IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Revue d'Intelligence Artificielle
Vol. 37, No. 4, August, 2023, pp. 985-995
Journal homepage: http://iieta.org/journals/ria

Non-Invasive Tongue-Based HCI System Using Deep Learning for Microgesture Detection

Dhuha F. Jasim*, Waleed F. Shareef

Department of Control and Systems Engineering, University of Technology-Iraq, Baghdad 10069, Iraq

Corresponding Author Email: cse.21.13@grad.uotechnology.edu.iq

https://doi.org/10.18280/ria.370420

Received: 31 May 2023
Revised: 11 July 2023
Accepted: 17 July 2023
Available online: 31 August 2023

Keywords:
human-computer interaction, microgestures, tongue gestures, wearable computing, hands-free computer interaction, deep learning

ABSTRACT

Tongue-based Human-Computer Interaction (HCI) systems have surfaced as alternative input devices offering significant benefits to individuals with severe disabilities. However, these systems often employ invasive methods such as dental retainers, tongue piercings, and multiple mouth electrodes. These methods, due to hygiene issues and obtrusiveness, are deemed impractical for daily use. This paper presents a novel non-invasive tongue-based HCI system that utilizes deep learning for microgesture detection. The proposed system overcomes the limitations of previous methods by non-invasively detecting gestures. This is accomplished by measuring tongue vibrations via an accelerometer positioned on the Genioglossus muscle, thereby eliminating the need for in-mouth installations. The system's performance was evaluated by comparing the classification results of deep learning with four widely-used supervised machine learning algorithms, namely K-Nearest Neighbors (KNN), Support Vector Machines (SVM), Decision Trees, and Random Forests. Raw data were preprocessed in both time and frequency domains to extract relevant patterns before classification. In addition, a deep learning Convolutional Neural Network (CNN) model was trained on the raw data, leveraging its proficiency in processing time series data and capturing intricate patterns automatically using convolutional and pooling layers. The CNN model demonstrated a 97% success rate in tongue gesture detection, indicating its high accuracy. The proposed system is also low-profile, lightweight, and cost-effective, making it suitable for daily use in various contexts. This study thus introduces a non-invasive, efficient, and practical approach to tongue-based HCI systems.

1. INTRODUCTION

Interaction (HCI) is an interdisciplinary field of study that investigates the dynamics of human interaction with computers. It incorporates various disciplines, including but not limited to, human factors, ergonomics, and psychology. HCI research probes diverse aspects that influence the user experience, such as technology perception, decision-making processes, and interactive visualizations.

As sensor technology continues to evolve and finds integration in an increasing number of applications, the need for user-friendly interactions becomes paramount. HCI systems have been developed to facilitate intuitive and straightforward interactions between human users and computers or other intelligent devices. By offering a user-centric interface, these systems strive to deliver an interaction experience that is both natural and accessible to the user.

The considerable advancements in sensor technology, coupled with its integration into wearable devices, have given rise to a novel interaction paradigm referred to as 'micro-interaction'. Micro-interactions can be defined as non-verbal communication methods employed in interactions with computers, smartwatches, and similar devices.

The improvement of micro-interactions with wearable devices has profound implications for user experience. Microgestures can serve as an alternative input method particularly beneficial for individuals with limited or no limb mobility, including those with quadriplegia. Moreover, for typical users, enhanced micro-interactions with wearable devices could result in a more seamless and intuitive experience.

Interest in HCI systems has surged, given their potential to significantly enhance user experience across various domains. Applications of HCI span a wide array of fields, such as medicine, education, cybersecurity, the military, and gaming, among others. The potential impact and utility of HCI systems underscore the importance of continuous research and development in this area. One of the human-computer interaction applications in the medical field is augmented and alternative communication or AAC. An AAC system aims to support individuals with speech or language impediments by providing various assistive tools, such as gestures and pictures, to facilitate effective communication. However, technical problems and poor interfaces caused to limit the adoption of such a system. To overcome these limitations, we can apply SCI system Approaches to Improve AAC systems [1, 2].

Another medical application that uses the HCI system is a human-in-the-loop system. The system is where humans oversee and control the operation, while the system provides feedback and assists humans in carrying out tasks. As the system can be utilized to help individuals with physical disabilities, as well as such systems can be utilized health monitoring and disease diagnosis [3-14].

Another application for HCI systems in developing assistive technologies for people with disabilities is the tongue computer interface or TCI. Such a system helps people with complete tetraplegia or quadriplegia to control electronic devices using tongue gestures. The main idea behind TCI is that cranial nerves innervate the tongue. Thus, it does not get affect by spinal or cord injuries. As a result, it can serve as an alternative means of Control [15-18].

In the education field HCI system is implemented by designing and developing interactive learning experiences. Therefore, HCI system applications in education can vary from interactive whiteboards that facilitate controlling collaborative learning to virtual reality systems that simulate real-world experiences for immersive learning.

As online learning platforms become more popular, improving the student experience during courses is important. Therefore, implementing HCI systems principles when designing an e-learning website can improve the student experience by optimizing user interaction and creating a tailored system that's created for users' need and preferences, moreover, by applying HCI principles lead to increased motivation and engagement of students, resulting in an improvement, learning experience [19-28].

The security research community acknowledged that human behavior is one of the main factors of security failure, where it is common to refer to humans as the weakest link in the security chain. Therefore, designers should consider user behavior to create and improve security systems. Therefore, researchers apply the knowledge and techniques of the HCI system to mitigate and resolve the issue and cyber security field. We utilize HCI systems main points when designing and developing security applications. This includes user authentication systems, intrusion detection systems, security awareness, training, and tools.

In the military application, we can utilize the principles to design and develop interactive systems that can help prove the effectiveness and efficiency of the military operation. An example of a military SCI system is comment on Control systems that enables real-time to distinguish making virtual training and unmanned aerial systems that provide remote surveillance [29-33].

Moreover, finally, we can utilize HCI principles to design and develop games, enhancing the gaming experience or design and engaging game mechanisms. There is a wide range of SCI system applications in gaming, including gesture paste, Control that enables the player to interact using hand gestures, and virtual r reality systems that immerse the player in the game environment. Moreover, by using intelligent gaming, billions, we can understand player motivation and create a tailored gaming experience that can be done by analyzing different aspects of user behavior, such as different gameplay modes or how humans interact with non-player characters [34-41].

The increasing influence of HCI systems has given rise to a novel domain known as tongue-based HCI systems, which serve as an alternative input method for individuals with limited or no limb mobility. The development of these systems presents significant challenges. The selection of appropriate sensors and sensor locations is necessary to detect tongue gestures accurately.

In this section, we viewed a brief introduction, and the next sections are organized as follows: Literature review, Methodology, Result and Discussion, and finally, Conclusions.

2. LITERATURE REVIEW

Previous research investigates several forms of human-computer interaction via wearable Technology due to its importance in improving people's life from regular use to helping people with quadriplegia or missing limb. HCI systems can be classified based on the body parts used; this study will focus on tongue and teeth gestures.

Hashimoto et al. [42] used an array of photo-reflective sensors mounted in a mouthpiece to measure the changes in distance between the tongue surface and the back of the upper teeth when the tongue moves.

Niu et al. [18] explore the tongue as an input device due to its high dexterity and resilience for people with upper limb impairment. The proposed system used a camera to identify tongue gestures.

Li et al. [43] utilized capacitive touch sensors placed on the palate to recognize tongue gestures during speech.

