

SOLENT UNIVERSITY, SOUTHAMPTON
FACULTY OF BUSINESS, LAW AND DIGITAL
TECHNOLOGY

MSc Applied AI and Data Science
2022

Student Name: Shany Stephen
Student ID: Q15725294

“Hand Chat - British Sign Language
Recognition System ”

Supervisor : Dr. Olufemi Isiaq
Date of submission : September 2022

Acknowledgements

Foremost I would like to express my sincere gratitude to my supervisor, Dr. Olufemi Isiaq for the long-time guidance, support, and encouragement throughout the work. I feel honoured to be part of this course and would like to thank Solent University, Southampton for giving me the chance to learn and upgrade my skills in the field of artificial intelligence and data science.

I would extend my gratitude to my colleagues and friends Ms Sony Abraham, Ms Viralben Raval, Ms Krishnaben Ghanshyambhai for supporting me in the real-time validation of the system.

Special thanks to my parents and my husband for their patience, personal support and motivating me in every stage of this project.

Above all, I thank God almighty for blessing me with good health and mental strength which was crucial for the completion of my work.

Abstract

Effective communication helps in conveying message with clarity and purpose. Sign language is used by the hard-of-hearing community to communicate using hand signs along with lip movement, facial expression, and body pose. Sign language is not universal and differs from country to country. British Sign Language used in the UK is recognised as an official language. There is a disproportionate number of registers interpreters to the BSL users which calls for the importance of having an electronic recognition system bridging the communication gap between hard-of-hearing and non-signers. A real-time vision-based sign language recognition system is developed using the techniques of computer vision and deep learning architecture to recognise fingerspelling and six BSL words. The proposed system used OpenCv for capturing the frames, MediaPipe framework to extract the features of the hands and built the LSTM models to predict the signs. The data is collected and trained for both left-hand and right-hand signers. The overall testing accuracy of 26 fingerspelling is 94.23% and six words is 99.07%. Real-time testing accuracy of 86% is obtained for fingerspelling. A graphical user interface is developed for the signers to interact with the developed system for real-time signing.

Keywords- British Sign Language, MediaPipe, LSTM, OpenCv, Real-time

Contents

<i>Acknowledgements</i>	1
<i>Abstract</i>	2
1. Introduction	7
1.1 Research question	9
1.2 Aim.....	10
1.3 Objectives.....	10
1.4 Project Plan	12
2. Literature Review	13
2.1 Artificial Intelligence & Computer Vision.....	13
2.2 Stages of SLR system development	13
2.3 Previous works with real-time recognition	15
2.4 BSL recognition using traditional methods.....	16
2.5 Sign Language Recognition Using MediaPipe	18
3. Methodology	22
3.1 Development Requisites:	25
3.2 Data collection	25
3.3 Data Cleaning and Pre-processing	34
3.3.1 Data Cleaning.....	34
3.3.2 Pre-Processing.....	36
3.4 Model Selection and Evaluation.....	37
3.4.1 Model 1: Long Short-Term Memory (LSTM).....	38
3.4.2 Model 2: Gated Recurrent Unit (GRU)	41
3.5 Evaluation and Tuning.....	43
3.6 Graphical User Interface	45
4. Results	48
4.1 Quantitative Evaluation	48
4.1.1 Accuracy & Loss Plots	48
4.1.2 Confusion Matrix	51
4.1.3 Multi-Label Confusion Matrix	52
4.1.4 F1 Score.....	53
4.2 Critical Evaluation.....	55

5. Discussion	57
6. Conclusions	59
7. Limitations & Future Works.....	60
7.1 Limitations	60
7.2 Future Works.....	60
8. Reference list.....	61
9. Appendices.....	A1
9.1 Appendix A: Ethics Application.....	A1
9.2 Appendix B: Development Codes	B1
9.3 Appendix C: BSL Words Results	C1

List of Tables

Table 1-Related work on MediaPipe and Sign Language	21
Table 2- List of methods.....	23
Table 3-Summary of collected data.....	28
Table 4-Train, Test & Validation split	37
Table 5-Train, validation & Test accuracy of LSTM, GRU.....	51
Table 6-F1 Score, Recall, Precision of LSTM & GRU.....	55
Table 7-Result of critical evaluation.....	56
Table 8-Comparison with related works	58

List of Figures

Figure 1-Distribution of different Sign Languages (Traynor, 2017).....	8
Figure 2- Project Plan	12
Figure 3- Architecture of proposed model.....	22
Figure 4-Stages covered for implementation.....	24
Figure 5-MediaPipe Hand Landmarks (source: Hands - mediapipe (google.github.io)) ..	27
Figure 6-Fingerspelling Chart (British Sign Language 2015).....	28
Figure 7-BSL words (British Sign Language, 2015).....	29
Figure 8-Python code for the data collection algorithm	31
Figure 9-Hand landmarks and connections	32
Figure 10-Creation of directories(A-Z) and subdirectories (0-120) for data collection....	32
Figure 11-NumPy array file of the hand landmark.....	33
Figure 12-Result of algorithm showing invalid frame	35
Figure 13-Result of algorithm after correcting the invalid instances	35
Figure 14-Numerical encoding of labels	36
Figure 15-One-hot encoding of labels	37
Figure 16-LSTM cell structure	39
Figure 17-Plotting of LSTM model summary.....	40
Figure 18-GRU cell structure (Kostadinov 2019)	42
Figure 19- Plotting of GRU model summary	43
Figure 20- Number of parameters in LSTM model.....	45
Figure 21-Number of parameters in GRU model	45
Figure 22- Home page of GUI.....	46
Figure 23- Real-time recognition of fingerspelling	47
Figure 24-Real-time recognition of BSL word.....	47
Figure 25-Accuracy plot of GRU	49
Figure 26-Loss plot of GRU	49
Figure 27-Accuracy plot of LSTM.....	50
Figure 28-Loss plot of LSTM.....	50
Figure 29-Normalised confusion matrix for LSTM	51
Figure 30-Normalised confusion matrix for GRU.....	52
Figure 31-Multilabel confusion matrix for LSTM	53

Figure 32-F1_Score of LSTM	54
Figure 33-F1_Score of GRU	54
Figure 34-Ethics 1.....	A1
Figure 35-Ethics 2.....	A2
Figure 36-Ethics 3.....	A3
Figure 37-Accuracy Plot of BSL words	C1
Figure 38-F1_Scores of words.....	C1
Figure 39-Recall of words	C2

1. Introduction

Hand Signs are used in regular daily existence for conveying a message where voice commands are not feasible. A few examples can be quoted from everyday life, for instance, an umpire during a cricket match uses different hand gestures to communicate with the team. Traffic control hand signals are helpful in busy traffic zones. The ability to communicate meaningfully and effectively is a significant skill associated with daily activities. People with unclear speech use a unique language called Makaton which uses hand signs along with speech. However, a section of the deaf-mute population in the world uses hand signs along with lip movement, facial expression, and body pose for communication. This visual means of communication used by the deaf-mute or people with hearing impairments is called Sign Language.

History shows that in 1620, the first instructional book in Madrid was published for the deaf by Spanish priest Juan Pablo de Bonet, the pioneer of education for the deaf. In 1771, the first free public school for the deaf named National Institute for Deaf-Mutes was established by Abbe Charles Michel de L'Epee, a French Catholic priest who used signs to teach his students French.

Like any other natural language, sign language has its own vocabulary, syntax, and expression. Due to its linguistic complexity, good training and practice are expected for deciphering the signs. There are famous inspiring personalities whose first language is sign language and achieved incredible success. William Ellsworth Hoy, an American Major Baseball player is credited with inventing the hand signs in baseball. Marlee Matlin, the only deaf actor who won the Golden Globe and Academy awards is worth quoting.

Sign languages are used globally, and 70 million people are estimated as deaf by the World Federation of the Deaf. Gesture-based communication is not the same around the globe, 200+ sign languages are being used by the hard-of-hearing community over the world. Out of which 41 countries have accepted sign language

as an official language. In addition to these facts, September 23rd is observed as the International Day of Sign language by the United Nation. Figure 1 adopted from an article by (Traynor 2017), illustrates the distribution of sign language families around the world.

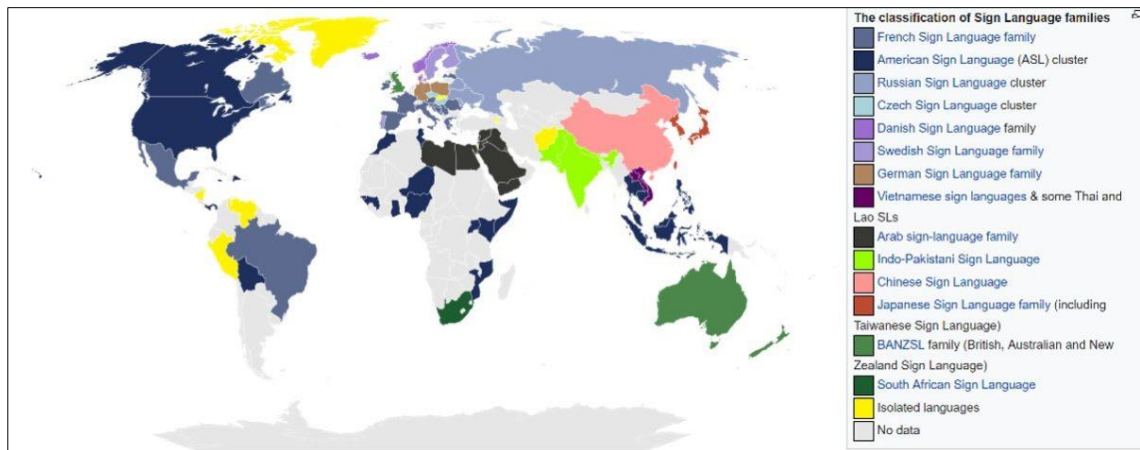


Figure 1-Distribution of different Sign Languages (Traynor, 2017)

Sign language cannot be generalised as it is not universal and differs from country to country and region to region. Every nation has its own sign language for instance, American Sign language (ASL) spoken by the deaf community in United states and few of the English-speaking parts of Canada uses single hand for finger spellings however the British Sign Language (BSL) used in Britain involves the use of both the hands to denote an alphabet except for the letter 'c.' BSL is part of BANZSL or British, Australian and New Zealand Sign Language family that uses two-hand finger spelling and used in Britain. BANZSL shares not many signs with ASL and shows around 30% of similarity. An experienced signer or human translator helps in understanding the sign language used in a region. As per the information given in British Deaf Association there are 151,000 individuals using BSL in the UK and 87,000 of these are deaf users. According to the information shared by National Registers of Communication Professionals working with Deaf and Deafblind People (NRCPD) shows there are 908 registered sign language

interpreters, 234 trainee sign language interpreters (TSLI) in the UK and 11 registered sign language translators.

In 2003, UK government has recognised BSL as official minority language like other minority language like Gaelic and Welsh. As per the data available in the UK government website, the Private Member's Bill introduced by Rosie Cooper MP on 16th June 2021, the BSL Bill has passed on its third reading in the House of Lords. This bill will recognise BSL as a language of England, Wales, and Scotland in its own right.

Recently in 2014, The Guardian detailed the news of a deaf couple who raised their grievance at a hospital for the lack of interpreter. An inexpensive electronic interpretation system can complement the person-to-person interpretation. Today there are applications that can convert a text/audio in one language to another, however sign language which is not in a written text format cannot be added with such applications. Moreover, the facts and findings elucidated in this section depicts the difficulty faced by the people using sign language to communicate with the public.

1.1 Research question

The unbalanced number of interpreters to the BSL users highlights the need for an easy inexpensive electronic system that can perceive the signs. Research is persistently conducted in sign language recognition, however little or less work is available for the BSL system. Hence the BSL recognition system development is chosen as the area of interest to add inputs and find techniques to solve the problem of hand sign recognition.

The research question raised from the facts and statistics is **'Can Artificial Intelligence and machine learning be applied in real-time visual recognition of British Sign Language?'**

A real-time Sign Language Recognition (SLR) system enabling the user to convert the hand signs to text can be developed using the concepts of artificial intelligence and machine learning techniques.

1.2 Aim

The fundamental aim of this project is to develop a system that can recognize the BSL hand signs and convert them to text and speech helping a deaf-mute user to translate the signs without the assistance of an interpreter.

The alphabets (A-Z) are spelled with both hands and the method is called fingerspelling. Fingerspelling helps in signing names using alphabets, introducing words with no equivalent sign, convey new concepts like computer mouse BSL is not all about finger spelling as there are different other dialects used to denote words and phrases.

A well-trained BSL recognition system will be helping the user in multiple ways in everyday life, below are a few use cases of a BSL recognition system:

- Help to bridge the communication gap between the deaf-mute and verbally speaking community
- The system can also assist the user to communicate with shop assistants while shopping
- The system can help a user in places like drive-thru and other kiosks
- BSL system can also help a user in public transport
- Assistance at ticket booking counters

1.3 Objectives

The key objectives of this work are listed below.

1. Develop a vision-based application to correctly detect the trained BSL signs in real-time.

2. Create a new BSL dataset with required signs for the study and discover solutions for challenges of data collection like hand detection, lighting, cluttered background, etc.
3. Detect the region of interest and extract the key features by solving the challenges of traditional image pre-processing techniques like hand segmentation, edge detection, and skin colour handling should be investigated to bring out a solution for processing the frames.
4. The system should be able to work for both right-hand and left-hand users. The sign language users can be either right-handed or left-handed, that is, the dominant hand of the user could be either right or left. The system should be able to recognize the signs from both category of users.
5. Implementation of Video Classification: Since sign language involves the classification of visual images, hence machine learning techniques and deep learning models for training these frames should be applied. The methods and concepts learned in the course will be leveraged to achieve this objective along with feature extraction technique (MediaPipe framework), and deep learning models like LSTM, GRU for modelling.
6. Evaluation of the developed model: The model should be able to meet defined success metrics where the accuracy score and confusion matrix- F1 Score, Recall, Precision. The accuracy of not less than 90% for the test data is expected as success. For the critical evaluation of the model, the ability to oversee the different classes in real-time will be verified.

