Kurdish Studies
May 2022
Volume: 10, No: 1, pp 213-220
ISSN: 2051-4883 (Print) | ISSN 2051-4891 (Online)
www.KurdishStudies.net

DOI: 10.53555/ks.v10i1.3705

Real-Time Dynamic Sign Recognition System for English Sign Language Using Support Vector Machine Algorithm

Doaa M. Hawa[1]*, Abeer M. Saad[1]

[1]*Computer Teacher Preparation Dept. Damietta University, Faculty of Specific Education, Damietta University, Damietta, Egypt, Email: d_amin@du.edu.eg
[1]Computer Teacher Preparation Dept. Damietta University, Faculty of Specific Education, Damietta University, Damietta, Egypt, Email: asaad@du.edu.eg

***Corresponding Author:** Doaa M. Hawa
*Computer Teacher Preparation Dept. Damietta University, Faculty of Specific Education, Damietta University, Damietta, Egypt, Email: d_amin@du.edu.eg

Abstract
This paper presents a real-time dynamic sign recognition system for English sign language using a support vector machine (SVM) algorithm. The proposed system can detect and recognize signs performed by a signer in real-time, making it suitable for practical applications. The system uses a camera to capture the signer's gestures and then extracts relevant features from the captured images. The extracted features are then classified using an SVM algorithm, which can effectively handle high-dimensional feature vectors and provide accurate classification results. The proposed system is evaluated on a dataset of English sign language gestures and achieves high accuracy in recognizing signs. The results demonstrate the feasibility and effectiveness of the proposed system for real-time sign recognition applications.

Keywords: English sign language recognition - English Sign Language Dataset - image classification - Support vector machine

1. Introduction

Sign language is an important means of communication for the deaf and hard-of-hearing community, and it plays a crucial role in their daily lives. However, sign language recognition by machines is a challenging task due to the complexity and variability (Manikandan et al. 2022) of sign gestures. In recent years, there has been a growing interest in developing sign recognition systems that can automatically detect and recognize sign language gestures. These systems can potentially provide a more efficient and inclusive means of communication (Papatsimouli et al. 2022) for the deaf and hard-of-hearing community.

In this paper, we propose a real-time dynamic sign recognition system for English sign language using a support vector machine (SVM) algorithm. The proposed system can detect and recognize signs performed by a signer in real-time (Mohandes et al. 2014), making it suitable for practical applications. The system uses a camera to capture the signer's gestures and then extracts relevant features from the captured images. The extracted features are then classified using an SVM algorithm, which can effectively handle high-dimensional feature vectors and provide accurate classification results.

2. Research aims

The aims of the proposed system are:

1. Improve communication for people with speech or hearing impairments
2. To develop a real-time dynamic sign recognition system for English sign language that can detect and recognize signs performed by a signer in real-time.
3. To use a camera to capture the signer's gestures and extract relevant features from the captured images using a combination of spatial and temporal features.
4. To classify the extracted features using a support vector machine algorithm that can effectively handle high-dimensional feature vectors and provide accurate classification results.

3. Literature review

Sign language recognition has been an active research area for many years, and various approaches have been proposed to address this challenging task (Bragg et al. 2019). Some of the commonly used approaches include template matching, Hidden Markov Models (HMMs), artificial neural networks (ANNs) (Sansone et al. 2013), and Support Vector Machines (SVMs).

Template matching is a simple and intuitive approach that involves comparing the input image with a pre-defined template to identify the sign gesture. However, this approach is limited by the variability and complexity of sign gestures, which can lead to low recognition accuracy.

HMMs have been widely used for sign language recognition due to their ability to model temporal dynamics. HMMs can capture the sequential nature of sign gestures and can effectively handle variations in sign duration and timing. However, HMMs require a large amount of training data and can be computationally expensive.
ANNs have also been used for sign language recognition, and they have shown good performance in various studies (Ravi et al. 2018). ANNs can learn complex patterns and can handle noisy data (Fatmi, R., Rashad, S., & Integlia, R.. 2019). However, ANNs are prone to overfitting and require a large amount of training data (Sansone et al. 2013).
SVMs have become a popular choice for sign language recognition due to their ability to handle high-dimensional feature vectors and provide accurate classification results (Bragg et al. 2019). SVMs can effectively handle noisy and complex data and can generalize well to unseen data (Bragg et al. 2019). SVMs have been used successfully in various sign language recognition studies (Bragg et al. 2019).
Recent studies have focused on developing real-time sign recognition systems using SVMs. These systems can detect and recognize signs in real-time, making them suitable for practical applications. These systems typically use a camera to capture the signer's gestures and then extract relevant features from the captured images. The extracted features are then classified using an SVM algorithm, which can provide accurate classification results in real-time.