Jiang et al. [16] introduced a system that utilizes a multi-contact detection algorithm to identify optical distance sensors placed on the roof of the mouth, enabling the detection of tongue gestures.

The study by Anaraki et al. [2] explores the potential of using video cameras to identify the faces of children with severe disabilities and communication challenges. The objective is to help overcome the limitations faced by these children in accessing assistive technology devices.

Yamashita et al. [44] introduced a novel method for using photo-reflective sensors to detect changes in skin deformation to control input for devices. The sensors are attached to the surface of the cheeks to capture and measure these changes.

Jingu et al. [45] Design a tongue-operated input and output device that uses Capacitive touch and electrotactile stimulation to enable eyes- and hands-free interactions.

In their study, Sun et al. [46] propose an alternative means of input for various scenarios and accessibility purposes. This system involves a wearable 3D-printed earpiece with an IMU sensor and a contact microphone behind both ears. These devices work together to capture jaw movements and sound data, respectively. The team utilized a KNN algorithm with dynamic time warping to classify different gestures.

Vojtech et al. [47] Investigate in the study two input methods using a video mouse and accelerometer and sEMG as an input device, where the accelerometer is located in a headband worn by the user to detect head acceleration. At the same time, the sEMG sensor is placed on the user's face to detect facial muscle movements. Moreover, the study shows a trade-off between speed and accuracy: ACC/sEMG system provided higher target selection accuracy than Camera Mouse, while the latter provided faster target selection.

Groll et al. [48] explore an alternative computer access device using accelerometry from head tilt to control cursor movement. The sensor is placed next to the eye for a blink and above the eyebrow for a brow raise to control cursor clicks.

Previous studies have explored tongue-based HCI systems; however, existing tongue-based HCI systems often rely on invasive methods, including dental retainers, tongue piercing, and multiple mouth electrodes. Concerns arise regarding the sensing technique, as it can be obtrusive, invasive, and intrusive, causing the system to be non-hygienic and impractical for everyday use. Hence, this work aims to present a low-profile non-invasive tongue-based HCI system without compromising the Accuracy of tongue detection. by employing a deep learning algorithm.

3. METHODOLOGY

Our system was designed with a primary goal: to be user-friendly without requiring any dental retainers or other invasive methods that may be intrusive or obtrusive. Therefore, we aimed to design a lightweight and small system suitable for everyday use or as an assistive input technology for people with limited or no limb movement without compromising accuracy. Therefore, one of the critical aspects of designing the system is: Selecting a suitable sensor, choosing the sensor position which helps us achieve our vision for the system, and selecting the gestures.

Regarding the available sensing technologies, such as textile pressure, X-band Doppler, and EEG /EMG sensors, these sensors can be bulky or obtrusive, which aligns differently from our vision for the system. Therefore, we selected an accelerometer since the sensor is small, lightweight, and can be concealed easily. Therefore, it serves our vision for a non-obtrusive non-invasive, and non-intrusive system without compromising accuracy.

Moreover, when designing the system, we must choose a suitable location for the sensor. Our system vision indicates that the forehead, tongue tip, and mouth are not valid positions for the sensor to detect tongue gestures. As a result, we have selected the lower jaw on the Genoglasses muscle (on the skin) as the location for the accelerometer. This location has been shown to accurately detect tongue gestures without being invasive or obtrusive.

To complete the system design, we had to choose the appropriate gestures. Hence, we carried out a series of experiments for the selection process. We picked out five tongue movements that can be done without opening the mouth, allowing the system to be used in public areas.

After shaping our design, we conduct machine learning, coupled with time domain and frequency domain data pre-processing and deep learning classification model to classify tongue gestures and compare the result of the two methods.

3.1 Gesture selection

Gestures are a method of nonverbal communication that is powerful for expressing intentions and meanings without using spoken language. In human-computer interaction, gestures can be used as an input method for controlling smart devices. Since tongue gestures are not a standard input method for users, we need to consider various factors, such as the anatomical features of the tongue, the ease of execution for users with varying abilities, and the system's accuracy in recognizing the gestures. Moreover, researchers also consider users' preferences for specific gestures, which can influence the system's overall usability and user experience. For example, in a study by Chen, Victor, et al., taxonomies were used to describe the formation and use of micro-mouth gestures. Twenty mouth gestures were identified as suitable for everyday software application activities [49]. Therefore, in this study, we examined multiple tongue gestures as an alternative input method for our system. One of the essential points when choosing tongue gestures is simplicity, ease of memorizing and selecting gestures that do not interfere with common tongue movement.

Further, we selected five gestures as our final selected gestures aligned with our goals for the system. The chosen gestures consist of an idle gesture and two variations of vertical tongue movements performed at different speeds. These movements occur when the mouth is closed and the tongue touches the upper and lower jaws. Additionally, two gestures involve the tongue moving swiftly or slowly from left to right, contacting the interior gums. By carefully selecting these gestures, we created an effective and user-friendly tongue-based HCI system that can provide individuals with disabilities, injuries, or other limitations with a new input method for interacting with smart devices.

3.2 Designing systems and conducting experiments

In this section, we discuss the system design and experiment process.

One of the main features we aimed to achieve in this study is to design a small, lightweight system that can be used in public without the social awkwardness of a bulky device. Therefore, in our design, we selected the 3×5 mm^2 ADXL 345 accelerometer as a sensing unit since it checks all the criteria we need in our design. The sensing unit is attached to the lower jaw on the Genoglasses muscle (on the skin), making the system non-obtrusive and non-invasive to use in public. The sensing unit is attached to a 29×58 mm^2 ESP32 microcontroller that processes the data from the sensing unit. After that, the collected data were stored on the computer.

We collected data through experiments on one test subject in a lab environment to minimize the noise. The sensing unit was attached to the participant's lower jaw on the Genoglasses muscle, using a gentle, double side tape, which provided a secure and none of crucify and non-invasive connection. We designed the sensing unit to collect at the sampling rate of 100 data better second. The experiment lasted for 100 seconds, resulting in a sizable dataset of 500,000 records where each gesture consisted of 100,000 records. x

3.3 Data classification

In our study, the design system relies on tongue gesture classification to distinguish multiple gestures as an alternative input method. Therefore, we utilized machine learning, deep learning models, and feature extraction methods to achieve the highest accuracy.

We use different supervised machine learning algorithms to achieve our goal, including Support Vector Machine, K-Nearest Neighbor, Decision tree, Random Forest, and Convolution neural network. Given their demonstrated efficacy in prior research, these machine learning and deep learning models have exhibited notable classification accuracy when applied to the task of classifying tongue-based HCI systems.

Deep learning models significantly impact HCI systems by learning convolution neural networks or CNN HCI systems can extract meaningful patterns and insights from complex data, enabling more advanced and intuitive interactions between humans and computers [50].

In the initial step, the dataset preprocessed using time and frequency domain methods to extract intricate relationships at a higher level of abstraction. Following the data preprocessing stage, the dataset is partitioned into a 70% training subset and a 30% testing subset. This division ensures an appropriate allocation of data for model training and evaluation.

During the training phase, the machine learning algorithm receives the training records and corresponding class labels, enabling the model to discern the underlying patterns and relationships that differentiate each class. Through this process, the model acquires the ability to classify new records in the test dataset, even those belonging to the Unknown class, facilitating robust and comprehensive classification.