1.4 Project Plan

The planned duration, start and end date of each stage is marked in the Gantt chart as shown in figure 2. The project plan has helped in the development and successful completion of the project.

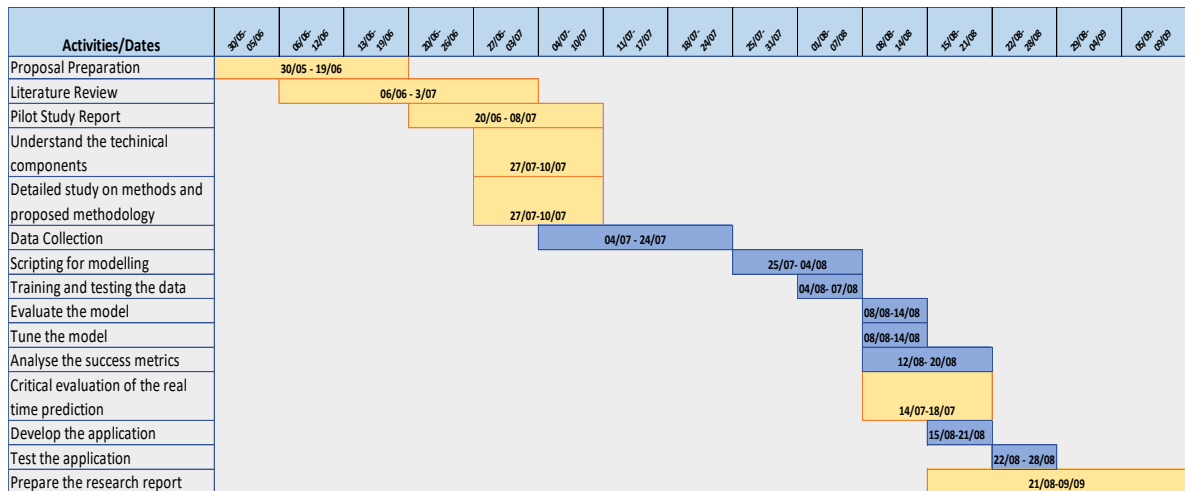


Figure 2- Project Plan

Following this introductory part are core sections that focuses on achieving the aim by covering the objectives of the work.

- Chapter 2 Literature Review, synthesise the previous works and latest techniques and methods used in SLR.
- Chapter 3 Methodology elaborates the implementation process and adopted methods.
- Chapter 4 Results illustrates the outcomes obtained for the implementation
- Chapter 5 Discussion justifies the results for deriving conclusions.
- Chapter 6 Conclusion summaries the work
- Chapter 7 Limitations of the current work and scope for future expansion are noted.

2. Literature Review

This section outlines the previous works and related findings for sign language recognition. The works related to the stages of development of a SLR system, common and traditional methods of implementation and the latest techniques and improvements discovered to handle the challenges of SLR are reviewed systematically.

2.1 Artificial Intelligence & Computer Vision

According to (IBM Cloud Education 2020), artificial intelligence (AI) uses computers to solve issues and make decisions like humans. The two major subsets of AI, Machine learning deep learning are employed to address complex problems. Deep learning can handle unstructured datasets like audio or video to solve supervised and unsupervised machine learning problems utilising neural networks and related algorithms. Computer vision is an AI technology aiming at developing frameworks and systems that can derive information from visual inputs captured by sensing devices. The machine learning algorithms can learn the patterns in the derived data for making predictions.

2.2 Stages of SLR system development

Sign language is not universal hence continuous investigation and research is carried out in the domain of SLR utilising the methods and techniques of AI and machine learning. There are several steps involved in developing a SLR system namely, data acquisition, pre-processing, detection of region of interest, feature identification and extraction, classification, and real time prediction.

First stage of SLR is data acquisition, the techniques used are broadly classified into vision-based and sensor-based method. (Ebrahim Al-Ahdal and Nooritawati 2012) has mentioned three methods for sign capturing:

- Vision-based: Computer vision and image processing techniques are used for detection. The challenges faced in this method are the noise like lighting, background, colour matching. Additionally, the vast computation and low detection confidence for signs with over lapping of hands.
- Data glove: Special glove with sensor to detect hands and tracker to detect location and movement with high reliability and eliminates pre-processing. However, this is inconvenient for the user to wear gloves and hinders the natural movement.
- EMG (electromyography): Sensor based method measures the electrical signals from muscle cells for detecting movements. Feature extraction, and pre-processing are difficult with this method and may include complex noise.

Next complex step is the pre-processing and feature extraction from the acquired images. (Suharjito *et al.* 2018) has presented a review on effective and compatible feature extraction methods.

- For hand segmentation edge detection methods like canny edge detector, elliptical Fourier descriptors can determine the edges and detect hands. Other methods are and skin detection combining with hand motion to distinguish hands from other area.
- Glove-based method can eliminate the complexity of feature extraction with help of sensors and accelerometers. However, the natural human interaction and convenience is hampered.
- Principal component analysis (PCA) is another dimensionality reduction technique to obtain the important features and simplifying the dataset.
- Scale Invariant Feature Transform (SIFT) Algorithm is a robust method handling rotation, translation, and scaling variations. Serial Particle Filter suited for tracking both hands with improvement in noise reduction.
- Microsoft Kinect SDK is a 3D sensor camera highly efficient to process every motion independent of lighting condition.

- Leap motion controller (LMC) is a sensor-based technique for detecting hand movements in high rate and converting hand movements to computer commands.

2.3 Previous works with real-time recognition

Real time prediction of sign language is challenging due to the complexity involved in each stage of SLR. Works related to vision based real-time recognition of hand signs are reviewed for study. (Patel and Patel 2019) recommends a real-time Indian Sign Language system with an overall recognition rate of 98.7% using Support Vector Machine (SVM) classifier. A self-made database of 7920 images is collected using webcam, face detection helps in removal of face and hand portion is detected by applying skin colour detection algorithm by thresholding in YcbCr colour space. Histogram of oriented gradients helped to get the feature vectors. Performance of three classifiers Support Vector Machine (SVM), K-Nearest Neighbours (KNN) and Linear Discriminant Analysis (LDA) were compared. The proposed system is implemented in MATLAB 2018a for real time detection. However, the system is trained for images and hence is static in nature and all the images shown in the proposed work have user wearing full sleeves, since it's a skin detection method the performance should have been tested for rolled up sleeves.

(Tanvir *et al.* 2021) presented the real-time Bangla sign language detection by integrating Adaptive thresholding with 2D Convolutional Neural Network (CNN). 3600 hand sign images were collected and resized to 64*64 and applied to the CNN architecture with softmax output layer and compiled with Adam optimiser. The accuracy obtained is 99.72% however this is applicable only to static signs as only images are considered for the implementation.

2.4 BSL recognition using traditional methods

Lot of work has been done for different sign language families. Here the focus is to systematically review the former works done for BSL recognition. Electronic search and screening are done on the Google Scholar database to identify closely related works.

One of the most cited BSL recognition by (Liwicki and Everingham 2009) presented the automatic recognition of fingerspelled BSL words using a new dataset of 1000 videos of 100 forenames. Colour models are used to estimate the skin, background, and clothing with spatial prior for hand segmentation. First the face is detected and masked, and a spatially varying colour model helps in detecting non-hand label. Hand shapes are recognized using the Histogram of Oriented Gradient (HOG) descriptors and the Hidden Markov Model (HMM) model is used for classification. 98.9% accuracy is obtained for 100 words; however, accuracy declines considerably as the lexicon size increases and words with similar spelling like ava/anna and Ella/Gemma fail during prediction showing the letter level recognition is not promising. The method treats both hands as a single shape descriptor and not tracked individually hence the palm-based letters L, M, N, R, V is difficult to be recognized in any orientation. The work has considered the application of fingerspelled words with edge detection technique (HOG) resulting in difficulty of recognising all the palm-signs like L, M, N, R, V and all the vowels. From the illustrations in the literature, it is noted from that the signer has covered the arms and testing with sleeves rolled up should have been validated as this is colour-based model. In comparison to this work, (Kumar 2022) has proposed a deep learning framework to tackle the hurdles of BSL recognition using CNN and techniques like conversion to grey scale, noise filtration by median filtering, improving hand texture using Gabor filtering, edge detection using Sobel edge filter. Otsu thresholding for conversion of image into binary image and Morphological filtering is to remove objects smaller or larger than a hand. The

training is done on the BSL finger spelled words and achieved a best accuracy of 98% and f1_score of 98.4. However, the model can be only applied to BSL fingerspelling and not tested for any other BSL signs. The model is trained and tested on the pre-saved videos and the real time testing and training for both left-hand and right-hand user is not performed.

(Rambhau 2013) proposed a real-time system for recognizing two hand gestures using image processing techniques – background subtraction, Skin Segmentation, Region Extraction, Feature Extraction (Gabor filtering), Pose Estimation, and HMM training. However, this is used for static alphabets in BSL, and success metrics are not explained.

(Olszewska and Quinn 2019) developed an intelligent BSL recognition system embedded in the smartphone. A self-made dataset with 2600 images of fingerspelling is pre-processed and gradients are obtained using the Histogram of Oriented Gradient (HOG) technique. Support vector machine algorithm is used for classification achieving an accuracy of 99%. Training and testing did not cover both left and right-handed signs. Testing is done on 13,066 images and not on sequence of frames or videos.

Similar work has been submitted by (RAJ and JASUJA 2018), using HOG and ANN for fingerspelling achieving an accuracy of 99.01%. The work has not considered testing for both right-hand and left-hand users. The implementation method shows the processing of a captured image, this limits the application to static images as video or sequences of frames are not considered in the for the experiment.

(Bird *et al.* 2020) has used image classification and feature extraction using VGG16 CNN and MLP along with the 3D hand features vector extracted from a leap motion sensor. The multimodality approach achieved an overall training accuracy of 94.44% by classifying 18 gestures. They have also done the transfer

learning from BSL to ASL, classifying the similar 18 gestures with an accuracy of 82.55%. However, it is observed that the accuracy of the unseen data is 76.5% because of the confusion in few classes.

(Buckley et al. 2021) presented the vision-based prediction of 19 static BSL signs (13 alphabets and 6 numbers) making use of CNN architecture. Testing achieved an average accuracy of 89% and the real-time evaluation was performed using OpenCV to evaluate the effectiveness of the model.

(Hameed *et al.* 2022) has presented a privacy-preserving method of BSL recognition using a radar sensor and deep learning techniques. The radar sensor collects the data and stores it as spectrograms and pre-trained models like InceptionV3, VGG16, and VGG19 were applied to classify the collected data. VGG16 outperformed the other two models with an accuracy of 93.33%. The user must stand 1.5 meters away from the radar and only 6 signs related to emotions are considered for this experiment.

These works on BSL does not mention about the consideration of both left-hand and right-handed signers. In this work both the left-handed and right-handed signing are considered to avoid the bias.

2.5 Sign Language Recognition Using MediaPipe

Above stated works uses traditional and common computer vision pre-processing techniques like background subtraction, skin non-kin segmentation, hand detection, and feature extraction and training mainly using HMM and CNN architecture. Research on latest techniques and models used for hand detection shows that MediaPipe is an effective framework launched by Google helping in easier detection and feature extraction of hand, pose, and face for Sign Language Detection. In 2019, MediaPipe was released for the public, allowing access for

researchers and developers with pretrained ML solutions for hand, face, pose detection. There are few works available for MediaPipe related to sign language recognition.

(Halder and Tayade 2021) used MediaPipe along with Support Vector Machine (SVM) to detect the American signs (26 alphabets and 10 numbers), Indian (24 alphabets), Italian (22 alphabets), and Turkey (10 numbers). Existing data sets of images are used for training and real time validation is performed. Average accuracy of 99% is achieved. However, the system is trained for static images and all the sample images shown in the work for the sign language families are of a single hand.

In the research article presented by (Duy Khuat *et al.* 2021), MediPipe with LSTM is used to detect Vietnamese Sign Language. The dataset is self-made, however, only 15 words are used for the work. The paper mentions the overfitting of data due to invalid data frames. This is because the data analysis and data cleaning are not performed to oversee the invalid videos. The overall accuracy of 63% was achieved for this multi-hand sign detection.

(Alvin *et al.* 2021) has applied MediaPipe hand solution with K-Nearest Neighbour (KNN) to classify and detect the American Sign Language. Work has included only the static signs and dynamic signs like 'J' and 'Z' are excluded. The dataset is self-created for all 24 ASL alphabet and a 94% accuracy is obtained during model evaluation. Due to the weakness of KNN to handle large dimensions the researcher has excluded the z coordinate of the hand landmarks.

A vision-based word recognition system is developed for Indian Sign Language by (Adhikary *et al.* 2021) and uses the concepts of MediaPipe and Random Forest Classifier. A self-made dataset of 2194 images for 11 classes are used in the work. The random forest classifier achieved 97.4% accuracy and performed better than

the decision tree and gradient boosting. It is noted the signs 10 out of 11 signs are single-handed.

Recently, (Subramanian *et al.* 2022) presented an article on solving the challenges of sign language recognition by integrating MediaPipe with an optimized Gated Recurrent Unit (MOPGRU). 12 signs with 30 videos per sign are used in the work and achieved an accuracy of 95%. The proposed model outperformed the other recurrent neural networks like simple RNN, standard GRU, LSTM, BiGRU, and BiLSTM for the selected dataset. The work states that training LSTM and BiLSTM models were not beneficial for the datasets because of limited data for sequence prediction.

Taking a deductive approach, considering the previous works and analysis, visual-based BSL recognition system can be built using MediaPipe and deep learning models. This method is adopted to meet the objectives in-scope for the experiment like new data creation, easier pre-processing, real time validation of the application for both left-hand and right-hand users.

