Metadata of the chapter that will be visualized in SpringerLink

Book Title	Artificial Intelligence and Speech Technology	
Series Title		
Chapter Title	Efficient Real-Time Indian Sign Language Fingerspelling Recognition in Natural Settings Using Heuristics	
Copyright Year	2025	
Copyright HolderName	The Author(s), under exclusive license to Springer Nature Switzerland AG	
Corresponding Author	Family Name	**Raghuveera**
	Particle	
	Given Name	**T.**
	Prefix	
	Suffix	
	Role	
	Division	Department of Computer Science and Engineering, College of Engineering Guindy Campus
	Organization	Anna University
	Address	Chennai, 600025, India
	Email	raghuveera@annauniv.edu
Author	Family Name	**Akshayalakshmi**
	Particle	
	Given Name	**V. K.**
	Prefix	
	Suffix	
	Role	
	Division	Department of Computer Science and Engineering, College of Engineering Guindy Campus
	Organization	Anna University
	Address	Chennai, 600025, India
	Email	
Author	Family Name	**Nisha**
	Particle	
	Given Name	**B. A.**
	Prefix	
	Suffix	
	Role	
	Division	Department of Computer Science and Engineering, College of Engineering Guindy Campus
	Organization	Anna University
	Address	Chennai, 600025, India
	Email	
Author	Family Name	**Easwarakumar**
	Particle	
	Given Name	**K. S.**

Prefix	
Suffix	
Role	
Division	Department of Computer Science and Engineering, College of Engineering Guindy Campus
Organization	Anna University
Address	Chennai, 600025, India
Email	

Abstract	Sign language is the primary mode of communication for Hearing and Speech Impaired (HSI) people. However, the complexity and intricate nature of Indian Sign Language, which includes a majority of double-handed signs, poses a challenge for HSI people to communicate effectively with others. Moreover, the expanding vocabulary of sign language makes it difficult for those without access to updates to communicate effectively. Fingerspelling is most widely used by the HSI for general and easy day-to-day communication. A real-time and efficient fingerspelling system is thus crucial to facilitate communication for HSI people in a natural setting. However, existing real-time recognition systems are cumbersome and inefficient as they employ complex deep-learning architectures and primarily use RGB image and video data that are sensitive to lighting and background conditions and therefore are more error prone and moreover do not perform well under natural settings. This study proposes a simple and efficient real-time fingerspelling system for recognizing static fingerspelling gestures using Leap Motion Controller. The study employs a random forest classifier with translation-independent features to recognize signs, while achieving comparable accuracy, making the overall system lightweight. We achieved a real-time validation accuracy of 71% while also predicting the sample instantaneously with an average response time of 3.02 ms. Since fingerspelling can introduce spurious signs during transitions and can be ambiguous when recognizing similar signs, our system also includes a word fine-tuning phase that uses a dictionary-based approach to simplify the recognition process, making our system well-suited for real-time deployment in natural settings.

Keywords (separated by '-')	Indian Sign Language - Fingerspelling - Natural Setting - Real-time - Dictionary approach - Heuristics

Efficient Real-Time Indian Sign Language Fingerspelling Recognition in Natural Settings Using Heuristics

T. Raghuveera(✉), V. K. Akshayalakshmi, B. A. Nisha, and K. S. Easwarakumar

Department of Computer Science and Engineering, College of Engineering Guindy Campus, Anna University, Chennai 600025, India
raghuveera@annauniv.edu

Abstract. Sign language is the primary mode of communication for Hearing and Speech Impaired (HSI) people. However, the complexity and intricate nature of Indian Sign Language, which includes a majority of double-handed signs, poses a challenge for HSI people to communicate effectively with others. Moreover, the AQ1 expanding vocabulary of sign language makes it difficult for those without access to updates to communicate effectively. Fingerspelling is most widely used by the HSI for general and easy day-to-day communication. A real-time and efficient fingerspelling system is thus crucial to facilitate communication for HSI people in a natural setting. However, existing real-time recognition systems are cumbersome and inefficient as they employ complex deep-learning architectures and primarily use RGB image and video data that are sensitive to lighting and background conditions and therefore are more error prone and moreover do not perform well under natural settings. This study proposes a simple and efficient real-time fingerspelling system for recognizing static fingerspelling gestures using Leap Motion Controller. The study employs a random forest classifier with translation-independent features to recognize signs, while achieving comparable accuracy, making the overall system lightweight. We achieved a real-time validation accuracy of 71% while also predicting the sample instantaneously with an average response time of 3.02 ms. Since fingerspelling can introduce spurious signs during transitions and can be ambiguous when recognizing similar signs, our system also includes a word fine-tuning phase that uses a dictionary-based approach to simplify the recognition process, making our system well-suited for real-time deployment in natural settings.

Keywords: Indian Sign Language · Fingerspelling · Natural Setting · Real-time · Dictionary approach · Heuristics

1 Introduction

Sign Language is the most natural and expressive way for Hearing and Speech Impaired (HSI) people. Communication through Sign Language is an act of conveying intended meanings from one peer to another using mutually understood signs and semiotic rules.