One of the supervised machine learning algorithms used in the study is K-Nearest Neighbor, a model usually used for classification and regression problems. This model relies on measuring the distance between the new input record from the test dataset and all the points in the training dataset. Further, to identify the nearest K neighbor, we use the Euclidean distance metric to calculate the nearest neighbor. The k value in this model represents the number of neighbors considered in the classification process, as shown in Algorithm 1.

Algorithm 1. KNN Algorithm

```
Input: Training samples as matrix
Output: classified Dataset where each point within the labelled class
1. Input:
  - Training dataset with features X_train and corresponding labels y_train
  - Test dataset with features X_test
  - The value of K (number of neighbors to consider)
2. For each test instance X_test_i in X_test:
  - For each training instance X_train_j in X_train:
    - Calculate the distance between X_test_i and X_train_j using Euclidean distance
    - Assign the calculated distance to the training instance X_train_j
3. Sort the training instances based on their distances in ascending order
4. Select the K nearest neighbors from the sorted training instances for each test instance X_test_i
5. Count the occurrences of each class label among the K nearest neighbors
6. Assign the class label with the highest count as the predicted label for each test instance X_test_i
7. Output the predicted labels for the test instances
```

Another popular machine-learning technique is Support Vector Machine (SVM), a supervised machine-learning algorithm often used to solve classification and regression problems. This model aims to find the ideal hyperplane to divide different classes. Moreover, this model aims to maximize the margin between classes, which is determined by measuring the distance between the nearest Datapoint in each class and the hyperplane. Furthermore, the hyperplane chooses to categorize brand-new data points according to their properties correctly. The SVM model utilizes the kernel function to map the data points into the high dimensional feature space, allowing the hyperplane to distinguish between different classes, as shown in Algorithm 2.

Algorithm 2. SVM algorithm

```
Input:
  - Training dataset with features X_train and corresponding labels y_train
  - Regularization parameter C
  - Kernel parameter gamma
2. Compute the kernel matrix K based on the training data:
  - For each pair of training instances (X_train_i, X_train_j):
  - Calculate the Gaussian (RBF) kernel value K(X_train_i, X_train_j) = exp(-gamma × ||X_train_i - X_train_j||^2)
3. Define the SVM optimization problem:
  - Initialize the weight vector w and bias b
  - Define the hinge loss function L(w, b) as per the SVM formulation
  - Define the regularization term R(w) as per the SVM formulation
  - Define the objective function J(w, b) = L(w, b) + C × R(w)
4. Solve the optimization problem to find the optimal weight vector w and bias b:
  - Use an optimization algorithm (e.g., quadratic programming) to minimize J(w, b)
  - Update w and b iteratively until convergence
5. Obtain the decision boundary and classify new instances:
  - For each test instance X_test:
    - Compute the decision function f(X_test) = sum(alpha_i × y_train_i × K(X_train_i, X_test)) + b
    - Assign the class label based on the sign of f(X_test)
      - If f(X_test) >= 0, assign the positive class label
      - If f(X_test) < 0, assign the negative class label
6. Output the predicted labels for the test instances
```

The Decision Tree model is a common type of supervised machine learning algorithm. It uses a tree-like structure to classify new data points by making decisions based on the potential outcomes. Decision nodes are used to make decisions, and they have multiple branches that lead to output in the form of leaf nodes, as shown in Algorithm 3.

Algorithm 3. Decision tree algorithm

```
1. Input:
  - Training dataset with features X_train and corresponding labels y_train
  - Maximum depth of the decision tree, max_depth
2. Define a function to build a decision tree:
  - If the stopping criteria are met:
    - Create a leaf node and assign it the most frequent class label in the current subset of training instances
    - Otherwise:
    - Find the best attribute to split the data based on a criterion (e.g., information gain, Gini index)
    - Create a new decision node for the chosen attribute
    - Split the training instances into subsets based on the attribute values
    - Recursively call the function to build a decision tree for each subset
    - Assign the decision nodes as children of the current node
3. Build the decision tree using the training dataset and the maximum depth constraint:
  - Call the function defined in step 2 to build the decision tree
4. Define a function to classify new instances using the decision tree:
  - For each test instance X_test:
    - Start at the root node of the decision tree
    - Traverse down the tree by evaluating the attribute conditions until reaching a leaf node
    - Assign the class label of the leaf node as the predicted label for the test instance
5. Classify new instances using the decision tree built in step 3:
  - Call the function defined in step 4 to classify new instances
6. Output the predicted labels for the test instances
```

This study uses another popular supervised machine learning algorithm called a Random Forest. Random forest is an effective machine learning algorithm that tackles classification and regression problems. This model utilizes multiple decision trees, improving generalizability and prediction accuracy. The main principle of random forest is to construct several decision trees where each is trained on the selection of training Dataset, then aggregate their production. Therefore, this method gets beyond some of the drawbacks of the individual decision trees, including bias or overfitting, and generates more solid and trustworthy outcomes, as shown in

Algorithm 4.

Algorithm 4. Random forest algorithm

```
1. Input:
   - Training dataset with features X_train and corresponding labels y_train
   - Maximum depth of the decision tree, max_depth
2. Define a function to build a decision tree:
   - If the stopping criteria are met:
     - Create a leaf node and assign it the most frequent class label in the current subset of training instances
     - Otherwise:
     - Find the best attribute to split the data based on a criterion (e.g., information gain, Gini index)
     - Create a new decision node for the chosen attribute
     - Split the training instances into subsets based on the attribute values
     - Recursively call the function to build a decision tree for each subset
     - Assign the decision nodes as children of the current node
3. Build the decision tree using the training dataset and the maximum depth constraint:
   - Call the function defined in step 2 to build the decision tree
4. Define a function to classify new instances using the decision tree:
   - For each test instance X_test:
     - Start at the root node of the decision tree
     - Traverse down the tree by evaluating the attribute conditions until reaching a leaf node
     - Assign the class label of the leaf node as the predicted label for the test instance
5. Classify new instances using the decision tree built in step 3:
   - Call the function defined in step 4 to classify new instances
6. Output the predicted labels for the test instances
```

Each machine learning model employed in this study exhibits distinct strengths and weaknesses. For instance, K-Nearest Neighbors (KNN) demonstrates effective handling of multi-class classification problems, yet its performance may be impacted by the curse of dimensionality when confronted with high-dimensional data. Support Vector Machines (SVM), on the other hand, possess the capacity to handle both linear and non-linear relationships between features and target variables through the use of diverse kernel functions. Nonetheless, the selection of an appropriate kernel function and the tuning of associated hyperparameters present challenges. Decision trees are adept at capturing non-linear relationships, but their sensitivity to minor variations in the training data can result in divergent tree structures. In contrast, Random Forests mitigate overfitting concerns by leveraging ensemble learning, yet their application to imbalanced datasets may introduce bias. In the results and discussion section, it becomes evident that the Random Forest model demonstrates a significant classification accuracy across various preprocessing methods.

Lastly, we utilized the convolution neural network (CNN), a deep learning model with impressive results in computer vision applications and partial image recognition. Moreover, researchers also utilized it in the time series classification of problems. In this model, the input time series data is transformed into two dimensions image-like structure where time is shown on one axis, while the values in the time series are representative on the other axis. Then the CNN model uses a sliding kernel across the time series data, set to apply filters and extract pertinent features. Further, the model uses a pulling layer to reduce the dimensionality of the relevant features, as shown in Algorithm 5.