Table 1 summarises the closely related work on sign language recognition with Mediapipe.

Author, Year	Title	Description
Halder and Tayade 2021	Real-time Vernacular Sign Language Recognition using MediaPipe and Machine Learning	MediaPipe along with Support Vector Machine for static images
Duy Khuat <i>et al.</i> 2021	Vietnamese sign language detection using Mediapipe.	MediPipe with LSTM is used to detect Vietnamese Sign Language for 15 words
Alvin <i>et al.</i> 2021	Hand Gesture Detection for Sign Language using Neural Network with Mediapipe	MediaPipe with KNN to detect 24 static American Sign Language
Adhikary <i>et al.</i> 2021	A Vision-based System for Recognition of Words used in Indian Sign Language Using MediaPipe	MediaPipe and Random Forest Classifier for 11 Classes
Subramanian <i>et al.</i> 2022	An integrated mediapipe-optimized GRU model for Indian sign language recognition	MediaPipe with an optimized Gated Recurrent Unit (MOPGRU) for 12 signs

Table 1-Related work on MediaPipe and Sign Language

3. Methodology

This section outlines methods and process of implementation of the proposed BSL recognition system. The section explains the method of data collection for BSL hand signs. Data cleaning and pre-processing of the collected data. Covers the details of two suitable models for solving the research question. Followed by the methods chosen for evaluation of the models.

The BSL recognition application can be implemented by integrating computer vision techniques with machine learning methods. A supervised machine learning model is trained to recognise the patterns from the BSL dataset. The trained model is saved and used in the proposed application for real time prediction. Figure 3 illustrates the architecture of the proposed model. In simple terms, by running the graphical user interface, camera gets activated and captures the hand sign frame-by-frame and the implemented machine learning process extracts the features, predict the label of the sign, and displays as a text on the screen.

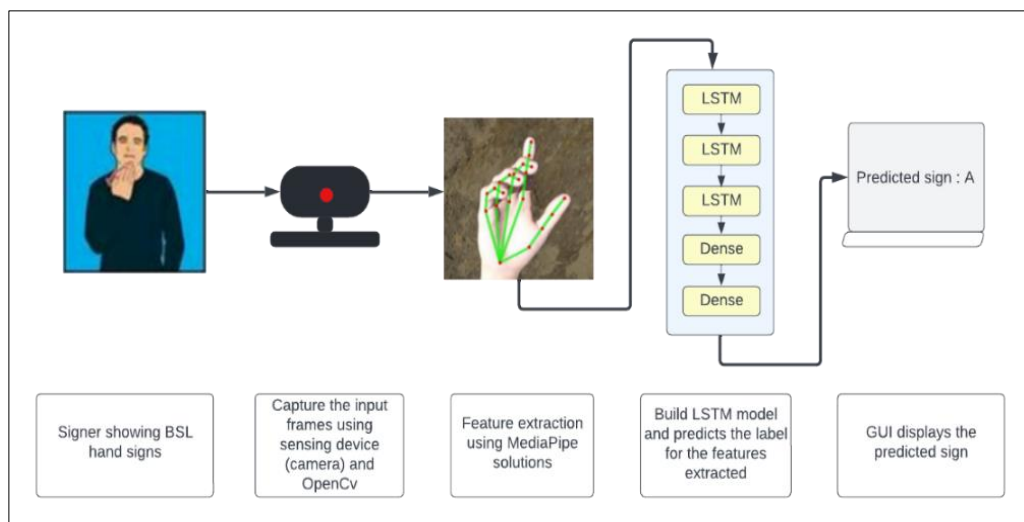


Figure 3- Architecture of proposed model

The complete work was planned and designed to cover three phases with defined methods. The phases along with the chosen methods are summarised in Table 2.

Phase	Method	Description	
Literature Review	Database: Google Scholar The advance search filters in google scholar were used to screen and identify relevant papers.	Google scholar is the search engine for credible journals and literature papers. For instance, relevant papers for Mediapipe were first filtered on papers with title "MediaPipe" and then screened to filters out based on "Mediapipe" AND "Sign Language"	
Development	Stage 1- Data Collection	<ul style="list-style-type: none"> • MediaPipe • OpenCv 	Self-made dataset utilising MediaPipe and OpenCv for required BSL signs. Data saved as NumPy file.
	Stage 2- Pre-processing	<ul style="list-style-type: none"> • Algorithm implemented to identify the invalid data • Numerical encoding • One-Hot encoding • Train-Test Split 	Cleaning the invalid frames and pre-processing for modelling for better performance. Enumerate function used for numerical encoding, to_categorical keras function for one-hot encoding Sklearn train_test_split for splitting the data
	Stage 3 – Selection of models	Two suitable models identified through research: <ul style="list-style-type: none"> • GRU • LSTM 	The models identified are suitable to handle the dataset instances of sequence of frames
	Stage 4- Evaluation	<ul style="list-style-type: none"> • Quantitative: Accuracy, F1_Score, Recall, Precision • Critical evaluation: Usability Check on real-time using 5 signers 	Performance of the models can be compared based on quantitative analysis and the best model is chosen for critical evaluation of the system
Real-Time implementation	Streamlit	User friendly web application for graphical interaction with the system	

Table 2- List of methods

The development of the proposed system covers four stages of machine learning process focussing on the DCSA approach which is covered in the study material of the course (define the problem, collect the data, select the model, and apply the model). Figure 4 summarises the various stages and steps involved; following sub-sections (3.2 - 3.6) explains each stage in detail.

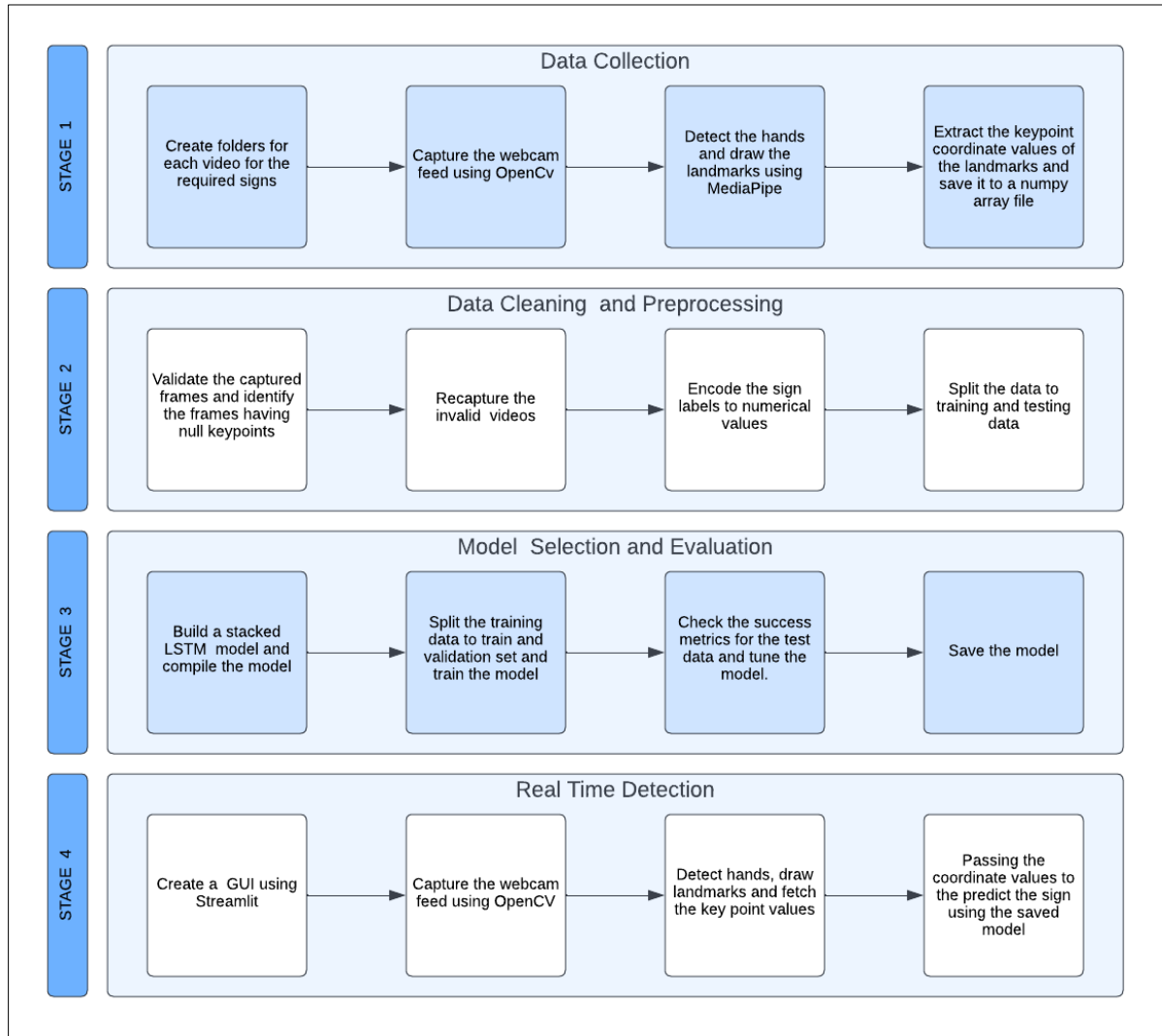


Figure 4-Stages covered for implementation

3.1 Development Requisites:

The details of the hardware component used for the implementation is listed below.

- Webcam: HP Wide Vision 720p HD camera
- Computer: Windows 11, 8 GB DDR4-3200 MHz RAM, GPU-Intel® Iris® Xe Graphics, Microprocessor -Intel Core i5-1155G7 (up to 4.5 GHz, 4 cores, 8 threads)

The software interpreter, python packages and libraries utilised for the building the application is enumerated.

- Interpreter: Python 3.8
- Computer Vision libraries: OpenCV-python v4.5.5.64, mediapipe v0.8.10
- Libraries for machine learning: scikit-learn v1.0.2, keras v2.8.0, tensorflow v2.8.0
- Libraries for visualisation: matplotlib v3.5.1
- Libraries for GUI: streamlit v1.8.1
- Other python libraries: pytsx3 v2.90, pandas v1.4.0, numpy v1.22.2

3.2 Data collection

Continuous research is being conducted by researchers for solving the problem of real-time sign language recognition. Unlike the other sign languages like ASL, BSL datasets are not easily available as unrestricted public dataset from any of the machine learning repositories. In the previous works on BSL recognition, researchers have utilised self-created dataset for their study and development. Hence the primary step for this work is the collection of data for the required signs. There are multiple approaches for collecting the data. Videos of the signs can be collected from video sharing websites like YouTube. Another method is to self-record the signs for a predefined frame rate and save the videos of each sign into corresponding folders. Both methods will also consume more memory to store the video data.

Less memory for data storage, privacy preservation and easy feature extraction and preprocessing are considered as the exit criteria for this stage. Hence the procedure adopted is to self-capture the required signs using webcam feed and directly process each frame and extract the key points from the region of interest (ROI) and save the co-ordinate values into a file instead of saving the frames as an image file. This approach will help in saving the space for data storage and easier feature extraction. The technique adopted for the approach is to extract the 3D coordinates of hand landmarks using MediaPipe framework.

MediaPipe is a free open-source customisable Machine learning solution for processing data live media and videos. It is a cross-platform framework that works across iOS, Android, desktop, cloud, web and IoT. Framework has different solutions for detecting face, hands, pose, iris, object, holistic etc. Here hands are considered as ROI and hence left and right hands landmarks from MediaPipe holistic solution is utilised for development. Hand component of holistic model has 21 3D landmarks each for left and right hand as shown in figure 5. A landmark is defined key point in hand consisting of the a (x, y, z) co-ordinate. The x and y are normalised values [0, 1] with respect to the height and width of the image, whereas z value represents the depth and helps to understand how close the landmark is to the camera. Similarly, the MediaPipe face solution has 468 landmarks and pose with 33 landmarks. The key advantage of MediaPipe is the easy processing of images helping in reducing the computational cost. For any computer vision programming the important pre-processing steps considered are image scaling, noise reduction, image segmentation, skin colour detection and creating bounding box for detecting ROI. (Raval and Gajjar 2021) has explained the image processing steps like ROI detection, background reduction, skin color detection, grayscale conversion and rescaling used for real-time ASL recognition. (Shangeetha. *et al.* 2012) has applied image segmentation, finger, and fingertip detection along with identification of palm identification, finger division and angle detection to identify the Indian Sign Language character recognition. MediaPipe holistic solution can

detect the ROI like hands, face, pose without being impacted by the background noise and the ability to work on RGB helps to irradiate the need of converting the image to grayscale for handling skin colour detection. Palm identification, finger detection with the finger division and tip detections are obtained from the hand landmarks.

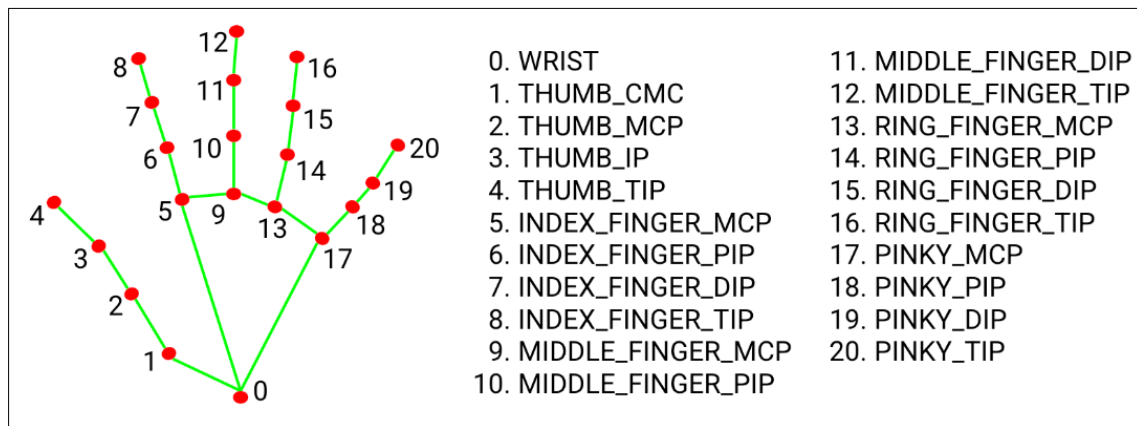


Figure 5-MediaPipe Hand Landmarks (source: Hands - mediapipe (google.github.io))

OpenCV is open-source computer vision library originally written in C and used for processing images and videos for accelerating the machine learning process. It can run on both desktop and mobile. The latest version of OpenCV module available in python is cv2, functions available in with the module will help in data collection and real-time validation. The other important cv2 functions used in the work are VideoCapture(0) helps to activate the inbuilt webcam and imread helps to read the images defaulted in BGR format. There are functions available in cv2 to convert the default colour format to other formats like RGB or grayscale. An image can be displayed on the window using imshow. Puttext will help to write text on an image, release and destroyAllWindows to close the camera.

