Chapter 4

Leveraging cooperative theory based feature selection for hand grasp classification using the minimal number of sEMG signals

This chapter delves into the second contribution of the thesis, as detailed in Section 1.2. Section 4.1 introduces the problem and provides an overview of the proposed solution. The experimental setup (including the materials and method used) is described in Section 4.2. Section 4.4 presents the experimental results and analysis achieved with the proposed technique. Section 4.4.4.5 discusses potential threats to the validity of the findings within this chapter. Finally, Section 4.5 concludes with a summary and outlines the future scope of this chapter.

4.1 Introduction

Deploying bio-electrical signals and image processing (visual) techniques are the two popular means to provide input to generate grasp control for robotic and

prosthetic devices. Visual perception-based techniques rely on computationally expensive image processing algorithms and are sensitive to lighting conditions. In contrast, grasp control based on bio-electric signals such as surface electromyography (sEMG) is invariant to lighting conditions and can reflect human intent to hand motion or grasp with lesser computational costs. In this chapter we propose an efficient machine-learning pipeline to classify hand grasp using the optimal number of sEMG sensors. A cooperative game theory-based feature selection technique is applied to find the representative feature subset. We use a modified marginal contribution based on the class distribution coefficient to generate feature ranking. Our proposed pipeline has been evaluated on a benchmark dataset and has achieved a classification accuracy of 98.20% using single-channel sEMG signals. Experimental results show that our work outperforms the other recent techniques, showing its effectiveness in classifying sEMG-based hand grasp classification tasks. Moreover, our proposed pipeline can be used in various domains such as Neurodiagnostic, Robotics, Human-Computer Interaction, rehabilitation, and prosthetics, thereby facilitating the building of cost-effective applications.

4.1.1 Problem Statement and overview of the solution

4.1.1.1 Problem Statement

The objective of this research is to develop an accurate and cost-effective machine learning model for classifying hand grasps using surface electromyography (sEMG) signals. Given a dataset D consisting of N sEMG time series recordings corresponding to different hand grasps, where each recording can be represented as $TS_i(u) = \{u_1, u_2, u_3, \ldots, u_N\}$ the task is to accurately predict the class label of each hand grasp using these signals. The primary challenge is to leverage cooperative game theory-based feature selection techniques to identify and select high-level features, thereby enhancing the accuracy and efficiency of hand grasp classification. Additionally, the goal is to minimize the number of sensors required, making the solution suitable for deployment in prosthetic devices

4.1.1.2 Methodology for the Proposed Solution

The solution primarily focuses on feature extraction and selection using advanced techniques. Cooperative game theory-based feature selection methods are employed to assess the contribution of each feature to the model's predictive performance. This approach ensures that only the most significant features are selected, thereby enhancing the accuracy and efficiency of the hand grasp classification model. By identifying and utilizing the high-level features that contribute the most to the classification task, the solution aims to achieve high predictive accuracy while minimizing the number of sensors required. This makes the model both effective and practical for deployment in prosthetic devices.

This chapter explores the improvement of hand grasp classification through a machine learning pipeline that utilizes a minimal number of sEMG signals and cooperative theory-based feature selection. To validate the effectiveness of the proposed model, we formulated the following research questions (RQs):

1. RQ5: *How effective is the cooperative game theory-based feature selection technique integrated with machine learning in classifying different hand grasps using sEMG signals?* (Answered in Section 4.4.1)

2. RQ6: *How effective is the proposed machine learning pipeline in classifying hand grasps using the minimal number of sEMG signals compared to other existing methods?* (Answered in Section 4.4.2)

3. RQ7 *How scalable is the proposed method, and how can it be integrated into real-world applications such as neurodiagnostic, robotics, human-computer interaction, rehabilitation, and prosthetics?* (Answered in Section 4.4.3)

4.1.2 Major contribution of the work

The key contributions of this chapter are:

- We have introduced an sEMG-based hand grasp recognition (HGR) pipeline that can effectively distinguish between six different hand movements using a minimal number of sEMG sensors.
- We have employed a game theory-based feature selection approach to determine the relevant feature subset by treating the feature selection as a cooperative game with a transferable utility function.
- We incorporated weighted class distribution-based feature contribution using SHAP values to rank the features.
- We have validated the proposed pipeline by using standard performance metrics (Matthew correlation coefficient, Area Under the Curve - Receiver Operating Characteristic (AUC-ROC), confusion matrix, and cross-validation accuracy) and have obtained a recognition accuracy of 98.20% on a publicly available data set.

Figure 4.1 presents a schematic of the proposed workflow for the recognition task.

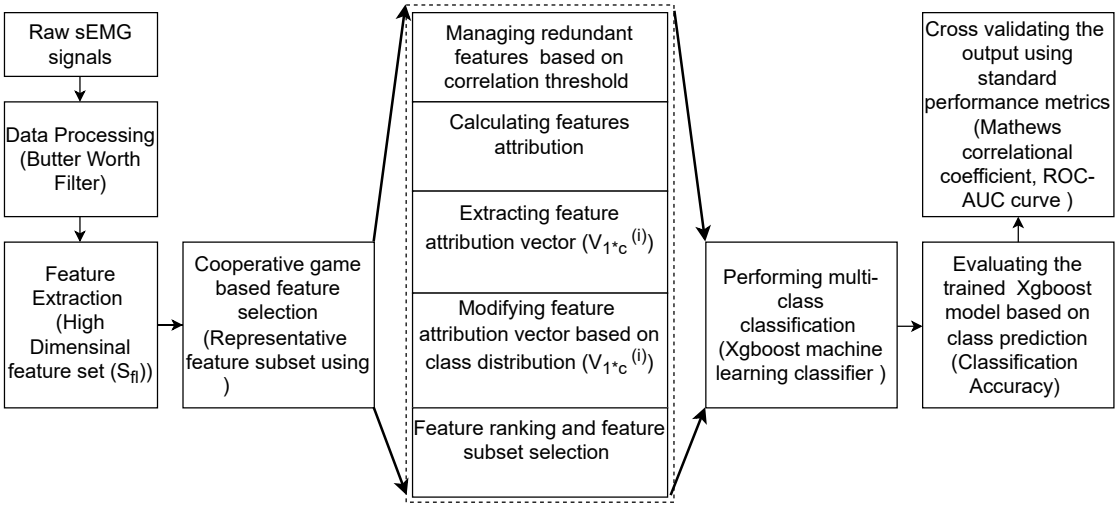

FIGURE 4.1: Schematic diagram of the pipeline used for hand grasp recognition

.

4.2 Materials and methods

4.2.1 Dataset

For experiments, we have utilized the publicly available database provided by Sapsanis et al. [153], which consists of sEMG signals generated for six basic hand movements/gasps. The sEMG signals were collected by applying two surface EMG electrodes on the muscles named Longus & Brevis and Flexor Capri Ulnaris Extensor Capri Radialis. The details of the dataset is provided in Section 2.3.5

4.2.2 Data Preprocessing

To reduce the effect of the noises added during the data collection, we have applied a digital third-order Butterworth Band Pass filter with a low and high cutoff at 15Hz and 500Hz, respectively [246] [74]. The low and high cutoff was used to deal with the motion artifact. Moreover, an additional notch filter at 50Hz was applied to eliminate the line interface.

