Classification Accuracy Comparison Between Machine Learning Algorithms and a Deep Learning Algorithm in Predicting Hand Gestures.

Shahed Alam, Saif Kabir, Md. Naveed Hossain, Quazi Rian Hasnaine, Md. Golam Rabiul Alam
BRAC University
Dhaka, Bangladesh
shahed.alam, saif.kabir@bracu.ac.bd

***Abstract*—In this paper four Machine Learning (ML) Algorithms have been implemented for the classification of four hand gestures using electromyography (EMG) dataset. The classifiers opted are Support Vector Machine (SVM), Random Forest (RF), Bagged tree and Extreme Gadient Boosting (XGBoost). The prediction accuracy of the machine learning algorithms were subsequently compared with Long Short-Term Memory (LSTM) which is a Deep learning based classification technique. Among the machine learning algorithms, XGBoost provided the highest accuracy of approximately 97% while LSTM provided a superior accuracy close to 99% which promises to provide the physiologically natural upper-limb movement control. In addition to the pursuit of improved accuracy in the research, the effect of removing the most noisy channel in the accuracy of the algorithms has been examined in order to decrease the volume of data processing.**

I. Introduction

Worldwide there is a mammoth number of people with amputated upper-limb conditions. In 2017, globally around 57.7 million people were living with amputated limb conditions due to injuries that resulted due to traumatic occurrences [1]. In the United States according to the data of 2020, of every 200 people a person was living with a lost limb [2]. The National Center for Health States reports exhibit that around 50,000 new hand amputated cases compound every year and among them the most common form of partial hand amputations is loss of one or more fingers [3]. Quite evidently as in most situations scientists and researchers are in the pursuit of finding a technological solution to the problem. Myoelectric prostheses is one such technology that have been designed to restore upper-limb movement. Electromyographic (EMG) signals are generated in the residual limb when muscle contractions are made. These signals are in turn detected by the electrodes that are placed in the prosthetic device socket. Upon detection of the signal, the prosthesis movement is incidented. In order to transform the signal into motion, a device controller processes and decodes the signal and consequently sends electrical signals to generate motion of the actuators [4].

Enacting dexterous hand movement is very difficult particularly because coordinating varied degrees of freedom (DOF) involved in hand movements is quite challenging. The prosthesis controlling can be categorized in two broad controlling strategies, namely, Conventional control strategies or Machine learning Control strategies [4]. Conventional control is achieved broadly in two categories 1. on/off of hand based on a minimum EMG signal value or above 2. using the proportionality concept to control the speed of the opening and closing of the hand in order to achieve a more subtle movement control. However, in this control strategy for additional joint movements more residual muscles are required to be employed [5]. Hence, the available independent number of muscle signals in the user's residual limb limits the control of each DOF [6]. In addition to limitation of the available muscle signals the inherent variability of the EMG signals also adds uncertainty in the prosthesis control. Hence, even though the conventional control is robust however, it does not provide the physiologically natural upper-limb movement control. Machine learning Control Strategies at present does not provide the robust performance compared to control strategies [5] but it promises to provide the desired physiologically natural upper-limb movement control because of the advancement in signal processing techniques, powerful processors and the enhanced battery management technologies. Hence a considerable amount of research is being undertaken in this field.

Detection of EMG signals is done by various invasive and noninvasive techniques [7]. Noninvasive technique because of the ease of implementation is used quite prevalently. In this particular paper a dataset from kaggle has been used [8]. Signals for four classes of motion have been read via the MYO armband and with the help of an app the data is extracted and stored [9]. In recent times, several techniques have been used for classification in the varied hand gestures. Support Vector Machine (SVM), Latent Dirichlet Allocation (LDA), Random forest (RF), k-Nearest Neighbors (KNN), Artificial Neural Network (ANN) have been employed for the classification of varied hand gestures and high values of accuracies have been attained ranging from 80-97% [10] [21]. This paper aspires to attain higher accuracy in order to attain the physiologically natural upper-limb movement control. In the pursuit the algorithms (SVM, RF, Bagged Tree) that are known to provide high classification accuracy have been employed and also few new algorithms such as XGBoost and LSTM. Effect of removing the most noisy channel on the overall accuracy was explored.

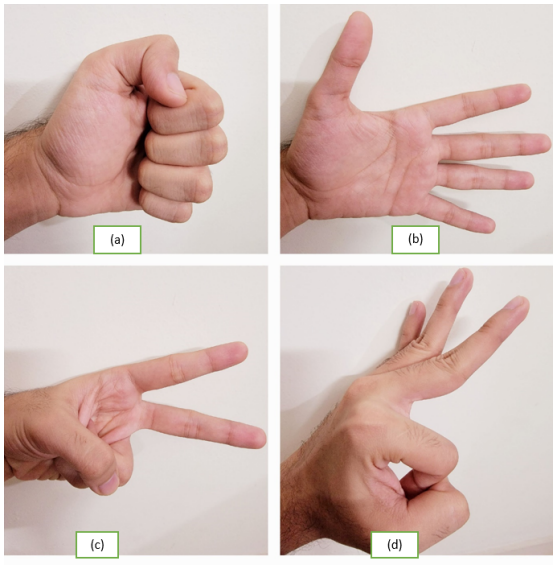

Fig. 1. Four Different Gestures of Hand Movements:Four Different Gestures of Hand Movements: (a) Rock (b) Paper (c) Scissor (d) Okay