Text to Speech conversion is applied in data collection helping the signer to change the action with a while signing. Pytsx3 python library is used to give the necessary voice commands. For fingerspelling, a total of 120 video sequence (60 videos for

right-handed signs and 60 videos left-handed signs) are collected for each sign. Whereas for BSL words, a total of 180 video sequence (90 videos for right-handed signs and 90 videos left-handed signs) are collected for each sign. The camera can pick 30 frames in 1 second hence for each class a sequence of 30 frames is captured to form a 30fps video instance. The size of captured frame is 640*480.

Category	Classes	Number of videos per class		Frames per video
		Right-handed signer	Left-Handed signer	
Fingerspelling	26	60	60	30
BSL Word Signs	6	90	90	30

Table 3-Summary of collected data

For the data collection the required signs are referenced from (British Sign Language 2015) that has the collection of BSL fingerspelling and other BSL signs. These signs are cross verified with other sources before self-capturing the signs to ensure the quality of data collection. Figure 6 and figure 7 shows the signs for the finger spelling and selected works for the proposed work.

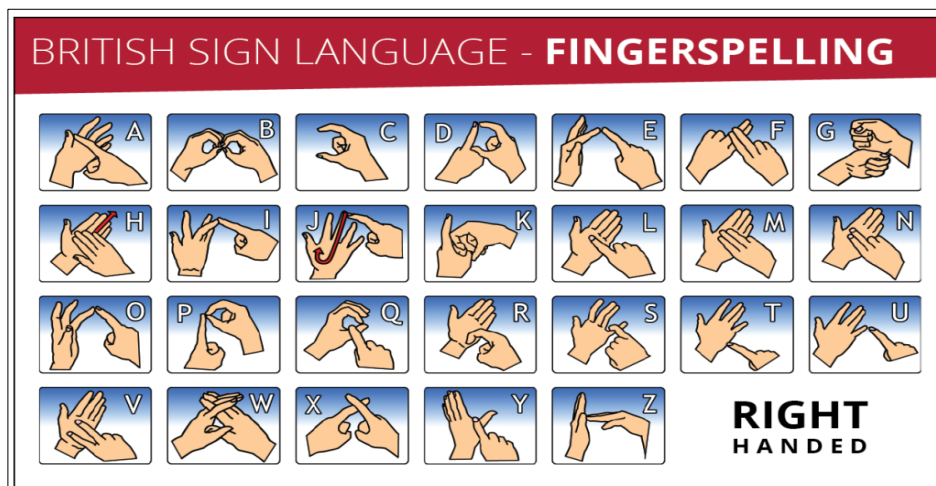


Figure 6-Fingerspelling Chart (British Sign Language 2015)

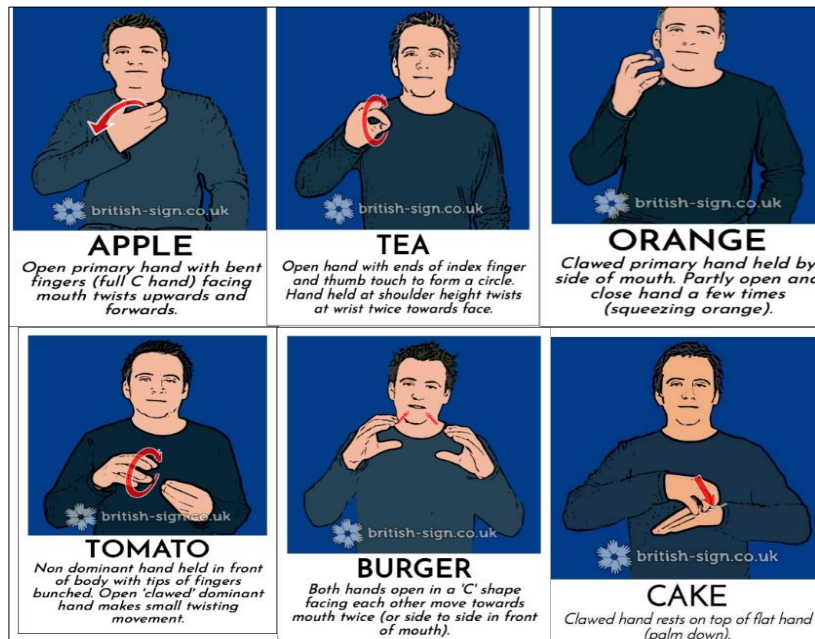


Figure 7-BSL words (British Sign Language, 2015)

An automation script in python is executed to collect and save the data for each class. A main directory Alphabet_Data2 is created to store the sub folders of each class and below algorithm is implemented for data collection.

Step 1: Instantiate the MediaPipe holistic solution and drawing utility module

Step 2: Define a function to process the captured frame

Step 2.1 Convert to captured BGR frame to RGB

Step 2.2 Process it with MediaPipe holistic solution defined in step 1.

Step 2.3 The function will return the image and landmark result.

Step 3: Define the function for drawing the landmarks on the detected frame

Step 3.1 Draw left-hand landmark using the drawing utilities in step 1

Step 3.2 Draw right-hand landmarks like step 3.1

Step 4: Define the function for fetching the key point values (x, y, z) of left and right hands

Step 4.1 Loop through each landmark extracted in step 2.3 and get the x, y, z values of each landmark flattened into an array

Step 4.2 If a hand is not detected, set the array elements as zero

Step 4.3 Return an array concatenated with left hand and right-hand key points

Step 5: Get the current working directory and set path for creating sub directories for each sign

Step 6: Declare a list with the required signs as its elements and set variables for required number of videos and frames

Step 7: For each sign,

Step 7.1 Loop through the number of videos

Step 7.1.1 Automatically create the folders for each sign and subfolders for each video with the path defined in step 5 using python function `makedirs`.

Step 8: Instantiate the function for text to voice conversion

Step 9: For each sign,

Step 9.1-Display the reference image of the sign

Step 9.2-Add a voice command while starting the collection sign for altering the signer

Step 9.3-For each video,

Step 9.3.1-Loop through the required number of frames,

- (i) Fetch the frame and display the frame number being captured on the output feed
- (ii) Pass the frame to MediaPipe holistic model and draw the hand landmarks using function defined in step 3
- (iii) Extract the key point values from the landmarks using function defined in step 4
- (iv) Save the extracted key point values as a NumPy array file in the respective video folder created in step 7.1.1

Step 10: Add necessary delay between the collection of videos and between each of the signs helping the signer to reposition

Snippet of python code for the Step 9 of the algorithm for collecting the data keypoints by looping through the signs and videos are shown in figure 8. The voice commands are activated for the starting of collection of a sign using 'say' command in pyttsx3 module and text instruction get displayed for the collected sign and frame using cv2.putText function. The waitKey (2000) will help in giving a delay of 2 seconds for the signer to reposition at the starting of each video.

```

# Instantiate mediapipe
with mediapipe_holistic.Holistic(min_detection_confidence=0.5, min_tracking_confidence=0.5, smooth_landmarks=True) as model_holistic

# Looping through the signs
for sign in signs:
    # Reading the reference image of each sign for the signers understanding
    ref_image=cv2.imread(f'{cwd}\\Ref_Images\\Alphabet\\Right\\{sign}.png', cv2.IMREAD_COLOR)
    cv2.imshow('Output Screen', ref_image)
    cv2.waitKey(1000) #adding waiting time for ref images
    #Adding voice to alter the user for changing the signed action
    voice.say(f"Starting the collection for {sign}")
    voice.runAndWait()

# Looping through videos
for vid in range(videos):

    # Loop through frames
    for frame_num in range(vid_frames):
        # Reading the CV2 capture
        ret, frame = capture.read()
        # Mediapipe detection
        image, results = detection(frame, model_holistic)
        # Drawing the Landmarks
        draw_custom_landmarks(image, results)

        # Logic for collection
        if frame_num == 0:
            cv2.putText(image, f'Starting the collection for {sign} for video {vid}', (30,100),cv2.FONT_HERSHEY_SIMPLEX,
            cv2.imshow('Output Screen', image)
            cv2.waitKey(2000)

        else:
            cv2.putText(image, f'Fetching frames for {sign} -Video {vid}', (20,12),
            cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 102, 204), 2, cv2.LINE_AA)
            # Display on screen
            cv2.imshow('Output Screen', image)
            # Extracting the keypoints
            keypointvalues = get_keypoints(results)
            numpyarray_path = os.path.join(file_path, sign, str(vid), str(frame_num))
            np.save(numpyarray_path, keypointvalues)

        # Logic for breaking the loop
        if cv2.waitKey(10) & 0xFF == ord('q'):
            break
    capture.release()
    cv2.destroyAllWindows()

```

Figure 8-Python code for the data collection algorithm

Figure 9 demonstrates the real-time collection of key points for a sample frame for sign 'A' captured by the webcam by executing the described algorithm. Left and right hands are detected by the MediaPipe solution, and the landmark drawings help to understand if the key points and related connections are appearing in the desired pattern. The keypoints are collected and saved as numpy array.

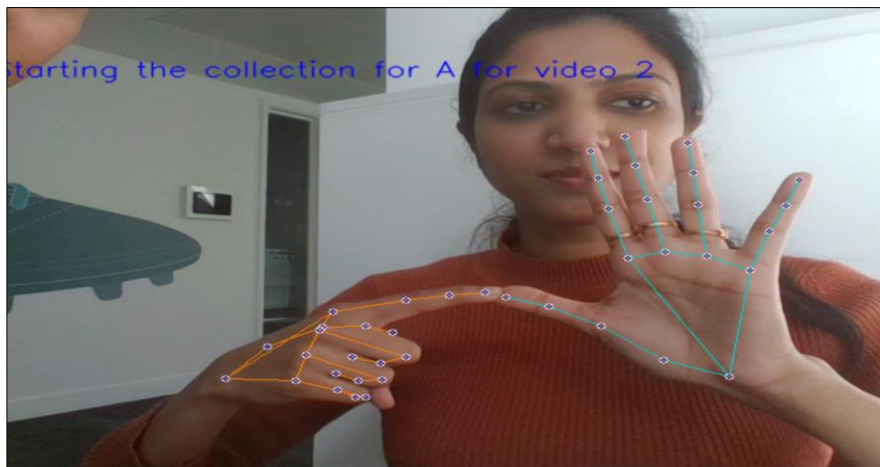


Figure 9-Hand landmarks and connections

Figure 10 illustrates the labelled directories created for each class (A-Z) and sub directories for corresponding videos (0-120). The 21 3D hand landmarks, each for right and left hand is extracted with a total of 126 key points i.e. $(21*3 + 21*3)$ from each frame and saved to respective video folder as a NumPy array file (.npz).

```

13 videos = 60
14 # 30 frames for each video
15 vid_frames = 30

<module 'ntpath' from 'C:\Program Files\Python38\lib\ntpath.py'
D:\Solent\MAIDS\Dissertation\Scripting\Appl

In 7 1 #Creating directories for videos for each sign
      2 for sign in signs:
      3     for video in range(videos):
      4         try:
      5             os.makedirs(os.path.join(file_path, sign, str(video)))
      6         except:
      7             pass
  
```

Figure 10-Creation of directories(A-Z) and subdirectories (0-120) for data collection

Figure 11 exemplifies the (x, y, z) coordinate values of the 21 hand landmarks of a frame saved as a NumPy array file. The first 63 values are for the left-hand landmarks and the last 63 are for the right-hand landmarks.

```
[ 7.20868945e-01  8.42984676e-01 -2.23428771e-07  6.72535241e-01
 8.24164867e-01 -1.95096694e-02  6.29754603e-01  7.92117536e-01
-3.81941944e-02  6.09728754e-01  7.62727797e-01 -5.81864156e-02
 6.35680020e-01  7.40267932e-01 -7.70966485e-02  6.38563395e-01
 6.51186764e-01 -2.59464793e-02  6.00063741e-01  5.78790069e-01
-5.61400056e-02  5.78423858e-01  5.32451749e-01 -7.46057257e-02
 5.64946115e-01  4.90434587e-01 -8.77983421e-02  6.75072193e-01
 6.39876008e-01 -3.30846310e-02  6.28143489e-01  6.46933317e-01
-7.44238570e-02  6.18823290e-01  7.08233535e-01 -8.33414793e-02
 6.23447955e-01  7.41795421e-01 -8.04855675e-02  7.07025111e-01
 6.54014409e-01 -4.33093607e-02  6.59487605e-01  7.05076635e-01
-7.64199644e-02  6.56473637e-01  7.56935120e-01 -6.85065016e-02
 6.65200055e-01  7.70590544e-01 -5.53936511e-02  7.29833841e-01
 6.85893118e-01 -5.66685200e-02  6.85983896e-01  7.42847860e-01
-7.49799311e-02  6.78764999e-01  7.83763647e-01 -6.82732984e-02
 6.84267879e-01  7.92470574e-01 -5.81554472e-02  3.75762403e-01
 5.62517822e-01 -1.94939780e-08  4.31523055e-01  5.45153737e-01
 1.71240896e-03  4.76122379e-01  5.09237468e-01 -7.49453297e-03
 5.14986575e-01  4.89873677e-01 -2.38088500e-02  5.50922632e-01
 4.72497165e-01 -3.89150381e-02  4.53407526e-01  4.00328219e-01
 1.26796644e-02  5.11351407e-01  3.92794967e-01 -1.52349332e-02
 5.42202532e-01  4.21562105e-01 -3.45347784e-02  5.61277211e-01
 4.50045943e-01 -4.42302972e-02  4.34205234e-01  3.86345655e-01
-2.22160527e-03  4.99807268e-01  4.19736326e-01 -2.96915323e-02
 4.79419947e-01  4.74019945e-01 -3.74179892e-02  4.56723452e-01
 4.77918386e-01 -3.72273400e-02  4.17580217e-01  3.89012158e-01
-2.05592215e-02  4.82579619e-01  4.27834898e-01 -4.75497209e-02
 4.60134506e-01  4.78055298e-01 -4.00565565e-02  4.37019169e-01
 4.79027718e-01 -2.84313094e-02  4.01392698e-01  4.01212156e-01
-3.95604745e-02  4.57469195e-01  4.26403970e-01 -5.33318333e-02
 4.46697444e-01  4.64686394e-01 -4.16891538e-02  4.26071763e-01
 4.69495684e-01 -2.86876038e-02]
```