4.2.3 Feature Extraction

The data collected from each sensor over a period of time was analyzed as a time series with k data points. Features were extracted from the time and frequency domains for a fixed window size. All of the features extracted for each hand grasp movement were consolidated into one file. Six unique classes were considered for the recognition task and each instance was assigned a distinct label for supervised classification purposes. The resulting set of labeled features for each class can be referred to as

$$Ins_i = \{f_{i,1}, f_{i,2}, f_{i,3}....f_{i,m}, l_k\} \tag{4.1}$$

Where, Ins_i is the i^{th} instance, l_k,$1 \leq k \leq 6$ is the k^{th} class label of hand grasp movement corresponding to the instance Ins_i. $f_{i,m}$ represents a feature extracted for a window size where m is the total number of features.

In our experiments, we aimed to identify the most appropriate feature set to represent the classification task. To encompass a wider range of significant features, we computed the main features listed in the literature for sEMG-based recognition of hand gestures [217]. The features obtained include those in the time, frequency, and time-frequency domains. A representative feature subset is created using a co-operative game-based feature selection. Statistical features such as Kurtosis, Mean, Variance, Absolute energy, Autocorrelation, and Standard deviation were calculated. Time domain sEMG signals were transformed into the frequency domain using the Fast Fourier Transform (FFT). Because various gestures would produce different frequency distributions so we chose 95 FFT coefficients as features [67] [218] [219]. Apart from these, other features such as Entropy, Benford Correlation, c3 nonlinearity statistics [220], Complexity-invariant distance (complexity information) [221], Mexican hat wavelet [222], Spectral centroid (absolute), skewness, and the kurtosis of the Fourier transform, Power spectral density based on Welch Method [223], Lempel-Ziv based complexity [224], One-dimensional Matrix profile [225], Partial autocorrelation, Root Mean square, Sample entropy, Friedrich coefficients [226], Absolute sum of changes (consecutive time series value changes, Langevin fixed point (Largest point of deterministic dynamics) and Qauntiles were extracted for each of the sEMG signals.

4.3 Feature selection

The feature selection problem can be described as a task of finding a suitable feature subset S_b from the extracted feature Set S_f, where $|S_b| = m$ & $|S_f| = n$, such that $n \subseteq m$ and the feature subset produce an optimal result for an evaluation function $\eta(S_b)$. Standard evaluation metrics, such as accuracy, precision, recall, and F1 score, can be used as an evaluation function.

For our case, we have used a concept from game theory to find the optimal feature subset. We initially considered the feature selection problem as a Cooperative game

with transferable utility, where each of the extracted individual features acts as a player. The prediction problem is treated as a cooperative game where the features work together to make a prediction. The prediction score is treated as the reward, and the goal is to determine the contribution of each feature to the prediction score. These players act cooperatively to achieve a common goal.

Shapley value, a solution concept in cooperative game theory, is used to ensure the fair distribution of the contribution of each feature to the prediction outcome. Shapley values assign feature attributions in a consistent and optimal way by considering all possible combinations of features. The aim of the experiment was to identify the subset of features that contribute the most to the overall performance of a machine-learning model. However, calculating exact Shapley values is computationally difficult and has been shown to be an NP-complete problem [247]. This means that finding the exact solution for the contribution of each player (or feature) in a cooperative game (or model prediction) is computationally infeasible for large games (or models). So we utilized the SHAP (SHapley Additive exPlanations) values which provide an approximation of Shapley values in computationally feasible time. SHAP values are used to rank the features based on their magnitude of contribution to the model prediction. By utilizing the ranking of the extracted features, we found the optimal feature subset for the hand grasp classification. Algorithm 6 describes the feature selection method in detail.

4.3.1 Preliminaries concepts and proposed method

The following subsections describe the background required to understand a cooperative game with a transferable utility function.

4.3.1.1 Cooperative game

A cooperative or coalitional game with transferable utility can be formally defined as a pair (X, v), where X represents a finite set with q players, and $v : 2^X \rightarrow \mathbb{R}$

is a characteristic function associated with each coalition $S \subseteq X$. The function v assigns a real-valued profit or payoff, denoted as $v(S)$, to each coalition S, representing the worth or value that the members of the coalition can divide among themselves. The characteristic function assigns a zero value to the empty set, denoted as ϕ, i.e., $v(\phi) = 0$. This property reflects the fact that an empty coalition generates no value. The payoff distribution results in a real-valued vector, where each player receives a share of the total value generated by the coalition

In a coalitional game with transferable utility, it is challenging to ascertain how payoffs should be distributed among the players. It is crucial to establish a method that is deemed "fair" to achieve an equitable allocation of each player's payoff. According to Lloyd Shapley, distributing payoffs proportional to their marginal contribution can ensure fairness in distribution.. This approach ensures that each player receives a portion that aligns with their individual impact on the overall outcome, thereby promoting fairness in the distribution process. The idea behind the method is to assign each player a value that represents their marginal contribution to all possible coalitions of players. The marginal contribution of a player in cooperative game theory refers to the change in the total payoff of the coalition (i.e., group of players) that results from the addition or removal of that player. It is calculated as the difference in the value of the coalition with and without the player divided by the number of players in the coalition. The marginal contribution, $\delta(S, j)$, of a player 'j' to a coalition is defined by the equation.

$$\delta(S, j) = [v(S \cup (j) - v(S))] \tag{4.2}$$

Where 'j' doesn't belong to S. On this basis, Lloyd Shapley proposed a new concept and termed it Shapley values. Shapley value is a method to fairly distribute the total contribution of a group of individuals to a collective outcome among the individuals themselves. This concept was first introduced in a game theory context by Lloyd Shapley in the 1950s.

The Shapley value of an individual is equal to the average marginal contribution of that individual to all possible coalitions. The value is determined by considering all possible permutations of players in a coalition and computing the contribution of each player to the total value of the coalition. It is an average of the marginal contribution of a player to all possible coalitions that they could have been a part of. The Shapley value takes into account not only the contribution of a player to the coalitions they are actually a part of but also the contribution they would have made to coalitions they were not part of. This gives a comprehensive measure of a player's overall contribution to the game. The significant advantage of Shapley values is that they can provide a fair contribution to each feature even if they contribute unequally in the collaborative game. Moreover, it can also efficiently deal with the interaction effect while evaluating the individual contribution.