Algorithm 5. CNN algorithm

```
Input:
   - Training dataset with features X_train and corresponding labels y_train
 2. Set hyperparameters:
   - verbose = 0
   - epochs = 10
   - batch_size = 32
3. Get the shape of the input data:
   - n_timesteps = shape(trainX)[1]
   - n_features = shape(trainX)[2]
   - n_outputs = shape(trainy)[1]
4. Create the CNN model:
   - Create an empty model object
   - Add a 1D convolutional layer with 64 filters, kernel size 3, and ReLU activation function, and specify input shape as (n_timesteps, n_features)
   - Add another 1D convolutional layer with 64 filters, kernel size 3, and ReLU activation function
   - Add a dropout layer with dropout rate 0.5
   - Add a max pooling layer with pool size 2
   - Add a flatten layer to flatten the output
   - Add a dense layer with 100 units and ReLU activation function
   - Add a dense output layer with n_outputs units and softmax activation function
5. Compile the model:
   - Compile the model using the Adam optimizer, categorical cross-entropy loss function, and accuracy as the metric
6. Fit the model to the training data:
   - Fit the model to the training data (trainX, trainy) for the specified number of epochs and batch size
   - Set verbose mode to 0 (no output during training)
7. Evaluate the model on the test data:
   - Evaluate the model on the test data (testX, testy) using the specified batch size
   - Get the accuracy of the model
8. Return the accuracy as the result of the function.
```

4. RESULT AND DISCUSSION

In this section, we present the result of our study in three-parts time domain processing and classification frequency domain, preprocessing and classification, and last convolution neural network (CNN) classification model.

The first part describes the result of preprocessing the data in the time domain and then using four different machine-learning algorithms to classify tongue gestures. In the second part, we show the preprocessing in the frequency domain, then classify it using four different machine-learning algorithms in the third part of our study. We report the results from the raw dataset as input to the CNN model.

Overall, we comprehensively analyze the performance of various machine learning and deep learning algorithms to classify tongue gestures accurately. The study demonstrates the importance of adequate data processing for feeding it to the machine. Learning algorithm further study shows the efficiency of using machine learning and deep learning algorithms to classify tongue gestures.

4.1 Frequency domain preprocessing and classification

One of the most important aspects of our study is to design an alternative tongue base input device suitable for everyday use without compromising accuracy. Therefore, one of the important steps in classifying tongue gestures using machine learning algorithms is adequate data processing. Initially, we ran experiment 5 for each gesture. In the experiment, we collected accelerometer data for 1,000 seconds, where is the sensor rate at an assembling rate of 100 per second, which resulted in a dataset of 100,000 per gesture in total; the dataset size for five gestures is 500,000. Then, we transformed the data set from Time domain to frequency domain, using Fourier transform, which decomposes time domain data into constituent frequency components this transformation allows us to extract more relevant feature from the dataset, but it resulted in complex values which are not suitable to be used in machine learning algorithms. Therefore, we used the absolute function to convert our dataset back to only real values. After feature extraction, we fed the dataset to four different machine learning which include: K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree, and Random Forest Classifier, as shown in Figure 1. Then we estimated the performance of each algorithm by using widely accepted evaluation metrics of Precision, Recall, F1 score, and accuracy, as shown in Tables 1-4.

Table 1. Accuracy matrices for the KNN classification algorithm and frequency domain pre-processing

Class	Precision	Recall	F1 Score
Idle	1.00	1.00	1.00
Horizontal fast	0.39	0.50	0.44
Horizontal slow	0.37	0.40	0.39
Vatical fast	0.43	0.39	0.41
Vatical slow	0.45	0.32	0.38
Accuracy			0.52
Weighted Avg.	0.53	0.52	0.52

Table 2. Accuracy matrices for the SVM classification algorithm and frequency domain pre-processing

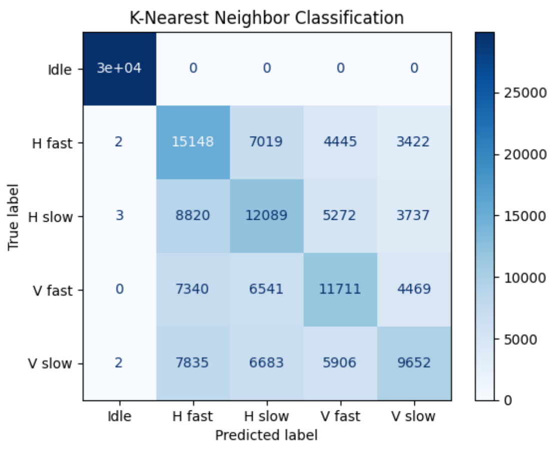
Class	Precision	Recall	F1 Score
Idle	0.92	1.00	0.96
Horizontal fast	0.50	0.52	0.51
Horizontal slow	0.48	0.45	0.47
Vatical fast	0.52	0.47	0.49
Vatical slow	0.47	0.47	0.47
Accuracy			0.58
Weighted Avg.	0.57	0.58	0.58

Table 3. Accuracy matrices for the decision Tree classification algorithm and frequency domain pre-processing

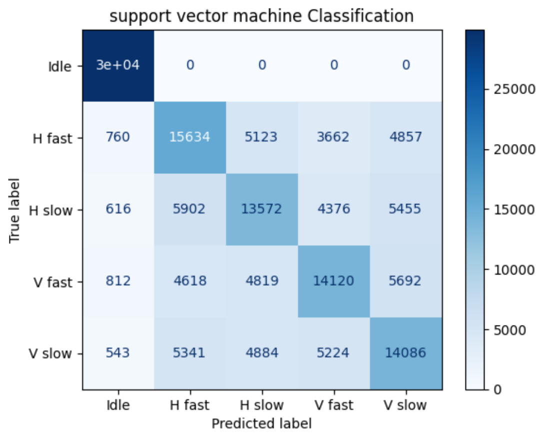
Class	Precision	Recall	F1 Score
Idle	1.00	1.00	1.00
Horizontal fast	0.79	0.79	0.79
Horizontal slow	0.78	0.78	0.78
Vatical fast	0.79	0.78	0.79
Vatical slow	0.78	0.78	0.78
Accuracy			0.83
Weighted Avg.	0.83	0.83	0.83

Table 4. Accuracy matrices for the random forest classification algorithm and frequency domain pre-processing

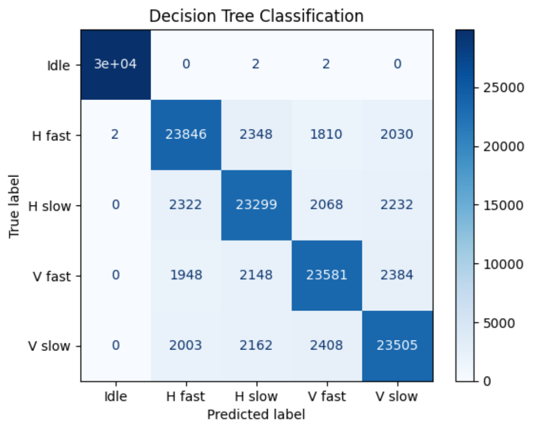
Class	Precision	Recall	F1 Score
Idle	1.00	1.00	1.00
Horizontal fast	0.79	0.81	0.80
Horizontal slow	0.79	0.78	0.79
Vatical fast	0.80	0.80	0.80
Vatical slow	0.79	0.79	0.79
Accuracy			0.84
Weighted Avg.	0.84	0.84	0.84