Figure 11-NumPy array file of the hand landmark

The fingerspelling dataset is pre-processed, trained, and tested for two suitable models; the results are then compared to find the best suited model for solving the research problem. The same process of data collection with all landmark points available from MediaPipe holistic solution left hand, right hand, pose, and face is collected for 6 BSL words helping in any future extension of the work from words to sentences and dataset can be reused. However, for the current scope only the first 126 landmarks corresponding to left-hand and right-hand are considered. The pre-processing and modelling are performed for the dataset of six BSL word signs to analyse and compare the performance of the suggested model on both smaller and larger number of classes. The following section in this chapter explains the process based on the fingerspelling dataset. Since the same process is applied to the BSL word signs, the details for the BSL words are provided in the Appendix C.

3.3 Data Cleaning and Pre-processing

The data needs to be cleansed before the training of the model, this is necessary for better performance and results. Pre-processing involves the transformation of the data making it suitable for the training.

3.3.1 Data Cleaning

Data analysis and data cleansing are early important steps in the machine learning process. The quality of the data ensures the performance of the model and effectiveness of prediction. Blurred frames and difference in lighting fails the detection of hands and hence the key point values get extracted as 0. These are invalid frames and needs to be handled before modelling. Analysis and validation of the captured frame is done to ensure the key point values of the left and right hand are extracted without invalid entries. Below algorithm is implemented as a python script to identify and print the list of frames failed to detect hands leaving the landmarks as zero. These invalid videos identified and recaptured to ensure the correctness of data.

Step 1 For every sign in list of signs, repeat step 2

Step 2 Declare an empty list to store the invalid video

Step 1.2 Loop through the number of videos

Step 1.2.1 Loop through the number of frames,

- (i) Read the NumPy array file and store it in a variable
- (ii) Check the elements in array are all zero
- (iii) If values are all zero and video number does not exist in array declared in step 1
- (iv) Append video number to list created in step 2

Step 1.3 Print the sign along with the list of invalid videos

Figure 12 show the invalid videos captured by executing the algorithm. There were three videos for alphabet 'C' where the hand landmarks were not captured for either of the hand. These videos are recaptured, and algorithm was re-executed to validate the videos are correctly captured. This check helps to ensure the correctness of data as illustrated in figure 13.

```
alp=['A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z']

for a in alp:
    invalid_video=[]
    for i in range(0,120):
        for fram in range(frame):
            img_arr= np.load (f'./Alphabet_Data2/{a}/{i}/{fram}.npy')
            all_zero=np.all ((img_arr==0))
            if all_zero:
                if i not in invalid_video:
                    invalid_video.append(i)
                else :
                    None
        print(f'all zero for {a}/{invalid_video}')

all zero for A/[90]
all zero for B/[]
all zero for C/[65, 99, 104]
all zero for D/[]
all zero for E/[]
all zero for F/[]
all zero for G/[]
```

Figure 12-Result of algorithm showing invalid frame

```
##CHECK 2: Checking if there are invalid frames
frame=30
video_checked=120
alp=['A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z']
for a in alp:
    invalid_video=[]
    for i in range(0,video_checked):
        for fram in range(fram):
            img_arr= np.load (f'./Alphabet_Data2/{a}/{i}/{fram}.npy')
            all_zero=np.all ((img_arr==0))
            if all_zero:
                if i not in invalid_video:
                    invalid_video.append(i)
                else :
                    None
        print(f'all zero for {a}/{invalid_video}')

all zero for A/[]
all zero for B/[]
all zero for C/[]
all zero for D/[]
all zero for E/[]
all zero for F/[]
all zero for G/[]
```

Figure 13-Result of algorithm after correcting the invalid instances

3.3.2 Pre-Processing

Pre-processing is the technique of preparing the raw data for modelling. The x, y coordinate values obtained from the landmarks are in the range [0,1] as these are normalised with respect to the width and height of the image. Hence the difference in hand size will not impact the real-time detection.

Numerical encoding the labels: Machine learning algorithms work with numerical predictors and so the categorical values need to be encoded to numerical values. Hence the class alphabets are converted to numerical values and stored as a dictionary of key-value pairs using enumerate function as shown in figure 14.

```
#Create a dictionary for signs with its encoded Label value
sign_label = {label: num for num, label in enumerate(alphabets)}
print(sign_label)

{'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4, 'F': 5, 'G': 6, 'H': 7, 'I': 8, 'J': 9, 'K': 10, 'L': 11,
'M': 12, 'N': 13, 'O': 14, 'P': 15, 'Q': 16, 'R': 17, 'S': 18, 'T': 19, 'U': 20, 'V': 21, 'W': 22,
'X': 23, 'Y': 24, 'Z': 25}
```

Figure 14-Numerical encoding of labels

One hot -encoding of labels: The feature and labels of all the video instances are stored into arrays for training the model. The feature data set will be the array of all the video instances of all the classes, hence it will have a total of 3120 i.e., 120*26 instances. Each video instance having 30 frames with 126 features. The label corresponding to each instance is stored into the label list. This label list needs to be one-hot encoded for setting the independent variable for training the neural network. To_categorical function from Keras helps in one-hot encoding of the label list, each element in the list is converted to a binary vector with '1' at position of its index value. The one-hot encoding of labels will help during training as the model will be compiled with categorical_crossentropy loss function which sends output in one-hot encoded form.

Sample of one-hot coded labels is illustrated in figure 15, first four rows show the one-hot coding of class A, and last four rows shows for class B.

[1 0]
[1 0]
[1 0]
[1 0]
[0 1 0]
[0 1 0]
[0 1 0]
[0 1 0]

Figure 15-One-hot encoding of labels

Train-Test Split: Splitting the data to train and test helps to check model performance on new set of unseen data. The data set is split to 80% train and 20% test data. Further 20% of train data is taken as validation data during training of the model. Table 4 shows the splitting of train and test data for fingerspelling, train data having 2496 videos and test data with 624 videos. Dependent variable y is converted to a matrix with number of columns equal to number of classes and rows equal to number of videos of binary values [0 ,1] using the one-hot coding.

Available Data	Features / Independent variable (X)	Labels/ dependent variable (y)
Total Data	(3120, 30, 126)	(3120, 26)
Train Data	(1997, 30, 126)	(1997, 26)
Validation Data	(499,30, 126)	(499,26)
Test Data	(624, 30, 126)	(624, 26)

Table 4-Train, Test & Validation split

3.4 Model Selection and Evaluation

This section outlines the implementation of Stage 3, first the history of the chosen models, architecture, advantages, and implementation of the models on BSL fingerspelling are explained. Followed by the details of the methods used for evaluation and model tuning.

(Yang *et al.* 2020) has done LSTM and GRU Neural Network Performance comparison study on Yelp dataset. The dataset used in the work is divided into categories of short text -small dataset (700), long text -small dataset (700), short text-large dataset(2K), and long text-large dataset(2K). The structure of GRU is simpler than LSTM, which reduces matrix multiplication and saves a lot of time without sacrificing performance. This holds true for only a small dataset. Hence the two chosen models for this work are LSTM, GRU to check the performance on the created dataset.

3.4.1 Model 1: Long Short-Term Memory (LSTM)

The project aims at predicting the signs from a real time video input. A video is a sequence of frames and hence the prediction should be based on the patterns obtained from the sequential data. LSTM being a sequential model is selected for the work because of its ability to learn the dependencies between the time steps that are far apart, unlike the standard Recurrent Neural Network lacking the ability to learn the long-term dependencies.

LSTM is an advanced version of Recurrent Neural Network (RNN) alleviating the shortcomings of RNN. A RNN network is suitable for recognising the patterns of a sequential data like text or videos with the help of hidden state storing the information from previous time step. However, RNN encounters the issue of vanishing gradient during the backpropagation through time (BPTT), hence vanilla RNN cannot handle long term dependencies. To mitigate the issue, Hochreiter, S. and Schmidhuber, J. introduced LSTM in 1997. Unlike RNN with single tanh layer, LSTM is a gated unit (forget gate, input gate and output gate) with four interacting layers (3 sigmoid and 1 tanh layer). Figure 16 shows the structure of a LSTM unit, each unit has three inputs hidden state of previous time step (h_{t-1}), cell state of previous time step (C_{t-1}) and current time step (x_t) and two outputs i.e., current hidden state (h_t) and current cell state (C_t). The inputs are regulated by three gates

i.e., forget gate, input gate and output gate. Forget gate with sigmoidal layer outputs (f_t) 0 or 1 for inclusion or exclusion of information from previous cell state. Input gate (i_t) with a sigmoidal and tanh layer determines what latest information to be added to the latest cell state. Output gate (o_t) with sigmoidal layer decides what part of the cell state information should be passed to the hidden state (h_t).

Formulation of a LSTM unit is mathematically explained in equations 1-6,

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (1)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2)$$

$$\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \quad (3)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (4)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (5)$$

$$h_t = o_t * \tanh(C_t) \quad (6)$$

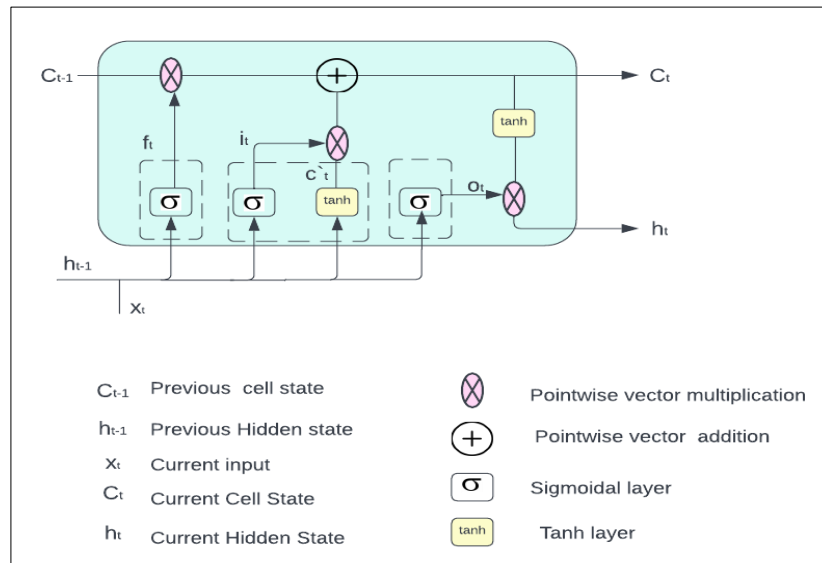


Figure 16-LSTM cell structure

Stacked LSTM: A deep neural network is created by stacking LSTM layers. Adding multiple LSTM layers will deepen the model and help to solve challenging sequence prediction problems. The sequence output of the first layer will be fetched as the input of the second layer and so on. Hence the hyperparameter `return_sequence` should be set to `True` for all layers except the last LSTM layer. Finally, the dense layers are added to the neural network to connect the neurons

from previous layers. The output of the final layer should have the same number of neurons as the number of classes here the final layer will return output from 26 neurons. The complexity of the model depends on the layers and cells, too many cells may overfit the LSTM and too few will not help in learning. Several combination of layers and LSTM units were tried to find a model that could return better performance on training and validation, this parameter tuning was performed while building the model.

Figure 17 illustrates the model summary plotted using Keras plot_model for ease of understanding the stacked layers.

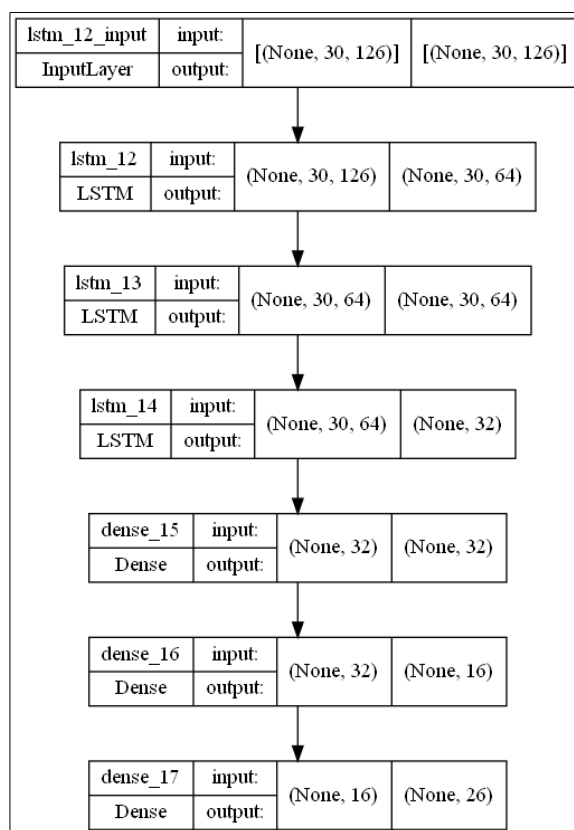


Figure 17-Plotting of LSTM model summary

A sequential model with three LSTM layers is stacked with three dense layers. The ReLu is chosen as the activation function for the LSTM layers. Softmax function is given for the final dense layer as this is a multiclass classification problem. As stated by Sharma *et al.* (2020), Softmax is a combination of multiple sigmoid function and values returned within the range 0 to 1 by the softmax function can be treated as the probability of a particular class. ReLu (rectified linear unit) is efficient and widely used in neural networks as these do not activate all the neurons at the same time.