The Shapley value can be computed by averaging all possible N! permutations. Shapley values for an arbitrary player 'j', in a coalitional game with transferable utility (X, v),is expressed as:

$$\varphi_j(X, v) = \frac{1}{|X|!} \sum_{S \subseteq X \setminus (j)} |S|! \, (|X| - |S| - 1)! \, [v(S \cup (j) - v(S))] \tag{4.3}$$

Basically, it assigns each player the average marginal contribution in the game. The Shapley values ensure fairness based on the following axioms.

a)Dummy player: For a coalition game with transferable utility (X, v), if a player j, does not contribute to pay off in any of the coalition, it is added. Such a player is known as a Dummy player. The Shapley value for such a player is zero.

$$(\forall X : v(X \cup (j)) = v(X)) \Rightarrow \varphi_j(X, v) = 0$$

b) Additivity: If for a coalition game with transferable utility (X, v), we can separate the game into two parts then we should be able to divide the payoff. Then

for each player, j, the Shapley value is given by the equation given below

$$\varphi_j(X, v_m + v_n) = \varphi_j(X, v_m) + \varphi_j(X, v_n)$$

Where, the game holds the characteristic $(v_m + v_n)(X) = v_m(X) + v_n(X)$.

c) Symmetry: In a coalition game with transferable utility (X, v), if for two of its players r and s, for all the coalitions formed without including these two players, $v(X \cup r) = v(X \cup s)$. Then we can say r and s are interchangeable

For any v, if players r and s are interchangeable then $\varphi_r(X, v) = \varphi_s(X, v)$

d) Efficiency: In a coalition game with transferable utility (X, v), the total payoff is distributed lossless among the players.

$$\sum_{j \epsilon X} \varphi_j(X, v) = v(X)$$

As discussed previously, computing the Shapley value for the larger dataset is NP-Complete. One of the possible ways to efficiently compute Shapley values is to use the Lundberg et al. [184] approach, which they refer to as Shapley additive explanations (SHAP). Basically, it helps to approximate the Shapley values in a computationally feasible manner.

4.3.1.2 SHAP (SHapley Additive exPlanations)

SHAP is a method generally used for explaining the output of any machine learning model. It combines the concept of Shapley values from cooperative game theory with local explanations, such as those provided by feature importance or partial dependence plots.

SHAP values are unique feature attribution values that have several attractive properties, including consistency with the model predictions, local accuracy, and the ability to handle interactions between features. For each feature, SHAP allocates a unique value, which denotes the contribution of that feature in the final prediction using a machine learning model. These values are commonly used to explain how the model achieves its prediction value and can even efficiently accommodate the interaction effect present between the features. This makes SHAP a powerful tool for feature selection, as it provides a unified way to quantify the contribution of each feature to the model's predictions. Additionally, SHAP values properties, such as local accuracy, missingness, and consistency, ensure the reliability and coherence of the feature importance measures. [184].

One key advantage of SHAP is its ability to handle complex models, including black-box models, using a unified framework. It provides model-agnostic explanations, meaning it can be applied to any model without requiring knowledge of its internal architecture. This flexibility enables SHAP to be used across various domains and machine-learning techniques.

In summary, SHAP follows additive feature attribution methods by decomposing a model's prediction into contributions from individual features using Shapley values. It considers all possible combinations of features and calculates the average marginal contribution of each feature, ensuring that the contributions are additive and providing interpretable explanations for model predictions.

Overall, SHAP's computational overhead scales linearly with the number of samples and, in its exact form, exponentially with the number of features. Nonetheless, the use of approximation techniques and model-specific optimizations allows SHAP to be applied effectively to real-world datasets. In our study, the number of features and samples were within a range that allowed the efficient use of SHAP. For larger-scale applications, additional optimizations such as limiting coalition size or

reducing feature dimensionality through preprocessing can help maintain computational feasibility.

4.3.2 Proposed feature Selection Algorithm

As outlined in the previous section, a comprehensive set of standard features commonly used in sEMG signal processing was extracted, resulting in a high-dimensional feature vector of size 623 per channel. To address the challenges posed by the curse of dimensionality and multicollinearity, a cooperative theory-based feature selection approach was employed to identify an optimal subset of informative features. This method is inspired by the work of Marcílio et al. [248].

The Algorithm 6 describes the overall description of our approach used for the feature section process. Whereas, Algorithms 7-12 provide the details of various functions used while performing feature selection.

Initially, the correlation between the extracted features was evaluated, and the highly correlated feature pairs were selected and replaced with a single feature between the two. This was done to reduce the effect of multicollinearity, which can negatively impact the model's performance and interpretability.

Multicollinearity occurs when two or more features are highly correlated, meaning they contain similar information. In such cases, it can be difficult for the model to determine which feature is actually contributing to the prediction because they are both influencing it in similar ways. Removing correlated features can help address this issue by reducing the redundant information in the data and improving the stability of the model. Algorithm 7 describes the steps involved while dealing with correlation. A threshold of 0.95 was used.

The next step was to find feature attribution, which reflects the importance or contribution of each feature to the outcome of the model. It is used to determine which features of a data sample have the most influence on the prediction made by the model. TreeSHAP was applied to evaluate the feature attribution. It results in a

matrix containing each feature's contribution for all instances. TreeSHAP is a modification of the SHAP algorithm designed specifically for tree-based machine learning models such as decision trees, random forests, and gradient-boosted trees. Initially intended as a fast, tailored alternative to KernelSHAP, it was later established that TreeSHAP could result in counterintuitive feature attributions. Algorithm 8 describes the steps used to calculate the feature attributions.

The SHAP values are a method for computing local feature attributions. In other words, SHAP values provide an explanation for the prediction made by a model for a specific data point by considering the contribution of each feature to that prediction. So, to find out a feature's importance in our dataset, we performed the column-wise summation of the feature attributions using the Matrix $X^{(i)}_{m*c}$ obtained in Algorithm 9. The column-wise summation resulted in the attribution vector $v^{(i)}_{1*c}$ for each class.

Algorithm 6 Feature Selection

Require: Labeled feature dataset $Ins = [f_1, f_2, f_3...f_n : l_i]$; $1 \leq i \leq 6$, $threshold = 0.95$

Ensure: Optimal feature subset $\tilde{f_k^\circ}$=$[f_1^{'}, f_2^{'}, f_3^{'}...f_k^{'}]$

1: $f_c \Leftarrow SEL\_UNCORR\_FEATURES(Ins, threshold)$ ▷ Algorithm 7
2: **for** Class C_i ; $1 \leq i \leq 6$ **do**
3: $W^{(i)}_{m*n} \Leftarrow FEATURE\_CONTRIBUTION(f_c^i)$ ▷ Algorithm 8
4: **end for**
5: **for** $W^{(i)}_{m*n}$; $1 \leq i \leq 6$ **do**
6: $V^{(i)}_{1*n} \Leftarrow FEATURE\_SUM(X^{(i)}_{m*n})$ ▷ Algorithm 9
7: **end for**
8: $z_i{'} \Leftarrow CALCULATE\_WEIGHT(feature\_set)$ ▷ Algorithm 10
9: $\tilde{V}^f_{1*n} \Leftarrow WEIGHTED\_FEATURE\_CONTRIBUTION(Vector, z_i)$ ▷ Algorithm 11
10: $\tilde{f_n^\circ} \Leftarrow RANK\_FEATURES(\tilde{V}^f_{1*n})$ ▷ Algorithm 12
11: $optimal\_selected\_features$ ▷ Selected features subset using shapley values

However, this feature contribution vector doesn't accommodate the class distribution information. As performing simple addition of feature attribution each class column-wise provides equal weight irrespective of the class distribution. We assume that the hypothesis is that the class with a larger number of instances should be given additional weightage. So for the same, we modified the feature attribution

vector using a coefficient calculated based on the number of instances in each Class. The algorithms 10-11 describe the calculation and implementation of this concept in detail.