II. Data Collection and Preprocessing

The open-source dataset was collected from Kaggle [8] provided by Kirill Yashuk, Wathek Loued and Hank Rearden. EMG data is obtained via a MYO wristband, which basically consist of a wearable gesture detection device that identifies gestures using eight sensors. In addition, the MYO wristband also comprise of gyroscope, magnetometer and accelerometers. Data is collected by these eight sensors with a signal frequency of 200Hz. Each sensor monitors the electrical activity generated by the muscles on the skin surface where it is placed and then the data is passed to an app for further processing and storing. Each row of dataset contains eight consecutive values from each of the eight sensors. As a result, there are 64 columns in the dataset. The last column identifies the gesture performed. The following gesture classes were opted: rock - 0, paper - 1, scissor - 2, and ok - 3 as shown in Fig. 1. Each gesture was recorded six times for a total of 20 seconds. Fig. 2 depicts the methodology of this paper.

III. Literature Review

Basically five algorithms are implemented in this research work. The algorithm being Support Vector Machine (SVM), Random Forest (RF), Bagged tree, Extreme Gadient Boosting (XGBoost) and Long Short-Term Memory (LSTM). However, while implementing Bagged tree algorithm, SVM, RF and pasting algorithms and methods were also employed to perform the classification of the dataset and consequently the results have been compared.

A. Support Vector Machine Classifier (SVM)

The SVM is basically a supervised learning model that attempts to separate the classes in consideration maximally. This is achieved by essentially mapping input data into separate feature spaces. The complexity of the algorithm is dependent

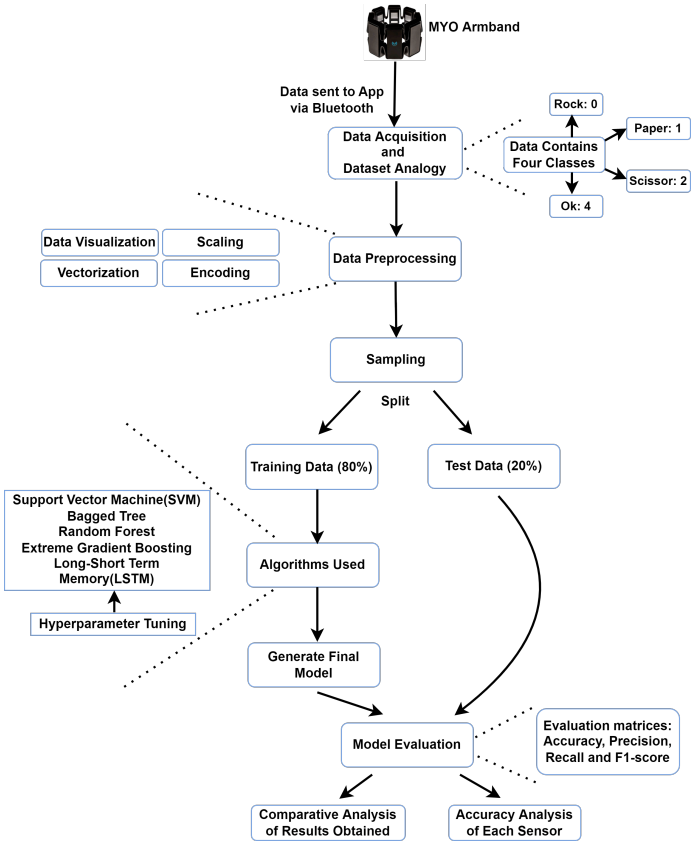

Fig. 2. Methodology

on the kernel chosen for transforming the data in the desired feature space. There are three kernel functions that are used in SVM [22]. The kernels are 1.) Linear kernel 2.) Polynomial kernel 3.) Radial basis function (RBF) kernel.In this paper RBF kernel provided better accuracy for which it has been implemented. Thus it can be concluded that the nature of data requires non-linear separation. The equation below shows the RBF function [23]:

$$K(x_n, x_i) = exp(-\gamma||x_n - x_i||^2 + C)$$

Here, $K(x_n, x_i)$, x_n, x_i, C, γ represents the RBF kernel function, support vector data, feature data, regulation or penalty for error parameter,γ parameter respectively. There were two parameters C, γ were tuned using iterative grid search and also trail and error method have been used to find the optimum value of C=10 and γ was as 'scale'. Larger values of C sets smaller margin is acceptable if the condition enables the decision function to classify the training data correctly. However, a lower value of C will result in a larger margin therefore a more relaxed decision function at the cost of training accuracy. Therefore, a trade-off needs to be struck. The behavior of the model is very sensitive to the value of the γ. The parameter γ controls the radius of the area of influence of the support vectors. If γ is too large the influence of C will be negligible, nearby points will have high influence. if γ is