Defined model is compiled and configured with Adam Optimizer, categorical crossentropy as loss metric and categorical accuracy as evaluation metric. Optimisers are algorithms that helps to adjust the weights of the network nodes by calculating the difference between true and predicted value. There are eight optimisers available in keras and Adam (Adaptive Moment Estimation) with default learning rate of 0.001, this is widely used in neural network training as it consumes little memory and efficient with large data or parameters. Categorical crossentropy loss function is used as this is a multi-class classification problem. Since the labels are one-hot encoded the categorical accuracy is used as the evaluation metric. Model is trained for the train dataset and validated for validation dataset in each epoch. The early stopping helps to monitor the validation loss and find the best epoch avoiding overfitting of the model. The epochs are set to 200 for training and hence the patience of early stopping is set to 15. According to article (Vijay 2020) the patience can be set to 10% of total epochs.

3.4.2 Model 2: Gated Recurrent Unit (GRU)

Gated Recurrent Unit is variation of the LSTM introduced by (Cho et al. 2014). Like LSTM the model can handle vanishing gradient with 2 gates, update and reset gate making computation and implementation simpler. Figure 18 shows the structure of an GRU unit with input X_t at time step t and hidden state of previous time step

h_{t-1} . Update gate is like memory cell of LSTM that controls how much information from previous hidden state goes current hidden state. It adds X_t and h_{t-1} and sigmoid activation function is applied resulting in an output (Z_t) ranging between 0 and 1. Reset gate decides whether the previous hidden state should be ignored. Output from input and reset gate will decide the current hidden state h_t .

Mathematically the GRU can be expressed as follows,

$$\begin{aligned}
 Z_t &= \sigma(W_z X_t + U_z h_{t-1}) \\
 r_t &= \sigma(W_r X_t + U_r h_{t-1}) \\
 h'_t &= \tanh(W X_t + r_t \cdot U h_{t-1}) \\
 h_t &= Z_t \cdot h_{t-1} + (1 - Z_t) \cdot h'_t
 \end{aligned}$$

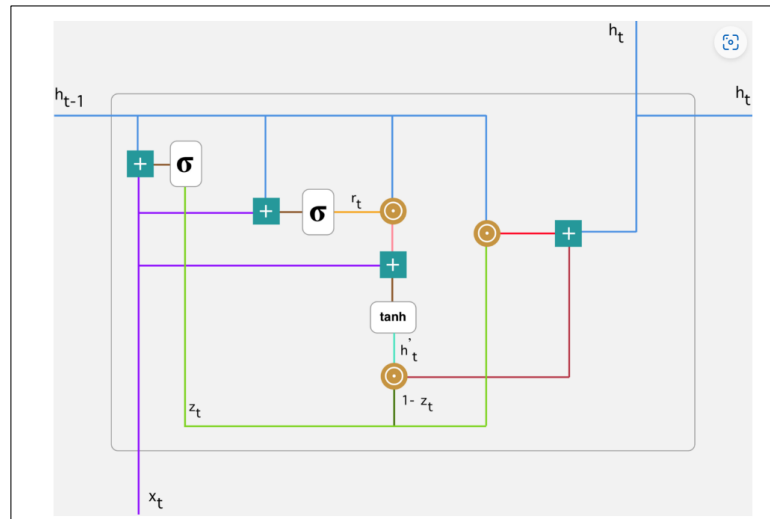


Figure 18-GRU cell structure (Kostadinov 2019)

Implementation of GRU model is illustrated in figure 19. A sequential model is instantiated to stack the GRU layers and the dense layers. The arguments used for the building the GRU model is same as that of LSTM model. This will help to compare the number of parameters and the performance of both the models. Three layers of GRU with units (64, 64, 32) are stacked along with 3 dense layers (32, 16, 26). Relu activation function is used in all layers except the final dense output layer where softmax function is applied.

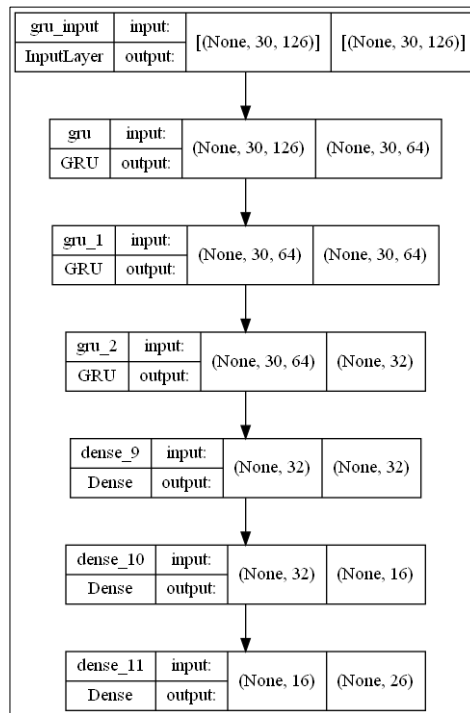


Figure 19- Plotting of GRU model summary

Model is compiled with Adam Optimiser, categorical_crossentropy as loss function and categorical accuracy as evaluation metric. The training loop is customised with EarlyStopping callback. Model is trained for train data set and the performance is monitored for the validation data (20% of train data).

3.5 Evaluation and Tuning

The performance of the model can be evaluated on the test data by assessing the success metrics such as accuracy, F1 score, precision, and recall. For a classification problem the F1_score, precision and recall can be extracted from the 2D confusion matrix in addition to the overall accuracy of the model. 2D Confusion matrix gives an overall idea on the quality of prediction of each class with respect to another. Seaborn heatmap visualisation technique is applied on confusion matrix for displaying the magnitude of prediction of classes with varying hues. (Heydarian *et al.* 2022) explains the multilabel confusion matrix used for multi class

classification. A confusion matrix displays the prediction of all class in one concise view, whereas a multilabel confusion matrix creates class-wise confusion matrix in one-vs-rest way.

The current work is a supervised classification problem with multiple labels, hence a multilabel confusion matrix is derived using from sklearn to see the performance of the model on each class. The class wise matrix helps in numerical analysis of each class. However, visualisation technique of matrix using heatmap is easy and simple to understand than this numerical analysis.

As defined by (Sokolova and Lapalme 2009), precision is the number of correctly classified positive instances to the number of instances labelled by the program as positive. Recall or sensitivity is the fraction of correctly classified positive instances to the number of positive instances in the data. Recall gives the effectiveness of the model to identify positive labels, whereas precision shows the class agreement of the data labels with the positive labels given by the model. F1_score is the widely used performance metric formulated from combination of recall and precision.

Mathematically,

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1\ Score = \frac{2 * precision * recall}{precision + recall}$$

The best value of f1_score is 1 and for this work f1_score > 0.7 is taken as the threshold and accuracy threshold is chosen to be 90%. Keeping in view of the success metric threshold the model is tuned by changing the hyperparameters like LSTM or GRU units, dense layer units and train_test split ratio to get the best model with desired success scores. The hyperparameter tuning was the challenging part of the work to get the best model and the weights are restored using from the best

epoch using early stopping. The trained model is then saved as .h5 file with the model weights. Figure 20 and figure 21 shows the model summary for the implemented LSTM and GRU models with the number of trainable parameters.

The models are given 200 epochs for training, GRU model training ran for 60 epochs whereas LSTM model ran for 103 epochs, the training stopped by monitoring the performance on validation data for patience of 15 epoch. At the end of the set the weights from the best epoch get restored, the epoch from which no improvement happened to the validation loss.

Apart from the training cost the performance metrics of the models should be considered for the selection of model. Results of performance metrics are detailed in chapter 4.

Layer (type)	Output Shape	Param #
lstm_12 (LSTM)	(None, 30, 64)	48896
lstm_13 (LSTM)	(None, 30, 64)	33024
lstm_14 (LSTM)	(None, 32)	12416
dense_15 (Dense)	(None, 32)	1056
dense_16 (Dense)	(None, 16)	528
dense_17 (Dense)	(None, 26)	442
Total params: 96,362		
Trainable params: 96,362		
Non-trainable params: 0		

Figure 20- Number of parameters in LSTM model

Layer (type)	Output Shape	Param #
gru (GRU)	(None, 30, 64)	36864
gru_1 (GRU)	(None, 30, 64)	24960
gru_2 (GRU)	(None, 32)	9408
dense_9 (Dense)	(None, 32)	1056
dense_10 (Dense)	(None, 16)	528
dense_11 (Dense)	(None, 26)	442
Total params: 73,258		
Trainable params: 73,258		
Non-trainable params: 0		

Figure 21-Number of parameters in GRU model

3.6 Graphical User Interface

The models are evaluated based on the defined success metrics and best suitable model is saved. Model is critically evaluated by testing the real-time prediction for five users by running the developed code in pycharm IDE. The python code developed for critical analysis of the saved model is integrated with graphical user interface (GUI) helping the user interact with the system through a web application. The adopted framework adopted for developing the GUI is Streamlit. The

advantage of this open-source framework is the easy to develop web application using python scripts and thereby reducing the development time. It also helps in creating multi-page application with user friendly widgets. However, during the development of application, the direct live streaming in streamlit is slow compared to the OpenCV feed performance. Hence the developed application will help the user to interact using the button widgets and open the camera in the OpenCv window.

A multipage streamlit application is developed with three pages. Figure 22 shows the Home page, briefing the details about the application with a designed logo. The sidebar provided in the page will help the user to navigate to the other pages.

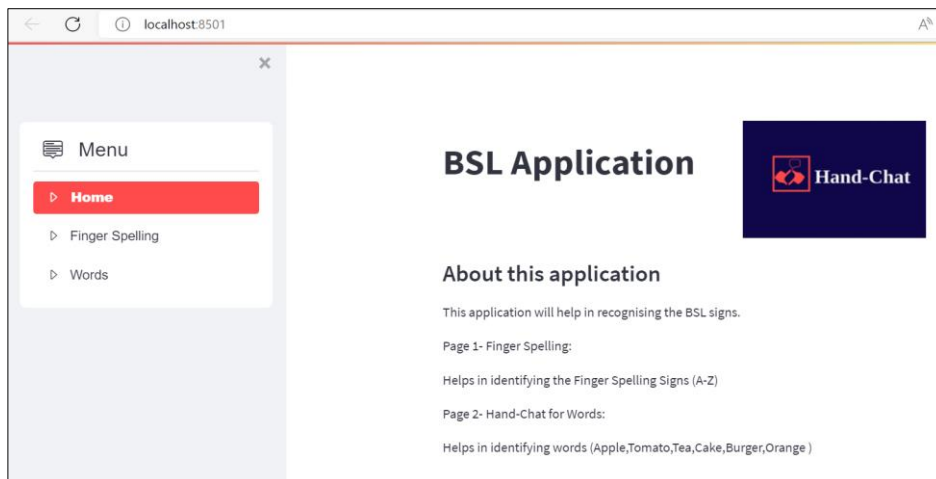


Figure 22- Home page of GUI

Clicking the Hand-Chat for fingerspelling will load the page for finger spelling detection where the instructions for activating and de-activating the camera is given. The push button will help to open the webcam as an OpenCv window, the user can start signing and the predicted output will be printed on the webcam feed as shown in figure 20. Option 'words' on the sidebar loads the page for signing for the trained BSL words (Apple, Tea, Orange, Burger, Tomato, Cake). The user can access the webcam by clicking the start button and start signing, the prediction is

printed on the feed. The page is enabled with text to speech conversion hence the predicted word is also given as an audio output along with the text.

Figure 23 shows the real-time fingerspelling recognition using the application, the signer is signing 'D' and the value is predicted and displayed as text along with the confidence bar. The voice commands are activated using pyttsx3 module by converting the predicted sign in text to speech.

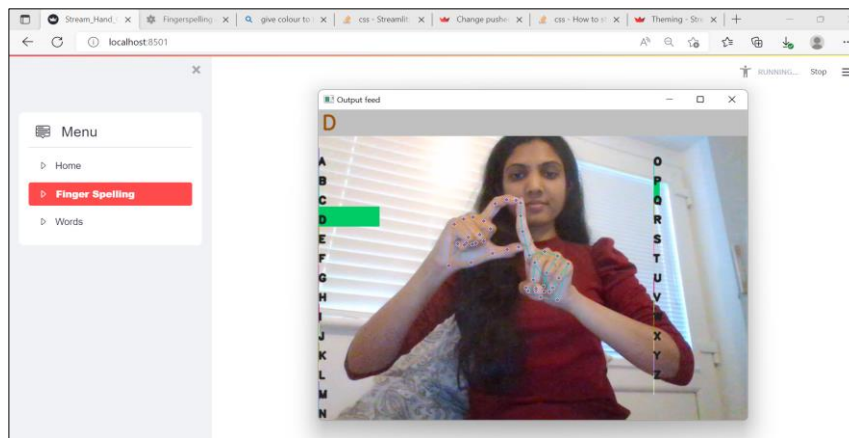


Figure 23- Real-time recognition of fingerspelling

Figure 24 shows the detection of a BSL word 'Tomato', the prediction is displayed as text with confidence bar.