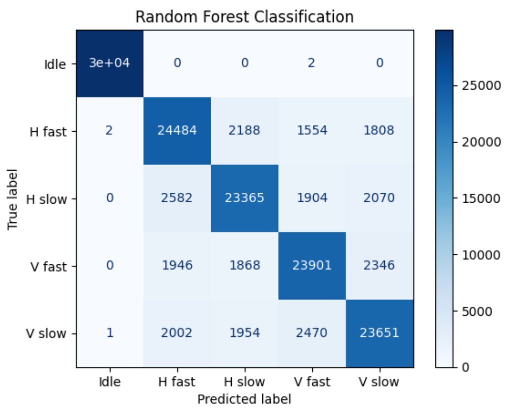
(a) K-Nearest neighbor confusion matrix



(b) Support Victor machine (SVM) confusion matrix



(c) Decision tree confusion matrix



(d) Random forest confusion matrix

Figure 1. Confusion matrix of training, validation and test

4.2 Time domain preprocessing and classification

In this study, we employed a time domain preprocessing. Furthermore, we segmented the data into chunks of 5-second intervals. Each chunk consists of 500 readings from each coordinate, namely X, Y, and Z, resulting in 1,500 features per record, with an additional feature for the class label. Our dataset originally contained 100,000 records for each class. By implementing this preprocessing technique, we reduced the number of records from 500,000 to 1,000 while generating 1,501 features, which include X0 to X499, Y0 to Y499, Z0 to Z499, and the class column.

Although we were able to make our machine-learning algorithms more efficient by reducing the amount of data inputted, we had to make a trade-off between classification accuracy and classification runtime.

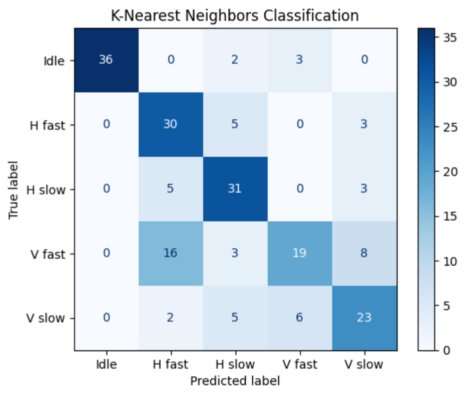
Table 5. Accuracy matrices for the KNN classification algorithm and time domain pre-processing

Class	Precision	Recall	F1 Score
Idle	1.00	0.88	0.94
Horizontal fast	0.57	0.79	0.66
Horizontal slow	0.67	0.79	0.73
Vatical fast	0.68	0.41	0.51
Vatical slow	0.62	0.64	0.63
Accuracy			0.69
Weighted Avg.	0.71	0.69	0.69

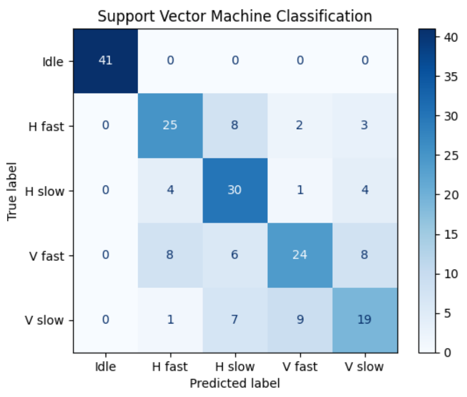
Table 6. Accuracy matrices for the SVM classification algorithm and time domain pre-processing

Class	Precision	Recall	F1 Score
Idle	1.00	1.00	1.00
Horizontal fast	0.66	0.66	0.66
Horizontal slow	0.59	0.77	0.67
Vatical fast	0.67	0.52	0.59
Vatical slow	0.56	0.53	0.54
Accuracy			0.69
Weighted Avg.	0.70	0.69	0.69

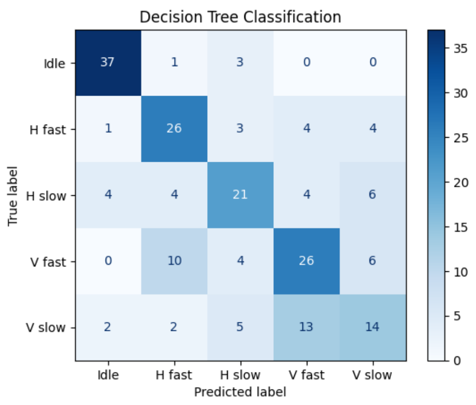
After conducting feature extraction on the raw time domain data, we used four distinct machine learning algorithms, including K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Decision Tree, and Random Forest Algorithms, to classify tongue gestures, as shown in Figure 2. Furthermore, to assess the performance of these algorithms, we compared their predictive accuracy using established evaluation metrics such as Precision, Recall, F1 score, and accuracy. Such metrics enable a comprehensive and reliable evaluation of the algorithms' effectiveness in accurately classifying the data, as shown in Tables 5-8.



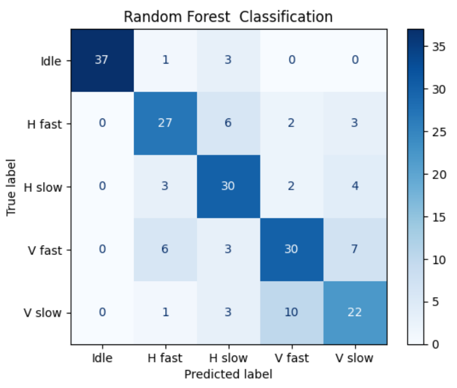
(a) K-Nearest neighbor confusion matrix



(b) Support victor machine (SVM) confusion matrix



(c) Decision tree confusion matrix



(d) Random forest confusion matrix

Figure 2. Confusion matrix of training, validation and test

Table 7. Accuracy matrices for the Decision Tree classification algorithm and time domain pre-processing

Class	Precision	Recall	F1 Score
Idle	0.84	0.90	0.87
Horizontal fast	0.60	0.68	0.64
Horizontal slow	0.58	0.54	0.56
Vatical fast	0.55	0.57	0.56
Vatical slow	0.47	0.39	0.42
Accuracy			0.62
Weighted Avg.	0.61	0.62	0.61

4.3 Deep learning CNN classifier

Recent studies have demonstrated that utilizing Convolutional Neural Networks (CNNs) for time series classification offers several advantages over alternative methods. These models exhibit remarkable resilience to noise and have the capacity to extract deep, informative features that are independent of time. In 1-D Convolution for Time Series analysis, a time series of length n and width k is considered. The length refers to the number of time steps, and the width represents the number of variables in a multivariate time series. The convolution kernels utilized always possess the same width as the time series, with the option to vary in length. This allows the kernel to move unidirectionally from the start of a time series to its end, performing convolution without movement to the left or right, as is the case with 2-D convolution applied to images. The architecture of our model is based on a convolutional neural network (CNN) with multiple layers where the CNN layers are responsible for extracting high-level features from the input data.

Table 8. Accuracy matrices for the random forest classification algorithm and time domain pre-processing

Class	Precision	Recall	F1 Score
Idle	1.00	0.90	0.95
Horizontal fast	0.71	0.71	0.71
Horizontal slow	0.67	0.77	0.71
Vatical fast	0.68	0.65	0.67
Vatical slow	0.61	0.61	0.61
Accuracy			0.73
Weighted Avg.	0.74	0.73	0.73

The model starts with two convolutional layers, each utilizing 64 filters with a kernel size of 3 and applying the ReLU activation function. These layers are responsible for extracting spatial features from the input time series. To prevent overfitting, a dropout layer with a rate of 0.5 follows the second convolutional layer, randomly dropping 50% of the connections.