A. Dev et al. (Eds.): AIST 2023, CCIS 2267, pp. 1–17, 2025.
https://doi.org/10.1007/978-3-031-75164-6_26

Sign language looks like manual communication and uses body language to convey meaning, as opposed to acoustically conveyed sound patterns, which involves the simultaneous combination of hand shapes, orientation and movement of hands, arms or body, and facial expressions to convey a speaker's thought. For an individual to progress in life and coexist with other individuals there is a need for effective communication. HSI makes up a sizable community with specific needs and has difficulty communicating efficiently and effectively. It is often seen as a hindrance to their growth and development despite their potential and abilities. Assistive technology gives confidence and improves the dignity and standard of living of the HSI. It also ensures non-discrimination on various grounds brings equal opportunities to the differently abled and provides them with a platform to join the mainstream. In contrast to other sign languages of the world including ASL, BSL, GSL etc., most of the signs in ISL are double-handed making it relatively complex. The latest dictionary of ISL includes 10000 words [1]. It is also known that over 70% of ISL gestures involve only hands (either one hand or both hands) and the rest of the signs involve hands, body and facial gestures combined.

In the context of sign language, it is not uncommon to come across words or phrases that do not have a corresponding sign. This can pose a challenge for individuals who are not proficient in sign language or are not exposed to the language frequently. In such situations, fingerspelling can prove to be an invaluable tool in facilitating communication. Fingerspelling involves representing each letter of the alphabet using its corresponding gesture. It is an integral aspect of sign language, and being able to finger-spell proficiently can greatly enhance one's ability to communicate effectively in sign language, especially in situations where there is no standard sign for a particular word or phrase. In this regard, a real-time fingerspelling recognition system is particularly helpful for HSI individuals. To develop an efficient real-time fingerspelling recognition system, it is imperative to have a system that can recognise individual letters with high accuracy under unconstrained natural settings. However, to meet the demands of real-time processing, it should be simple and portable all while ensuring efficiency and effectiveness. Existing sign language recognition methods employ complex deep learning architectures such as LSTM and HMM, which take significant time to recognize signs. From the existing research, it is observed that, simple machine learning models have not been experimented with for their feasibility for real-time sign recognition. Additionally, their performance under natural settings is not explored.

Another significant challenge in developing a simple and deployable fingerspelling system is the presence of similar-looking signs and the introduction of spurious signs while transitioning between signs. To address the challenge of recognition of similar looking signs, broader contextual understanding of the signs and signers would be of great utility. Understanding this context requires a complex language model that deals with semantic relationships and computations with embeddings. This complexity makes the system bulky and unsuitable for real-time deployments.

Therefore, we introduce a feasible, efficient, yet lightweight system for the near in- stantaneous and accurate recognition of signs in Indian sign language under natural settings. We suggest a Random Forest classifier using direction vectors of fingers as distinguishing features of signs. Our research indicates that the direction vectors used as features for our model are especially effective at handling translational differences in

the signer's hand position while performing a sign. This is a crucial consideration for sign language recognition, as signers may not always perform signs in the exact same position relative to the recording device. The use of direction vectors as features enables our model to recognize signs accurately regardless of small differences in position and general anthropometry, making it a valuable tool for real-world applications. We find that our model predicts 71% of performed signs accurately with an average response time of 3.02 ms. Our system can perform even better if similar-looking signs are removed.

To address the challenge of similar looking signs and spurious signs, we propose a simple word fine-tuning phase utilizing a dictionary consisting of the most frequently used words assuming a closed-world concept. This makes our system more suitable for real-time deployments under natural settings. Experiments were done with the actual beneficiaries to establish the efficacy of the system and thereby enabling the HSI community to better communicate with members of society in their natural way. Furthermore, the system is easily portable to smart phones and wearables, making it suitable for real-world deployments.

2 Related Works

Hand gestures are an expressive way of communication between humans using signs that are interpreted by others. However, this mode of communication is primarily con- fined to the HSI community since the general population seldom learns sign language. This highlights the necessity for a sign recognition system, to facilitate effective communication of their ideas with the broader public.

Initial methods and experiments used cameras for sign recognition. G. A. Rao et al., [2] proposed a CNN architecture and compared it against AdaBoost and ANN for recognizing the Indian Sign language. Their model achieved an accuracy of 92.88%. Bhagat et al. [3] utilized 3D Construction and affine transformation on images to extract features, and trained the data using a CNN. This approach achieved an accuracy of 98.81% on ISL and 97.71% on ASL. In [4], a web-based dataset of static signs was created in which over 50 deep learning models with distinct optimizers have been tested. Among them, CNN achieved the highest accuracy of 99.72%. Madhuri et al. [5] designed a system for recognition of Indian Sign language numerals from images, and used neural network and KNN to classify the signs achieving an accuracy of 97.10%. But in general while using images for sign recognition, gloves must be worn which might cause inconvenience and image recognition requires special computation to segment the hand and fingers. In addition, this poses a difficulty for overlapping signs and signs that involve dynamics. Alternatively, cyber gloves were used for the same by Kong and Ranganath [6] for continuous Sign Language recognition.

In later stages of experiments, Microsoft Kinect was used. Cao Dong et al. [7] used images from Microsoft Kinect and used Random Forest (RF) classifier for sign language recognition. Ansari and Harit [8] developed a Kinect-based ISL recognition method in which they achieved above 90% recognition rate for 13 alphabets, and 100% recognition for 3 signs. The method proposed in [9] uses depth-based information from Kinect along with video frames and performs classification using a support vector ma- chine with an accuracy of 97.5%. The system proposed by Raghuveera et al. [10] utilized Microsoft

Kinect to capture both RGB and depth data. Three kinds of features Speeded Up Robust Features (SURF), Histogram of Oriented Gradients (HOG), Local Binary Patterns (LBP) are extracted and SVM classifier is utilized for ISL recognition achieving an accuracy of 71.85%.