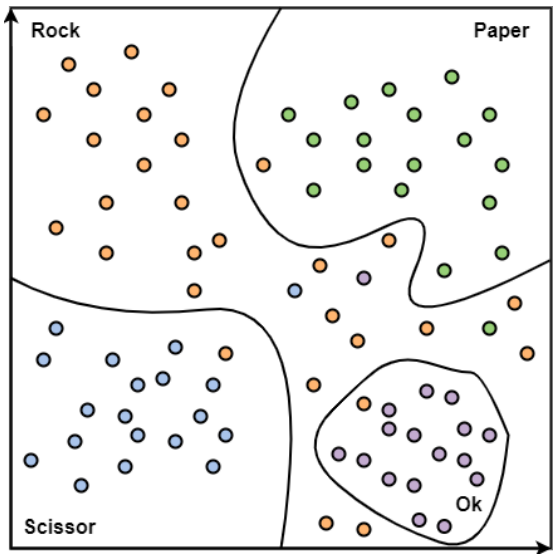

Fig. 3. Support Vector Machine Classification boundary representing RBF Kernel

too low the model will be very constrained and will not be able to capture the shape of the data [24] [25].

B. Bagged Tree

Bagging is a meta-algorithm that is used to increase the stability and accuracy of machine learning algorithms that are used in statistical classification and regression. It also helps to minimize variance and prevent over fitting by reducing the number of observations. Bagging basically reduces the size of the data and the classification is attained with the help of other machine learning algorithms. Fig. 4 portrays the classifcation process. Despite the fact that it is most often associated with decision tree techniques, other techniques can also be opted as well.

A single decision tree only has one training data set used to build the model and perform the classification of the features [27]. To elaborate on the method the steps of the process is provided below:

• In the Bagging process flow, the data is first separated into randomized samples, which is known as bootstrapping. Fig. 5 depicts the process of bootstrapping. Bootstrapping is a re-sampling strategy used in statistics and machine learning that involves repeatedly drawing samples from the source data with replacement, usually to estimate a population parameter."With replacement" actually means the same data point may appear numerous times in the re-sampled dataset.

• Each sample should be subjected to another algorithm such as Decision Trees, SVM, pasting etc. Training is done parallelly.

• Calculates the aggregated output by taking the average of all the outputs.

The accuracy in percentage is shown in the table I:

The tables-I reveals that random subspaces provided maximum accuracy 91.72%. The parameter required tuning, the base estimator value was set to 50, the n estimator value was set to 5, the random state value was set to 42, bootstrapping was false, bootstrap features was true, max samples was set to 1.0, lastly max features was 0.5. Hence, in the overall result the random subspaces accuracy is included only.

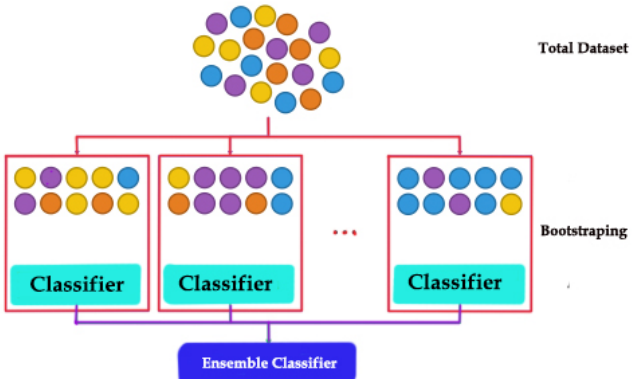

Fig. 4. Bagged Tree [26].

TABLE I
APPLIED ALGORITHMS VS ACCURACY

Serial	Name of the Methods	Accuracy
1	Decision_Tree	90.38%
2	SVM	84.86%
3	Pasting	78.25%
4	Random Subspaces	91.72%
5	Random Patches	91.09%

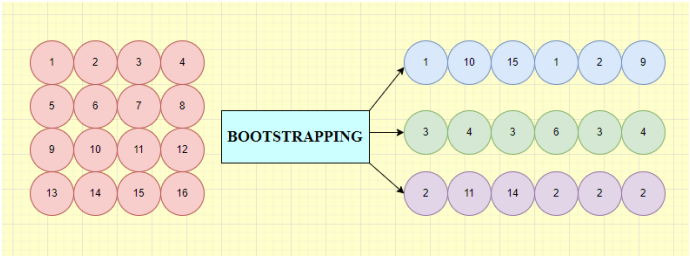

Fig. 5. Bootstrapping

C. Random Forest

The random forest method, as the name indicates, builds a forest with a large number of decision trees. It is an algorithm for supervised classification, often desired due to its fast execution speed. A random forest is formed by combining many decision trees and averaging the forecasts of each component tree to make predictions. A Decision Tree (DT) usually starts from the parent node which is the entire data-set, then divides into two data-sets known as child node. Consequently, each of these nodes will be further divided into other smaller data-sets (child node). However, random forest often outperforms a single decision tree in terms of predicted accuracy. The more trees there are in a forest, the more accurate it appears [31].

