8	2245	0.47	14.9%	Accuracy
			12.9%	F1
			13.7%	Precision
			14.4%	Recall
600(<900) * 8	53948	11.24	98.6%	Accuracy
			98.4%	F1
			98.5%	Precision
			98.3%	Recall
600(<900) * 16	53948	5.62	98.4%	Accuracy
			98.1%	F1
			98.3%	Precision
			97.9%	Recall

Accuracy, which is a state-of-the-art result compared to other methods results on the same dataset NinaPro DB5 as shown in Table 3.

Table 3

Comparing Accuracy of the proposed framework to other methods used NinaPro DB5 dataset

Method	Number of classes	Accuracy
HVPN [15]	41	88.60%
HVPN-maxpool[15]	41	88.10%
HVPN-average[15]	41	88.10%
HVPN-2-view[15]	41	90.10%
SVM[5]	41	69.00%
ShenNet[16]	40	72.10%
MV-CNN[17]	41	90.00%
Transfer Learning[18]	53	68.98%
XGBoost	53	98.40%

We did the training with the usage of GPU, Table 4 shows the time needed, which is little concerning the big number of features considered.

Table 4

Training Time using XGBoost with GPU option. *: the considered signals are of the length in the range [600, 900]

Samples number	Features number	Time
3180	200 *8	1m6s
3180	400 *8	2m2s
2245	600 (<900) *8*	4m53s
2789	600 *8	2m44s
96990	200 * 8	4m7s
96990	400 * 8	12m37s
53948	600(<900) * 8	7m14s
76225	600 * 8	9m59s
96990	200 * 16	8m25s
96990	400 * 16	15m30s
53948	600(<900) * 16	13m34s

We could not train the model with 76225 samples where each sample has 600*16Features using the GPU option, because of the lack of needed storage space.

We recommend:

- Using the longest signals length possible so we have enough features to help the model generalize better on real signals (taking in consideration having enough number of samples).
- The model is not immune to displacement of the armband, so to use it in real applications the armband should be placed the same way described in [5].
- Using balanced classes (similar number of samples) to get better performance.

Our next step will be adapting XGBoost to handle variable lengths samples; so we do not have to cut off lots of data as we did when we shrunk the signals to one chosen length.

Conclusions

In this work, we presented a new framework for recognizing hand gestures using sEMG signals and XGBoost, we presented the results of using averaging as an augmentation method to help enlarge small datasets. We generated new samples by averaging every two samples with the same label, getting for each class $C_n^2 = \frac{n(n-1)}{2}$ new samples, where n is the number of samples of the considered class. It proved to be a simple, fast, and helpful method. We used the new dataset to train the model with XGBoost and got state-of-the-art results.

References

1. Sadikoglu F., Kavalcioglu C., Dagman B. Electromyogram (EMG) signal detection, classification of EMG signals and diagnosis of neuropathy muscle disease // Procedia Computer Science. – 2017. – Vol. 120. – Pp. 422–429. – DOI: 10.1016/j.procs.2017.11.259.

2. Reaz M.B.I., Hussain M.S., Mohd-Yasin F. Techniques of EMG signal analysis: detection, processing, classification and applications // Biol. Proced., 2006. – Vol. 8:163. – Pp. 11–35. – doi: 10.1251/bpo115.

3. Visconti P., Gaetani F., Zappatore G., Primiceri P. Technical features and functionalities of Myo armband: an overview on related literature and advanced applications of Myoelectric armbands mainly focused on arm prostheses // International Journal on Smart Sensing and Intelligent Systems. – 2018. – Vol. 11. – Pp. 1–25. – DOI: 10.21307/ijssis-2018-005.

4. Mendez I., Hansen B., Grabow Ch., Smedegaard E., Skogberg N., Uth X., Bruhn A., Geng Bo, Kamavuako E. Evaluation of the Myo armband for the classification of hand motions // Proceedings of the IEEE ... International Conference on Rehabilitation Robotics. – 2017. – Pp. 1211–1214. – DOI: 10.1109/ICORR.2017.8009414.