The modified feature attribution vector is derived as V^{f}_{1*c}. The values of vector V^{f}_{1*c} represent the impact of each feature on the model prediction. Sorting the vector provides the list of the most influential features for the sEMG-based classification task.

We ranked these values and selected the optimal feature subset by using Algorithm 12. This approach provided the top most influential features for the classification task. However, for choosing the top-k features we performed an empirical analysis. In this, we iteratively chose the value of 'k' and evaluated the performance of the top-k features subset for the classification task. Using the trade-off between the number of features and the performance evaluation for the feature subset, the final value of k is chosen. For our dataset, we achieved maximum accuracy at k=200.

Algorithm 7 Finding Uncorrelated features

Require: Feature dataset $f_{sl} = [f_1, f_2, f_3 ... f_n]$; $1 \leq i \leq 6$; threshold δ.
Ensure: Reduced feature subset f_c
1: **function** SEL_UNCORR_FEATURES(feature set,threshold)
2: // compute correlation matrix of (feature set f_{sl}) and drop the features from the (feature set f_{sl}) based on a user-defined threshold
3: **return** features set (f_c)
4: **end function**

Algorithm 8 Finding feature contribution

Require: Labeled reduced feature dataset $f_{cl} = [f'_1, f'_2, f'_3 ... f'_c : l_i]$; $1 \leq i \leq 6$
Ensure: Feature attribution Matrix $X^{(i)}_{m*c}$
1: **function** FEATURE_CONTRIBUTION(feature set)
2: // compute features attribution matrix using SHAP Tree explainer
3: **return** Matrix $X^{(i)}_{m*c}$
4: **end function**

Algorithm 9 Extracting feature attribution vector

Require: Feature attribution Matrix $X^{(i)}_{m*c}$
Ensure: Feature attribution Vector $v^{(i)}_{1*c}$
function FEATURE_SUM(Matrix)
  for $X^{(i)}_{m*c}$; $1 \leq i \leq 6$ **do**
    for Column $C^{(i)}$; $1 \leq i \leq c$ **do**
      // perform column-wise summation
    end for
  end for
  return Vector $v^{(i)}_{1*c}$;
end function

Algorithm 10 Calculating coefficients based on class distribution

Require: Labeled feature dataset $f_{cl} = [f_1^{'}, f_2^{'}, f_3^{'}...f_c^{'} : l_i]$; $1 \leq i \leq 6$
Ensure: Coefficient z_i,$1 \leq i \leq 6$
function CALCULATE_WEIGHT(feature_set)
  $|C_i| \Leftarrow No\_of\_instances\_in\_each\_class$
  $|N| \Leftarrow Total\_no\_of\_instances\_in\_all\_classes$
  for Class C_i ; $1 \leq i \leq 6$ **do**
    $z_i \Leftarrow |C_i|$ / $|N|$
    return z_i
  end for
end function

Algorithm 11 Evaluating modified feature contribution

Require: Feature attribution Vector $v^{(i)}_{1*c}$; Coefficient z_i,$1 \leq i \leq 6$
Ensure: Modified feature attribution vector V^{f}_{1*c}
function WEIGHTED_FEATURE_CONTRIBUTION(Vector, z_i)
  $V^{f}_{1*c} = \sum_{i=1}^{6} z_i * v^{(i)}_{1*c}$
  return V^{f}_{1*c}
end function

Algorithm 12 Performing feature ranking

Require: Modified feature attribution vector V^{f}_{1*n}
Ensure: Feature Ranking; Optimal feature subset $\tilde{f_k^{\circ}}$=$[f_1^{'}, f_2^{'}, f_3^{'}...f_k^{'}]$
function RANK_FEATURES(Vector)
  Sort the element of $\tilde{V}^{f}_{1*c}$ in descending order
  Select top-k features based on empirical analysis
  return feature subset (f_k°)
end function

4.4 Results and discussions

The objective of the proposed method was to improve the classification accuracy of the sEMG-based hand grasp recognition task using an optimal number of sEMG signals. For this purpose, we separately analyzed the two channels of a benchmark dataset [153]. Channel-I was derived from the signals of the Flexor Capri Ulnaris and Extensor Capri Radialis electrodes. On the other hand, Channel-II was obtained from the signals of the Longus and Brevis electrodes.

Our proposed pipeline showed acceptable results in classifying six hand grips using a single channel. The feature selection strategy helped to obtain a feature subset with adequate discriminatory power to distinguish between different hand grasps.

4.4.1 Performance Measure

We validated our proposed pipeline using standard performance metrics such as Accuracy, MCC, F1-score, Recall, and precision. With the XGBoost classification algorithm, the proposed pipeline achieved an average classification accuracy of 87.60% and 98.2% for channel-I and channel II, respectively. Ten-fold cross-validation was used to ensure the correctness of our result. Other baseline machine-learning algorithms were also evaluated on the dataset for each sEMG signal. Table 4.1 depicts the Accuracy and average MCC scores obtained using these classifiers. While Table 4.2 highlights the other performance metrics, such as Precision(P), Recall(R), and F1-Score(F1), obtained while using our feature selection approach and XGBoost classifier for Channel-I. Similarly, Table 4.3 highlights the Precision, Recall, and F1-Score, obtained while using our feature selection approach and XGBoost classifier for Channel II.

Moreover, normalized confusion matrices were plotted for the channels classification result while considering the 80-20 train test split on the dataset. Figure 4.2 and Figure 4.3 represent the normalized confusion matrix obtained using the

XGBoost classifiers for channels I and II, respectively. Considering channel-I, normalized confusion suggests that the classifier achieves a decent prediction accuracy for most classes. However, a few instances labeled as class 1 (CL) were predicted as class 5 (SL) by our trained model. Moreover, the instances labeled as class 2 (HK) were also predicted as class 5 (SL). However, we assume that providing the model with a sufficient number of training examples could improve the efficiency of our pipeline.

TABLE 4.1: Mean Accuracy and MCC scores achieved using baseline classifiers

CH2			CH1		
Classifiers	Accuracy	Av. MCC	Classifiers	Accuracy	Av. MCC
K-nn	50.12 ± 0.68	48.68	K-nn	58.40 ± 1.10	49.28
RF	82.50 ± 1.00	80.14	RF	98.00 ± 0.50	97.39
SVM	57.13 ± 1.30	56.70	SVM	91.90 ± 0.70	87.96
Catboost	86.10 ± 1.00	83.08	Catboost	98.10 ± 0.40	97.44
XGBoost	87.60 ± 0.46	84.36	XGBoost	98.20 ± 0.36	97.80

Additionally, the ROC-AUC curve was plotted to evaluate the classification result concerning true and false positive rates. Figure 4.4 and Figure 4.5 illustrate the Micro and Macro ROC AUC curve obtained using the XGBoost classifier for channel I and channel II, respectively. The micro ROC AUC computes the aggregated True Positive Rate (TPR) and False Positive Rate (FPR) across all classes, treating all classes as one combined class. The TPR and FPR values are computed based on the total number of true positive, false positive, true negative, and false negative predictions for all classes. The resulting ROC AUC score summarizes the overall performance of the classifier across all classes. For channel I, the ROC-AUC for all six classes was reported to be 1. This leads to the micro-average and macro-average of ROC-AUC being 1.