Next, a max pooling layer with a pool size of 2 is added, reducing the spatial dimensions and retaining the most relevant information. The output from the pooling layer is then flattened into a one-dimensional vector using the Flatten layer, preparing it for the subsequent fully connected layers. The model continues with a dense layer of 100 units and employs the ReLU activation function. This layer learns to identify more complex patterns by combining the features extracted by the previous convolutional layers.

Finally, the model concludes with a dense layer representing the output layer, where the number of units in this layer represents the number of classes. The SoftMax activation function is applied to provide a probability distribution over the classes. As shown in Figure 3, the model consists of several CNN layers that progressively learn and represent more complex patterns. The input goes through multiple layers that apply filters to capture various aspects of the data. These layers play a crucial role in capturing spatial dependencies and ensuring accurate predictions by the model. Table 9 provides more information on the hyperparameter settings that were made during model training.

In our investigation, we leveraged CNN to classify time series data by inputting the raw dataset, which consisted of 500,000 records, for model training and testing. We evaluated the model's Accuracy ten times, calculating the average accuracy and standard deviation of the ten accuracy values obtained.

We compared different techniques for classifying time series data. We found that the CNN model is a high classification accuracy of 97% due to its ability to extract informative features independent of time, as shown in Figure 4. However, machine learning models with frequency domain preprocessing can achieve satisfactory accuracy by converting time series data to the frequency domain. Furthermore, Time-domain preprocessing produces the fastest classification runtime for identifying tongue gestures.

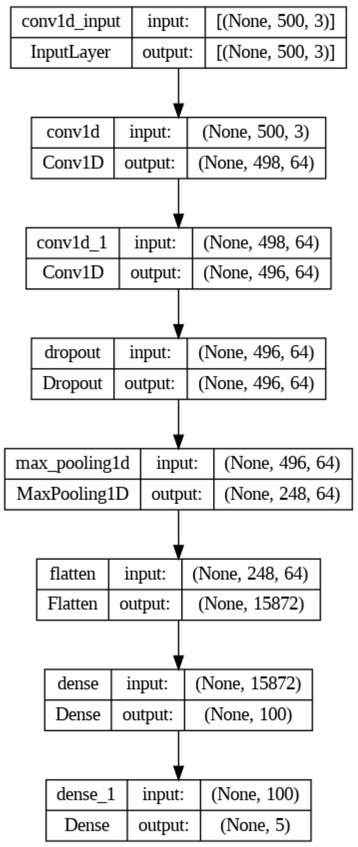

conv1d_input | input: | [(None, 500, 3)]
InputLayer | output: | [(None, 500, 3)]
conv1d | input: | (None, 500, 3)
Conv1D | output: | (None, 498, 64)
conv1d_1 | input: | (None, 498, 64)
Conv1D | output: | (None, 496, 64)
dropout | input: | (None, 496, 64)
Dropout | output: | (None, 496, 64)
max_pooling1d | input: | (None, 496, 64)
MaxPooling1D | output: | (None, 248, 64)
flatten | input: | (None, 248, 64)
Flatten | output: | (None, 15872)
dense | input: | (None, 15872)
Dense | output: | (None, 100)
dense_1 | input: | (None, 100)
Dense | output: | (None, 5)

Figure 3. Convolutional neural network (CNN) layers architecture

Table 9. CNN model hyperparameters

Hyperparameters	Value
Learning rate	0.001
Batch size	32
The hidden layer activation function	ReLU
Optimizer	Adam optimizer
epochs	10
Dropout	0.5

Table 10. Accuracy matrices for the random forest classification algorithm and time domain pre-processing

Author	Year	Input Method	Sensor Location	Classification Method	Intrusive	Invasive	Obtrusive	Accuracy
[42]	2018	An array of photo-reflective sensors	mounted in a mouthpiece	SVM	YES	NO	YES	77.50%
[18]	2019	Camera-based tongue computer interface	camera mounted on the computer	pattern recognition method	NO	NO	NO	83%
[43]	2019	Capacitive touch sensors	the roof of the mouth	SVM	YES	NO	YES	91.01%
[16]	2021	Optical distance sensors	the roof of the mouth	multi-contact detection algorithm	YES	NO	YES	86.99%
[2]	2020	Video cameras	Head	CNN	NO	NO	NO	99%
[44]	2017	Photo-reflective sensors	Head (cheaks)	SVM	NO	NO	YES	80.45%
[45]	2023	Capacitive touch and electrotactile stimulation	Head (on the lips)	centroid-based estimator	YES	NO	YES	93%
[46]	2021	Accelerometer and microphone	Head (next to the ear)	KNN WITH DTW	NO	NO	NO	90.90%
[47]	2020	Accelerometer and sEMG	head	not define	YES	NO	YES	70.30%
[48]	2020	Accelerometer and Surface Electromyography (sEMG)	Head (forehead)	not define	YES	NO	YES	95%

Ultimately, a comparison between the proposed system and existing work can be made across several categories, such as sensor type, sensor location, methodology used, system application, classification method, and the degree of intrusiveness, invasiveness, and obtrusiveness, as shown in Table 10.

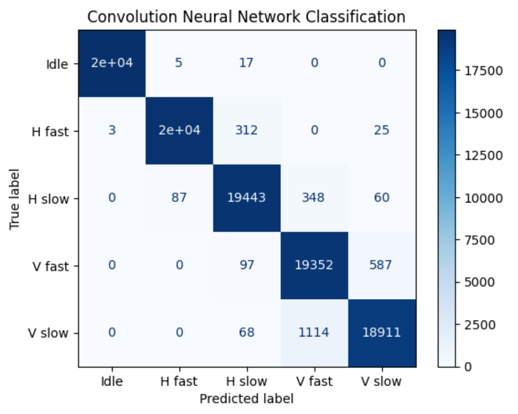


Figure 4. Confusion matrix of training, validation and test

5. CONCLUSIONS

This study explores the future of hands-free human-computer interaction utilizing tongue gestures as an alternative method of interacting with smart devices. We primarily focus on the improvements achieved by utilizing a Convolution neural network to classify tongue gestures. Our system is designed to be non-intrusive, non-obtrusive, and non-invasive, ensuring user comfort and ease of use in everyday scenarios. We detected five distinct tongue gestures using an accelerometer sensor attached to the lower jaw's Genioglossus muscle. To enhance the accuracy of gesture classification, we implemented three preprocessing methods, namely time domain and frequency domain preprocessing.

Additionally, we applied four different machine-learning algorithms. Our initial results with time domain preprocessing achieved a 73% accuracy rate, while frequency domain preprocessing improved the accuracy to 84%. However, the most significant advancement was observed when utilizing the CNN model on the raw data, resulting in an impressive accuracy of 97% in accurately classifying tongue gestures. The outcomes of our study have led to the development of an exceptionally effective HCI system. This system not only serves as an alternative input device for individuals with limited or no limb mobility but also presents a practical solution for daily use by anyone.

It is worth noticing that the data used for this study was obtained in a controlled lab environment. In future work, it would be valuable to address potential challenges related to real-world scenarios, such as noise when users are in motion.