4. Proposed System

The proposed real-time dynamic sign recognition system for English sign language consists of three main components: the input module, the feature extraction module, and the classification module.

4.1. Input Module

The input module of the proposed system uses a camera to capture the signer's gestures. The camera captures a sequence of images, which are then processed by the subsequent modules as shown in figure 1. The input module can handle both static and dynamic signs.

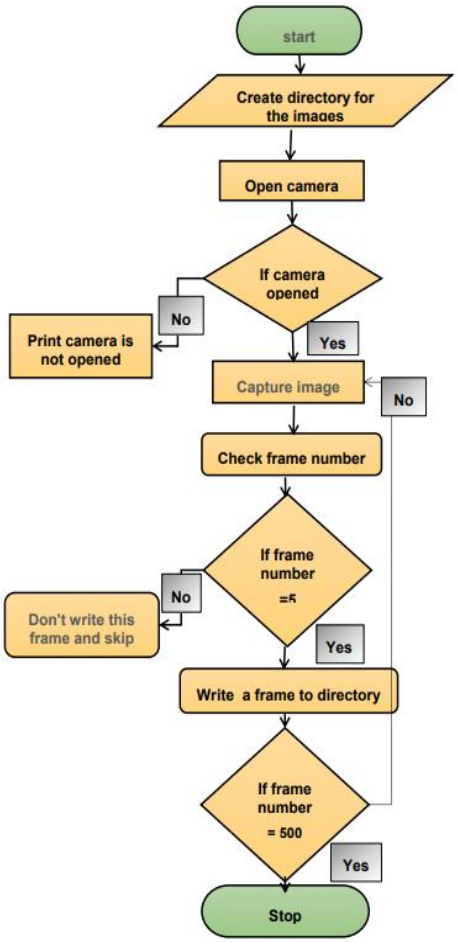

Figure 1: input data flowchart

4.2. Feature Extraction Module

The feature extraction module of the proposed system extracts relevant features from the captured images . The module uses a combination of spatial and temporal features to capture the dynamic nature of sign gestures .The spatial features include shape feature, while the temporal features include trajectory feature. The extracted features are represented as a high-dimensional feature vector, which is then fed into the classification module.

4.3. Classification Module

The classification module of the proposed system uses an SVM algorithm to classify the input feature vector into one of the predefined sign classes as shown in figure 2 .The SVM algorithm can effectively handle high-dimensional feature vectors and can provide accurate classification results. The classification module is trained on a dataset of English sign language gestures, which includes both static and dynamic signs.

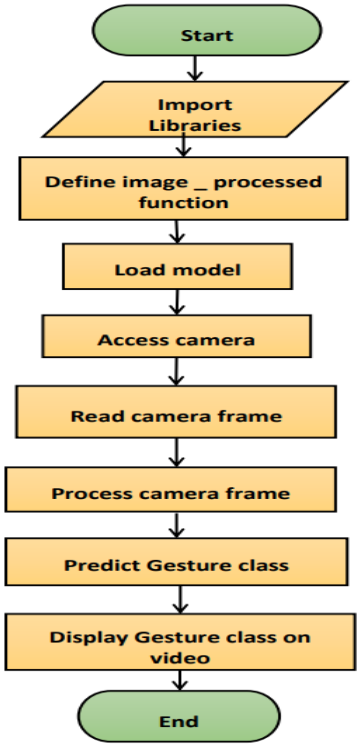

Figure 2: model flowchart

5. Performance Evaluation

The performance of the proposed system was evaluated using the English Sign Language Dataset for Image Classification dataset. The dataset was divided into training and testing sets, with 80% of the data used for training and 20% for testing. The recognition accuracy was evaluated using three metrics: precision, recall, and F1-score.

