CLASSIFIER FOR SINHALA FINGERSPELLING SIGN LANGUAGE

Akalanka Weerasooriya

(209392B)

Thesis submitted in partial fulfillment of the requirements for the degree Master of Science in Computer Science

Department of Computer Science and Engineering

University of Moratuwa

Sri Lanka

June, 2022

DECLARATION

I declare that this is my own work and this dissertation does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my thesis, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Signature : UOM Verified Signature P.A.A.Weerasooriya 7/5/2022 Date: 05.07.2022

Name: P. A. A. Weerasooriya

The above candidate has carried out research for the Masters Dissertation under my supervision.

Signature of the supervisor: UOM Verified Signature Date: 05-07-2022

Name: Dr. Thanuja D. Ambegoda

Abstract

Computer vision based sign language translation is usually based on using thousands of images or video sequences for model training. This is not an issue in the case of widely used languages such as American Sign Language. However, in case of languages with low resources such as Sinhala Sign Language, it's challenging to use similar methods for developing translators since there are no known data sets available for such studies.

In this study a sign language translation method is developed using a small data set for static signs of Sinhala Fingerspelling Alphabet. The classification model is simpler in comparison to Neural Networks based models which are used in most other sign language translation systems.

The methodology presented in this study decouples the classification step from hand pose estimation and uses postural synergies to reduce dimensionality of features. This enables the model to be successfully trained on a data set as small as 122 images. As evidenced by the experiments this method can achieve an average accuracy of over 87% . The size of the data set used is less than 12% of the size of data sets used in methods which have comparable accuracies.

Code and datasets are available at: `https://github.com/aawgit/signs`
Keywords: Vision, Sign language, Sinhala, fingerspelling

ACKNOWLEDGEMENT

I would like to convey my gratitude to my supervisor Dr. Thanuja D. Ambegoda for his guidance, continuous supervision and assistance. I have been extremely fortunate to have him as my supervisor, whose guidance and knowledge helped me to improve. His motivation, persuasion and patience helped me to overcome many obstacles and complete this research successfully.

I'm grateful to Mr. Buddika Gunathilaka, Ms. Samantha Madurangi and Ms. Geshani Amila who are professional sign language teachers and interpreters who contributed to building the data set amidst various challenges. Furthermore, Mr. Gunathilaka helped me to understand the background of sign languages in Sri Lanka and the usefulness of translation systems. Their knowledge and experience were vital to lay the foundation for the research. I also thank Ms. Sunitha Karunaratna of The Ceylon School for the Deaf & Blind, Mr. Brayan Susantha and Mr. Janaka Perera of Sri Lanka Central Federation of the Deaf for connecting me with resource people of their organisations.

This wouldn't have been possible without the love and support of my family. They have been a continuous source of love, concern, support and strength all these years. I am grateful to them for believing in me, cheering for me and being patient and supportive during this critical period in my academic life, which motivated me to successfully complete the research.

Contents

List of Figures

List of Tables

Acronyms and Abbreviations

Abbreviation	Description
ASL	American Sign Language
ANN	Artificial Neural Networks
CNN	Convolutional Neural Networks
CPM	Convolutional Pose Machine
DIP	Distal interphalangeal
DNN	Deep Neural Networks
DoF	Degree of freedom
DTW	Dynamic Time Warping
ELM	Extreme Learning Machine
FASSL	Fingerspelling Alphabet of Sinhala Sign Language
FPS	Frames per second
FV	Fisher Vector
HPT	Hyper Parameter Tuning
IP	Interphalangeal
KLT	Kanade–Lucas–Tomasi
KNN	K-Nearest Neighbour
LR	Logistic Regression
LSTM	Long Short Term Memory
MCP	Metacarpophalangeal
PCA	Principal Component Analysis
PIP	Proximal interphalangeal
RF	Random Forest
SORT	Simple Online and Realtime Tracking
SSD	Single Shot Detector
SSL	Sinhala Sign Language
ST-LSTM	Spatio-Temporal LSTM
TMC	Trapeziometacarpal

1 INTRODUCTION

1.1 Background

Sign language is a system of communication that uses visual gestures and signs. Sign languages consist of three main parts which are manual features consisting of gestures made with hands, non-manual features such as facial expressions, and fingerspelling [2].

Fingerspelling is a method of spelling words using hand movements [3]. Fingerspelling signs can be divided into two types, one being dynamic signs which are defined by a sequence of poses and the other being static signs which are defined by a single pose which doesn't vary with time. The Fingerspelling Alphabet of Sinhala Sign Language (FASSL) has signs for vowels and consonants of Sinhala Language. Some of these signs are static, while others are dynamic. Some signs used in the FASSL are shown in Fig. 1.

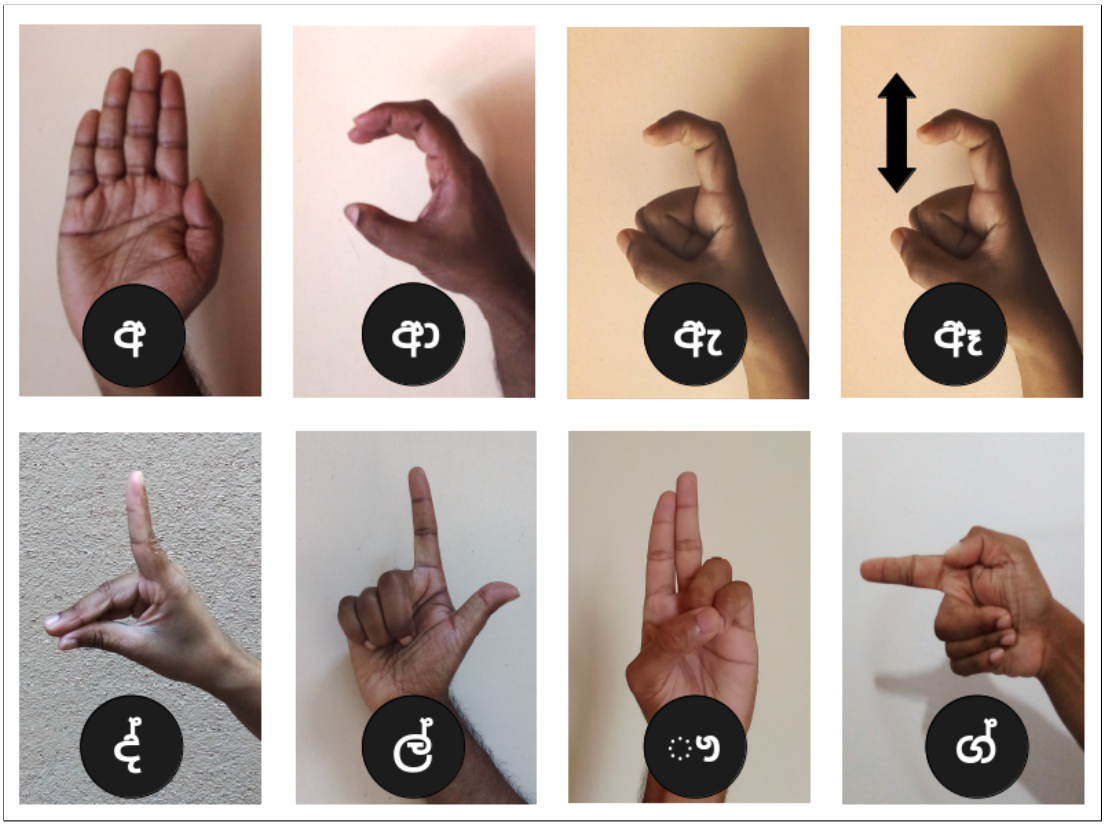

Figure 1.1: Some signs from Fingerspelling Alphabet of Sinhala Sign Language

1.2 Research problem

Sign language is used for communication with and among deaf and mute people in Sri Lanka as the preferred language [4] [5]. There is a Sinhala Sign Language (SSL) as well as a Tamil Sign Language. Further, there are several dialects adopted by different teaching institutes in different areas of the country [6].

The majority of the Sri Lankan population doesn't understand SSL [5]. There have been some attempts in developing systems to capture the hand and body movements and translate them to Sinhala Language. Fernando et al. [5] have developed a system which can translate 15 words. Although it has a good accuracy for the selected words, fingerspelling is not included. Further, there has been no known study carried out to develop a translator for the FASSL.

Many studies have been done for classification of widely used sign languages such as American Sign Language (ASL). Hence, there are datasets containing thousands of data points for such languages. However, in the case of FASSL there is no such known dataset.

In recent years, there have been significant improvements in the field of human body pose estimation using images as inputs. Such pose estimations can be used in various applications including sign language translation. The aim of this research is to develop a computer vision based translator for static signs of FASSL utilizing hand pose estimation techniques.