A significant number of works used Leap Motion Controllers [11] owing to their 3D skeletal representation of the scene that provides much knowledge on the input data. Naglot and Kulkarni [12] proposed a system using Leap to extract features based on Euclidean distances between various key points of fingers. This work achieved an accuracy of 96.15% for ASL using a multilayer perceptron. Ching-Hua Chuan and Eric Regina proposed a KNN and SVM system [13] for ASL recognition using leap. They classified 26 alphabets of ASL and reported an average classification rate of 72.78% and 79.83% for KNN and SVM respectively.

Since LSTM can process sequences better, they were experimented for Indian sign language recognition. The framework proposed by Kumar et al [14] uses a combination of two sensors Kinect and Leap to capture gestures. They extracted fingertip positions and fingertip directions and used HMM and Bidirectional LSTM classifiers for improving the accuracy of the recognition. Their approach achieved an accuracy of 97.85% and 94.55% for single and double handed signs respectively. In [15] an approach that uses a block list classifier after segmentation to remove all non-gesture frames was proposed and a random forest classifier was used for recognition using Leap Motion Sensor.

The realm of real-time Indian sign language recognition is notably less experimented with. In [16] H. Muthu Mariappan and V. Gomathi used fuzzy clustering to recognize the hand gestures made by the user in real-time, which resulted in an accuracy of 75%. However, it exhibited limitations under various conditions, particularly in terms of lighting variations and clothing constraints. A similar approach was tried in sign-language videos in [17] where the authors used a combined LSTM-GRU to classify signs and acquired an accuracy over 97% for 11 signs. Bird et al., [18] proposed a fusion based approach for British Sign Language (BSL) recognition towards implementing a real-time system. They used both data from a camera and a Leap motion sensor to accurately recognise signs and reported that the stand-alone Leap data performed comparatively poor (72.73%) which dropped even below to 41.78% when tested with un-seen data. So, they proposed a fusion approach which improved their performance to 94.44%. Their claims also substantiated the challenges associated with developing a real-time system capable of generalizing across a diverse user base. A similar transfer learning approach was utilized by [19] where the authors have found that the accuracy of their Irish Sign-Language Translator was better when they tried transfer learning with a larger dataset. In [20], a game-based interface for learning ASL through real-time ASL recognition was proposed. They used an LSTM based Classifier to predict signs from features such as the hand's sphere radius, fingertip positions and other features and obtained a testing accuracy of 99.44% and a validation accuracy of 91.82%. Hisham et al., [21] proposed a real-time Arabic sign language recognition system with 20 single-handed and 10 double-handed gestures using an AdaBoost classifier with Dynamic Time-Wrapping and achieved an accuracy of 93%. From the above-mentioned works, it is evident that building a real-time system, which can generalise across a wide user-base, is challenging and inherently requires a trade-off between efficiency and accuracy. Therefore, we

propose a simple and efficient real-time Indian sign language fingerspelling recognition system, which is robust under natural settings while achieving comparable accuracies.

3 Architecture

The proposed fingerspelling system's architecture is shown in Fig. 1. The signs performed by the signers are recorded using a Leap Motion Controller (see Fig. 1). The frame-sampling module captures two frames per second from the device. The captured data is then sent for classification to the recognition module, which utilizes a simple Random Forest classifier to classify the sign performed in each frame, after pre-processing and feature selection.

The recognized stream of letters is forwarded to the word generation module to generate the final word. This module analyses the stream of letters as they are received to determine the performed word. Once the performed word is identified, in order to address the ambiguity caused by signs that look similar, it is passed to a word fine-tuning module.

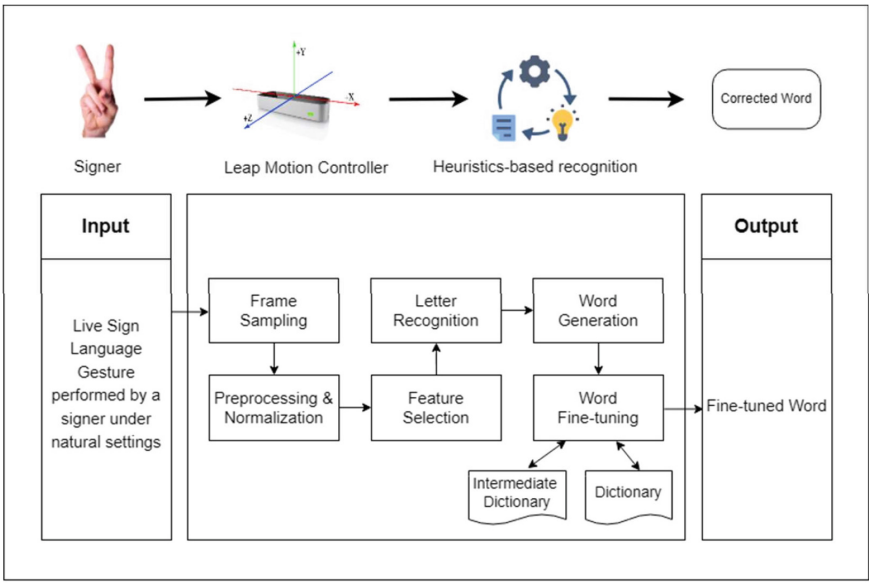

Fig. 1. Architecture of our proposed system

The fine-tuning module's main function is to eliminate noise and jitter while transitioning between signs. A dictionary comprising 114 words commonly used in an academic context has been created specifically for this purpose. The module compares the Levenshtein distance between the identified word and all the words in the dictionary, returning the most probable candidate with the least distance.