5. Pizzolato S., Tagliapietra L., Cognolato M., Reggiani M., Müller H., Atzori M. Comparison of six electromyography acquisition setups on hand movement classification tasks // PLOS ONE. – 2017. – Vol. 12. – e0186132. – DOI: 10.1371/journal.pone.0186132.

6. Chen Tianqi, Guestrin C. XGBoost a scalable tree boosting system. – 2016. – URL: https://arxiv.org/abs/1603.02754v3 (date of access: 11.10.2022).

7. An Guozhong. The Effects of adding noise during backpropagation training on a generalization performance // Neural Comput. – 1996 – Vol. 8(3). – Pp. 643–674.

8. Um T., Pfister F., Pichler D. Ch., Endo S., Lang M., Hirche S., Fietzek U., Kulic D. Data augmentation of wearable sensor data for Parkinson's disease monitoring using convolutional neural networks, 2017. – URL: arXiv:1706.0 (date of access: 11.10.2022).

9. Rashid Khandakar, Louis Joseph. Time-warping: a time series data augmentation of IMU data for construction equipment activity identification // International Symposium on Automation and Robotics in Construction (ISARC). – 2019. – DOI: 10.22260/ISARC2019/0087.

10. Le Guennec A., Malinowski S., Tavenard R. Data Augmentation for Time Series Classification using Convolutional Neural Networks // ECML/PKDD Workshop on Advanced Analytics and Learning on Temporal Data, Sep 2016, Riva Del Garda, Italy. ffhalshs-01357973f. – URL: https://shs.hal.science/halshs-01357973/document (date of access: 11.10.2022).

11. Petitjean F., Ketterlin A., Gancarski P. A global averaging method for dynamic time warping, with applications to clustering // Pattern Recognition. – 2011. – Vol. 44. – P. 678. – DOI: 10.1016/j.patcog.2010.09.013.

12. Forestier G., Petitjean F., Dau A., Webb G., Keogh E. Generating synthetic time series to augment sparse datasets // IEEE International Conference on Data Mining (ICDM), 2017. – Pp. 865-870. – DOI: 10.1109/ICDM.2017.106.

13. Iwana B. K., Uchida S. Time series data augmentation for neural networks by time warping with a discriminative teacher // 25th International Conference on Pattern Recognition (ICPR), 2020. – DOI: https://doi.org/10.48550/arXiv.2004.08780.

14. Iwana B. K., Uchida S. An empirical survey of data augmentation for time series classification with neural networks // PLOS ONE. – 2021. – Vol. 16. – e0254841. – DOI: 10.1371/journal.pone.0254841.

15. Wei W., Hong Hong, Wu Xiaoli. A hierarchical view pooling network for multichannel surface electromyography-based gesture recognition // Computational intelligence and neuroscience. – 2021. – 6591035. – DOI: 10.1155/2021/6591035.

16. Shen Shu, Gu Kang, Chen Xin-Rong, Yang Ming, Wang Ru-Chuan. Movements classification of multi-channel sEMG based on CNN and stacking ensemble learning // IEEE Access. – 2019. – Pp. 1-1. – DOI: 10.1109/ACCESS.2019.2941977.

17. Wei Wentao, Dai Qingfeng, Wong Yongkang, Hu Yu, Kankanhalli Mohan, Geng Weidong. Surface-electromyography-based gesture recognition by multi-view deep learning // IEEE Transactions on biomedical engineering. – 2019. – PP. 1-1. – DOI: 10.1109/TBME.2019.2899222.

18. Côté A., Ulysse Fall C. L., Drouin A., Campeau-Lecours A., Gosselin C., Glette K., Laviolette F., Gosselin B. Deep learning for electromyographic hand gesture signal classification using transfer learning // IEEE transactions on neural systems and rehabilitation engineering: a publication of the IEEE Engineering in Medicine and Biology Society. – 2019. – DOI: 10.1109/TNSRE.2019.2896269. – URL: https://arxiv.org/abs/1801.07756 (date of access: 11.10.2022).