1.3 Research Objectives

Objectives of this study are as follows:

- Build a labelled dataset which can be used for developing a classifier for FASSL.
- Develop a hand pose based classification method for fingerspelling which can be trained using a substantially smaller dataset relative to the existing sign language translators, and contextualize the said classifier for static signs of FASSL.
- Implement an end-to-end real time translator using the above classifier such that it can run on general purpose computers such as laptops.

2 LITERATURE REVIEW

The literature review is carried out under 4 main areas, which are different approaches to sign language translation, hand detection & tracking methods, pose estimation methods, and pose classification methods.

2.1 Different approaches to sign language translation

A widely used approach in the domain of vision based sign language translation is extracting features from images and classifying using Artificial Neural Networks (ANN) such as Convolutional Neural Networks (CNN). The feature extraction is usually preceded by image segmentation to extract the hand or arms.

The method presented by R. Akmeliawati et al in [7] involves 3 steps. Segmentation & tracking the hand, extracting gesture, and then classifying the gesture. Segmentation is done based on skin colour and tracking is done by a method called Mean-Shift Algorithm. An ANN with 45 inputs and 14 outputs has been used for classification. It identifies 13 gestures with an accuracy of 96.02%.

Lean Karlo S. Tolentino et al's [8] system is capable of detecting 35 signs including 10 numbers. The system finds a hand placed in the predefined area of the image based on skin colour and feeds it to a CNN, which does the feature extraction and classification. According to experimental results, the overall accuracy is 93.667% and the best performance in terms of recognition time is 2.66s. However, some signs and angles of signs have been modified in order to be used with the system. Further the system needs a background with uniform colour, and the lighting level has to be calibrated.

In [9] Lionel Pigou et al present a method to recognize 20 signs used in Italian language with an accuracy of 91.70%. Their training set is a set of human pose videos containing depth and joint position information. The training process consists of a preprocessing step to crop a hand by using joint positions. The proposed model comprises two CNNs for extraction of hand features & upper body features.

Ruslan Kurdyumov et al present a Support Vector Machine (SVM) based method in [10] to recognize 25 static signs. At the preprocessing stage the silhouette of the hand is extracted from the background and then its size and position are normalized.

According to the experimental results, the highest accuracy is 93.5%.

Rekha et al have used Dynamic Time Warping (DTW) in [11] for recognizing dynamic gestures. Their approach consists of 3 steps which are segmentation using skin colour, hand feature extraction, and classification of gestures. Recognition rate is 86.3% for dynamic gestures.

An alternative approach to the image feature based recognition of signs is deriving the underlying hand/ body pose as a skeleton model and recognizing signs based on features extracted from the said model. Both methods follow the same steps up to feature extraction for classification.

Sanalohit et al [12] have followed the said approach for recognizing Thai Sign Language and have achieved an accuracy of 84.57%. Although there aren't many published studies for the pose based approach in the context of fingerspelling sign language, it is a method successfully used in gesture recognition, which is a superset of the sign language recognition problem. In this study, a pose based approach is adopted and the rest of literature review is focused on main steps involved in such an approach.

2.2 Hand detection and tracking

As the first step of the classification pipeline, the position of the hand i.e. the Region of Interest (RoI) has to be determined. There are two broad areas of solutions. First one is using skin colour and motion based detection. The second is using Deep Neural Networks (DNN). In case of video data, the RoI has to be determined for each video frame. Instead of running a hand detection process for each frame, in some studies a tracking method is employed once a hand is detected [13] [14] [15].

In [13] if there is a face in the image, it is detected and removed first. Then the difference among first the 3 frames is calculated in color and grayscale. This difference and results of the skin filtering is combined to detect a hand. Once a hand is detected, it's tracked by using a method called modified Kanade–Lucas–Tomasi (KLT) algorithm.

A similar approach has been followed by Singha et al [16] for detecting hands where the input RGB image is converted to the HSV first and then the HSV image is filtered for skin colour. To eliminate other skin colored objects, the largest binary linked object (BLOB) is selected as the hand.

Zhang et al [14] have used a DNN based approach for hand detection. They have

used a RetinaNet neural network architecture, which is a CNN based model. The model had been pre trained on hand image data sets and then trained again on hand image data specific to their problem domain. Hand tracking is done using the Simple Online and Realtime Tracking (SORT) algorithm.

In [15] the palm is detected instead of the hand by using a Single Shot Detector (SSD) Model. According to the said study, estimating bounding boxes of rigid objects like palms is simpler in comparison to detecting hands. They have achieved an average precision of 95.7% in palm detection.

2.3 Pose estimation

The review on pose estimation and classification methods described below is not limited to hand pose. There are interesting methods presented in the literature relating to body pose as well. Both hand pose and body pose literature were reviewed since findings of the latter can be applied to the former.

Pose estimation step takes an image or video frame as the input and returns an estimation of the skeleton in 2D or 3D space. There are two main approaches in estimating a pose using images, based on the input type. One is using depth images which contain depth values for each pixel in addition to its RGB values. The other is using only RGB values without information on the depth. This research aims to use images without depth information, hence the literature on the latter was reviewed. The amount of data required for training a RGB image based model is larger than the amount needed to train using depth values [17]. Therefore studies in the area of RGB image based estimation have paid special attention to solving the problem of making large data sets.

In Pose Machines introduced in [18] there are multiple stages, at each of which a classifier predicts confidences for locations of each anatomical landmark (body part). A classifier in a certain stage takes features of the image data and the result of the previous classifier as inputs. There is a hierarchical approach in detection of parts such that at the coarsest level, the part is the whole body, and at the finest level parts are atomic parts such as a knee. At each stage a Random Forest (RF) model is used as the classifier.

The Convolutional Pose Machine (CPM) presented in [19] is based on the same architecture as [18] but uses a sequence of CNNs for detecting landmarks.

Simon et al proposed a method to improve the performance of a pose estimator using multiple cameras in [20]. The detectors architecture used here is based on PMs [19], except it uses the convolutional stages of a pre-initialized VGG-19 network - which is a CNN model, as a feature extractor. A detector is initially trained on a labelled dataset. This data set is not enough to generate a detector with the required accuracy. Then they used multiple cameras to capture a hand and carry out a process called triangulation. In the triangulation process the detections for each view of the hand is used to construct a 3D model of the pose. The 3D points for each point generated by the triangulation step are then projected to badly estimated views of the same pose to create a new set of labelled data. Then a detector is trained on the extended dataset. There are 6 stages in detection in series. Input at each stage is image features and the score maps from the previous stage. With this set up the system was able to estimate poses of multiple hands for a group of people by using 31 cameras.

The estimation method followed in [21] is based on a 2.5D representation, which is a set of (x, y, z) coordinates where x and y are pixel coordinates of the image and z is the depth relative to the palm. First, a 2.5D pose is generated from a RGB image using a CNN, and then the result is used to generate a 3D pose. Panteleris et al [22] proposed a similar approach where they use a pretrained model to estimate 2D pose for a RGB image and then lift the estimations to 3D by using camera parameters. This method works with commodity RGB cameras. However, it's assumed that the parameters of the camera such as focal length, distortion etc., are known.

2.4 Pose classification

Pose a classification is a problem having $P \times$C input features for static gesture recognition, where P is the number of points in the skeleton model and C is the number of coordinates representing a point. C is 2 and 3 for 2D and 3D models respectively. In most of the recent studies, P is taken as 21 [15], [21], [22], [17] for hand skeleton models. When there are dynamic gestures, temporal features also need to be taken into account in the classification.

Prior to the classification step, the inputs need to be pre-processed to ensure that it's suitable to define a good descriptor. The said pre-processing steps involve normalizing the coordinates in terms of position, size and rotation.

Zhang et al [15] have employed a simple algorithm where the state of each finger, e.g. bent or straight, is determined from pose data, and then states are mapped to a set of predefined gestures for classification.

In [23] a sequence of 3D coordinate sets is taken as input, and to preserve temporal data, two more components are added which represents the difference of a frame relative to an arbitrary offset frame and the difference between two consecutive frames. The final descriptor, i.e., the concatenation of the said 3 components is used as the input to the training and testing pipeline which consist of Principal Component Analysis (PCA), Fisher Vector (FV) encoding, and Extreme Learning Machine (ELM) classifier. PCA is used to reduce the dimensions of the descriptor. FV encoding is applied in each sequence to all features resulting from the PCA step, and the resulting fixed sized FVs are used for the classification. These FVs consist of partial derivatives of mean and standard deviation of the descriptors. ELM is a learning algorithm which analytically determines the output weights for feedforward neural networks having a single-hidden layer [24].