The node size, number of trees and features sampled are the main hyper-parameters that is tuned when solving regression and classification-based problems. The upgraded variant of bagged decision trees is random forest. The combined effect of several predictions from different models can improve the prediction of uncorrelated sub-models or weakly correlated prediction models. DTs find the best split to divide the data and are often trained by the Classification and Regression Tree (CART) algorithm. The difficulty with CART algorithm is that it is a greedy algorithm, therefore it chooses a splitting variable which prioritizes optimization of the current node split, without considering how that split affects the whole tree. Nevertheless, a greedy method speeds up DTs but exposes them to over-fitting i.e. the classifier memorizes the noise in the training data but fails to capture the essential patterns. [30] DTs consider all possible splits while random forests select a subset of those features. By taking into consideration all sorts of conditions under different splits, random forest reduces problems such as bias and over-fitting.

Since a random forest is made up of many DTs, each tree in the ensemble is made up of a bootstrap sample which will be utilized numerous times in a single tree as already discussed in section II. Ensemble modeling is a process in which several separate models are developed to predict a result, either via the use of many different modeling techniques or through the use of numerous training data sets. Fig. 6 shows how the algorithms are modified as sub-trees learned during the training phase, resulting in a reduced correlation between these sub-trees in their predictions [30]. Hyper parameters were tuned to obtain higher accuracy. The parameter n was set to 200 as small values such as 10 or 20 gives a lower accuracy. Similarly, the value for test size was set as 0.25.

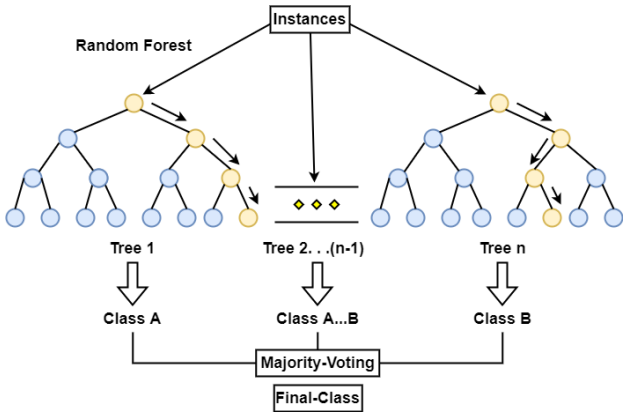

Fig. 6. Random Forest based Classification

D. Extreme Gradient Boosting (XGBoost)

Tree boosting algorithms are often used due to their efficiency and effectiveness. XGBoost is a scalable tree-boosting algorithm that is extensively utilized by data scientists to generate state-of-the-art performance on several machine learning problems. It is an ensemble algorithm based on gradient boosted trees [37] that combines the predictions of "weak" classifiers to create "strong" classifiers [32]. Features of XGBoost that accounts for its superior efficiency are:

1) Parallelization of trees: During training, tree construction is parallelized using all available CPU cores. The collection of statistics for each column is parallelized, resulting in a parallel split finding process [33]. Its ability to perform parallel processing on a single machine is what makes it fast.

2) Cache-aware Access: XGBoost is designed to use the computer hardware efficiently. CPU cache is used to store calculated gradients and Hessians (cover). This enables faster calculations to calculate the gain in each split.

3) Parallel learning using Column block: Getting the data into sorted order is the most time-consuming element of tree learning. In order to cut down on sorting costs, in XGboost, the data is stored in memory units called block where the data is kept in compressed column (CSC) format, with each column in sorted order [33]. This enables split finding and quantile calculations in all leaf branches in a single scan.

4) Out-of-core computation: The algorithm divides large data into multiple blocks for out-of-core computation and stores each block on disk space. Block Compression and Block Sharding is carried out to improve disk reading performance [33], [38], [39].

TABLE II
BEST HYPERPARAMETERS

Parameter	Value
min_child_weight	4
max_depth	5
learning_rate	0.25
gamma	0.4
colsample_bytree	0.9

The objective function of XGBoost, L , at iteration t , that needs to be minimized is :

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

Here , l is a function of CART learners, a sum of the current and previous additive trees. It is the training loss function that measures the difference between the prediction $\hat{y_i}$ and the target (real labelled value from the training data) y_i. $\hat{y_i}^t$ is the prediction of the i-th instance at the t-th iteration and everytime the algorithm will need to add f_t to minimize the objective.The second term Ω is the Regularization term that measures complexity of model (i.e., the regression tree functions) and avoid the problems of multicollinearity and overfitting [33], [34].

Hyperparameters were tuned using cross-validation techniques like RandomizedSearchCV and GridSearchCV. It involved selecting a dictionary for some parameters that will pass on the values to the function. In every iteration, the cross-validation score for different combinations of parameters were checked and the best set of parameters,that gave the highest accuracy was obtained. The best parameters are shown in table II above. Compared to other classifiers, XGBoost has a large

number of hyper-parameters that can be tuned and these makes it a complex structure.

E. Long Short-Term Memory (LSTM)

In the realm of deep learning, Long Short-Term Memory (LSTM) is essentially an artificial Recurrent Neural Network (RNN), capable of learning long-term dependencies. They are widely used as they work exceedingly well on a wide range of problems. LSTMs are specifically developed to prevent the problem of long-term dependency. Just like all other RNNs, LSTM also have a chain of repeating modules that consist of four interacting, neural network layers as shown in Fig. 7. The LSTM may erase or add information to the cell state, which is carefully controlled by gates [35].