4 Implementation

4.1 Data collection and Sampling

For the purpose of this study, a sign language dataset was constructed from 16 volunteers (8 male, 8 female) who were sign language learners and instructors (see Fig. 2). The sign language gestures were recorded using a Leap Motion Controller and the coordinates were fetched using the Orion 3.2.1 SDK controller. The recorded data consists of 428 attributes, including finger direction, finger angle in relation to the palm, finger length, and other related features from a 3D hand skeleton model (Fig. 3). Data of signs corresponding to the 24 static alphabets (A – Z, except J and Y) (see Fig. 4) and 10 digits 0 to 9 (see Fig. 5) were collected.

Since recording and processing 75 frames per second using the Leap Motion Controller may cause a significant performance overload, a more efficient approach is implemented. On identifying that many of these frames are redundant and contain similar data due to the negligible time lapse between them, the number of frames sampled is reduced and only 2 frames are sampled per second. These frames are uniformly selected, with an approximate offset of 0.5 s between them. Thus, the system performs effectively by reducing the space, time and other resources spent in processing all the frames.

Fig. 2. Data recording in progress at the HSI site

4.2 Data Pre-processing and Normalization

The collected data undergoes pre-processing to ensure its accuracy and reliability. Sign language data exhibits significant variation among individuals and even across different fingers of the same individual. Thus, normalizing the data across individuals or across fingers of the same individual may miss crucial information. To address this, a fingerwise normalization method is adopted which normalizes the data across the three coordinates of each attribute. The normalization formula used is

$$\mathrm{An} = (\mathrm{Ax} + \mathrm{Ay} + \mathrm{Az})/3 \tag{1}$$

$$\mathrm{Ax} = \mathrm{Ax}/\mathrm{An} \tag{2}$$

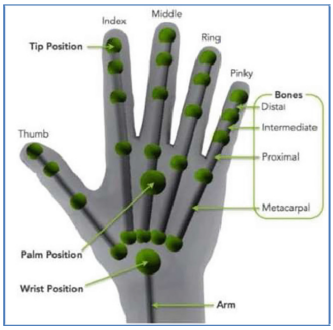

Fig. 3. Image of the hand-skeleton with some of the attributes recorded by the Leap Motion Sensor

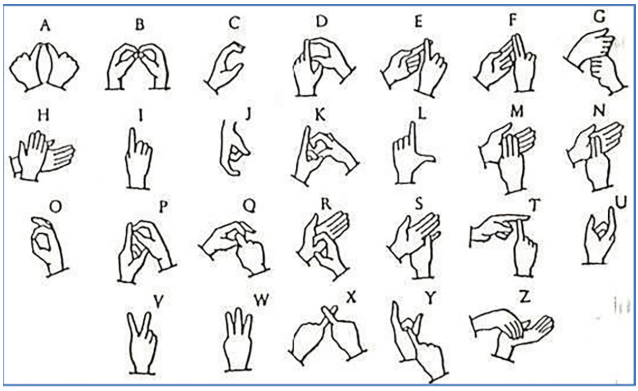

Fig. 4. ISL alphabets [1]

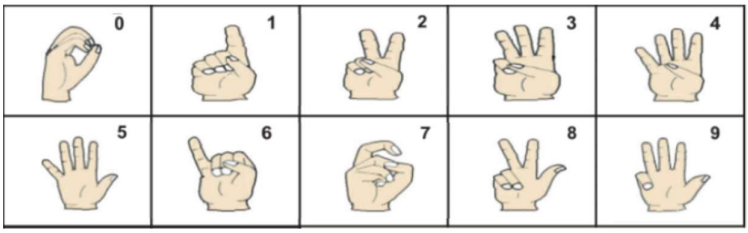

Fig. 5. ISL Numbers

$$\mathrm{Ay} = \mathrm{Ay}/\mathrm{An} \tag{3}$$

$$\mathrm{Az} = \mathrm{Az}/\mathrm{An} \tag{4}$$

where A_i represents any attribute recorded by the Leap Motion sensor with three components (A_x, A_y, A_z) along the coordinate axes respectively.

4.3 Feature Selection

To build an accurate and efficient model for real-time sign language recognition, careful selection of relevant features is essential. Out of the 428 features recorded by the Leap Motion Controller, not all are equally relevant for differentiating between different signs. Using all features can increase processing time and introduce unwanted bias and noise. Therefore, the study selects specific features that convey the distinguishability of signs. Directions serve as the ideal distinguishing feature, determined through careful analysis of the signs. The direction vectors of all fingers taken as a whole provide unique information for each sign, except for similar-looking signs like 'v' and '2'. These direction vectors are translation independent and can be unit-normalized, enabling sign recognition regardless of the signer's position and avoiding scaling issues in the model. Moreover, direction vectors are not dependent on signers' anthropometry, making the model more robust with new testers. The model utilizes the following 30 features comprised of the direction vectors of 10 fingers counted across their three coordinate axes:

$$f = (f_1, f_2, ...f_{10}) \tag{5}$$

where f denotes the Feature Set and f_1 to f_{10} are direction vectors of distal bones of the 10 fingers with $f_i = (f_{ix}, f_{iy}, f_{iz})$

4.4 Letter Recognition

Letter recognition is accomplished through a machine learning model that displays the recognized letter from each captured frame as shown in Fig. 6. To determine the most effective model, various types of machine learning models were tested, and their accuracy and precision were considered as the decisive metrics. Specifically, the models listed in Table 1 were tested, and their hyperparameters were carefully selected using the Grid Search Optimizer. Among these models, the Random Forest Classifier demonstrated superior performance in terms of accuracy and precision as can be seen from our results.