Temporal data is captured using a CNN in [25] by first converting a series of poses into a 2D image where the vertical axis and horizontal axis correspond to time and location of each point respectively. Different channels are used for 3 coordinates of joints of the pose. 2D convolutions are used to extract patterns directly from a temporal sequence of body joints from the said 2D image. Subsequently, a max plus min pooling followed by a Softmax activation is performed on the action heat map.

Liu et al have presented a Spatio-Temporal Long Short Term Memory (ST-LSTM) model in [26] which simultaneously models the spatial and temporal information of a series of poses represented in 3D coordinates. Each ST-LSTM unit corresponds to one of the skeletal joints. Each of these units receives the hidden representation of the previous joint and also the hidden representation of its own joint from the previous frame. They have also introduced a gating mechanism to filter out noise, which analyzes the reliability of the input considering the position of its adjacent joint and its own position in the previous frame. If the gate determines that an input has wrong 3D coordinates, it will prevent the memory cell from updating with that input.

Based on ST-LSTM, Liu et al have presented another method for dynamic 3D pose classification in [27] which has a different approach to filter out the less important in-

puts. In addition to the ST-LSTM network, there is a second LSTM network, which takes the hidden representations from the first network as its inputs. There is a global context memory which maintains an overall representation for the whole action sequence. The memory cells of the second ST-LSTM layer are updated only if the inputs are important in relation to the global context. The output of the second network is used to update the global context memory, which in turn is used for classification.

In [28] Baradel et al have proposed a model which takes two data streams as inputs to classify activity sequences.The model utilizes a stream of 3D pose information and a stream of RGB images and classifies activities involving one or two people and their interactions. The proposed system consists of a convolutional model, input to which are coordinates of each joint concatenated over time. The Inputs consist of 3 channels, corresponding to raw coordinates, derivatives of coordinates (velocities), and the second derivatives (accelerations) respectively. The output of the CNN model is used to crop the images in RGB stream at attention point and feed them to a RNN model which classifies actions.

In [29] Pham et al generate a 2D image using pose and motion, which represents the action. This representation is termed Enhanced-SPMF (Enhanced Skeleton Pose-Motion Feature) and comprises two main aspects, pose, and motion. First, the distances between each joint are calculated and normalized to obtain values between 0 and 1. Then each distance value is mapped to a colour scale, and represented in 3 channels for RGB colours. The orientation is measured by taking the vector difference of two joints and mapped to the colour space. The resulting two 3D vectors (position and orientation) are concatenated to form the pose feature. Motion features are calculated in a similar manner by taking joints of two consecutive frames to calculate distance and orientation. Finally, the pose features and motion features are concatenated to form a feature vector which represents the action from beginning to end, which is used for classification after further processing. A CNN is used for classification.

In [30] Wang et al first identify different poses a body part can have and create dictionaries for such different poses. Each dictionary contains poses for the relevant body part, which are head arms and legs. The said poses are indexed, and the body pose is represented with five indices corresponding to the poses of 5 body parts. The poses are learnt using data mining techniques, considering a table containing poses as

a transaction table and mining patterns for classes of poses. A histogram of the poses is used as features for the classification using SVMs.

A summary of reviewed techniques on hand detection & tracking, pose estimation and pose classification are presented in Table 1.

Table 2.1: A summary of hand detection, tracking, pose estimation and classification techniques

Task	**Techniques**	**Reference**
Hand detection	Skin filtering by color	[16], [13]
	CNN-based (RetinaNet [31], SSD [32])	[14], [15]
Hand tracking	Modified KLT algorithm.	[13]
	SORT algorithm [33]	[14]
	The bounding box from the landmark prediction of the previous frame as input for the current frame	[15]
Pose estimation (hand/ body)	A sequence of classifiers, each taking score maps from the previous stage, and image features as input. (RF and CNN models used as the classifier in different papers).	[18], [20], [19]
	Generating a 2.5D pose from a RGB image using a CNN and converting to 3D using camera parameters.	[21]
	Estimating 2D joints using a CNN-based method and then lifting the 2D estimations to 3D as an inverse kinematics problem.	[22]
Pose classification (hand/ body)	The state of each finger (bent or not) mapped to a set of predefined gestures.	[15]
	ELM classifier [24] with features as the current pose and temporal differences of poses with FV encoding.	[23]
	Temporal convolution (Encoding spatial and temporal information to image form and using that as a input for a CNN)	[25], [28], [29]
	LSTM (ST LSTM, Global Context-Aware Attention LSTM)	[26], [27]
	Bag of visual words	[30], [34]
	Dynamic Time Warping	[34]
	Hidden Markov Models	[35]

3 METHODOLOGY

The proposed system first generates a 3D hand pose from a video/ image input and then classifies the pose to recognize the signs. The input video/ image is generated from a single view. Depth information or parameters of the camera are not used. The system consists of the following 3 main steps.

1. Hand detection, tracking and pose estimation
2. Pre-processing
3. Pose classification

The high level architecture of the system is presented in Figure 3.1.

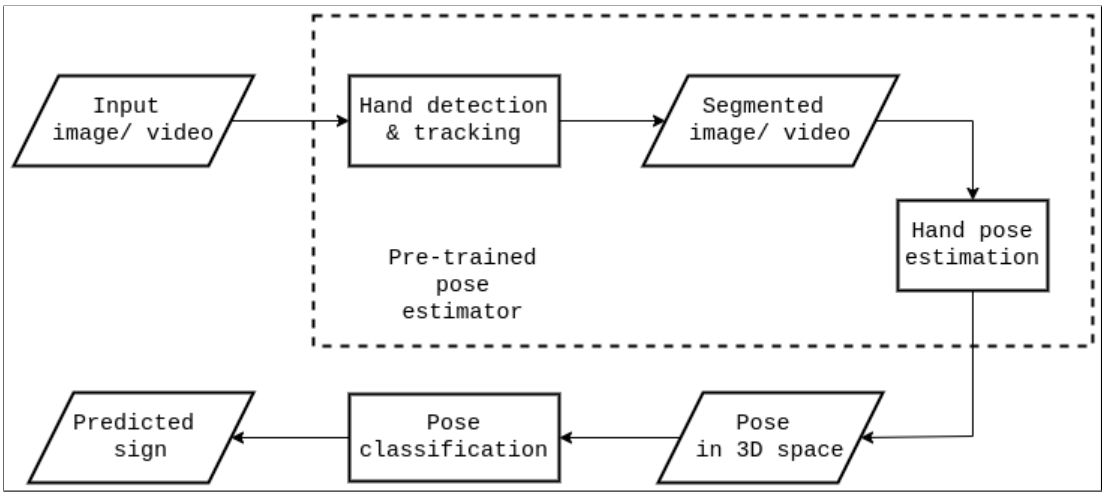

Figure 3.1: High level architecture of the proposed system

This modular approach reduces the complexity of the pose classification step as opposed to training a pose classifier directly on images, thus making it possible to train the system on a smaller dataset in comparison to the latter approach. Further, this approach utilizes the existing methods and pre-trained models for up and including the pose estimation step.

3.1 Dataset

The dataset was created using 2 educational videos sourced from YouTube, 3 videos created by sign language teachers, and 4 photo sets created by novices. The videos

contain all the signs (58), while photo sets contain only the static signs (27). There is a variety in frames rates, resolutions and backgrounds among sources. In each video a signer performs letters in the alphabet sequentially. Each photo set contains up to 6 images per sign. The dataset was validated before annotating, and incorrect signs were removed. Each letter in the alphabet was given a unique index for annotation and internal functions of the system. The Images have been annotated using the image name against its index whereas videos have been annotated using the start frame number and the end frame number against the relevant index. All annotation files are in Comma Separated Value (CSV) format and available with the dataset.