Each line transfers data from one node's output to the inputs of others. The orange boxes represent trained neural network layers, whereas the green circles represent component-wise operations like addition and multiplication. Concatenation happens where lines merge,and forking represents the copying of content to various locations.The sigmoid layer produces values ranging from zero to one, indicating how much of each element should be allowed to pass to the next layer. To preserve and regulate the cell state, an LSTM contains three of these gates.

The primary step in the LSTM is to select which information from the cell should be excluded in a particular time step via a sigmoid function.The second layer consists of both sigmoid function and tanh function that determines which values are allowed to pass (0 or 1) and assigns weight to the passed values (-1 to 1), respectively. The final step determines which part of the current cell state gets to the output via a sigmoid function followed by a tanh function. Quite rationally, the key objective is to allow each repeating step of an RNN to pick information to examine from a huge pool of information.

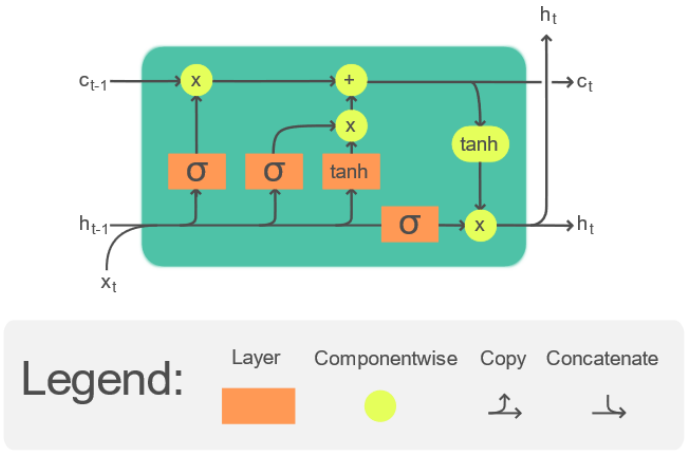

Fig. 7. Schematic of an LSTM cell [36].

IV. Results and Discussion

In this paper, for analyzing the performance, the metrics used are Accuracy, Precision, Recall and F1-Score. In all the algorithms implemented, the training and test data is split in the ratio of 80:20. The hyperparameters were tuned in all the classification models to achieve the best possible metrics.

Table III shows the precision, recall, and F1-score performance of the five classification models investigated in this study.

- For SVM, the class "Paper" has the highest recall and f1-score of 0.97 and 0.95, respectively. The highest precision of 0.94 is observed for the gesture "Scissors".
- In case of Bagged Tree, the highest precision of 0.95 is observed for the class "Paper", whereas the recall and f1-score is highest for the class "Rock", having values of 0.96 and 0.94 respectively.
- For Random Forest classifier, highest precision of 0.95 is observed for the class "Paper". Highest recall values of 0.96 and f1-score of 0.95 is obtained respectively for the class "Rock".
- In case of Extreme Gradient Boosting (XGBoost), all the performance metrics demonstrate much higher values compared to the other ML algorithms. Highest values of precision, recall and f1-score are observed for the class "Rock".
- Identical values are also observed in case of LSTM.

The accuracy of the various classifiers investigated for the proposed study is shown in Fig. 8.

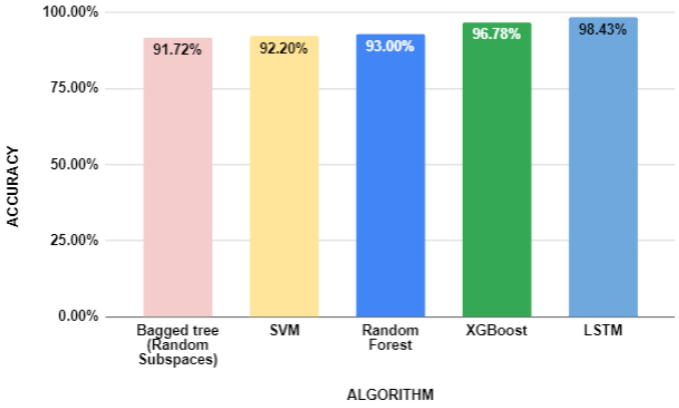

Fig. 8. Accuracy of Applied Algorithms.