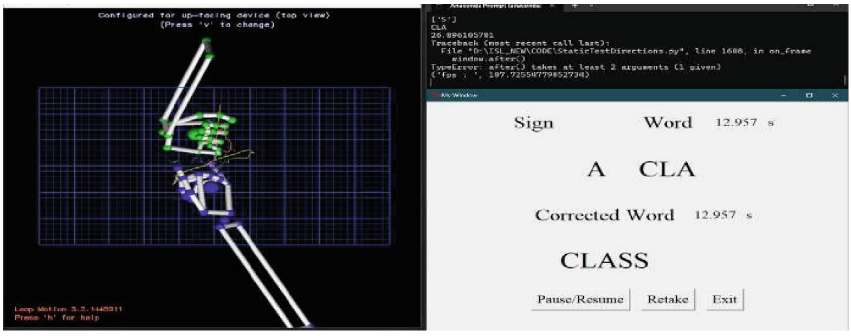

Fig. 6. Word "Class" being performed along with its output. In this picture, 'Word' refers to the output from Recognition module and 'Corrected Word' refers to the output of the Fine-tuning module

Table 1. Models and Hyperparameters

Classifier	Hyperparameters
Multi-layer perceptron	hidden_layer_sizes = (34,24,34),max_iter = 260, activation = 'relu',solver = 'adam'
Ada-Boost	n_estimators = 92, learning_rate = 0.03
Gradient Boost	n_estimators = 68, learning_rate = 0.01
Support vector machine	kernel = 'poly', degree = 5
K-nearest neighbours	n_neighbors = 5, metric = 'minkowski'

4.5 Word recognition

Generating a word through fingerspelling requires the careful performance of each letter, one by one. During this process, the signer changes his/her hand positions to perform the next character. However, these transitions between letters can introduce turbulence into the generated stream of letters, causing random characters to be included due to slight variations in hand orientation. Moreover, the Leap Motion Sensor captures 75 frames per second thereby increasing the complexity of operations. To reduce this and improve accuracy, we select only two frames per second, thereby minimizing the impact of hand orientation changes. Algorithm 1 explains the whole process. In order to identify a letter belonging to a word, we require four consecutive recognitions of the same letter, requiring a signer to hold each letter pose for at least 2 s. This approach is consistently applied to record the remaining letters, resulting in the generation of the predicted word.

Algorithm 1: Word Recognition

Input: *Recognised letter from the recognition module*
Output: *Recognised Word*
Initialisations: *word = '', previous_letter = '', count = 0*
Process:
  for every letter from the recognition module
    if the current letter is same as previous_letter
      Increment count by one
    else
      Update previous_letter to current letter
      Initialise count to 1
    if count equals four: //Since two frames are captured per second
      Append the letter to the word
      Initialise count as zero
  return word

4.6 Word Fine Tuning

Based on our observations, we noticed that similar-looking signs (such as “m” and “n,” “e” and “f,” “0” and “o”) and signs characterized by finger or hand obstruction during performance (like “x”) were often recognized incorrectly, resulting in the formation of non-existent words.

To address these issues, we introduced a word fine-tuning phase that employs a carefully curated dictionary. This phase plays a pivotal role in rectifying the inaccuracies in word predictions made by the system. The selection of words in the dictionary is based on their relevance to specific contexts, considering that communication often revolves around words closely tied to certain closed contexts. By leveraging this contextually relevant dictionary, we can significantly enhance the system’s accuracy. Furthermore, this dictionary can be expanded, to allow support for broader contexts by incorporating additional words.

For our study, we developed a custom dictionary consisting of 114 different words commonly used in academic contexts. The majority of these words were 4 to 7 letters long, while some exceeded 7 letters. Some of the words in the dictionary can be found in Table 2.

Table 2. Some words from the Dictionary

Exam	Books	College
Pass	School	Research
Quiz	Degree	Academic

Each time the system generates a word, the fine-tuning model calculates the Levenshtein distance (edit distance) between the generated word and the words in the dictionary. Levenshtein distance between two words is defined as the number of letters to be inserted, deleted or replaced in order to transform the first word into the second one. It is defined as

$$lev_{a,b}(i,j) = \begin{cases} max(i,j) & if \ \ \min(i,j) = 0 \\ min \begin{cases} lev_{a,b}(i-1,j)+1 \\ lev_{a,b}(i,j-1)+1 \\ lev_{a,b}(i-1,j-1)+1_{(a_i \neq b_i)} \end{cases} & otherwise. \end{cases} \tag{6}$$

where, a, b – words to be compared

i – position of terminal character of word a

j – position of terminal character of word b

After calculating the distance, the system selects the word with the minimum edit distance as the fine-tuned word. However, one major challenge in building a fingerspelling system with fine-tuning is handling proper nouns, as they are fingerspelled entirely and thus differ from common words. To address this, we established a threshold for the Levenshtein distance after careful experimentation. If the edit distance is greater

than approximately 65% of the word length, the generated word is returned without fine-tuning. Following the same process, in order to accommodate new words, the system records any new word identified while recognition in an intermediate dictionary along with its frequency. The dictionary is constantly updated if any new word is performed more than three times. Figs. 6 and 7 shows two words being performed along with their predicted word and fine-tuned word. The working of the fine-tuning phase is elaborated in Algorithm 2 and the dictionary updation is explained in Algorithm 3.