2 image sets and 2 videos were used for training & validation and the rest were used as the test set. The size of the training set is 122 and its distribution is shown in Figure 3.3

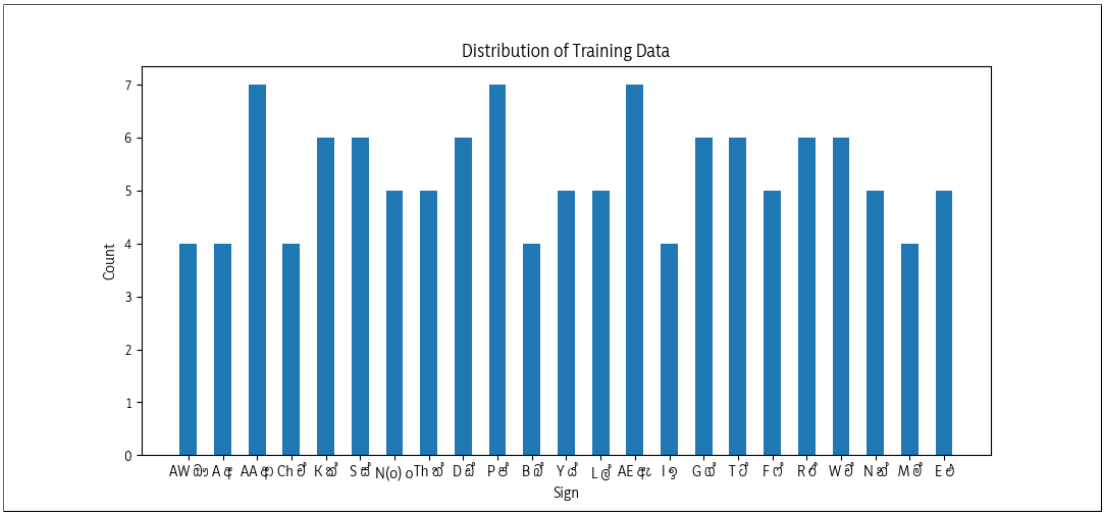

Figure 3.2: Distribution of training data by sign

3.2 Hand detection, tracking and pose estimation

There are multiple pre-trained hand pose estimators available, and most of these estimators [15], [21], [22] return the pose as a set of coordinates of 21 points which is termed as the hand landmark. Considering the accuracy, speed, availability, and effort to set up & integrate, the estimator in [15] was selected. The estimator uses a CNN model for pose estimation, the architecture of which is a Convolutional Pose Machine [19]. There is an integrated hand detecting & tracking component which also uses a

CNN model. There is a pretrained model of the estimator available as a Python module. Details of the selected estimators implementation are shown in the Table 3.1 .

Table 3.1: Details of the selected pose estimator. Source: [1]

Criteria	**Value**	**Description**
Mean 3D error	1.4 cm	Mean Absolute Error in Euclidean 3D metric space
Speed	48 - 120 FPS	On laptop/ desktop computers

The estimator also returns whether the detected hand is left or right (handness). This information becomes very useful in the classification model as it can be used independent of the handness by using a mirrored pose if its a left hand. Further, the estimator is capable of detecting 2 hands in an image. However, since the signs of FASSL are performed using only one hand, the estimator was configured not to detect 2 hands.

3.3 Feature selection

The output of the pose estimator is the locations of the skeletal joints of the hand in the form of 3D cartesian coordinates. Useful features for classification were derived using the said coordinates. Identifying important features and reduction of dimensions are normally done using methods such as PCA. However, as the aim of this study is to develop a classifier using a small dataset, methods such as PCA are not applicable since the success of such depends heavily on the availability of a large dataset. Therefore an alternative approach was adopted.

The human hand has a degree of freedom (DoF) of 27 [36]. However, studies have shown that a hand pose can be represented with a model of lower DoF because movements of some limbs are correlated with each other [36], [37], [38]. This is termed as postural synergies. In the referred studies this has been analyzed using PCA and most important features describing a pose have been identified.

In [37] Cobos et al show that one extension/flexion angle of all fingers and adduction/ abduction angle of thumb and index finger are important features for gesture reconstruction. Based on findings of the said studies we developed our baseline classifier using the Metacarpophalangeal (MCP) joint angles of all fingers, Proximal interphalangeal (PIP) joint angles of fingers except for thumb, Trapeziometacarpal (TMC) joint angle of thumb, and adduction/ abduction angle of thumb and index finger as

features. However, it was observed that the accuracy marginally improved when Interphalangeal (IP) angle of thumb, and Distal interphalangeal (DIP) angles of other fingers were added. Therefore, all extension/flexion angles of finger joints and adduction/ abduction angle of thumb & index finger were used.

Further, the said feature vector was concatenated with flattened coordinates of non stationary joints, which improved the accuracy.

Furthermore, to distinguish the signs which only differ in the orientation of the palm, it taken as a feature.

Selected angles and landmark points for features are shown in Figure 3.3

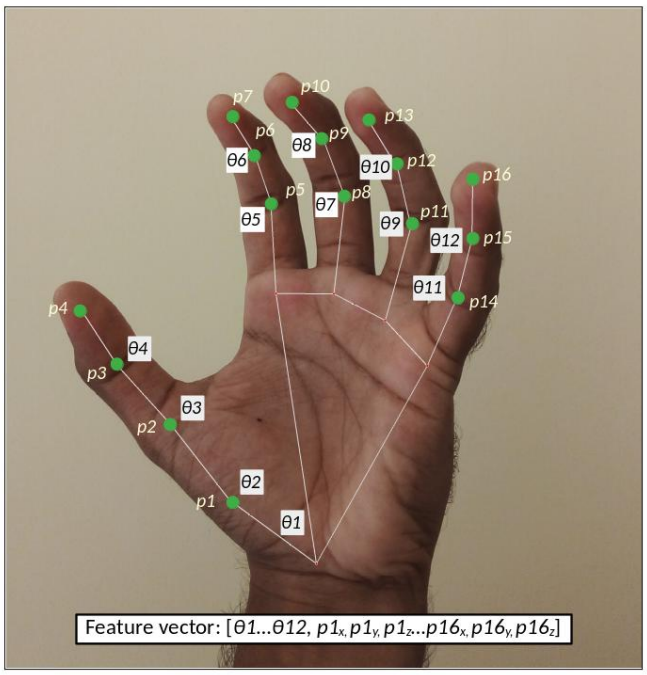

Figure 3.3: Selected angles and landmark points for features

3.4 Pre-processing

With respect to the angle based features, no preprocessing was necessary. However, to use coordinates, they had to be normalized in order to remove variations resulting from camera angle, distance between camera and the signer, size of the hand, and the position of the hand in the image frame. Therefore following preprocessing steps are done:

- Removing movement: Hand was fixed on the wrist joint and it was taken as the origin of the coordinate system

- Removing rotations: 2 reference lines were identified for removing rotations. Reference line 1 was selected as the imaginary line which connects the wrist joint to the mean of the bases of index, middle, ring and small fingers, whereas the reference line 2 is the imaginary line which connects the small fingers base to the index fingers base. Rotations were removed in such a way that the resulting hand model's reference line 1 becomes coincident with the y axis and the projection of reference lines 2 on the x, z plane becomes coincident with the x axis. It was observed that when the reference line 1 is set this way it results in a more symmetrical pose around the y axis, which reduces some unnecessary variations in the poses.

- Scaling the dimensions to convert to a predefined size: The length of the line connecting the wrist joint to the base of the middle finger was taken as the reference and the model was scaled such that this length becomes 1.

Although the rotations are removed, they are extracted and fed to the classifier at a later stage in the pipeline. There are limitations in the pose which could not be corrected by preprocessing which are differences in limb size ratios from person to person, personal variations in signs, and the estimators inaccuracies.

Figure 3.4 shows the inputs and outputs of the pose estimation and preprocessing steps.

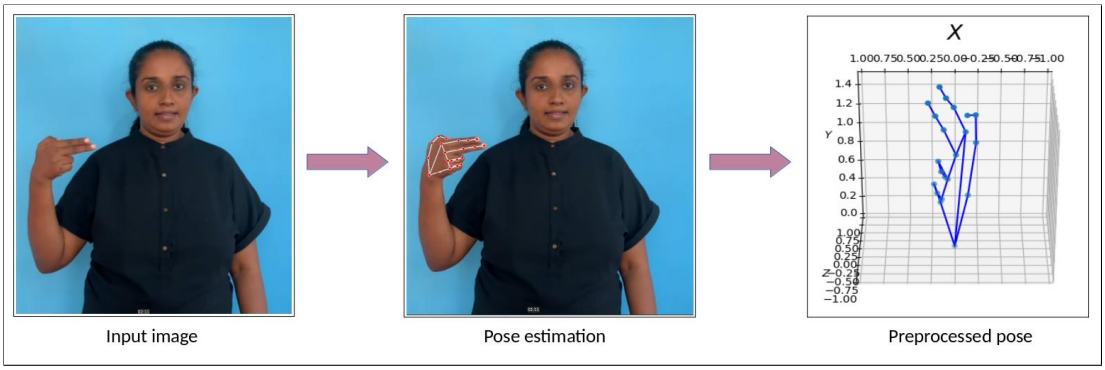

Figure 3.4: Pose estimation and preprocessing

3.5 Pose classification

Classification of poses in the context of FASSL involves dynamic pose classification, and hence, a classifier which solely relies on the hand pose of a single frame is not

capable of identifying the complete alphabet. There are 58 signs in the FASSL and 27 of them are static. In this study, 26 static signs out of the 27 were classified. The symbol ◌a: was omitted because the thumb is hidden between other fingers in the sign, and therefore the estimator failed to give a correct output in most cases.