REFERENCES

[1] Ascari, R.E.S., Pereira, R., Silva, L. (2020). Computer vision-based methodology to improve interaction for people with motor and speech impairment. ACM Transactions on Accessible Computing (TACCESS), 13(4): 1-33. https://doi.org/10.1145/3408300

[2] Anaraki, J.R., Orlandi, S., Chau, T. (2020). A deep learning approach to tongue detection for pediatric population. https://doi.org/10.48550/arXiv.2009.02397

[3] Richhariya, P., Chauhan, P., Kane, L., Pasricha, A., Dewangan, B.K. (2022). Recognition of hand motion trajectory gestures for novel input interfaces. Revue d'Intelligence Artificielle, 36(6): 919. https://doi.org/10.18280/ria.360613

[4] Mahmoud, A.G., Hasan, A.M., Hassan, N.M. (2021). Convolutional neural networks framework for human hand gesture recognition. Bulletin of Electrical

Engineering and Informatics, 10(4): 2223-2230. https://doi.org/10.11591/eei.v10i4.2926

[5] Chu, F.J., Xu, R., Zhang, Z., Vela, P.A., Ghovanloo, M. (2018). The helping hand: An assistive manipulation framework using augmented reality and tongue-drive interfaces. In 2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Honolulu, HI, USA, pp. 2158-2161. https://doi.org/10.1109/EMBC.2018.8512668

[6] Kumar, R., Sharma, K., Assaf, M., Sharma, B., Naidu, S. (2019). Development of an assistive tongue drive system for disabled individuals. In PRICAI 2019: Trends in Artificial Intelligence: 16th Pacific Rim International Conference on Artificial Intelligence, Fiji, pp. 506-511. https://doi.org/10.1007/978-3-030-29894-4_41

[7] Rada, H.M., Abdul Hassan, A.K., Al-Timemy, A.H. (2023). Recognition of upper limb movements based on hybrid EEG and EMG signals for human-robot interaction. Iraqi Journal of Computers, Communications, Control and Systems Engineering, (In press).

[8] Wali, S.S., Abdullah, M.N. (2021). Integrating wearable devices for intelligent health monitoring system. Iraqi Journal of Computers, Communications, Control and Systems Engineering, 21(4): 1-14. https://doi.org/10.33103/uot.ijccce.21.4.1

[9] Ali, M.J., Ali, A.H., Mahmood, A.I. (2020). The design and simulation of FBG sensors for medical application. Iraqi Journal of Computers, Communications, Control and Systems Engineering, 20(4): 1-8. https://doi.org/10.33103/uot.ijccce.20.4.1

[10] Nasser, A.R., Hasan, A.M., Humaidi, A.J., Alkhayyat, A., Alzubaidi, L., Fadhel, M.A., Santamaría, J., Duan, Y. (2021). IoT and cloud computing in health-care: A new wearable device and cloud-based deep learning algorithm for monitoring of diabetes. Electronics, 10(21): 2719. https://doi.org/10.3390/electronics10212719

[11] Nasser, A.R., Mahmood, A.M. (2021). Cloud-based Parkinson’s disease diagnosis using machine learning. Mathematical Modelling of Engineering Problems, 8(6): 915-922. https://doi.org/10.18280/mmep.080610

[12] Croock, M.S. (2014). LTE based E-health monitoring system. Iraqi Journal of Computers, Communications, Control and Systems Engineering, 14(2): 37-45.

[13] Riyaz, L., Butt, M.A., Zaman, M. (2022). A novel ensemble deep learning model for coronary heart disease prediction. Revue d'Intelligence Artificielle, 36(6): 825-832. https://doi.org/10.18280/ria.360602

[14] Kateb, Y., Meglouli, H., Khebli, A. (2023). Coronavirus diagnosis based on chest X-ray images and pre-trained DenseNet-121. Revue d'Intelligence Artificielle, 37(1): 23-28. https://doi.org/10.18280/ria.370104

[15] Kirtas, O., Mohammadi, M., Bentsen, B., Veltink, P., Struijk, L.N.A. (2021). Design and evaluation of a noninvasive tongue-computer interface for individuals with severe disabilities. In 2021 IEEE 21st International Conference on Bioinformatics and Bioengineering (BIBE), Kragujevac, Serbia, pp. 1-6. https://doi.org/10.1109/BIBE52308.2021.9635238

[16] Jiang, B., Kim, J., Park, H. (2020). A new approach of minimizing midas touch problem for a tracer-free tongue-controlled assistive technology. IEEE Sensors Journal, 21(1): 743-754. https://doi.org/10.1109/JSEN.2020.3013858

[17] Mohammadi, M., Knoche, H., Bentsen, B., Gaihede, M., Struijk, L.N.A. (2020). A pilot study on a novel gesture-based tongue interface for robot and computer control. In 2020 IEEE 20th International Conference on Bioinformatics and Bioengineering (BIBE), Cincinnati, OH, USA, pp. 906-913. https://doi.org/10.1109/BIBE50027.2020.00154

[18] Niu, S., Liu, L., McCrickard, D.S. (2019). Tongue-able interfaces: Prototyping and evaluating camera based tongue gesture input system. Smart Health, 11: 16-28. https://doi.org/10.1016/j.smhl.2018.03.001

[19] Ahmed, S.T., Al-Hamdani, R., Croock, M.S. (2020). Enhancement of student performance prediction using modified K-nearest neighbor. TELKOMNIKA (Telecommunication Computing Electronics and Control), 18(4): 1777-1783. http://doi.org/10.12928/telkomnika.v18i4.13849

[20] Susilawati, K., Suarna, N., Amalia, D.R. (2022). Analisis Usabilitas Pengguna E-learning Menggunakan HCI di SMKAl-Musyawirin. INTERNAL (Information System Journal), 5(1): 40-52. https://doi.org/10.32627/internal.v5i1.514

[21] Sinche, S., Hidalgo, P., Fernandes, J., Raposo, D., Silva, J., Rodrigues, A., Armando, N., Boavida, F. (2020). Analysis of student academic performance using human-in-the-loop cyber-physical systems. Telecom, 1(1): 18-31. https://doi.org/10.3390/telecom1010003

[22] Jeske, D., Bagher, M., Pantidi, N. (2018). HCI expertise needed! Personalisation and feedback optimisation in online education. In Proceedings of the 31st International BCS Human Computer Interaction Conference (HCI 2017), Sunderland, United Kingdom, pp. 1-4. https://doi.org/10.14236/ewic/hci2017.15

[23] Wilde, A., Vasilchenko, A., Dix, A. (2018). HCI and the educational technology revolution # HCIEd2018: A workshop on video-making for teaching and learning human-computer interaction. In Proceedings of the 2018 International Conference on Advanced Visual Interfaces, Italy, pp. 1-3. https://doi.org/10.1145/3206505.3206600

[24] Atreya, A. (2021). Analysis of human computer interaction (HCI) model in SMEs. International Journal of Advanced Information` and Communication Technology, 8(1): 47-53. https://doi.org/10.46532/ijaict-202108003

[25] Hu, B., Hong, X.Q. (2022). Application of challenging learning based on human-computer interaction under machine vision in vocational undergraduate colleges. Computational Intelligence and Neuroscience, 2022: 4667387. https://doi.org/10.1155/2022/4667387

[26] Santana-Mancilla, P.C., Rodriguez-Ortiz, M.A., Garcia-Ruiz, M.A., Gaytan-Lugo, L.S., Fajardo-Flores, S.B., Contreras-Castillo, J. (2019). Teaching HCI skills in higher education through game design: A study of students’ perceptions. Informatics, 6(2): 22. https://doi.org/10.3390/informatics6020022