However, for channel II, ROC for the different classes was reported to be 0.99, resulting in the micro-macro ROC curve and AUC being 0.99.

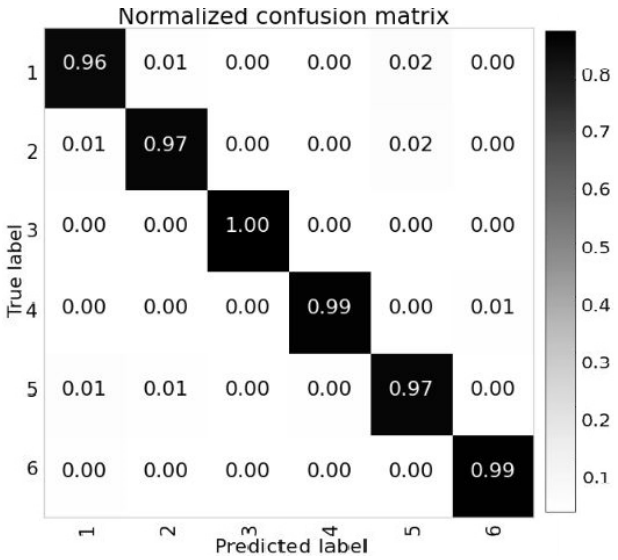

FIGURE 4.2: Normalized confusion matrix archived for Channel-I

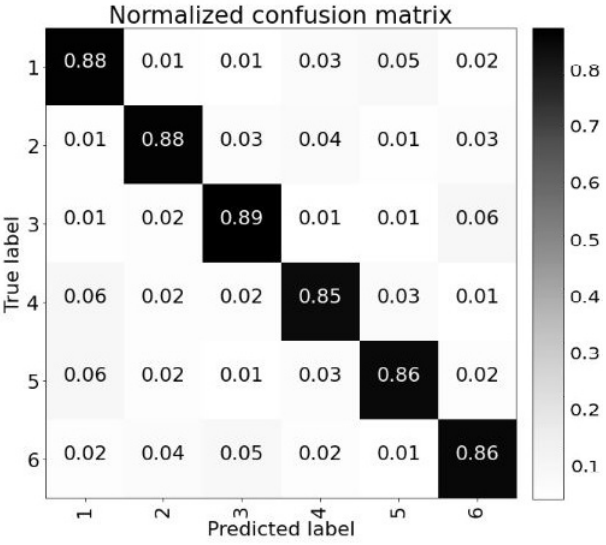

FIGURE 4.3: Normalized confusion matrix obtained for Channel-II

TABLE 4.2: Performance metrics for baseline classifiers for Channel-I

	K-NN			RF			SVM			Catboost			XGBoost		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
CL	0.46	0.66	0.55	0.98	0.95	0.97	0.86	0.87	0.86	0.97	0.97	0.97	0.97	0.96	0.97
HK	0.50	0.45	0.48	0.97	0.96	0.97	0.82	0.82	0.82	0.97	0.97	0.97	0.98	0.97	0.98
LL	0.55	0.61	0.58	1.00	1.00	1.00	0.98	1.00	0.99	0.99	1.00	1.00	1.00	1.00	1.00
PR	0.67	0.58	0.62	0.99	0.99	0.99	0.91	0.94	0.92	1.00	0.99	0.99	0.99	0.99	0.99
SL	0.81	0.66	0.73	0.93	0.98	0.95	0.85	0.80	0.82	0.95	0.96	0.96	0.95	0.97	0.96
TP	0.55	0.49	0.52	1.00	0.99	0.99	0.98	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99

TABLE 4.3: Performance metrics for baseline classifiers Channel-II

	K-NN			RF			SVM			Catboost			XGBoost		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
CL	0.77	0.57	0.65	0.88	0.79	0.83	0.78	0.53	0.63	0.87	0.85	0.86	0.88	0.86	0.87
HK	0.65	0.48	0.55	0.81	0.80	0.80	0.61	0.66	0.63	0.85	0.83	0.84	0.87	0.85	0.86
LL	0.51	0.69	0.58	0.84	0.87	0.85	0.72	0.80	0.76	0.89	0.87	0.88	0.87	0.89	0.88
PR	0.46	0.60	0.52	0.92	0.84	0.88	0.57	0.62	0.60	0.90	0.87	0.89	0.90	0.88	0.89
SL	0.66	0.62	0.64	0.76	0.88	0.81	0.61	0.63	0.62	0.82	0.86	0.84	0.84	0.88	0.86
TP	0.50	0.47	0.49	0.83	0.83	083	0.59	0.60	0.60	0.84	0.87	0.86	0.86	0.86	0.86