Algorithm 2: Word Fine-tuning

Input: *Recognised Word (output from Algorithm 1), Dictionary of contextual words*
Output: *Fine-tuned Word*
Process:

```
for each word in the dictionary:
    Compute the Levenshtein distance between this dictionary word and the recognised word
Choose the dictionary word with minimum distance as the probable candidate
if the edit distance of candidate is less than two-third of the length of the word
    return the candidate as the tuned word
else
    return the generated word
Update the dictionary
```

Algorithm 3: Dictionary Updation

Input: *Fine-tuned Word (output from Algorithm 2), Dictionary of contextual words, Intermediate Dictionary*
Output: *Updated Dictionary, Updated Intermediate Dictionary*
Process:

```
for every word in the intermediate dictionary:
    Compute the Levenshtein distance between the Fine-tuned word and this dictionary word
Choose the word from intermediate dictionary with minimum distance as the probable candidate
if the edit distance of candidate is less than two-thirds of the length of the Fine-tuned word:
    Increase the frequency of the candidate by one
    if the frequency equals three:
        Add candidate to the dictionary
        Remove candidate from the intermediate dictionary
else
    Add this new word to the intermediate dictionary
```

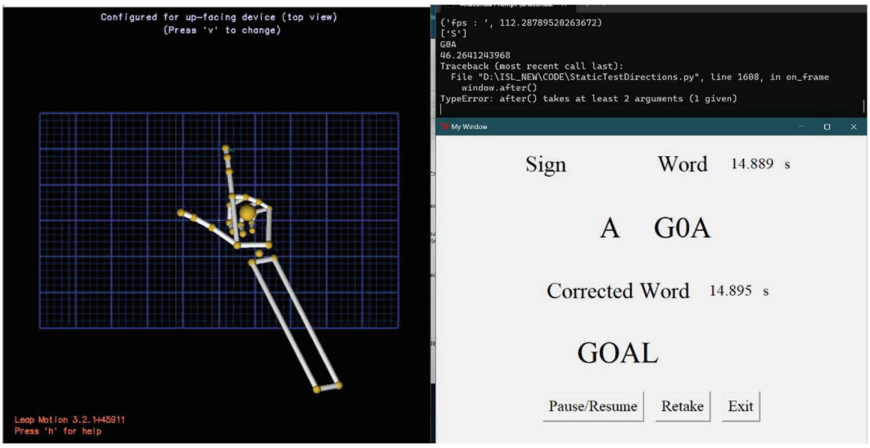

Fig. 7. Word "Goal" being performed. In this picture, 'Word' refers to the output from Recognition module and 'Corrected Word' refers to the output of the Fine-tuning module

5 Evaluation and Results

5.1 Evaluation of Features

Since we primarily focus to develop a system that can be used to recognise the signs accurately and quickly, we experimented with different feature sets to identify the best feature set which uses simple features and thus are appropriate for efficient real-time predictions under natural settings.

To compare the performance of the proposed feature set, we extracted aspect ratio and distance features that have been known to effectively distinguish between Indian sign language gestures and have been extensively utilized in existing systems. The study uses 24 such features, which are derived from the selected points. The features are divided into three categories.

1. The aspect ratio between the thumb and the other four fingers: Eight features, with four features per hand, are calculated.
2. The distance between the palm centre and the distal end points of the fingers: Ten features, with five features per hand, are calculated.
3. The distance between the corresponding fingers and palm centres of both hands: Six features are computed.

While the first set of features corresponds to the 24 widely used features extracted from the dataset as mentioned above, the second set of features is the coefficients of direction vectors, which are heuristically chosen, based on our approach. The model is tested against samples from existing signers and also from new signers recorded under natural settings.

5.2 Results and Observations

In order to evaluate the system performance in recognising signs under natural settings, we tested the system with data recorded from two new signers under conditions that were different from the training data. Accuracy and precision are chosen to be the metrics for comparison of models under natural settings for both sets of features (commonly used, proposed). From the results as shown in Table 3, we found that the Random Forest Classifier performed better in both settings. Our model was able to perform better in both the settings. The model using the traditional feature set predicted the signs with an average response time of 3.7 ms whereas the proposed model took 3.02 ms to predict the sign.