The accuracy values of the Bagged tree classifier (using Random Subspaces) and the SVM classifier is 91.72% and 92.20% respectively. In case of RF and the XGBoost classifier, the accuracy is 92.64% and 96.78%, respectively. However, the highest accuracy of 98.43% is observed for LSTM. It took 40 epochs to reach this value of accuracy. Fig. 9 shows number of Epochs versus accuracy dependency for both training and validation dataset. XGBoost possess the potential of outperforming the other ML algorithms since it can create classifiers sequentially and thus reduce the classification errors. For any algorithm, finding the optimum values of weights that minimizes prediction error is the key to obtain an good classifier with precise predictions. And in case of deep neural networks, this is accomplished via the backward

TABLE III
PRECISION, RECALL AND F1-SCORE OF APPLIED ALGORITHMS

	SVM			Bagged Tree			Random Forest			XGBoost			LSTM		
Class	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
ROCK	0.91	0.94	0.92	0.92	0.96	0.94	0.94	0.96	0.95	0.99	0.99	0.99	0.99	0.99	0.99
PAPER	0.93	0.97	0.95	0.95	0.87	0.91	0.95	0.91	0.93	0.98	0.97	0.97	0.98	0.99	0.98
SCISSOR	0.94	0.92	0.93	0.91	0.92	0.91	0.92	0.95	0.94	0.96	0.96	0.96	0.97	0.96	0.97
OK	0.91	0.86	0.88	0.85	0.87	0.86	0.89	0.87	0.88	0.96	0.95	0.95	0.96	0.97	0.96

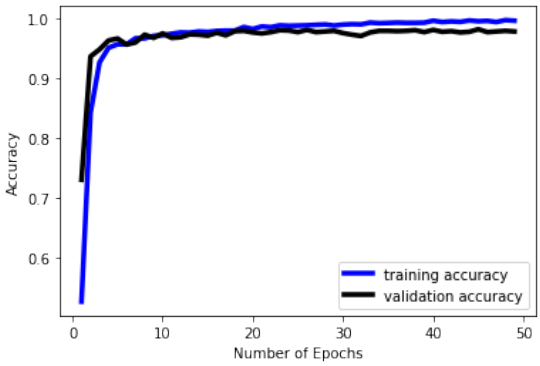

Fig. 9. Epoch vs Accuracy for LSTM

TABLE IV
ACCURACY OF EACH SENSOR

Sensor	Accuracy Obtained
Sensor 1	0.512
Sensor 2	0.507
Sensor 3	0.433
Sensor 4	0.483
Sensor 5	0.414
Sensor 6	0.503
Sensor 7	0.677
Sensor 8	0.437

propagation algorithm which gives artificial neural network an edge over ML algorithms by establishing a bi-directional feedback network [40]. LSTM gives the highest accuracy due to their ability to memorize patterns selectively over long periods of time. The underlying reason is the fact that LSTM cell enhances long-term memory by allowing the learning of even more parameters, thus enabling a better classification, especially, if the data has a longer-term trend.

Confusion matrix of XGBoost and LSTM , shown in Fig. 10 indicated that most of the errors were made in recognizing the sign “OK” and it was often misread as “Paper” or “Scissors”. For addressing the misclassifications in classes “Scissors” and “OK”, one solution might be adding another transducer or sensor in the thumb that can measure the activity of the thumb directly, which must bend when doing “Ok” (muscular contraction means higher electrical magnitude). “Paper” has all fingers open (relaxed) and thus thumb will have low magnitude. Hence, addition of another sensor near the thumb might considerably improve our prediction accuracy even more.

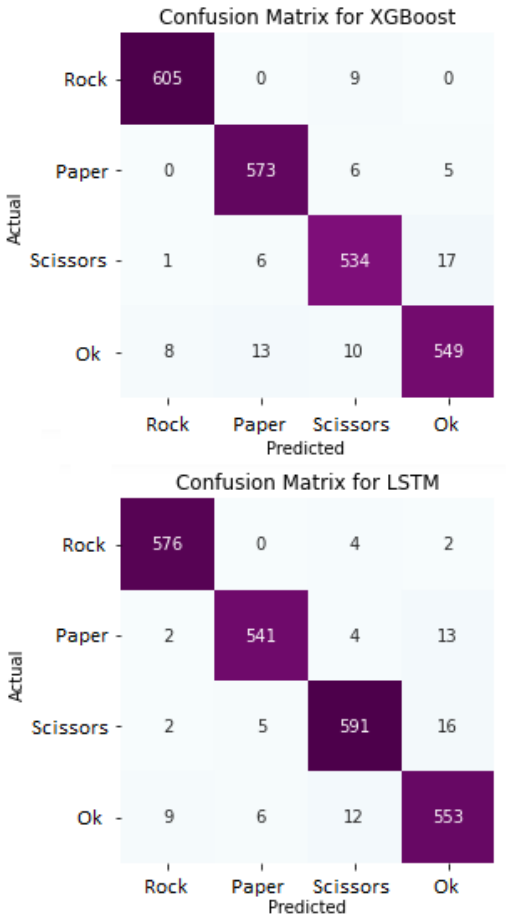

Fig. 10. Confusion Matrix for XGBoost and LSTM

V. ACCURACY ANALYSIS OF DIFFERENT CHANNELS

In order to investigate if there is any further improvement in accuracy when the noisiest channel is removed , the accuracy of each sensor was calculated. XGBoost classifier was used since it outperformed the other ML classifiers. Sensor 7 is most accurate whereas sensor 5 is least accurate. As a result, columns containing data of Sensor 5 were filtered out to check if accuracy improves after removal of noisiest channel. Table IV above shows the accuracy of each channel.