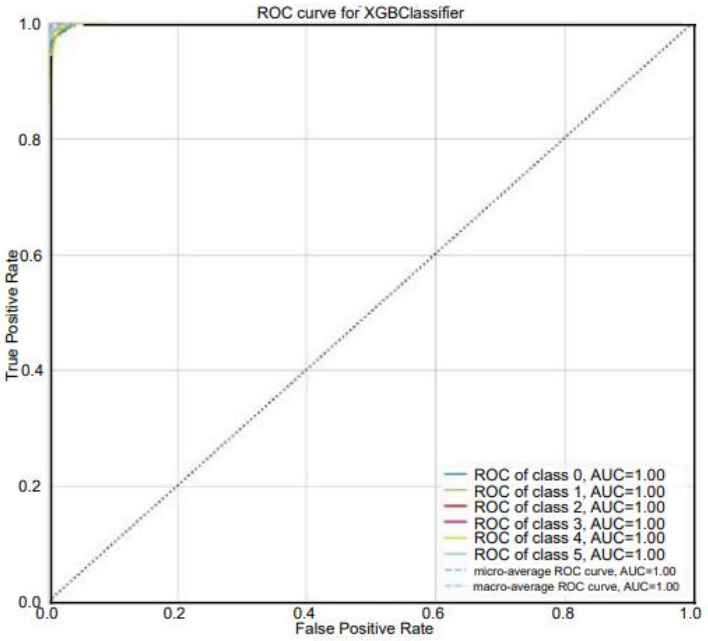

FIGURE 4.4: ROC AUC curve obtained for Channel-I

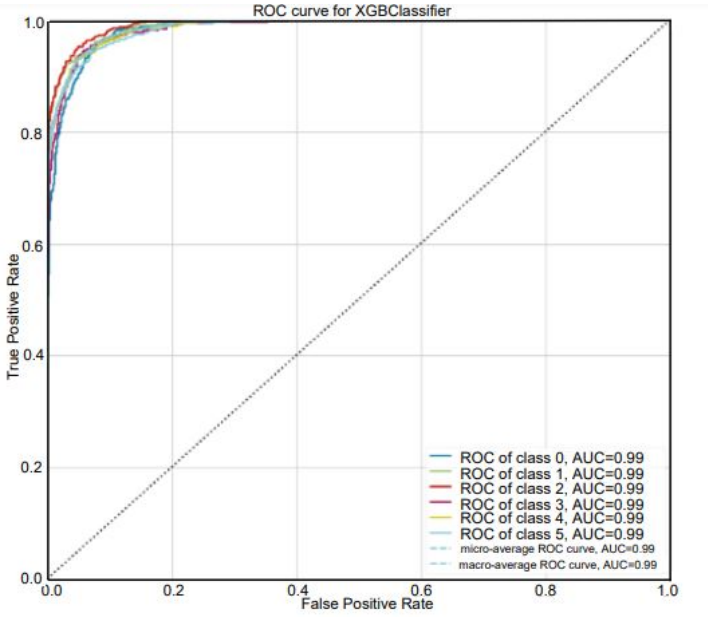

FIGURE 4.5: ROC AUC curve obtained for Channel-II

4.4.2 State-of-the-Art comparison

To show the effectiveness of our proposed pipeline, we have compared its performance with that of other published works on the same dataset. In such a work, Sapsanis et al. [148] applied a linear classifier for classifying six hand grasps using

two-channel sEMG signals. The authors applied Empirical mode decomposition for feature extraction, followed by the use of RELIEF and Principal Component Analysis (PCA) for feature reduction. Their model was able to achieve a classification accuracy of 86.64%.

Later, on a similar dataset, Zhou and Feng [155] demonstrated the efficiency of the multi-grained cascade forest approach for classifying six-hand grasp movements. This novel approach used a decision tree-based ensemble for representation learning. The authors achieved a classification accuracy of 71.3%.

Iqbal et al. [156] applied Singular value decomposition along with PCA to classify six different hand movements. The authors divided the sEMG signals into overlapping sub-frames. Using K nearest neighbor (KNN) classifier they achieved a classification accuracy of 86.71%.

Nisad et al. [157] initially applied a Tunable-Q Wavelet Transform-based (TQWT) filter bank and extracted Kraskov entropy (KRE) features from raw sEMG signals. Using the K-nearest neighbor classifier the authors achieved a classification accuracy of 98.55% across six primary hand grasp movements.

Toledo-Pérez et al. [158] utilized zero crossings feature and support vector machine to classify six different hand grasps. The authors highlighted the efficiency of the Zero crossing as a feature and its behavior with and without a threshold. Using the proposed method, they achieved a classification accuracy of 85.66%.

Recently, Coskun et al. [152] applied a deep learning model, a one-dimensional Convolution neural network to classify similar six-hand grasp movements using sEMG signals. The proposed deep learning technique automatically extracted relevant features and was further used for classification. Using the proposed architecture, the authors achieved a classification accuracy of 94.94% on a similar dataset.

The comparative performance of all the aforesaid studies has been summarized in Table 4.4. The table shows that our proposed pipeline achieves a better classification accuracy than most of the state-of-the-art methods. However, the model proposed by Nisad et al. [157], achieved a result marginally greater than our achieved accuracy. However, it is worth noting that their model utilized two-channel sEMG signals, while our proposed pipeline used only a single-channel sEMG signal for the classification task. Similarly, Baygin et al. [161] and Erazo et al. [160] reported classification accuracies of 98.89% and 99.12%, respectively, employing TQWT and Bi-LSTM architectures. These accuracies exceed ours when evaluated on the same dataset [153], but it should be noted that their methods relied on two-channel sEMG signals, unlike our single-channel approach.

Obtaining higher classification accuracy using an minimal number of sensors is beneficial as it can be used to build cost-effective systems with reduced hardware expenditure.

Table 4.4 reveals that among the studies mentioned, only Coskun et al. [152] and Alsawaf et al. [154] have utilized single-channel sEMG signals for classification purposes.

We have achieved better accuracy than both of these approaches. However, it's worth noting that the dataset employed by Alsawaf et al. [154] is related to finger/hand gestures, which differentiates it from our study. Notably, Our method demonstrates an improved classification accuracy compared to that of Coskun et al. [152].

4.4.3 Model Scalability and Training Sample Sufficiency

In a supplementary experiment to evaluate the scalability and robustness of our proposed pipeline, we generated a set of graphical representations highlighting the interrelationships among the training and cross-validation scores, the time expended for model training (fit_times), and the number of training examples. By plotting

TABLE 4.4: Summary of state-of-the-art methods for classifying different hand grasp movements.

Article	Dataset	sEMG channels	No of hand gestures	Methodology	Accuracy
[148]	[153]	2	Basic hand grasps	Linear classifier on feature extracted using Intrinsic Mode Functions.	86.64%
[155]	[153]	2	Basic hand grasps	Decision tree-based ensemble	71.30%
[156]	[153]	2	Basic hand grasps	K-nn classifier with SVD.& PCA	86.71 %
[157]	[153]	2	Basic hand grasps	K-nn classifier with TQWT.	98.55%
[158]	[153]	2	Basic hand grasps	SVM classifier with Zero crossing.	85.66%
[152]	[153]	1	Basic hand grasps	1-D CNN for classification and feature extraction.	94.94%
Proposed method	[153]	1	Basic hand grasps	Cooperative game based feature selection and XGB classifier	98.20%

these graphs, we aimed to gain insights into the performance of our pipeline in terms of its ability to handle increasing amounts of training data and the associated computational time.

The training and validation scores were calculated using accuracy (the ratio of correctly classified instances to the total number of instances) as an evaluation metric. Moreover, We also examined the relationship between the evaluation metrics (Accuracy) and the time taken to train (fit_times) our machine learning model. These results are consolidated in Figure 4.6.

In Figure 4.6 a, we present the learning curves for Channel I. Our pipeline achieved a favorable validation accuracy, exceeding 95%, even with less than 2000 samples. This indicates that the model is able to learn and generalize well from a limited amount of data. On increasing the number of training samples to 8000, the model smoothly converges towards its peak recognition accuracy. However, further increasing the training examples did not significantly impact the accuracy, and it remained stable. Similar observations were made for Channel II (Figure 4.6 b) to its reported recognition accuracy. The observed property in the learning curves highlights the efficiency and reliability of our pipeline. It demonstrates that the model is not overly sensitive to variations or fluctuations in the training data, as it maintains a consistent and high level of accuracy. This robustness is essential for real-world applications where the model needs to perform reliably even with limited or diverse datasets.