Classification happens at two levels. At the first level, the normalized hand pose is classified using the selected angles and coordinates. Since rotation of the hand is removed during normalization, another level of classification is used which takes the orientation of the palm and prediction of the classifier at the first level as features. The architecture of the classifier is shown in Figure 3.5.

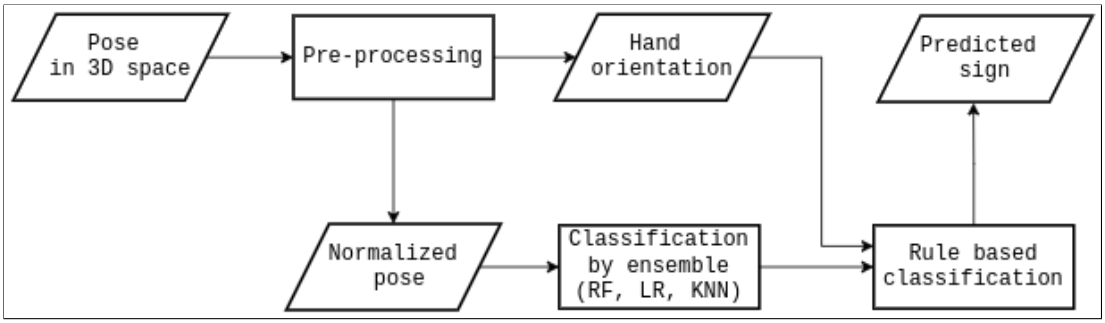

Figure 3.5: Architecture of the static sign classifier

Details of the classifiers are as follows.

3.5.1 Level 1

An ensemble consisting of K-Nearest Neighbour (KNN), Random Forest (RF), and Logistic Regression (LR) is used for classifying the normalized pose.

Hyper Parameter Tuning (HPT)

HPT was done on 2 levels. First for individual models and then for the ensemble. In the first case the training set of 122 pose estimations was used to carry out repeated stratified 4 fold cross validation. 4 was selected as each sign had at least 4 samples. For each model a grid search was carried out. Details of the determined parameters of the models are as follows:

- KNN classifier:
  - No. of neighbours: 3
  - Weight function: Distance
  - Distance matric: Minkowski with p = 4
- RF classifier:
  - No. of trees: 100
  - Criterion (function to measure the quality of a split): Entropy
  - No. of features to consider for splitting: 6 (Logarithm of the total number of features to the base 2)
- LR classifier
  - C (Inverse of regularization strength): 10
  - Penalty: L2
  - Algorithm: Newton-CG

LR models are used for binary classification problems. In order for it to be used in a multi-class classification, the problem was treated as set of separate binary classifications and separate classifiers were trained for all classes. This option was validated in comparison to Softmax Regression (SR), which is a generalization of the LR model which supports multinomial classification. The average accuracy for the cross validation were 0.7891 and 0.7788 for LR and SR respectively. The LR model was selected considering its marginally better accuracy.

Having a small training dataset increases the risk of overfitting. Following measures contributes to reducing that. Having 3 neighbours in KNN model as opposed to a smaller number helps to avoid overfitting in addition to giving the best accuracy during HPT. In case of the LR model, L2 regularisation reduces the possibility of overfitting. The value of Inverse of regularisation strength (C), and selection between l1 and l2 options were determined at the HPT stage. RF is an ensemble method in which the decision trees are learnt on random sub-samples independently drawn from the training set. Further, limiting the number of features considered at each split adds more

randomness.

In addition to the aforementioned 3 models, XGBoost and Decision Tree models were evaluated, but they were not used in the ensemble considering the lower accuracy. Performance of the said models on the train & validation set are given in Table 3.2.

Table 3.2: Performance of all considered models

Model	**Best average accuracy**
LR - One-vs-Rest	0.7886
LR - Multinomial (Softmax Regression)	0.7788
RF	0.7690
KNN	0.7607
XGBoost	0.5667
Decision Tree	0.4980

For combining the outputs of the above classifiers and deriving the output of the ensemble, following strategy was used. Instead of always using a majority vote, if the probability of the predicted class of the LR classifier is greater than a threshold value, that prediction is taken as the output without considering output of the other 2 classifiers. Otherwise, the majority vote is taken. This helps to marginally improve the accuracy as well as to significantly reduce the time taken for classification. The threshold value was determined by using cross validation on the training data set in the same manner as the individual models HPT and set at 0.6. Comparison on accuracy, precision and speed by using this approach and majority vote is shown in Table 3.3.

In case of the signs which only differ in the orientation of the palm, only one out of such a group was used for HPT since the pose representations are identical for all the signs in such a group as the palms orientation is not considered at the level 1 classification.

Table 3.3: Performance gain over majority vote strategy

Method	**Accuracy**	**Precision**	**Time (ms)**
Majority vote	0.7808	0.7021	9.65
Alternative	0.7889	0.7211	2.68

3.5.2 Level 2

There is a rule based classifier which distinguishes signs which only differ in palms orientation. It checks if the prediction is one of the predefined signs, and if so determines the correct prediction considering the palms orientation. An example of such signs are හ් and ඔ෭ where the fingers are bent similarly and the only difference between the poses is the angle of the palm makes with the vertical axis. This is shown in Figure 3.6.

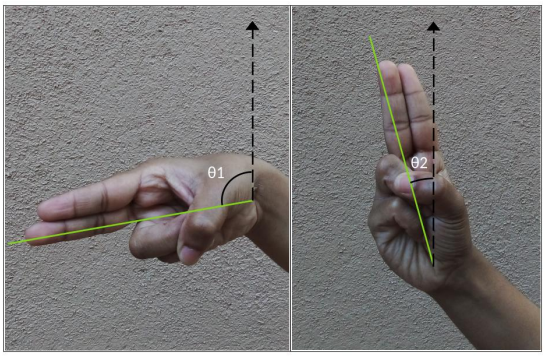

Figure 3.6: Difference between හ් (left) and ඔ෭ (right) being the angle of the reference line makes with the vertical axis

There are three pairs of such signs, and they are distinguished based on palms rotation angle with reference to the vertical axis. The algorithm of the rule base classification is shown in Algorithm 1:

Algorithm 1 Level 2 (rule based) classifier

Input : Output of level 1 classifier, $Sign_{in}$,
Rotation angle w.r.t. Y axis *Angle*
Output: Predicted sign, $Sign_{out}$

$Sign_{out} \leftarrow Sign_{in}$
if $Sign_{in}$ = ේ *or* $Sign_{in}$ = ේ **then**
    if *Angle>45°* **then**
        $Sign_{out} \leftarrow$ ේ
    else
        $Sign_{out} \leftarrow$ ේ
    end
else
    if $Sign_{in}$ = ේ *or* $Sign_{in}$ = ේ **then**
        if *Angle>45°* **then**
            $Sign_{out} \leftarrow$ ේ
        else
            $Sign_{out} \leftarrow$ ේ
        end
    else
        if $Sign_{in}$ = ේ *or* $Sign_{in}$ = ේ **then**
            if *Angle>45°* **then**
                $Sign_{out} \leftarrow$ ේ
            else
                $Sign_{out} \leftarrow$ ේ
            end
        else
        end
    end
end
return $Sign_{out}$

4 RESULTS AND DISCUSSION

The classifier was evaluated for 2 cases. First with all the available test data and then after removing images/ video frames which gave incorrect estimations. In the second case, estimations with distorted limb sizes and wrong angles have still been used as long as it's possible for a human to classify them correctly.

The test set consists of 2 photo sets and 3 videos. Each sign appears once in one video, but due to slight variations of the pose, movements and noise, the resulting estimation could vary. Therefore the duration of a sign was divided into 2 equal parts and the mode result of each part was used for the evaluation. Accuracy, precision and recall were used as the evaluation criteria. Confusion matrices for the test set are shown in 4.1 and 4.2.