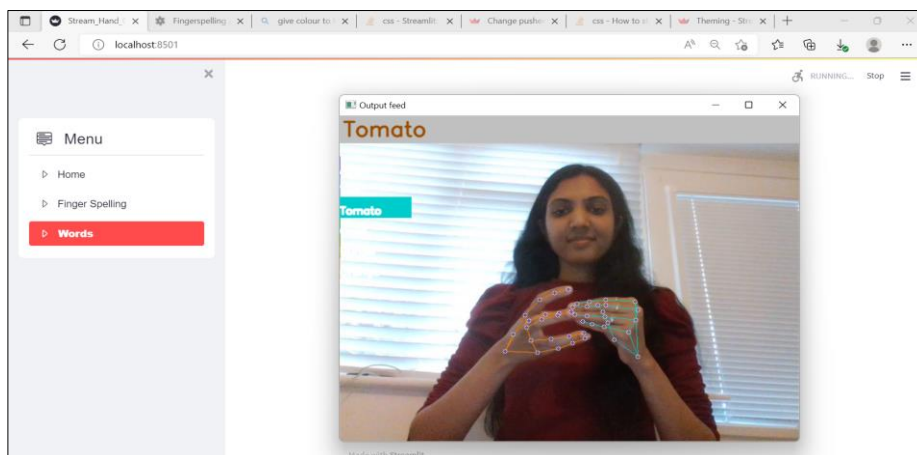


Figure 24-Real-time recognition of BSL word

4. Results

This chapter details the results obtained during the training of LSTM and GRU models for the BSL fingerspelling dataset. The methods and results for quantitative analysis and the output of critical evaluation is documented.

4.1 Quantitative Evaluation

The model evaluation helps to assess the performance of the model on the unseen test data. In this section the results of training vs validation and the defined success metrics are evaluated on the test data using visualisation techniques for easier analysis and understanding.

4.1.1 Accuracy & Loss Plots

Accuracy plots and loss plots for LSTM and GRU models for training and validation dataset is plotted using matplotlib. These plots help to visually analyse if the model is underfitting, overfitting or best fitting. The value of evaluation metric 'categorical accuracy' obtained at the end of each epoch for the training and validation data is plotted for obtaining the accuracy plot, whereas the categorical_crossentropy values at each epoch is plotted to obtain loss plot. With the EarlyStopping callback, the validation loss is monitored at each epoch. This is a regularisation technique to avoid overfitting, if the validation loss doesn't change after certain epoch or if the validation loss starts to increase, then the model training is stopped. The training stopped for GRU model at the end of 60th epoch, the weights are restored from the best epoch resulting in training accuracy of 94.24% and validation accuracy of 90.4%. The model is then evaluated on the unseen test data, resulting in an accuracy of 91.5% for GRU.

Figure 25 shows the accuracy plot of GRU model, the model training accuracy grew from 57% in 10th epoch to 94% in epoch 60th.

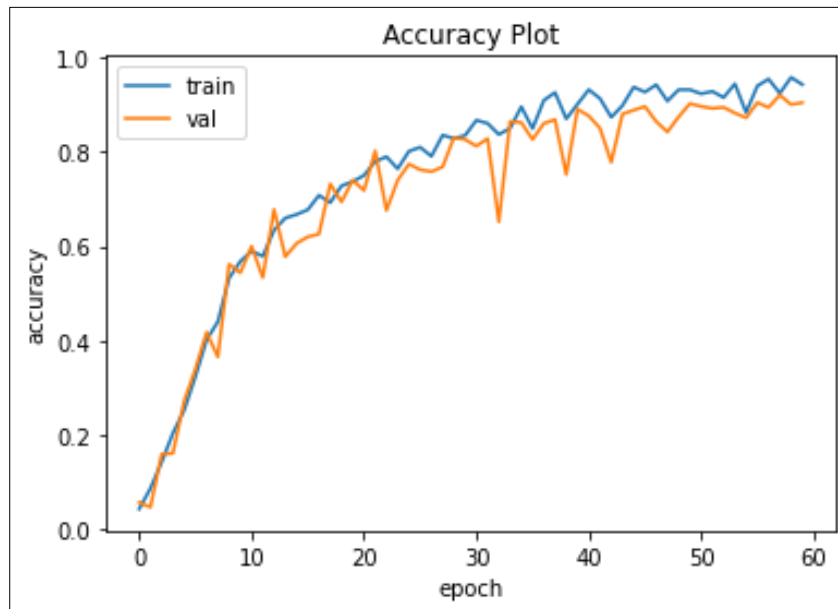


Figure 25-Accuracy plot of GRU

Figure 26 displays the loss plot of GRU model, the model validation loss decreased from 1.2095 in 10th epoch to 0.3077 in 60th epoch.

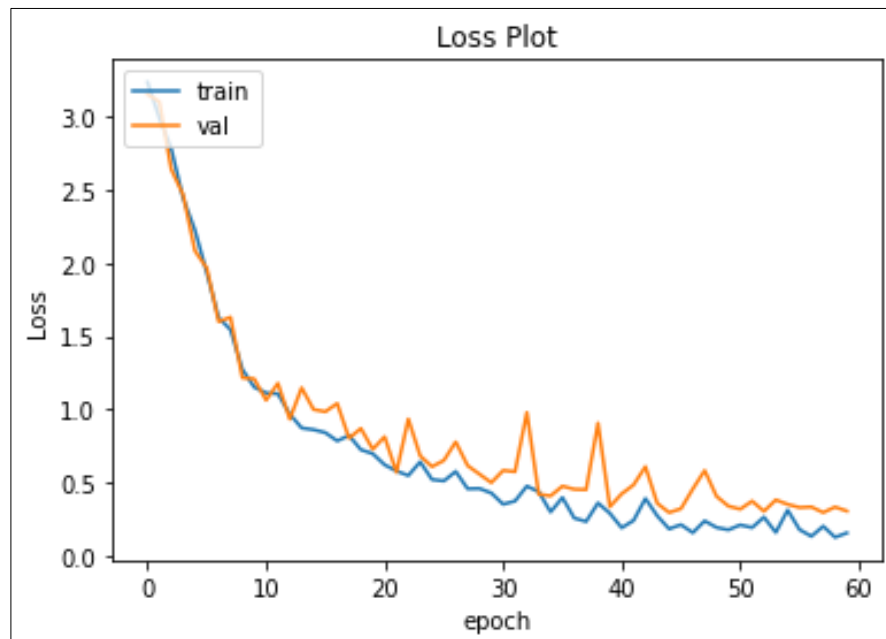


Figure 26-Loss plot of GRU

Figure 27 illustrates the LSTM model resulted in training accuracy of 95.29% and validation accuracy of 88.20%. The training accuracy grew from 20.24% in 10th epoch to 95.29% in 103rd epoch. The performance of the model is evaluated on the test dataset, LSTM gave the test accuracy of 94.23%.

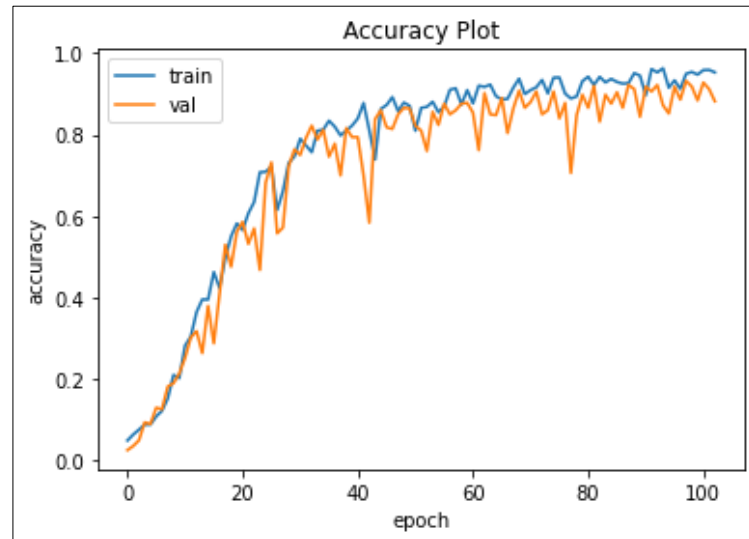


Figure 27-Accuracy plot of LSTM

Figure 28 shows the loss plot of LSTM model, the model validation loss decreased from 2.3666 in 10th epoch to 0.3832 in 103rd epoch.

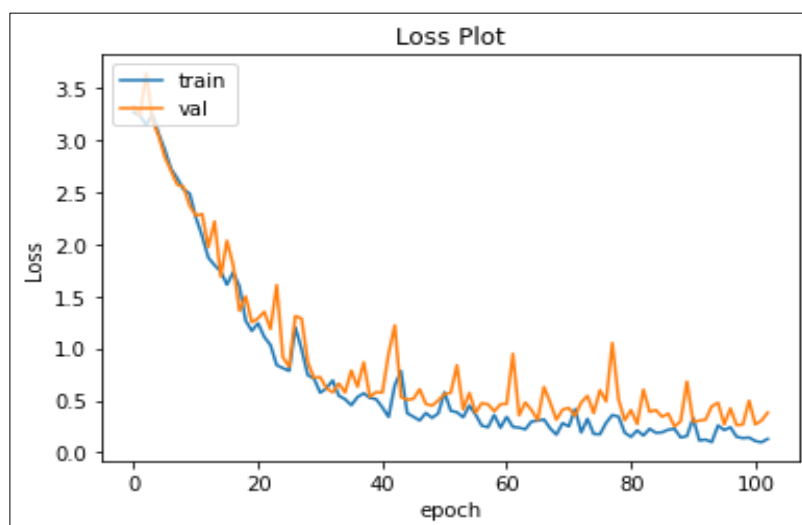


Figure 28-Loss plot of LSTM

Table 5 summarises the accuracy obtained for train, validation, and test data for LSTM and GRU.

Model	Train	Validation	Test
LSTM	95.25	88.2	94.23
GRU	94.24	90.4	91.5

Table 5-Train, validation & Test accuracy of LSTM, GRU

4.1.2 Confusion Matrix

A 2D confusion matrix (26*26) for 26 alphabets plotted for predicted vs true value, the diagonal elements represent the number of accurately predicted instances per class (True Positive) and the non-diagonal components represents false negative and false positive cases. The test data has an imbalance nature as the number of instances is not the same for all classes, hence a normalised confusion matrix is chosen over the simple confusion matrix for making a proper conclusion on the percentage of prediction. A normalised confusion matrix is plotted as shown in figure 29.

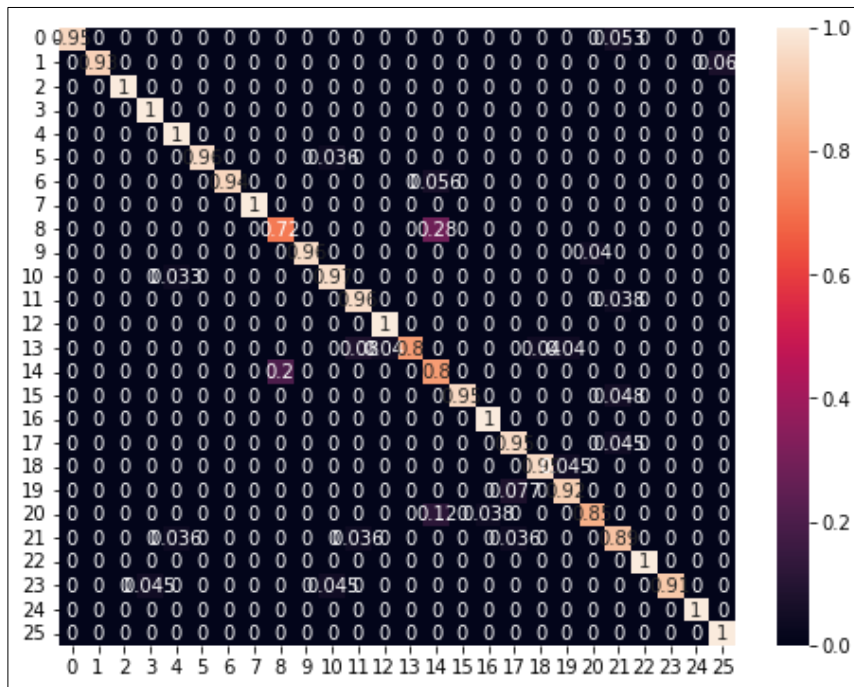


Figure 29-Normalised confusion matrix for LSTM

The normalised confusion matrix for LSTM model shows there are nine classes with TP= 1, twelve classes with 0.9-1, four classes with 0.8-0.9 and one class with 0.72.

Figure 30, Normalised confusion matrix for GRU shows there are seven classes with TP= 1, nine classes with 0.9-1, eight classes with 0.8-0.9 and one class with 0.73 and one class with 0.68.