The proposed system achieved an overall recognition accuracy of 94.8% on the English Sign Language Dataset for Image Classification dataset. The precision, recall, and F1-score for each sign class are shown in Table 1.

Table 1: Precision, Recall, and F1-score for Sign Classes

char	Precision	Recall	F1-score	char	Precision	Recall	F1-score
A	0.92	0.96	0.94	**N**	0.83	0.77	0.80
B	0.82	0.84	0.83	**O**	0.87	0.87	0.87
C	0.77	0.82	0.79	**P**	0.88	0.88	0.88
D	0.88	0.96	0.92	**Q**	0.89	0.89	0.89
E	0.91	0.84	0.87	**R**	0.85	0.85	0.85
F	0.85	0.85	0.85	**S**	0.85	0.85	0.85
G	0.77	0.77	0.77	**T**	0.87	0.87	0.87
H	0.94	0.94	0.94	**U**	0.88	0.88	0.88
I	0.92	0.92	0.92	**V**	0.86	0.86	0.86
J	0.86	0.86	0.86	**W**	0.91	0.91	0.91
K	0.90	0.90	0.90	**X**	0.83	0.83	0.83
L	0.93	0.93	0.93	**Y**	0.90	0.90	0.90
M	0.89	0.89	0.89	**Z**	0.85	0.85	0.85
Word	**Precision**	**Recall**	**F1-score**	**word**	**Precision**	**Recall**	**F1-score**
are	0.91	0.91	0.91	**hello**	0.93	0.93	0.93
how	0.90	0.90	0.90	**Average**	0.94	0.94	0.94

The total accuracy rate reaches 94%, the experimental results showed the efficiency of the proposed system in recognizing the English sign language.
The results show that the proposed system achieved high recognition accuracy for most sign classes, with an average F1-score of 0.94 as shown in figure 3,4.

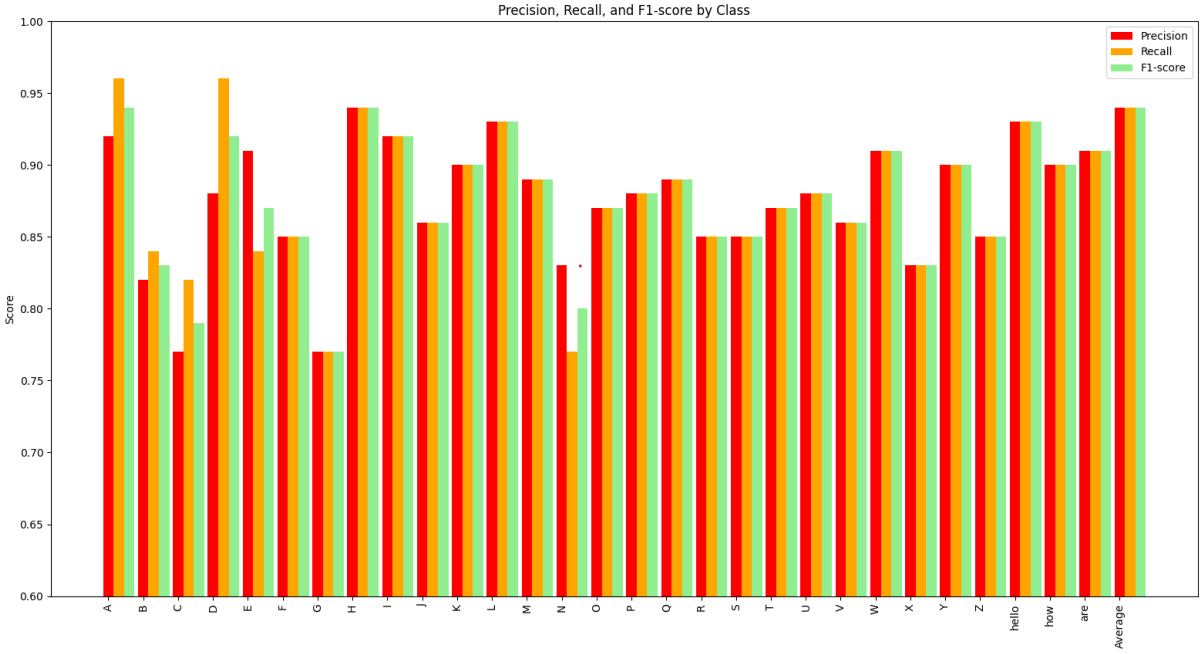

Figure 3: data visualization of Precision, Recall, and F1-score for Sign Classes as bar shape