When the data of Sensor 5 was dropped, the overall accuracy was reduced to 95.03% whereas the accuracy is 97% when all the sensors are used. Hence, the results show that removal of noisiest channel does not give superior accuracy, but it is able to attain close values when all the channels are

considered. Hence it can be considered that all the channels are important and carries relevant information for classification.

VI. Comparative Analysis

The comparison table depicts that SVM six times were able to provide the highest accuracy ranging from 92.4% to around 97%. Quite recently in [21] the XBoost provided the highest accuracy when compared with K-Nearest Neighbors, Decision Tree and Random Forest classifiers in dealing with the same dataset. [21] agrees with the results attained in this research in respective to the ML algorithm that gave the maximum accuracy which is XGBoost. However, the inclusion of the Deep Learning algorithm LSTM has provided a much better accuracy than XGBoost. Table V shows that LSTM provided the maximum accuracy compared to the 12 research work that is cited [10]- [21].

TABLE V
Comparison Table

Paper Reference	Gestures	Classifier	Accuracy	Year
[10]	5	SVM	92.4%	2017
[11]	17	SVM	96.8%	2018
[12]	6	Neuro-Fuzzy	96%	2007
[13]	5	SVM	92%	2017
[?]	9	LDA	91.95%	2017
[14]	5	K-NN	86%	2017
[15]	7	SVM	95.26%	2019
[16]	50	SVM	95%	2017
[17]	48	C 4.5	97.11%	2018
[18]	6	LDA	92.8%	2020
[19]	4	QDA	83.9%	2021
[21]	5	XGBoost	85%	2022
This Study	4	XGBoost, LSTM	96.7%, 98.4%	2022

VII. Conclusion

The primary objective of this work was to compare the accuracy of machine learning algorithms with a deep learning algorithm (LSTM). In addition to that,the research also pursued very high accuracy, observed the effect of different algorithms on Bagged tree accuracy and also the effect of the most noisy channel on the overall accuracy. With the ML algorithm, the highest accuracy of 96.7% has been achieved using XGBoost. However, almost 99% accuracy has been attained using deep learning. The paper was able to highlight that DL algorithm is capable of providing very close to 100% accuracy which promises to provide the aspired physiologically natural upper-limb movement control.

References

[1] McDonald C. L., Westcott-McCoy S., Weaver M. R., Haagsma J., Kartin D. Global prevalence of traumatic non-fatal limb amputation. Prosthetics and Orthotics International. 4 December 2020.

[2] Diane W. B., Jennifer N. Y. M., "Upper Limb Amputations," Essentials of Physical Medicine and Rehabilitation(Fourth Edition),Elsevier,2020,page 651-657.

[3] Limb Loss Stats "https://u.osu.edu/wheelbarrows/upper-limb-injury-statistics/"

[4] Shehata A. W., Williams H. E., Hebert J. S. and Pilarski P. M., "Machine Learning for the Control of Prosthetic Arms: Using Electromyographic Signals for Improved Performance," in IEEE Signal Processing Magazine, vol. 38, no. 4, pp. 46-53, July 2021, doi: 10.1109/MSP.2021.3075931.

[5] Kyranou I., Vijayakumar S., and Erden M. S., "Causes of performance degradation in non-invasive electromyographic pattern recognition in upper limb prostheses," Front. Neurorobot., vol. 12, p. 58, Sept. 2018.

[6] Scheme E. and Englehart K., "Electromyogram pattern recognition for control of powered upper-limb prostheses: State-of-the-art and challenges for clinical use," J. Rehabil. Res. Dev., vol. 48, no. 6, pp. 643–659, 2011.

[7] Li, G.F.; Jiang, D.; Zhou, Y.L.; Jiang, G.Z.; Kong, J.Y.; Manogaran, G. Human lesion detection method based on image information and brain signal. IEEE Access 2019, 7, 11533–11542.

[8] Yashuk, K., Loued A., Rearden, H., Online accessed 2022 https://www.kaggle.com/datasets/kyr7plus/emg-4

[9] Enshov, K., Bulgahov, M., Perikov, I., Yashuk, K., Online accessed 2022 https://github.com/cyber-punk-me/nukleos

[10] Krishnan, K.S., Saha, A., Ramachandran, S., Kumar, S., 2017. Recognition of human arm gestures using Myo armband for the game of hand cricket. In: 2017 IEEE International Symposium on Robotics and Intelligent Sensors (IRIS). IEEE, Ottawa, ON Canada, pp. 389–394.

[11] Phinyomark, A., Khushaba, R.N., Scheme, E., 2018. . Feature extraction and selection for myoelectric control based on wearable EMG sensors. Sensors (Switzerland)18 (5).

[12] Khezri, M., Jahed, M., Sadati, N., 2007. Neuro-fuzzy surface EMG pattern recognition for multifunctional hand prosthesis control. In: 2007 IEEE International Symposium on Industrial Electronics. IEEE, Vigo, Spain, pp. 269–274.

[13] Morales, L., Cepeda, J., 2017. Feature Extraction from sEMG of Forearm Muscles, Performance Analysis of Neural Networks and Support Vector Machines for Movement Classification. In: Proceedings of the 14th International Conference on Informatics in Control, Automation and Robotics. SCITEPRESS - Science and Technology Publications, Madrid, Spain, pp. 254–261.