Figure 4.6 c presents the relationship between the achieved accuracy and the corresponding delay. Conversely, Figure 4.6 d provides a graphical representation of the time delay, measured in unit time, experienced during the model training process relative to the number of training instances. The data shows a nearly linear correlation between the delay and the number of samples once the 65,000-sample threshold is reached. Drawing insights from these figures (Figure 4.6 a, Figure 4.6 b, Figure 4.6 c and Figure 4.6 d), we can confidently assert that our pipeline has been trained effectively using a reasonable number of samples. Furthermore, it is less likely that the inclusion of additional training instances would have any substantial impact on the average recognition accuracy.

4.4.4 Insights on Significant Feature Contributions

In addition, we identified and plotted the most significant features using SHAP values [184]. The resulting key features are emg_friedrich_coefficients_coeff_2_m_3_r_30, emg_friedrich_coefficients_coeff_0_m_3_r_30, emg_friedrich_coefficients_coeff_1_m_3_r_30, emg_matrix_profile_feature_"mean"_threshold_0.98 and fourier_entropy_bins_2. Figure 4.7 illustrates the contribution of these features as the average impact on the model output. The details of these features are described below.

4.4.4.1 Friedrich coefficients

These coefficients are derived from the algorithm proposed in Friedrich et al. [226], which is used to analyze the time series of nonlinear systems. The algorithm extracts the stochastic and deterministic part of system dynamics using observation data. The part of the time series under consideration is fitted with the deterministic dynamics of the Langevin Model. The Langevin equation [249] [250] is a differential equation that is used to represent a wider class of dynamic systems. It is denoted by the following equations:

$$\frac{\mathrm{d}}{\mathrm{d}x} X_d(t_n) = G(X_d(t_n), t_n) + H(X_d(t_n), t_n) * \tau(t_n) \tag{4.4}$$

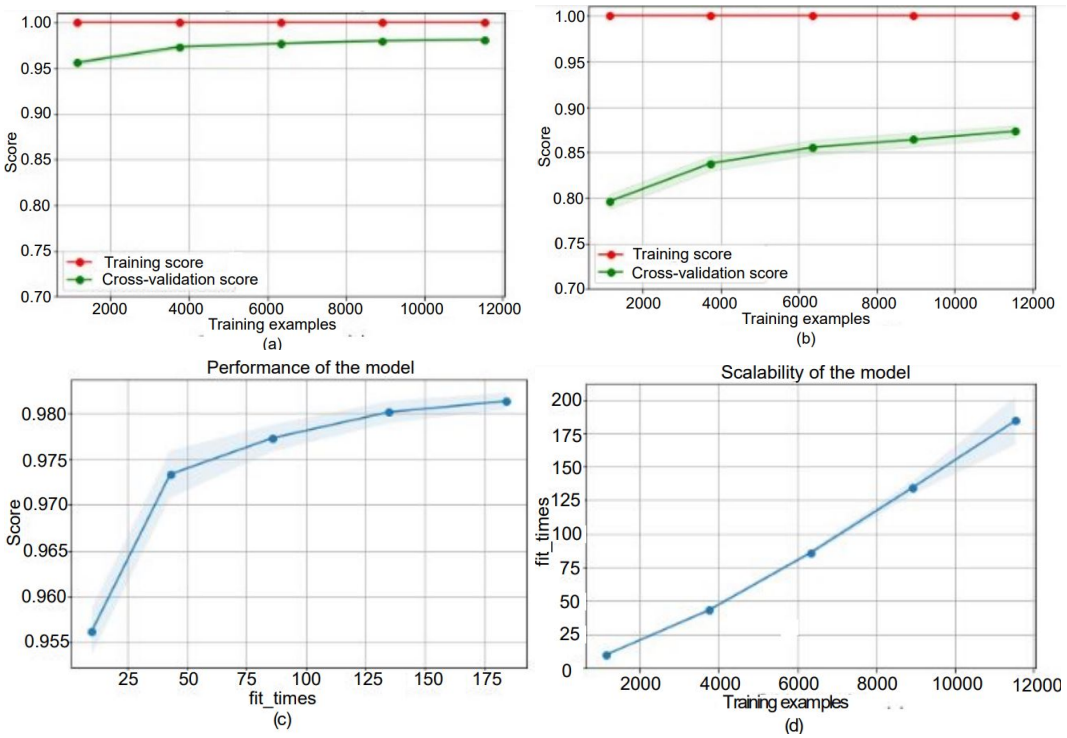

FIGURE 4.6: Various curves to access the scalability and performance of the two channels, Channel-I and Channel-II

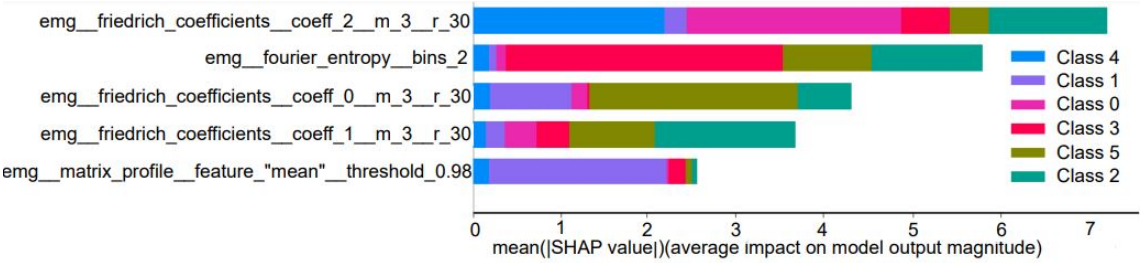

FIGURE 4.7: Figure illustrating the most significant features and their impact on model output for the six classes

Where $X_d(t_n)$ is the d-dimensional time-dependent vector. These vectors are stochastic in nature and are used to characterize the system under observation. Its derivative with respect to time is represented as a summation of $G(X_d(t_n), t_n)$ deterministic part and $H(X_d(t_n), t_n)\tau(t_n)$ stochastic part. H is a $d*d$ dimensional matrix used to deal with noise. $\tau(t_n)$ is related to white Gaussian noise. The algorithm suggests that the deterministic part can be calculated by finding the difference of different states of vector $X_d(t_n)$ and taking the average over the whole time series.

The function incorporates the polynomial order used to fit the deterministic dynamics and the number of quantiles used for averaging. Further details can be found in article [226]. The Friedrich coefficients were calculated based on the implementation given in library [251].

4.4.4.2 Fourier entropy

Fourier entropy is a feature used in time series analysis to capture the complexity and randomness of a signal. It is a measure of the entropy of the Fourier spectrum of a time series signal. In other words, it measures the disorder or randomness in the frequency components of a time series signal [252] [251].

4.4.4.3 Matrix profile

The matrix profile is a feature used in time series analysis that represents a compact summary of a time series dataset. It is a powerful tool for discovering patterns and correlations in time series data and identifying and characterizing anomalies and changes in the data [251,253].The matrix profile is a vector of distances between each data point in the time series and its nearest neighbor in the time series. The distance is computed using a suitable metric, such as Euclidean distance. The matrix profile provides a compact representation of the entire time series as it summarizes the local similarities and dissimilarities between all pairs of data points in the series.