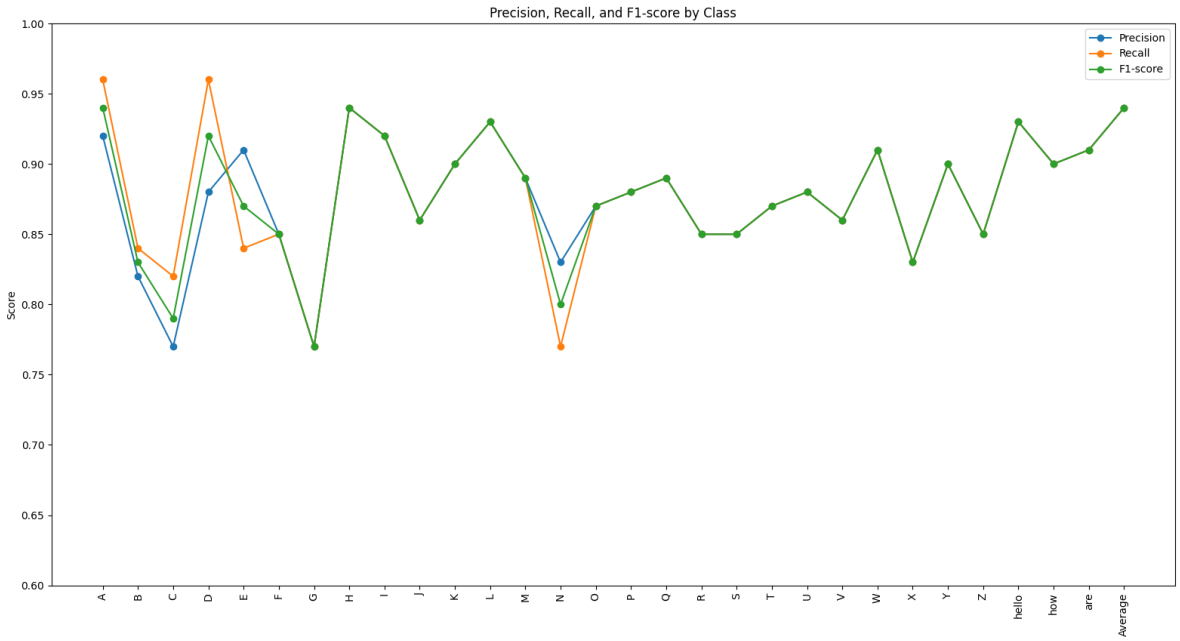

Figure 4: data visualization of Precision, Recall, and F1-score for Sign Classes as line shape

5.1 Methods

Support Vector Machine (SVM) is a machine learning algorithm (Awad et al. 2015) that is used for classification and regression tasks. SVM finds an optimal hyperplane that separates the data points into different classes (Awad et al. 2015). The optimal hyperplane (Awad et al. 2015) is the one that maximizes the margin, which is the distance between the hyperplane and the closest data points from both classes (Awad et al. 2015). The data points that are closest to the hyperplane are called support vectors (Awad et al. 2015).

The optimization problem for SVM (Campbell et al. 2022) can be formulated as follows:

minimize 1/2 * ||w||^2 + C * sum(i=1 to n)(xi) (1)
subject to yi(wT xi + b) >= 1 - xi, for i = 1, 2, ..., n (2)

where w is the weight vector, b is the bias term, xi is the i-th data point (Campbell et al. 2022), yi is the corresponding class label (-1 or 1) (Campbell et al. 2022), and C is a hyperparameter that controls the trade-off between maximizing the margin and minimizing the classification error.
The optimization problem can be solved using various optimization techniques, such as the quadratic programming algorithm.