Table 3. Accuracy and Precision (%)

Model	With commonly used features		With proposed features	
	Accuracy	Precision	Accuracy	Precision
Multi-layer perceptron	63.7	44.0	51.9	56.8
Ada-Boost	26.4	9.0	22.5	18.8
Gradient Boost	53.9	45.1	49.0	50.9
Support vector machine	46.0	57.3	63.7	43.8
K-nearest neighbours	50.0	50.0	54.9	45.2
Random Forest	**67.7**	**65.7**	**71.0**	**65.7**

Table 4. Misclassified letters and their predictions

Misclassified signs	
{0, ‘o’}	{‘e’, ‘f’}
{‘d’, ‘p’}	{‘m’, ‘n’}
{2. ‘v’}	{‘i’, ‘t’, ‘x’}
{1, 7, ‘l’, ‘c’}	

According to the findings shown in Table 4, the signs that were similar such as (O, 0) were not easily distinguishable by the model. However, the model performed robustly in classifying the other signs. Specifically, M and N differ only by the presence of an extra ring finger in ‘M’ which is absent for ‘N’. In most of the cases, the presence of this finger is occluded by the signer’s palms or miscalculated due to minor variations in the orientation, leading to its misclassification. The justification for misclassification of ‘E’ and ‘F’ also follows the same reason. These minor shortcomings are effectively addressed and corrected in the word fine-tuning phase. Despite these inherent difficulties, the model demonstrates a robust functioning.

5.3 Performance

The performance of the model is assessed with the following metrics along with the model's accuracy and precision. The values are given in Table 5.

Negative Predicted Value. Negative Predictive Value measures the ratio of true negative predictions considering all negative predictions.

$$Negative\ Predicted\ Value = \frac{TN}{TN + FN} \tag{7}$$

Sensitivity. Sensitivity is a measure of how well a model can detect positive instances

$$Sensitivity = \frac{TP}{TP + FN} \tag{8}$$

Specificity. Specificity is a measure of how well a model can detect negative instances

$$Specificity = \frac{TN}{TN + FP} \tag{9}$$

F1 Score. F1 measures a model's accuracy. It is a harmonic mean of accuracy and precision

$$F1\ Score = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \tag{10}$$

False Positive Rate. False positive Rate is the proportion of negative cases incorrectly classified as positive

$$False\ Positive\ Rate = \frac{FP}{FP + TN} \tag{11}$$

Table 5. Performance metrics

Metric	Value
Accuracy	0.71
Precision	0.65
Negative Predicted Value	0.98
Sensitivity	0.62
Specificity	0.98
F1 score	0.66
False Positive Rate	0.01

5.4 Comparative Analysis

We conducted a comprehensive comparative analysis against existing methodologies in ISL recognition, as well as similar recognition methods in other sign languages in Table 6. The realm of real-time sign language recognition is notably less experimented with. So, our primary focus was on recent advancements in real-time ISL recognition employing the Leap Motion Controller [16], with due consideration for the state-of-art real-time implementations in other sign languages like [18, 20] and [21] to provide a broader context.

While the existing approaches demonstrated relatively higher accuracies compared to our proposed approach, it is noteworthy that the performance is primarily due to the predominance of single-handed signs in those languages. Some methods were also successful due to the use of limited dataset, which overlooked similar signs. However, some of these approaches did not explicitly validate their models with data from new signers recorded in natural settings, raising concerns about their generalizability.

It is also notable that many published studies rely on camera-captured images and videos as primary data sources, making it vulnerable to lighting variations, and certain constraints on clothing requirements (e.g., full sleeves). Furthermore, several existing methods predominantly relied on extensive datasets and intricate feature engineering, often lacking experimentation with real-time scenarios. In contrast, our method attains comparable real-time accuracy levels to these real-time techniques present in both ISL and other sign languages, all while using a simpler architectural design and a remarkable ability to generalize to new testers and settings. Our average response time for a single letter is 3.02 ms seconds without fine-tuning and 2 s with fine-tuning which outperforms the published the average time of [24], which is 4 to 7 s. Moreover, our system is also robust to position and anthropometric changes. This underlines the efficiency and effectiveness of our approach in obtaining comparable performance metrics.

Table 6. Comparative Analysis

Authors	Signs	No. of Features	Method	Real-time/Natural Setting	Validation	Validation Accuracy (%)
Mariappan et. al.,	ISL (80 words, 50 sentences)	Visual feature vectors of video frames	Fuzzy c-means	Yes/No clustering	Not Available	75 (testing accuracy)
Bird et. al.,	BSL(18 gestures)	>150	DNN	No/No	Unseen data	41.78

(*continued*)

Table 6. (*continued*)

Authors	Signs	No. of Features	Method	Real-time/Natural Setting	Validation	Validation Accuracy (%)
Lee et. al.,	ASL (26 single-handed)	30	LSTM	Yes/No	Not Available	91.82
Hisham et. al.,	Arabic (20 single hand + 10 double hand)	70	Ada-Boost + Dynamic Time Wrapping	Yes/No	Yes/No	93 (testing accuracy)
Our approach	ISL (15 singlehanded, 19 double-handed)	30	Random Forest Classifier with Word fine-tuning	Yes/Yes	Unseen data	71

6 Conclusion

We proposed and successfully implemented an efficient ISL fingerspelling recognition system for real-time use under natural settings. The system is much simpler than the existing ones and shows comparable performance in terms of speed and accuracy. Intelligent feature selection and heuristics-based word fine-tuning have enabled faster, accurate yet simpler system while avoiding the need for delimiters and other cumbersome practices. The accuracy is close to 71% when tested with unseen data under natural settings. Additionally, we proposed a word refinement phase to address potential misclassifications arising from similar signs. Future works can include word level continuous sign language recognition, profiling the users by including regional language pidgins and exploring newest sensors for multimodal recognition with support of multiple platforms.