The matrix profile can be calculated using various algorithms, including brute-force search, approximate nearest neighbor search, and index-based search, depending on the size and complexity of the time series data. The choice of algorithm and distance metric will depend on the specific needs and requirements of the application. However, we used the implementation given in the library [251].

4.4.4.4 IoT based Real-World Application Scenario

The efficiency and scalability of the system can be consistently improved through integration with IoT infrastructure. In such a scenario, the cloud maintains a copy of

the trained machine learning model. During a hand grasp, signals are generated and sent simultaneously to both the cloud and the prosthetic module. These signals are used as control signals to generate the appropriate hand grasp locally. Meanwhile, the signals sent to the cloud are stored and used to update the machine learning model, as depicted in Figure 4.8. Moreover, a centralized module can be created where sEMG signals from different locations can be uploaded to the server to accommodate the user-specific and generalized hand-grasp information. This updated model can then be downloaded to the server for future use.

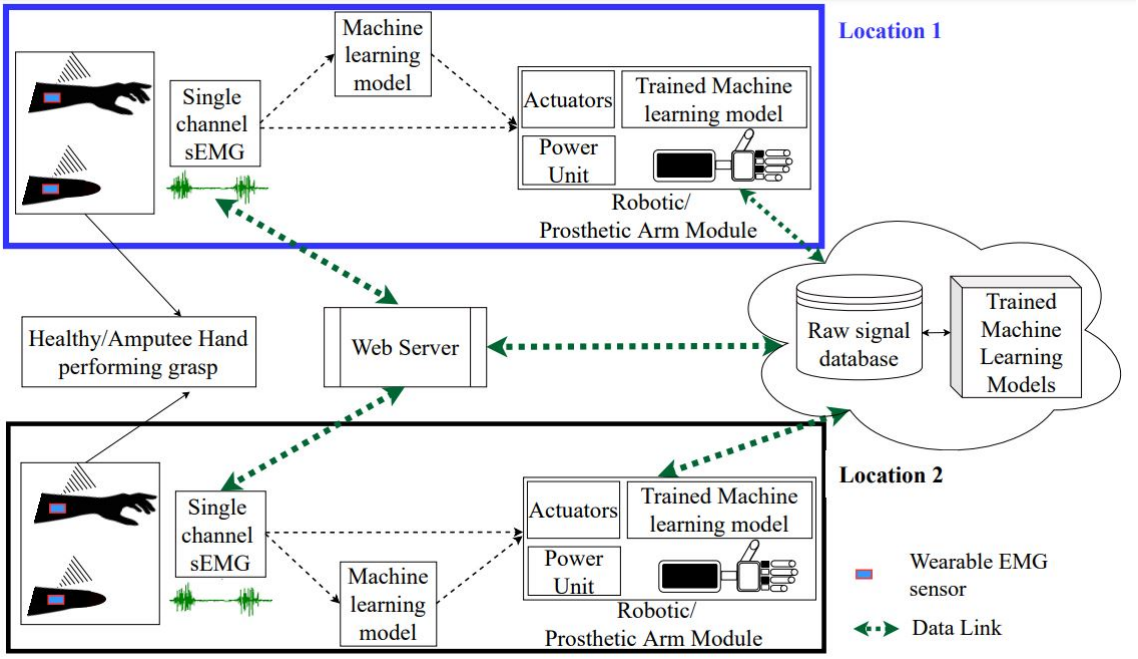

FIGURE 4.8: Real world application scenario utilizing IoT infrastructure

4.4.4.5 Threats to validity

Although the cooperative theory-based feature selection approach has demonstrated promising performance, several factors may introduce threats to the validity of the results. These are discussed below.

Internal Validity: The effectiveness of the method is strongly influenced by the quality of sEMG signal acquisition. Variability arising from sensor noise, physiological differences among users, and inconsistent recording conditions can affect the

reliability of feature extraction. While signal preprocessing mitigates some of these factors, uncontrolled variations during acquisition may still impact the consistency of the results.

External Validity: The generalizability of the approach may be limited by its reliance on single-channel sEMG data and the controlled nature of the experimental setup. Although high classification accuracy was achieved, the need for user-specific calibration and environmental tuning suggests that performance may differ across users or real-world conditions, reducing the method's applicability without further adaptation.

Construct Validity: The proposed system approaches hand grasp classification using sEMG-derived features that primarily reflect static muscle activation patterns. However, natural grasping involves dynamic coordination across multiple joints, variations in force, and context-dependent adjustments based on object properties and task intent. By focusing solely on static signal characteristics, the model may fall short in fully representing the underlying concept of functional grasp behavior, limiting its alignment with the real-world complexity of hand use.

Conclusion Validity: The performance metrics reported are based on controlled datasets and available computational resources. The complexity of the feature selection algorithm, particularly when scaling to larger datasets or more complex feature spaces, may influence reproducibility and runtime efficiency. As such, practical constraints should be considered when generalizing results to other experimental setups or applications.

4.5 Summary

This chapter presented a novel approach for static hand grasp recognition using sEMG signals, incorporating cooperative game theory for feature selection. The primary objective was to achieve high classification accuracy while minimizing the

number of sensors required, thereby creating a system that is both cost-effective and practical for real-world deployment. The proposed machine learning pipeline effectively identifies and utilizes the most informative features, enabling accurate discrimination between various static hand positions. This contributes meaningfully to the field of gesture recognition by reducing hardware complexity and enhancing user accessibility.

Key features driving the model's success include Friedrich coefficients, Fourier entropy, and matrix profile features. These features capture critical aspects of the sEMG signals: the deterministic dynamics of muscle activity, the complexity of the frequency spectrum, and local time-series patterns, respectively. Their integration significantly boosts the model's capability to distinguish between different hand gestures.

Experimental results demonstrated that the proposed method outperforms existing techniques, achieving a classification accuracy of 98.2% on a benchmark dataset using only a single-channel sEMG signal. This result emphasizes the practicality and efficiency of the approach for applications in prosthetics, robotics, and human-computer interaction.

The cooperative game theory-based feature selection method proved to be robust and scalable, adaptable to varying datasets and user conditions. By prioritizing the most impactful features, the approach supports efficient computation and maintains high performance even with limited training data. Moreover, the system's compatibility with IoT platforms suggests strong potential for real-time, intelligent, and adaptive prosthetic devices.

Despite its effectiveness, the chapter also highlights challenges such as variability in signal acquisition quality and the computational cost of feature selection methods. Addressing these limitations will be essential for improving system robustness and enabling broader application. Future work will focus on further optimization

and real-world validation to strengthen the system's performance and expand its usability.

Additionally, findings from this chapter support the effective use of Explainable AI (XAI) models—particularly SHAP (SHapley Additive exPlanations)—for feature selection. By leveraging SHAP's feature importance rankings, the system gains not only performance improvements but also enhanced interpretability.

Building on the insights gained from static gesture recognition, the subsequent chapters will shift focus toward dynamic hand gesture recognition. This presents greater complexity due to the involvement of temporal and spatial motion patterns. Addressing this challenge is crucial for enabling natural human-computer interaction and extending the system's versatility in practical, real-world environments.