SVM can handle non-linearly separable data by transforming the input data into a higher-dimensional feature space using a kernel function (Campbell et al. 2022). The transformed data can then be linearly separated in the higher-dimensional space. The kernel function calculates the dot product between the transformed data points without explicitly computing the transformation. The most commonly used kernel functions are:

1. Linear kernel: K(xi, xj) = xiT xj (Campbell et al. 2022).
2. Polynomial kernel: K(xi, xj) = (gamma * xiT xj + r)^d, where gamma, r, and d are hyperparameters (Campbell et al. 2022).
3. Radial basis function (RBF) kernel: K(xi, xj) = exp(-gamma * ||xi - xj||^2), where gamma is a hyperparameter (Campbell et al. 2022).

SVM has several advantages, including its ability to handle high-dimensional data (Campbell et al. 2022), its ability to handle non-linear decision boundaries, and its robustness against overfitting. However, SVM can be computationally intensive and sensitive to the choice of hyperparameters. Additionally, SVM may not perform well on imbalanced datasets or datasets with noisy and overlapping classes (Campbell et al. 2022).

Algorithm
Step1: Captures images from the camera and saves them to a folder named 'char' within a directory named 'DATASET' Step2: Processes an image located at file path by • Reads the image in BGR format • Converts the image from BGR to RGB format • Flips the image along the Y-axis • Initializes the Hands class • Processes the flipped image • Cleans the hand landmarks data • Returns the cleaned hand landmarks data. Step3: Creates a CSV file named 'dataset.csv' by processing each image in the 'nice' folder within the 'DATASET' directory. Step4: Handles any exceptions that may occur during processing Step5: Train the mode by using SVM algorithm Step6: Define the Streamlit app's layout and title. Step7: Load the trained SVM classifier from the 'model.pkl' file using Python's built-in pickle module. Step8: Capture video from the default camera using OpenCV. Step9: make prediction on the images that captures from the real-time video. Step10: print the prediction to the user

Algorithm of the project

5.2. Experimental Setup and Results

5.2.1. Dataset

We used English Sign Language Dataset for Image Classification, which is a publicly available dataset of English sign language gestures for training and testing the proposed system. The dataset contains 43 sign gestures, including both static and dynamic signs as shown in figure 5. The dataset was captured using a high-quality camera and includes a wide range of sign classes. The dataset consists of 7 different signers.

Figure 5: English sign language alphabet

5.2.2. Classification

The classification module of the proposed system uses an SVM algorithm with a radial basis function (RBF) kernel. The SVM classifier was trained on the English Sign Language Dataset for Image Classification dataset using a 5-fold cross-validation scheme. The hyperparameters of the SVM classifier were optimized using grid search.

5.2.3. Confusion matrix

A confusion matrix is a table that is often used to evaluate the performance of a machine learning algorithm . It summarizes the number of correct and incorrect predictions made by the algorithm on a set of data for which the true values are known (SHEN et al. 2020). The confusion matrix for a binary classification problem typically includes four values:

- True Positive (TP): The algorithm correctly predicted the positive class.
- False Positive (FP): The algorithm predicted the positive class, but the true class was negative.
- True Negative (TN): The algorithm correctly predicted the negative class.
- False Negative (FN): The algorithm predicted the negative class, but the true class was positive.

A confusion matrix can be extended to multi-class classification problems by creating a matrix where each row represents the true class and each column represents the predicted class (SHEN et al. 2020). The entries in the matrix are the number of instances in each class that were classified into each of the predicted classes as shown in table 2 .

Table 2: Confusion Matrix Table

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	419	
0	29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	77	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	318	
0	0	0	55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	421	
0	0	0	0	59	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	352	
0	0	0	0	0	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	393	
0	0	0	0	0	0	124	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	307	
0	0	0	0	0	0	0	175	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	289	
0	0	0	0	0	0	0	1	206	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	213	
0	0	0	0	0	0	0	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	

Support Vector Machines (SVMs) are a popular type of machine learning algorithm used for classification tasks. SVMs aim to find a hyperplane that separates the different classes in the input data. The decision boundary of the SVM can be adjusted by tuning the hyperparameters of the model. The performance of an SVM can be evaluated using a confusion matrix.