Acknowledgement. This work is supported by the Department of Science and Technology (DST), Government of India, under the scheme: SEED/TIDE/2018/21. The authors would like to thank the management, teachers, staff and students of CSI Higher Secondary school for Deaf, Santhome High Road, Mylapore, Chennai-600004, for their wholehearted support during recording of the data. Authors would like to thank the Research Scholars and students of Dept. of CSE, Anna University, Chennai – 600025, Mr. Selvin Ebenezer, Ms. Akshara M S, Ms. Jesima A for their active participation and contribution during recording of the signs and later.

References

1. Indian Sign Language Research and Training Centre (ISLRTC). http://www.islrtc.nic.in/
2. Rao, G.A., Syamala, K., Kishore, P.V.V., Sastry, A.S.C.S.: Deep convolutional neural networks for sign language recognition. In: Proceedings of the 2018 Conference on Signal Processing and Communication Engineering Systems (SPACES), pp. 194–197. IEEE, Vijayawada, India (2018)

3. Bhagat, N.K., Vishnusai, Y., Rathna, G. N.: Indian sign language gesture recognition using image processing and deep learning. In: Proceedings of the 2019 International Conference on Digital Image Computing: Techniques and Applications (DICTA), pp. 1–8. IEEE, Perth, Australia (2019)
4. Wadhawan, A., Kumar, P.: Deep learning-based sign language recognition system for static signs. Neural Comput. Appl. **32**, 7957–7968 (2020)
5. Sharma, M., Pal, R., Sahoo, A.K.: Indian sign language recognition using neural networks and kNN classifiers. J. Eng. Appl. Sci. **9**, 1255–1259 (2014)
6. Kong, W.W., Ranganath, S.: Towards subject independent continuous sign language recognition: a segment and merge approach. Pattern Recogn. **47**(3), 1294–1308 (2014)
7. Dong, C., Leu, M. C., Yin, Z.: American Sign Language Alphabet Recognition using Microsoft Kinect. In: Proceedings of 2015 Computer Vision and Pattern Recognition Workshops (CVPR), pp. 44–52. IEEE, Boston, USA (2015)
8. Ansari, Z.A., Harit, G.: Nearest neighbour classification of indian sign language gestures using kinect camera. Sādhanā **41**(2), 161–182 (2016)
9. Raheja, J.L., Mishra, A., Chaudhary, A.: Indian sign language recognition using SVM. Pattern Recognit. Image Anal. **26**, 434–441 (2016)
10. Raghuveera, T., Deepthi, R., Mangalashri, R., Akshaya, R.: A depth-based Indian Sign Language Recognition using Microsoft Kinect. Sādhanā **45**, 34 (2020). https://doi.org/10.1007/s12046-019-1250-6
11. Leap Motion Controller. https://www.leapmotion.com
12. Naglot, D., Kulkarni, M.: Real time sign language recognition using the leap motion controller. In: Proceedings of 2016 International Conference on Inventive Computation Technologies (ICICT), pp. 1–5. IEEE, Coimbatore, India (2016)
13. Chuan, C.-H., Regina, E., Guardino, C.: American sign language recognition using leap motion sensor. In: Proceedings of 13th International Conference on Machine Learning and Applications, pp. 541–544. IEEE, Detroit, USA (2014)
14. Kumar, P., Gauba, H., Pratim Roy, P., Prosad Dogra, D.: A multimodal framework for sensor based sign language recognition. Neurocomputing **259**, 21–38 (2017)
15. Chavan, P., Ghorpade, T., Padiya, P.: Indian sign language to forecast text using leap motion sensor and RF classifier. In: Proceedings of 1st IEEE Symposium on Colossal Data Analysis and Networking (CDAN), pp.1–5. IEEE, Indore, India (2016)
16. Muthu Mariappan, H., Gomathi, V.: Real-time recognition of indian sign language. In: Proceedings of 2019 International Conference on Computational Intelligence in Data Science (ICCIDS), pp. 1–6. IEEE, Chennai, India (2019)
17. Kothadiya, D., Bhatt, C., Sapariya, K., Patel, K., Gil-González, A.B., Corchado, J.M.: Deepsign: sign language detection and recognition using deep learning. Electronics **11**, 1780 (2022)
18. Bird, J.J., Ekárt, A., Faria, D.R.: British sign language recognition via late fusion of computer vision and leap motion with transfer learning to American sign language. Sensors **20**(18), 5151 (2020)
19. Holmes, R., et al.: From scarcity to understanding: transfer learning for the extremely low resource irish sign language. In: IEEE/CVF International Conference on Computer Vision, pp. 2008–2017 (2023)
20. Lee, C.K., Ng, K.K., Chen, C.H., Lau, H.C., Chung, S.Y., Tsoi, T.: American sign language recognition and training method with recurrent neural network. Expert Syst. Appl. **167**, 114403 (2021)
21. Hisham, B., Hamouda, A.: Arabic sign language recognition using ada-boosting based on a leap motion controller. Int. J. Inf. Technol. **13**, 1221–1234 (2021)

Author Queries

Chapter 26

Query Refs.	Details Required	Author's response
AQ1	This is to inform you that corresponding author has been identified as per the information available in the Copyright form.	
AQ2	Reference [24] is cited in text but not provided in the reference list. Please provide reference in the list or delete this citation.	