[14] Benalcazar, M.E., Jaramillo, A.G., Jonathan, Zea, A., Paez, A., Andaluz, V.H., 2017. Hand gesture recognition using machine learning and the Myo Armband. In: 2017 25th European Signal Processing Conference (EUSIPCO). IEEE, Kos, Greece,pp. 1075–1079.

[15] Raurale, S.A., McAllister, J., del Rincon, J.M. Real-time embedded EMG signal analysis for wrist-hand pose identification. IEEE Trans. Signal Process. 2020, 68, 2713–2723.

[16] Pizzolato, S., Tagliapietra, L., Cognolato, M., Reggian, M., Muller, H., Atzori1, M. Comparison of six electromyography acquisition setups on hand movement classification tasks. PLoS ONE 2017, 12, e0186132.

[17] Ka ´ntoch, E. Recognition of sedentary behavior by machine learning analysis of wearable sensors during activities of daily living for telemedical assessment of cardiovascular risk. Sensors 2018, 18, 3219.

[18] Cao, T., Liu, D., Wang, Q., Bai, O., Sun, J. A wearable and portable real-time control gesture recognition system for bionic manipulator. J. Phys. Conf. Ser. 2020, 1549, 52060.

[19] H. Javaid et al., "Classification of Hand Movements Using MYO Armband on an Embedded Platform", Electronics, vol. 10, no. 11, p. 1322, 2021. Available: 10.3390/electronics10111322.

[20] M. Hasan, M. Islam, M. Zarif and M. Hashem, "Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches", 2022.

[21] P. Geethanjali, "Myoelectric control of prosthetic hands: State-of-the-art review," Med. Devices Evid. Res., vol. 9, pp. 247–255, July 2016.

[22] M. A. Oskoei and H. Hu, "Support vector machine-based classification scheme for myoelectric control applied to upper limb," IEEE Trans. Biomed. Eng., vol. 55, no. 8, pp. 1956–1965, 2008.

[23] A. Dellacasa Bellingegni, E. Gruppioni, G. Colazzo, A. Davalli, R. Sacchetti, E. Guglielmelli, and L. Zollo, "NLR, MLP, SVM, and LDA: A comparative analysis on EMG data from people with trans-radial amputation," J. Neuroeng. Rehabil., vol. 14, no. 1, p. 82, 2017. doi: 10.1186/s12984-017-0290-6.

[24] https://scikit-learn.org/stable/autoexamples/svm/plotrbfparameters

[25] https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms

[26] Pawa, "Bagging", https://www.geeksforgeeks.org/xgboost/, 24 Oct, 2021

[27] Vihar K., "Introduction to Bagging and Ensemble Methods" https://blog.paperspace.com/bagging-ensemble-methods/.

[28] Dietterich, Thomas G. 2000b. "Ensemble Methods in Machine Learning." In International Workshop on Multiple Classifier Systems, 1–15. Springer.

[29] Harrison Jr, David, and Daniel L. R. 1978. “Hedonic Housing Prices and the Demand for Clean Air.” Journal of Environmental Economics and Management 5 (1). Elsevier: 81–102.

[30] Javaid H. E. A., “Classification of Hand Movements Using MYO Armband on an Embedded Platform”, Electronics, vol. 10, no. 11, p. 1322, 2021. Available: 10.3390/electronics10111322.

[31] Hasan M.,Islam M.,Zarif M., and Hashem M., ”Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches”, 2022.

[32] P. Li. “Robust Logitboost and adaptive base class (ABC) Logitboost” In Proceedings of the Twenty-Sixth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI’10), pages 302-311, 2010.

[33] Chen, T., and Guestrin, C. (2016), “XGBoost: A Scalable Tree Boosting System” , Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785–794). New York, NY, USA: ACM. https://doi.org/10.1145/2939672.

[34] Zhang T. and Johnson R., “Learning nonlinear functions using regularized greedy forest”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 36(5), 2014

[35] Christopher O., Colah’s blog, “Understanding LSTM Networks”, https://colah.github.io/posts/2015-08-Understanding-LSTMs/,27 August, 2015.

[36] Guillaume Chevalier, “https://commons.wikimedia.org/wiki/File :The_LSTM_Cell.svg”, 16 May, 2018

[37] Friedman J., “Greedy function approximation: a gradient boosting machine”, Annals of Statistics, 29(5):1189-1232, 2001.

[38] Panda B.,Herbach J. S.,Basu S., and Bayardo P. R. J., “Massively parallel learning of tree ensembles with mapreduce”, Proceeding of VLDB Endowment, 2(2):1426-1437, Aug. 2009

[39] Ye J.,Chow J. H.,Chen J., and Zheng Z.,“Stochastic gradient boosted distributed decision trees”, Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM ’09.

[40] Jahnavi M.,“Introduction to Neural Networks, Advantages and Applications”, https://towardsdatascience.com/introduction-to-neural-networks-advantages-and-applications-96851bd1a207, 10 July, 2017