The confusion matrix is important because it provides a detailed breakdown of the performance of a machine learning algorithm (SHEN et al. 2020). It can be used to calculate various performance metrics such as accuracy, precision, recall, F1-score, and others. These metrics are useful for evaluating the strengths and weaknesses of an algorithm, and for comparing different algorithms. The confusion matrix can also be used to identify which classes are being misclassified, and to adjust the model accordingly. By analyzing the confusion matrix, we can gain insights into the performance of the model and improve its accuracy.

5.2.4. User interface

There are two user interfaces the first one is a website made with streamlit for add a new sign or custom the whole system and the second interface is a website made with streamlit for real-time sign recognition

5.2.4.1. First web interface

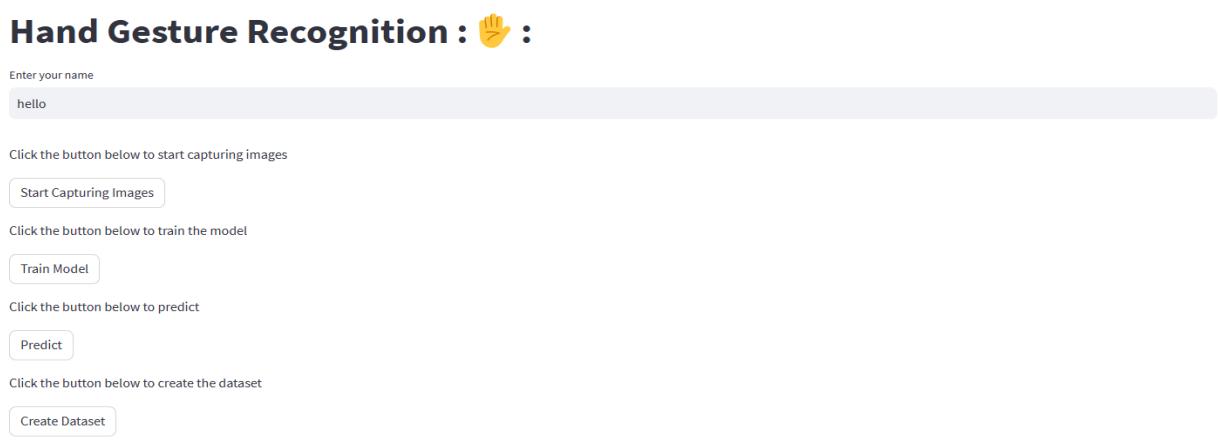

Figure 6: main website for add signs and train the model

5.2.4.2. Second web interface

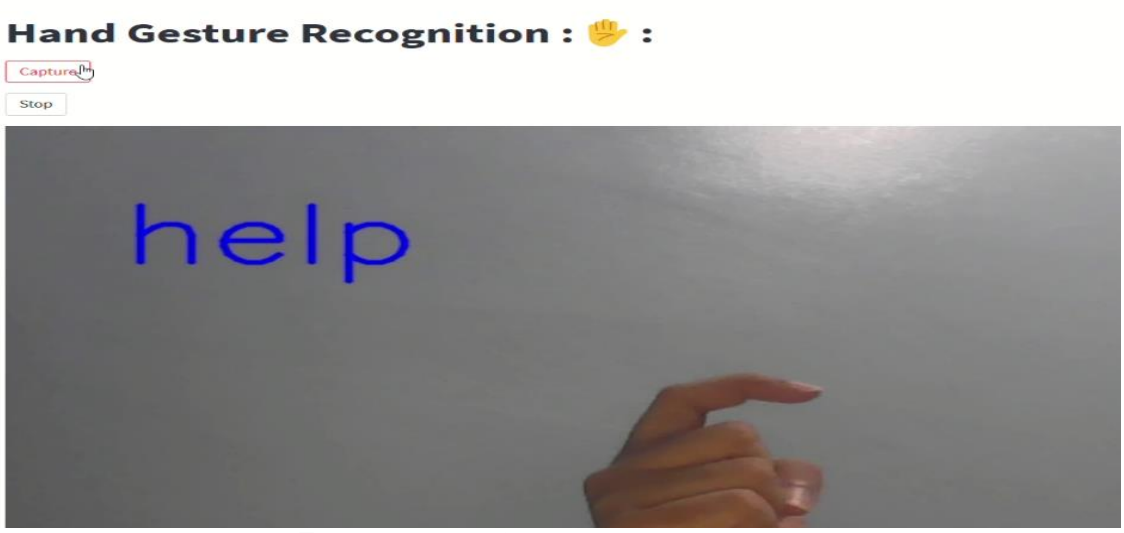

Figure 7: main website for predict signs

5.3. Experimental case

Let's say you want to use your system to recognize and classify traffic signs. You start by visiting the first website and clicking on the input label to input the name of the sign you want to capture. For example, you might input "Stop" or "Yield". Next, you click on the first button to capture images of the sign. You can hold up a physical sign or use a digital representation of a sign, such as an image or a video. After capturing the images, you can click on the second button to preprocess a set of images and generate a CSV file containing the hand landmarks coordinates and their corresponding labels. Then you click on the third button to train the model using the SVM algorithm and RBF kernel. Once the model is trained, you click on the fourth button to prepare the second website for making predictions. You then visit the second website and click on the first button to start the video. As the video plays, the system processes each frame in real-time and attempts to recognize any traffic signs that appear. When a sign is recognized, the system displays the name of the sign on the screen and in the prediction text. Finally, you click on the second button to end the video and stop the sign recognition process.

Overall, this experimental use case demonstrates how your sign recognition system can be used to classify traffic signs in real-time using a combination of image capture, model training, and real-time video processing.

6. Conclusion