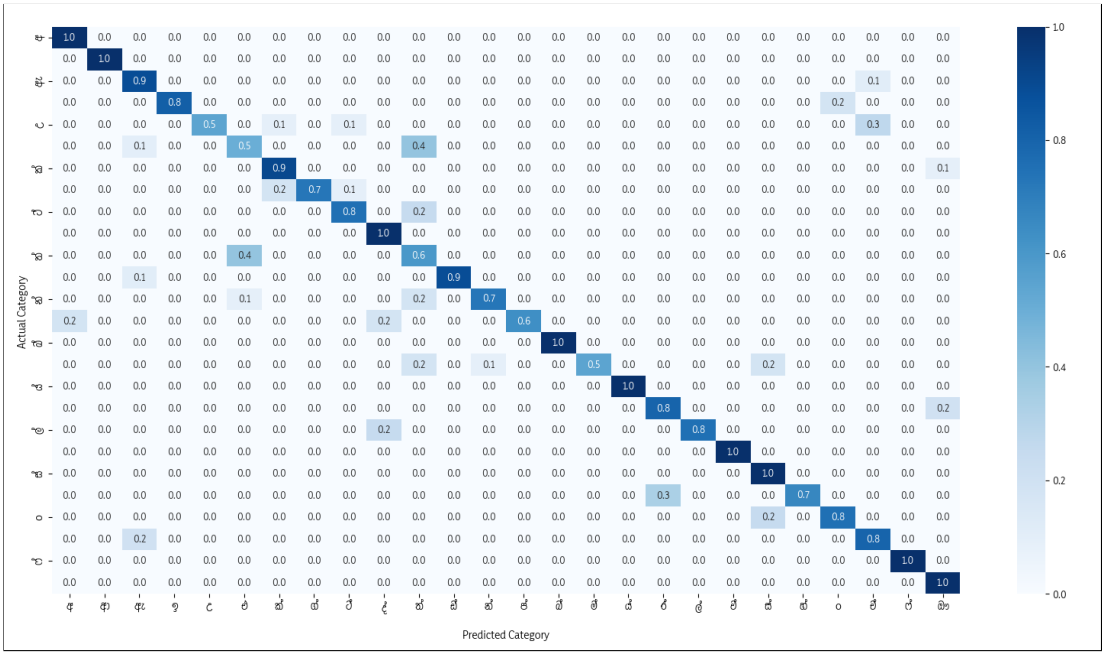

Figure 4.1: Confusion matrix for test data having wrong estimates

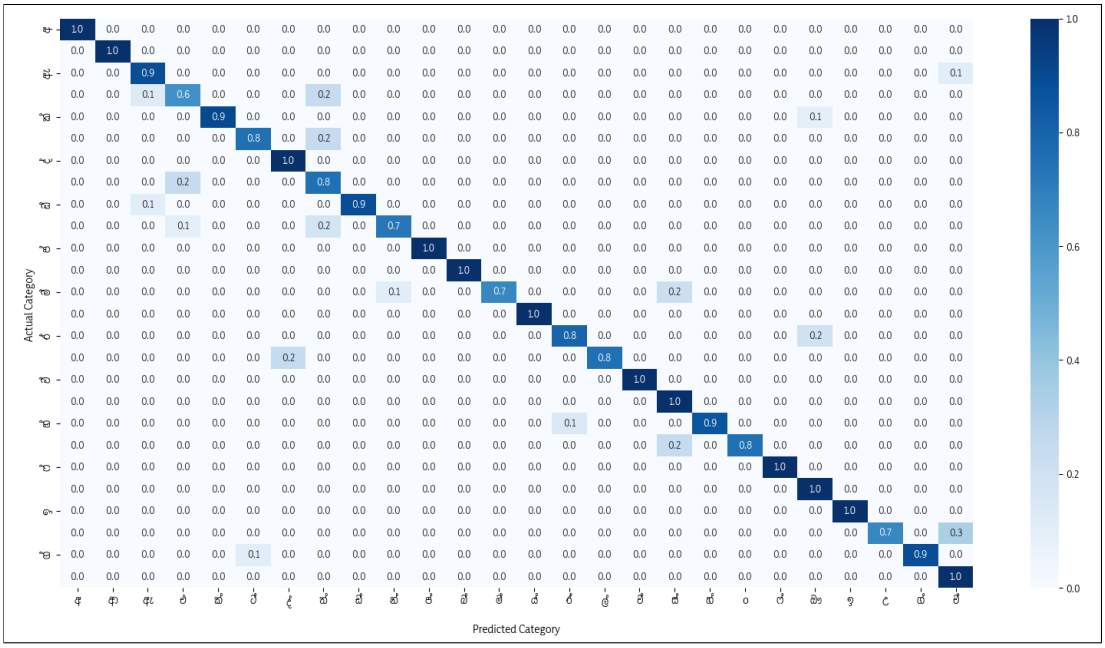

Figure 4.2: Confusion matrix for test data without wrong estimates

Details of results for different data sets are shown in the Table 4.1. Precision for each data set has been calculated by taking unweighted average over the classes. Precision and recall values for each class is shown in Figure 4.3.

Table 4.1: Performance of the classifier on the test set.

		All data		Without wrong estimates	
Dataset	**Type**	**Accuracy**	**Precision**	**Accuracy**	**Precision**
Subject 5	Video	0.7308	0.6308	0.8636	0.8106
Subject 6	Video	0.7447	0.6875	0.8974	0.8571
Subject 7	Video	0.8043	0.7246	0.8810	0.8413
Subject 8	Photo	0.9111	0.8800	0.9111	0.8800
Subject 9	Photo	0.8519	0.8427	0.8519	0.8427
All		0.8074	0.8479	0.8795	0.9035

Although the limitations of the pose estimator reduces the accuracy of the end to end system, it still performs at over 80% even with wrong estimations as input to the classification step. Also, it shows consistent performance over all the data set which vary by video/ image quality, attributes of the person and background.

Removing estimation errors results in about a 7% increase in the accuracy. When analyzing the classifier independent of the estimation errors it can be seen that 22 out of 26 signs are being recognized with a recall value over 75%. Signs ಲ, ಒ, and

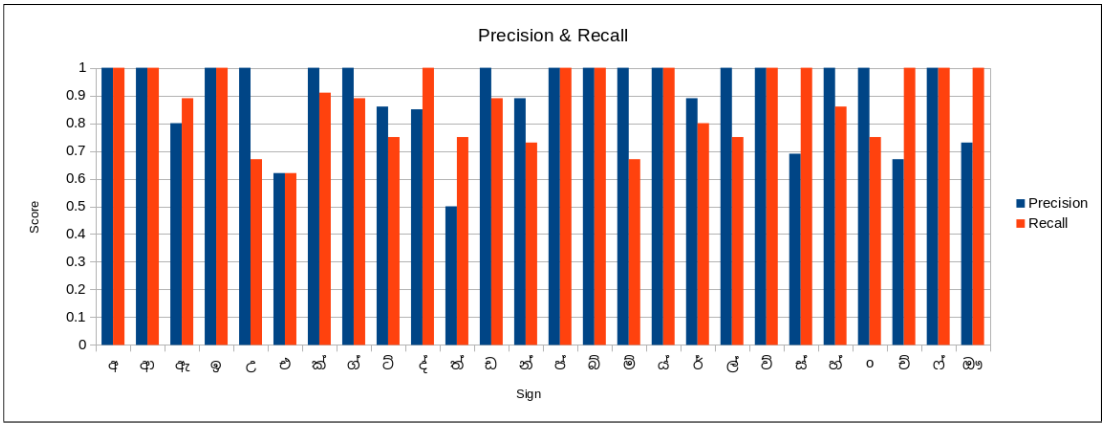

Figure 4.3: Precision and Recall for each class

මි have the lowest scores which are 62.50%, 66.67%, and 66.67% respectively. The signs එ, and න් differ from each other mostly only by the position of the thumb, thus making their pose estimations close to each other. Therefore about 20% of the time එ is being misclassified as න් and න් is being misclassified as එ. Likewise, මි is being misclassified as ස් and න් due to similarities between them. Although උ and ච් are not very similar to the eye, in the normalized pose their difference are not very prominent in some cases. Therefore 30% of the time, උ is misclassified as ච්. The said similarities are shown in Figures 4.4, 4.5 and 4.6

Figure 4.4: Similarity between signs එ (left), and න් (right)

Figure 4.5: Similarities between signs මි (left), ස් (middle), and න් (right)

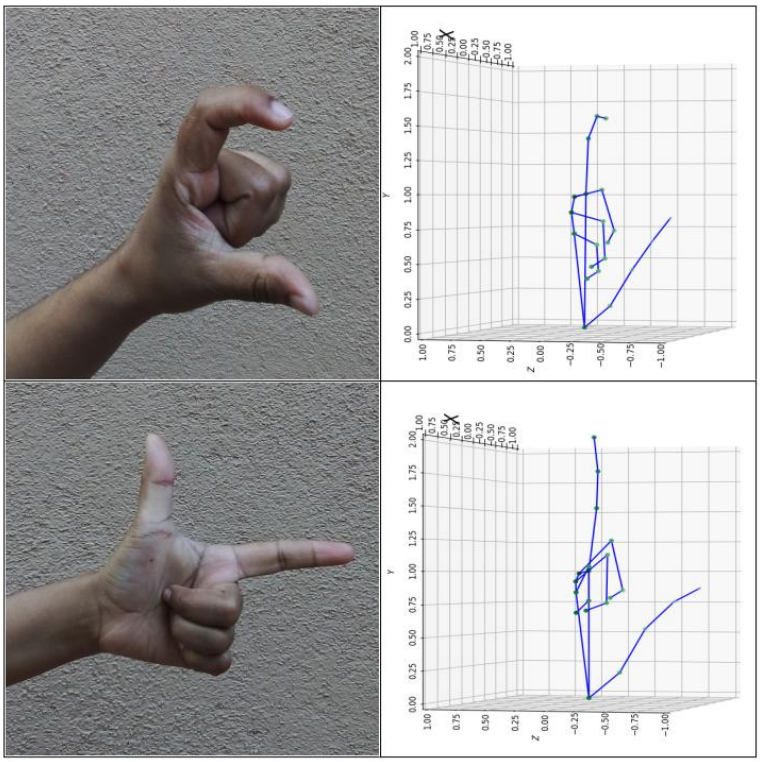

Figure 4.6: An instance where ඒ (top) and ඍ (bottom) signs having approximately similar normalized pose estimations.