[27] Bollin, A., Pasterk, S., Kesselbacher, M., Reci, E., Wieser, M., Lobnig, N. (2021). HCI in K12 computer science education–using HCI as a topic and a didactic tool. In CHItaly 2021: 14th Biannual Conference of the Italian SIGCHI Chapter, pp. 1-8. https://doi.org/10.1145/3464385.3464717

[28] Nguyen, D.S., Le, Q.M. (2020). Hacking user in human-computer interaction design (HCI). In 2020 3rd International Conference on Information and Computer Technologies (ICICT), USA, pp. 230-234.

https://doi.org/10.1109/ICICT50521.2020.00042

[29] Kulshreshtha, N., Basak, S., Monika, M. (2021). HCI: Use in cyber security. International Journal for Research in Applied Science and Engineering Technology, 9(VII): 109-113. https://doi.org/10.22214/ijraset.2021.36246

[30] Grobler, M., Gaire, R., Nepal, S. (2021). User, usage and usability: Redefining human centric cyber security. Frontiers in Big Data, 4: 583723. https://doi.org/10.3389/fdata.2021.583723

[31] Moallem, A. (2019). HCI for cybersecurity, privacy and trust. First International Conference, HCI-CPT 2019, Held as Part of the 21st HCI International Conference, Denmark, Vol. 11594. https://doi.org/10.1007/978-3-030-77392-2

[32] Akinsola, J.E.T., Akinseinde, S., Kalesanwo, O., Adeagbo, M., Oladapo, K., Awoseyi, A., Kasali, F., Heimgartner, R. (2021). Application of artificial intelligence in user interfaces design for cyber security threat modeling. IntechOpen. https://doi.org/10.5772/intechopen.96534

[33] Razzak, M.A., Islam, M.N. (2020). Exploring and evaluating the usability factors for military application: A road map for HCI in military applications. Human Factors and Mechanical Engineering for Defense and Safety, 4: 1-18. https://doi.org/10.1007/s41314-019-0032-6

[34] Zeng, Y.L. (2021). How human centered AI will contribute towards intelligent gaming systems. Proceedings of the AAAI Conference on Artificial Intelligence, 35(18): 15742-15743. https://doi.org/10.1609/aaai.v35i18.17868

[35] Fang, X.W. (2019). HCI in games. First International Conference, HCI-Games 2019, Held as Part of the 21st HCI International Conference, USA, Vol. 11595. https://doi.org/10.1007/978-3-030-22602-2

[36] Rau, P.P., Ji, Y.G. (2018). Games and HCI. International Journal of Human-Computer Interaction, 34(8): 681-681. https://doi.org/10.1080/10447318.2018.1461766

[37] Dias, S.B., Diniz, J.A., Konstantinidis, E.I., Savvidis, T., Zilidou, V., Bamidis, P.D., Grammatikopoulou, A., Dimitropoulos, K., Grammalidis, N., Jaeger, H., Stadtschnitzer, M., Silva, H., Telo, G., Ioakeimidis, I., Ntakakis, G., Karayiannis, F., Huchet, E., Hoermann, V., Filis, K., Theodoropoulou, E., Lyberopoulos, G.L., Kyritsis, K.A., Papadopoulos, A., Depoulos, A., Trivedi, D., Chaudhuri, R.K., Klingelhoefer, L., Reichmann, H., Bostantzopoulou, S., Katsarou, Z., Iakovakis, D., Hadjidimitriou, S., Charisis, V.S., Apostolidis, G., Hadjileontiadis, L.J. (2021). Assistive HCI-serious games co-design insights: The case study of i-prognosis personalized game suite for Parkinson’s disease. Frontiers in Psychology, 11: 1-16. https://doi.org/10.3389/fpsyg.2020.612835

[38] Guglietti, B., Hobbs, D.A., Wesson, B., Ellul, B., McNamara, A., Drum, S., Collins-Praino, L.E. (2022). Development and co-design of NeuroOrb: A novel “serious gaming” system targeting cognitive impairment in Parkinson’s disease. Frontiers in Aging Neuroscience, 14: 266. https://doi.org/10.3389/fnagi.2022.728212

[39] Zeng, Y.L., Shah, A., Thai, J., Zyda, M. (2021). Applied machine learning for games: A graduate school course. In Proceedings of the AAAI Conference on Artificial Intelligence, 35(17): 15695-15703. https://doi.org/10.1609/aaai.v35i17.17849

[40] Ninaus, M., Moeller, K., McMullen, J., Kiili, K. (2017). Acceptance of game-based learning and intrinsic motivation as predictors for learning success and flow experience. International Journal of Serious Games, 4(3): 15-30. http://dx.doi.org/10.17083/ijsg.v4i3.176

[41] van de Weijer, S.C., Kuijf, M.L., de Vries, N.M., Bloem, B.R., Duits, A.A. (2019). Do-it-yourself gamified cognitive training. JMIR Serious Games, 7(2): e12130. https://doi.org/10.2196/12130

[42] Hashimoto, T., Low, S., Fujita, K., Usumi, R., Yanagihara, H., Takahashi, C., Sugimoto, M., Sugiura, Y. (2018). Tongueinput: Input method by tongue gestures using optical sensors embedded in mouthpiece. In 2018 57th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), Nara, Japan, pp. 1219-1224. https://doi.org/10.23919/SICE.2018.8492690

[43] Li, R., Wu, J., Starner, T. (2019). Tongueboard: An oral interface for subtle input. In Proceedings of the 10th Augmented Human International Conference 2019, pp. 1-9. https://doi.org/10.1145/3311823.3311831

[44] Yamashita, K., Kikuchi, T., Masai, K., Sugimoto, M., Thomas, B.H., Sugiura, Y. (2017). CheekInput: Turning your cheek into an input surface by embedded optical sensors on a head-mounted display. In Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology, pp.1-8. https://doi.org/10.1145/3139131.3139146

[45] Jingu, A., Tanaka, Y., Lopes, P. (2023). LipIO: Enabling lips as both input and output surface. In Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, pp. 1-14. https://doi.org/10.1145/3544548.3580775

[46] Sun, W., Li, F.M., Steeper, B., Xu, S., Tian, F., Zhang, C. (2021). Teethtap: Recognizing discrete teeth gestures using motion and acoustic sensing on an earpiece. In 26th International Conference on Intelligent User Interfaces, pp. 161-169. https://doi.org/10.1145/3397481.3450645

[47] Vojtech, J.M., Hablani, S., Cler, G.J., Stepp, C.E. (2020). Integrated head-tilt and electromyographic cursor control. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 28(6): 1442-1451. https://doi.org/10.1109/TNSRE.2020.2987144

[48] Groll, M.D., Hablani, S., Vojtech, J.M., Stepp, C.E. (2020). Cursor click modality in an accelerometer-based computer access device. IEEE Transactions on Neural Systems and Rehabilitation Engineering, 28(7): 1566-1572. https://doi.org/10.1109/TNSRE.2020.2996820

[49] Chen, V., Xu, X.H., Li, R., Shi, Y.C., Patel, S., Wang, Y.T. (2021). Understanding the design space of mouth microgestures. In Designing Interactive Systems Conference 2021, pp. 1068-1081. https://doi.org/10.1145/3461778.3462004

[50] Lv, Z., Poiesi, F., Dong, Q., Lloret, J., Song, H. (2022). Deep learning for intelligent human-computer interaction. Applied Sciences, 12(22): 11457. https://doi.org/10.3390/app122211457