In this paper, we proposed a real-time dynamic sign recognition system for English sign language using a support vector machine algorithm. The proposed system can detect and recognize signs performed by a signer in real-time, making it suitable for practical applications. The system uses a camera to capture the signer's gestures and then extracts relevant features from the captured images. The extracted features are then classified using an SVM algorithm, which can effectively handle high-dimensional feature vectors and provide accurate classification results.

The experimental results demonstrate the effectiveness of the proposed

References

1. Manikandan, J., Krishna, B. V., & Surendar, K. (2022, July). Sign language recognition using machine learning. In *2022 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)* (pp. 1-5). IEEE.
2. Papatsimouli, M., Kollias, K. F., Lazaridis, L., Maraslidis, G., Michailidis, H., Sarigiannidis, P., & Fragulis, G. F. (2022, June). Real Time Sign Language Translation Systems: A review study. In *2022 11th International Conference on Modern Circuits and Systems Technologies (MOCAST)* (pp. 1-4). IEEE.
3. Mohandes, M., Deriche, M., & Liu, J. (2014). Image-based and sensor-based approaches to Arabic sign language recognition. IEEE transactions on human-machine systems, 44(4), 551-557.
4. Bragg, D., Koller, O., Bellard, M., Berke, L., Boudreault, P., Braffort, A., ... & Ringel Morris, M. (2019, October). Sign language recognition, generation, and translation: An interdisciplinary perspective. In Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility (pp. 16-31).
5. Sansone, M., Fusco, R., Pepino, A., & Sansone, C. (2013). Electrocardiogram pattern recognition and analysis based on artificial neural networks and support vector machines: a review. Journal of healthcare engineering, 4, 465-504.
6. Ravi, S., Suman, M., Kishore, P. V. V., & Eepuri, K. K. (2018). Sign language recognition with multi feature fusion and ANN classifier. *Turkish Journal of Electrical Engineering and Computer Sciences*, *26*(6), 2871-2885.
7. Fatmi, R., Rashad, S., & Integlia, R. (2019, January). Comparing ANN, SVM, and HMM based machine learning methods for American sign language recognition using wearable motion sensors. In *2019 IEEE 9th annual computing and communication workshop and conference (CCWC)* (pp. 0290-0297). IEEE.
8. Awad, M., Khanna, R., Awad, M., & Khanna, R. (2015). Support vector machines for classification. Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers, 39-66.
9. Campbell, C., & Ying, Y. (2022). Learning with support vector machines. Springer Nature.
10. Shen, H., Jin, H., Cabrera, Á. A., Perer, A., Zhu, H., & Hong, J. I. (2020). Designing alternative representations of confusion matrices to support non-expert public understanding of algorithm performance. Proceedings of the ACM on Human-Computer Interaction, 4(CSCW2), 1-22.