The results of the ablation study done on the test set is shown in Table 4.2. It can be observed that the accuracy for RF, KNN and LR models without rule based classification are lower than the results of the cross validation of same on the training set during HPT, given in Table 3.2. The reason behind this is distinguishing between signs which differ only in palms orientation had not being considered in the latter evaluation as there's no information on the palms orientation available at the level 1. Therefore it's more fair to compare the validation accuracy with test accuracy combined with the rule based classifier, which shows slightly better results for the test data. This is an indicator of the classifier being generalized well.

Table 4.2: Ablation study

Classifier	**Features**	**Accuracy**
RF	Angles + Coordinates	0.6964
KNN	Angles + Coordinates	0.6696
LR	Angles + Coordinates	0.7410
RF + Rule Based	Angles + Coordinates	0.7902
KNN + Rule Based	Angles + Coordinates	0.7723
LR + Rule Based	Angles + Coordinates	0.8482
Ensemble + Rule Based	Angles	0.8080
Ensemble + Rule Based	Angles + Coordinates	0.8795

The results were compared with some other sign language classifiers. In order to do a fair comparison, following criteria were considered when selecting alternative methods:

- Developed for static signs
- Have at least 20 signs
- Use images/videos without depth information
- Accuracy and the size of the training data set are published

Comparison of accuracy to other method are shown in Table 4.3.

Table 4.3: Comparison to other fingerspelling classifiers.

Authors/ year	**Training set size**	**No. of signs**	**Accuracy**	**Context**
Pugeault et al [39]/ 2011	24000	23	75%	ASL Fingerspelling
Sanalohit et al [12]/2022	2518	30	84.57%	Thai Sign Language Fingerspelling
Wang et al [40]/2020	1050	30	89.48%	Chinese Sign Language Fingerspelling
Nguen et al [41]/2019	10420	41	94.1%	Japanes Sign Language Fingerspelling
Ours - all	122	26	80.7%	FASSL
Ours - correct estimates	122	26	87.9%	FASSL

While the compared methods have higher and lower accuracies than the proposed method, it should be noted that all the said methods have been developed using much

larger training datasets. In the case of the smallest compared training set, it's still about 9 times larger, and in the case of the largest, it's about 197 times larger than the proposed methods training set. When compared with methods which have close accuracies as the proposed method such as [12] 84.57%, [40] (89.48%) it can be seen that the proposed method achieves results in par with them using a training set of size of 4.8% and 11.6% of the compared training sets respectively.

The hardware used for the experimental setup was a computer having a 8 core 1.6 GHz processor, 8 GB of RAM and no GPUs. Average processing speed for pose estimation was 30.7 frames per second (FPS). Average speed for pre-processing and classification was 89.20 FPS. The system was implemented in such a way that pose estimation and classification run parallelly utilizing 2 cpu cores. The estimator's output is put on a queue which is read by the classifier. Therefore the speed of the pose estimation step becomes the speed of the end to end system.

5 CONCLUSION

This study presents a method for developing a classifier for FASSL using a small training data set which consists of 122 images/video frames sourced from 4 people. Since there is no known data set available for developing a vision based classifier for the FASSL, the key idea was to develop a simpler classification model such that it doesn't need a large data set for training, and yet performs well enough to use in real world applications.

The results show that it can recognize static signs used in the FASSL with an average accuracy over 87%. This is a major leap forward in the context of FASSL where there has been no previous attempts. The results also show that the presented method achieves an accuracy which is on par with that of the methods developed for other languages' fingerspelling alphabets using much larger datasets.

Further, the classifier's speed on commodity hardware enables it to be used with common video formats in real time applications. The implementation of the end-to-end classification system is available as a free and open source software, which can be used in laptop or desktop computers. This software also supports collecting more data, labelling them and using them to improve the classification model.

It was observed that there are only a few experts in the field of Sinhala Sign Language. The data set was developed in consultation with such experts from 2 institutions. It was particularly challenging to develop the data set as it was done amidst restrictions on travelling and while the said institutions were closed. The labelled data set used for this study has been made available to the public to support future studies.

References

[1] “3D Hand Pose with MediaPipe and TensorFlow.js.” [Online]. Available: https://blog.tensorflow.org/2021/11/3D-handpose.html

[2] H. Cooper, B. Holt, and R. Bowden, “Sign Language Recognition,” p. 23.

[3] “Fingerspelling Alphabet - British Sign Language (BSL).” [Online]. Available: https://www.british-sign.co.uk/fingerspelling-alphabet-charts/

[4] M. Punchimudiyanse and R. G. N. Meegama, “Computer Interpreter for Translating Written Sinhala to Sinhala Sign Language,” *OUSL Journal*, vol. 12, no. 1, p. 70, Jul. 2017. [Online]. Available: https://ouslj.sljol.info/article/10.4038/ouslj.v12i1.7377/

[5] P. Fernando and P. Wimalaratne, “Sign Language Translation Approach to Sinhalese Language,” *GSTF Journal on Computing (JoC)*, vol. 5, no. 1, p. 9, Sep. 2016. [Online]. Available: http://www.globalsciencejournals.com/article/10.7603/s40601-016-0009-8

[6] M. Punchimudiyanse and R. G. N. Meegama, “Animation of Fingerspelled Words and Number Signs of the Sinhala Sign Language,” *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 16, no. 4, pp. 1–26, Sep. 2017. [Online]. Available: https://dl.acm.org/doi/10.1145/3092743

[7] R. Akmeliawati, F. Dadgostar, S. Demidenko, N. Gamage, Y. Kuang, C. Messom, M. Ooi, A. Sarrafzadeh, and G. SenGupta, “Towards real-time sign language analysis via markerless gesture tracking,” in *2009 IEEE Intrumentation and Measurement Technology Conference*. Singapore. Singapore: IEEE, May 2009, pp. 1200–1204. [Online]. Available: http://ieeexplore.ieee.org/document/5168637/

[8] L. K. S. Tolentino, R. O. S. Juan, A. C. Thio-ac, M. A. B. Pamahoy, J. R. R. Forteza, X. J. O. Garcia, and Department of Electronics Engineering, Technological University of the Philippines and the University Extension Services Office, Technological University of the Philippines, Philippines, “Static Sign Language Recognition Using Deep Learning,” *International Journal of Machine Learning and Computing*, vol. 9, no. 6, pp. 821–827, Dec. 2019. [Online]. Available: http://www.ijmlc.org/content-103-1026-1.html

[9] L. Pigou, S. Dieleman, P.-J. Kindermans, and B. Schrauwen, “Sign Language Recognition Using Convolutional Neural Networks,” in *Computer Vision - ECCV 2014 Workshops*, L. Agapito, M. M. Bronstein, and C. Rother, Eds. Cham: Springer International Publishing, 2015, vol. 8925, pp. 572–578, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/978-3-319-16178-5_40

[10] R. Kurdyumov, P. Ho, and J. Ng, “Sign Language Classification Using Webcam Images,” p. 4.

[11] J. Rekha, J. Bhattacharya, and S. Majumder, “Shape, texture and local movement hand gesture features for Indian Sign Language recognition,” in *3rd International Conference on Trendz in Information Sciences & Computing (TISC2011)*. Chennai, India: IEEE, Dec. 2011, pp. 30–35. [Online]. Available: http://ieeexplore.ieee.org/document/6169079/

[12] J. Sanalohit and T. Katanyukul, “TFS Recognition: Investigating MPH]{Thai Finger Spelling Recognition: Investigating MediaPipe Hands Potentials,” *arXiv:2201.03170 [cs]*, Jan. 2022, arXiv: 2201.03170. [Online]. Available: http://arxiv.org/abs/2201.03170

[13] J. Singha, A. Roy, and R. H. Laskar, “Dynamic hand gesture recognition using vision-based approach for human–computer interaction,” *Neural Computing and Applications*, vol. 29, no. 4, pp. 1129–1141, Feb. 2018. [Online]. Available: http://link.springer.com/10.1007/s00521-016-2525-z

[14] M. Zhang, X. Cheng, D. Copeland, A. Desai, M. Y. Guan, G. A. Brat, and S. Yeung, “Using Computer Vision to Automate Hand Detection and Tracking of Surgeon Movements in Videos of Open Surgery,” *arXiv:2012.06948 [cs]*, Dec. 2020, arXiv: 2012.06948. [Online]. Available: http://arxiv.org/abs/2012.06948

[15] F. Zhang, V. Bazarevsky, A. Vakunov, A. Tkachenka, G. Sung, C.-L. Chang, and M. Grundmann, “MediaPipe Hands: On-device Real-time Hand Tracking,” *arXiv:2006.10214 [cs]*, Jun. 2020, arXiv: 2006.10214. [Online]. Available: http://arxiv.org/abs/2006.10214

[16] J. Singha and K. Das, “Indian Sign Language Recognition Using Eigen Value Weighted Euclidean Distance Based Classification Technique,” *International Journal of Advanced Computer Science and Applications*, vol. 4, no. 2, 2013. [Online]. Available: http://thesai.org/Publications/ViewPaper?Volume=4&Issue=2&Code=IJACSA&SerialNo=28

[17] B. Doosti, “Hand Pose Estimation: A Survey,” *arXiv:1903.01013 [cs]*, Jun. 2019, arXiv: 1903.01013. [Online]. Available: http://arxiv.org/abs/1903.01013

[18] V. Ramakrishna, D. Munoz, M. Hebert, J. Andrew Bagnell, and Y. Sheikh, “Pose Machines: Articulated Pose Estimation via Inference Machines,” in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, vol. 8690, pp. 33–47, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/978-3-319-10605-2_3

[19] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, “Convolutional Pose Machines,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, Jun. 2016, pp. 4724–4732. [Online]. Available: http://ieeexplore.ieee.org/document/7780880/

[20] T. Simon, H. Joo, I. Matthews, and Y. Sheikh, “Hand Keypoint Detection in Single Images using Multiview Bootstrapping,” *arXiv:1704.07809 [cs]*, Apr. 2017, arXiv: 1704.07809. [Online]. Available: http://arxiv.org/abs/1704.07809

[21] U. Iqbal, P. Molchanov, T. Breuel, J. Gall, and J. Kautz, “Hand Pose Estimation via Latent 2.5D Heatmap Regression,” in *Computer Vision – ECCV 2018*, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, vol. 11215, pp. 125–143, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/978-3-030-01252-6_8

[22] P. Panteleris, I. Oikonomidis, and A. Argyros, “Using a single RGB frame for real time 3D hand pose estimation in the wild,” *arXiv:1712.03866 [cs]*, Dec. 2017, arXiv: 1712.03866. [Online]. Available: http://arxiv.org/abs/1712.03866

[23] S. Agahian, F. Negin, and C. Köse, “An efficient human action recognition framework with pose-based spatiotemporal features,” *Engineering Science and Technology, an International Journal*, vol. 23, no. 1, pp. 196–203, Feb. 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2215098618312345

[24] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: Theory and applications,” *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, Dec. 2006. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0925231206000385

[25] D. C. Luvizon, D. Picard, and H. Tabia, “2D/3D Pose Estimation and Action Recognition Using Multitask Deep Learning,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT, USA: IEEE, Jun. 2018, pp. 5137–5146. [Online]. Available: https://ieeexplore.ieee.org/document/8578637/

[26] J. Liu, A. Shahroudy, D. Xu, and G. Wang, “Spatio-Temporal LSTM with Trust Gates for 3D Human Action Recognition,” *arXiv:1607.07043 [cs]*, Jul. 2016, arXiv: 1607.07043. [Online]. Available: http://arxiv.org/abs/1607.07043

[27] J. Liu, G. Wang, P. Hu, L.-Y. Duan, and A. C. Kot, “Global Context-Aware Attention LSTM Networks for 3D Action Recognition,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, HI: IEEE, Jul. 2017, pp. 3671–3680. [Online]. Available: http://ieeexplore.ieee.org/document/8099874/

[28] F. Baradel, C. Wolf, and J. Mille, “Pose-conditioned Spatio-Temporal Attention for Human Action Recognition,” *arXiv:1703.10106 [cs]*, Aug. 2017, arXiv: 1703.10106. [Online]. Available: http://arxiv.org/abs/1703.10106

[29] H. H. Pham, H. Salmane, L. Khoudour, A. Crouzil, P. Zegers, and S. A. Velastin, “Spatio–Temporal Image Representation of 3D Skeletal Movements

for View-Invariant Action Recognition with Deep Convolutional Neural Networks," *Sensors*, vol. 19, no. 8, p. 1932, Apr. 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/8/1932

[30] C. Wang, Y. Wang, and A. L. Yuille, "An Approach to Pose-Based Action Recognition," in *2013 IEEE Conference on Computer Vision and Pattern Recognition*. Portland, OR, USA: IEEE, Jun. 2013, pp. 915–922. [Online]. Available: http://ieeexplore.ieee.org/document/6618967/

[31] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," *arXiv:1708.02002 [cs]*, Feb. 2018, arXiv: 1708.02002. [Online]. Available: http://arxiv.org/abs/1708.02002

[32] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single Shot MultiBox Detector," *arXiv:1512.02325 [cs]*, vol. 9905, pp. 21–37, 2016, arXiv: 1512.02325. [Online]. Available: http://arxiv.org/abs/1512.02325

[33] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple Online and Realtime Tracking," *2016 IEEE International Conference on Image Processing (ICIP)*, pp. 3464–3468, Sep. 2016, arXiv: 1602.00763. [Online]. Available: http://arxiv.org/abs/1602.00763

[34] A. A. Chaaraoui, J. R. Padilla-Lopez, and F. Florez-Revuelta, "Fusion of Skeletal and Silhouette-Based Features for Human Action Recognition with RGB-D Devices," in *2013 IEEE International Conference on Computer Vision Workshops*. Sydney, Australia: IEEE, Dec. 2013, pp. 91–97. [Online]. Available: http://ieeexplore.ieee.org/document/6755884/

[35] L. Lo Presti, M. La Cascia, S. Sclaroff, and O. Camps, "Gesture Modeling by Hanklet-Based Hidden Markov Model," in *Computer Vision – ACCV 2014*, D. Cremers, I. Reid, H. Saito, and M.-H. Yang, Eds. Cham: Springer International Publishing, 2015, vol. 9005, pp. 529–546, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/978-3-319-16811-1_35

[36] M. Santello, M. Flanders, and J. F. Soechting, "Postural Hand Synergies for Tool Use," *The Journal of Neuroscience*, vol. 18, no. 23, pp. 10 105–10 115, Dec. 1998. [Online]. Available: https://www.jneurosci.org/lookup/doi/10.1523/JNEUROSCI.18-23-10105.1998

[37] S. Cobos, M. Ferre, M. Ángel Sánchez-Urán, J. Ortego, and R. Aracil, "Human hand descriptions and gesture recognition for object manipulation," *Computer Methods in Biomechanics and Biomedical Engineering*, vol. 13, no. 3, pp. 305–317, Jun. 2010. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/10255840903208171

[38] N. Bhatt and V. Skm, “Posture similarity index: a method to compare hand postures in synergy space,” *PeerJ*, vol. 6, p. e6078, Dec. 2018. [Online]. Available: https://peerj.com/articles/6078

[39] N. Pugeault and R. Bowden, “Spelling it out: Real-time ASL fingerspelling recognition,” in *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*. Barcelona, Spain: IEEE, Nov. 2011, pp. 1114–1119. [Online]. Available: http://ieeexplore.ieee.org/document/6130290/

[40] X. Jiang, B. Hu, S. Chandra Satapathy, S.-H. Wang, and Y.-D. Zhang, “Fingerspelling Identification for Chinese Sign Language via AlexNet-Based Transfer Learning and Adam Optimizer,” *Scientific Programming*, vol. 2020, pp. 1–13, May 2020. [Online]. Available: https://www.hindawi.com/journals/sp/2020/3291426/

[41] N. T. Nguen, S. Sako, and B. Kwolek, “Deep CNN-Based Recognition of JSL Finger Spelling,” in *Hybrid Artificial Intelligent Systems*, H. Pérez García, L. Sánchez González, M. Castejón Limas, H. Quintián Pardo, and E. Corchado Rodríguez, Eds. Cham: Springer International Publishing, 2019, vol. 11734, pp. 602–613, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/978-3-030-29859-3_51