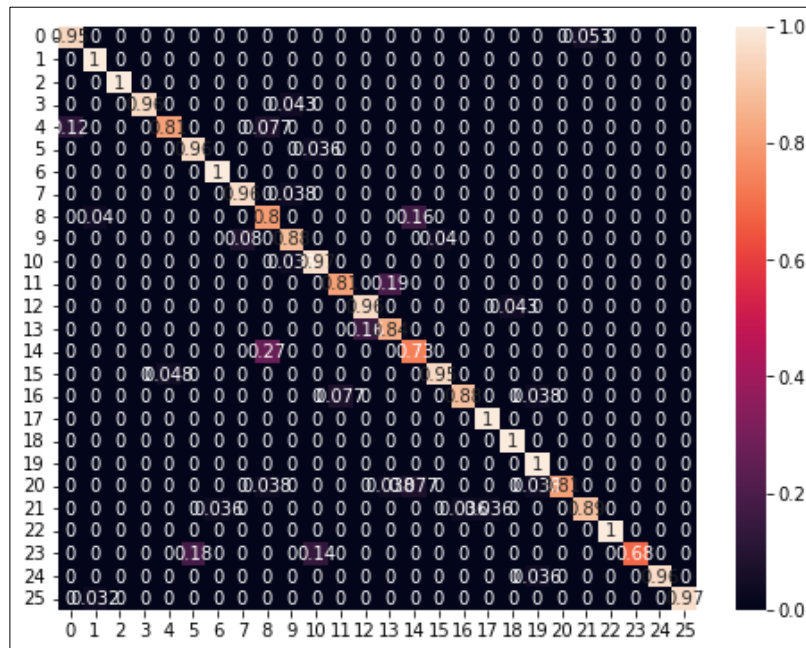


Figure 30-Normalised confusion matrix for GRU

4.1.3 Multi-Label Confusion Matrix

Figure 31 shows the multilabel confusion matrix for LSTM model prediction with first few classes. The sub matrix of the classes is sorted within the confusion matrix according to the label. For example, class 0 -alphabet A, 18 instances are correctly predicted (TP). 1 instance of False negative and 0 instances of False positive. On manual analysis of the 26 classes, 'l', 'N','O', 'V' labels gave a greater number of False negatives (FN) and False positive (FP). For instance, class 'l' with 3 False Positives and 7 False negatives 18 True positives and 598 True Negatives.

```

from sklearn.metrics import confusion_matrix,multilabel_confusion_matrix
#Accuracy of test data
print('Accuracy of Test Data: ',accuracy_score(ytrue2, y_pred2))
print('\nmultilabel_confusion_matrix')
multilabel_confusion_matrix(ytrue2, y_pred2)

Accuracy of Test Data:  0.9423076923076923

multilabel_confusion_matrix

array([[605,  0],
       [ 1, 18]],

       [[609,  0],
       [ 1, 14]],

       [[598,  0],
       [ 0, 26]],

       [[600,  1],
       [ 0, 23]],

```

Figure 31-Multilabel confusion matrix for LSTM

Multilabel confusion matrix for GRU, the classes like 'E', 'I', 'J', 'L', 'N', 'O','X' has more FN and (FP) count over the other classes. For example, class 'I' has 7 FP, 5 FN ,21 TP and 597 TN instances. The balance of FN & FP can be analysed by calculating the F1 Score, a critical metric for evaluation of any classification problem.

4.1.4 F1 Score

Accuracy and True positives alone cannot determine the performance of the model. The False negatives and False positives are also key factors for evaluating the model to understand the misses and incorrect classification. Recall focusses false negative whereas precision on false positive. To consider a harmonic mean of both these parameters the F1_score is considered critical as it can be applied for imbalanced dataset where both false negative and false positives are important. Figure 32 illustrates the bar graph obtained for f1_score of each class for LSTM model. F1_score value range between 0 to 1, where f1_score =1 shows the best prediction. F1_score value of 0.9 is kept as the threshold, keeping an eye on the

this there are four classes (alphabets 'I', 'N', 'O', 'V') falling under 0.9. The minimum value is for class 'O' with 0.63.

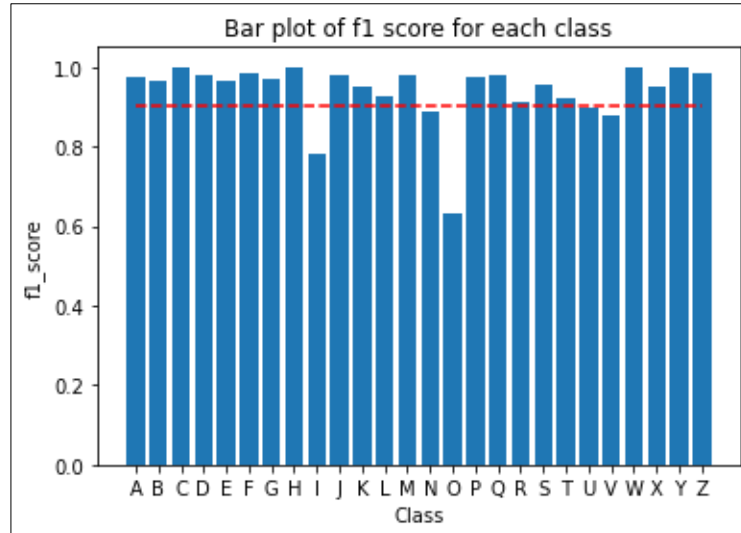


Figure 32-F1_Score of LSTM

Figure 33 shows the f1 scores of each class obtained for the GRU model prediction. There are eight classes (alphabets – 'E', 'I', 'J', 'L', 'N', 'O', 'U', 'X') with f1 score less than 0.9. Class 'O' gives the minimum f1 score value of 0.69.

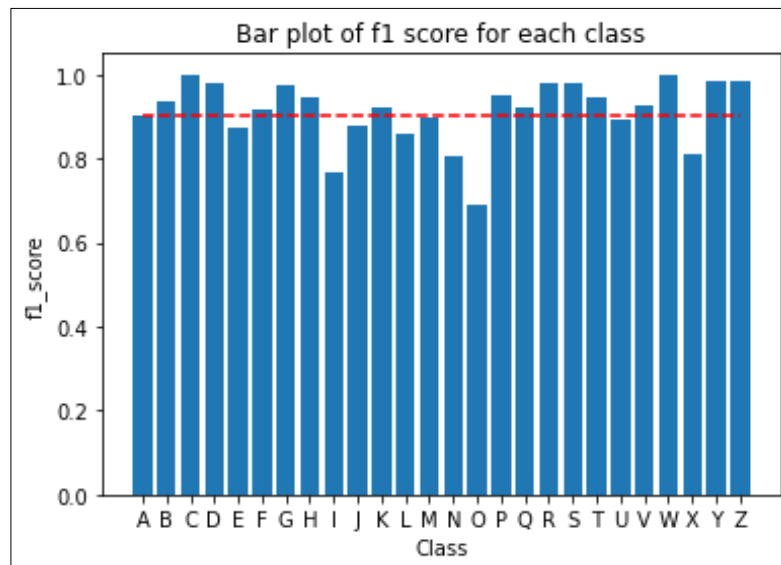


Figure 33-F1_Score of GRU

Table 6 compares the F1 Score, recall and precision values obtained from LSTM and GRU model prediction.

Class	Label	LSTM			GRU		
		F1_score	Recall	Precision	F1_score	Recall	Precision
0	A	0.97	0.95	1.00	0.90	0.95	0.86
1	B	0.97	0.93	1.00	0.94	1.00	0.88
2	C	1.00	1.00	1.00	1.00	1.00	1.00
3	D	0.98	1.00	0.96	0.98	0.96	1.00
4	E	0.96	1.00	0.93	0.88	0.81	0.95
5	F	0.98	0.96	1.00	0.92	0.96	0.87
6	G	0.97	0.94	1.00	0.97	1.00	0.95
7	H	1.00	1.00	1.00	0.94	0.96	0.93
8	I	0.78	0.72	0.86	0.77	0.80	0.74
9	J	0.98	0.96	1.00	0.88	0.88	0.88
10	K	0.95	0.97	0.94	0.92	0.97	0.88
11	L	0.93	0.96	0.89	0.86	0.81	0.91
12	M	0.98	1.00	0.96	0.90	0.96	0.85
13	N	0.89	0.80	1.00	0.81	0.84	0.78
14	O	0.63	0.80	0.52	0.69	0.73	0.65
15	P	0.98	0.95	1.00	0.95	0.95	0.95
16	Q	0.98	1.00	0.96	0.92	0.88	0.96
17	R	0.91	0.95	0.88	0.98	1.00	0.96
18	S	0.95	0.95	0.95	0.98	1.00	0.96
19	T	0.92	0.92	0.92	0.95	1.00	0.90
20	U	0.90	0.85	0.96	0.89	0.81	1.00
21	V	0.88	0.89	0.86	0.93	0.89	0.96
22	W	1.00	1.00	1.00	1.00	1.00	1.00
23	X	0.95	0.91	1.00	0.81	0.68	1.00
24	Y	1.00	1.00	1.00	0.98	0.96	1.00
25	Z	0.98	1.00	0.97	0.98	0.97	1.00
Overall		0.94	0.94	0.94	0.91	0.91	0.92

Table 6-F1 Score, Recall, Precision of LSTM & GRU

4.2 Critical Evaluation

The real-time validation of the selected model (LSTM) is carried out to evaluate how well the model can perform in real time prediction before deploying the model to the graphical user interface. Testing is performed with five signers, where webcam is activated, and the signer will start fingerspelling a sign. The frames are

captured from the feed and features get extracted, the saved model is used to predict the label of the action. If the predicted output matches the expected sign, the user enters an input value 'T' (True) else enters it as 'F' (False) and this gets stored to an array. Similarly, all signs are captured with its label and input status.

Table 7 records the results obtained from the five signers for both right- and left-hand fingerspelling signs. The model is then integrated to the web application GUI for user access.

Alphabets	Right					Left					Percentage
	Signer 1	Signer 2	Signer 3	Signer 4	Signer 5	Signer 1	Signer 2	Signer 3	Signer 4	Signer 5	
A	T	T	T	T	T	T	T	T	T	T	100%
B	T	F	F	F	F	T	T	T	F	T	50%
C	T	T	T	T	T	T	T	T	T	T	100%
D	F	T	T	T	T	T	T	T	T	T	90%
E	T	T	T	T	T	T	T	T	F	F	80%
F	T	T	T	T	T	T	T	T	T	T	100%
G	T	T	T	T	T	T	T	T	T	T	100%
H	T	T	T	T	T	T	T	T	T	T	100%
I	F	T	T	T	F	T	F	F	T	F	50%
J	T	F	T	T	T	T	T	F	T	T	80%
K	T	T	T	T	T	T	T	T	T	T	100%
L	T	T	T	T	T	T	T	T	T	T	100%
M	T	T	T	T	T	T	T	T	T	T	100%
N	T	T	T	T	T	T	T	T	T	F	90%
O	T	T	F	T	F	F	F	F	F	F	30%
P	T	T	T	T	T	T	T	T	T	F	90%
Q	T	T	T	T	T	T	T	T	F	T	90%
R	T	F	T	T	F	T	F	T	T	T	70%
S	T	T	T	T	T	T	T	F	T	F	80%
T	T	T	T	T	T	T	T	T	T	T	100%
U	T	T	T	T	F	T	T	T	F	F	70%
V	T	F	T	T	T	T	T	F	T	T	80%
W	T	T	T	T	T	T	T	T	T	T	100%
X	T	F	T	T	T	T	F	T	T	T	80%
Y	T	T	T	T	T	T	T	T	T	T	100%
Z	T	T	T	T	T	T	T	T	T	T	100%
Overall Accuracy											86%

Table 7-Result of critical evaluation

5. Discussion

The metrics and results are analysed to choose the best model for solving this research problem. The test accuracy obtained for LSTM and GRU are comparable, accuracy of LSTM is 94.23 whereas GRU accuracy is 91.5. The other performance metrics are important for the selection of the best model.

The results of normalised confusion matrix has given insights on the true positive of each class, LSTM model all the classes are predicted with more than 80% accuracy, except for class 8 corresponding to 'I' with 72%.Whereas for GRU the alphabets 'X' (class 23) is predicted 68% correctly and 'O' (class 14) 73% correctly and all other fingerspelling are predicted with more than 80.This shows LSTM and GRU could detect most of the classes accurately.

The manual analysis of multilabel confusion matrix for the 26 classes, for LSTM the classes 'I', 'N','O', 'V' labels gave a greater number of False negatives (FN) and False positive (FP). For instance, class 'I' with 3 False Positives and 7 False negatives 18 True positives and 598 True Negatives. Whereas for GRU, the classes like 'E', 'I', 'J', 'L', 'N', 'O','X' has more FN and (FP) count over the other classes. For example, class 'I' has 7 FP, 5 FN ,21 TP and 597 TN instances. Hence LSTM had less misclassification.

Comparing the F1 score values of two models, both performed the least for class 'O' LSTM with F1_score =0.63 and GRU with 0.69. However, LSTM gave better scores for 22 classes whereas GRU performed well for 18 classes. Overall f1_score for LSTM is 0.94 and GRU is 0.91 Hence LSTM is selected and critically evaluated in real-time validation.

In terms of training, LSTM has a greater number of trainable parameters (96362) than GRU (73258 trainable parameters), this shows the training of GRU will be slightly faster as the number of parameters increase the calculation and weight

adjustment during each epoch. However, the training process is not complex for the used dataset and the computational cost is not high.

It is also noted that for a smaller data set of 6 BSL words with 180 videos per class the LSTM model could give 99.5% accuracy and f1_score=1, this shows t more data instances for BSL fingerspelling will help to achieve better performance with LSTM model.

Author, Year	Description	Performance
Halder and Tayade 2021	MediaPipe along with Support Vector Machine for static images: American signs (26 alphabets and 10 numbers), Indian (24 alphabets), Italian (22 alphabets), and Turkey (10 numbers)	99%
Duy Khuat <i>et al.</i> 2021	MediPipe with LSTM is used to detect Vietnamese Sign Language for 15 words	63%
Alvin <i>et al.</i> 2021	MediaPipe with KNN to detect 24 static American Sign Language	94%
Adhikary <i>et al.</i> 2021	MediaPipe and Random Forest Classifier for 11 Classes	97.4%
Subramanian <i>et al.</i> 2022	MediaPipe with an optimized Gated Recurrent Unit (MOPGRU) for 12 signs	95%
Current Work	MediaPipe with LSTM for BSL hand signs	
	26 BSL alphabet	94.23%
	6 BSL words	99.0%

Table 8-Comparison with related works

Table 8 shows the comparison with the previous works on MediaPipe and sign language recognition. The LSTM model could achieve a good accuracy of 94.23% for 26 classes compared to the (Adhikary *et al.* 2021) with 11 classes, (Subramanian *et al.* 2022) with 12 classes and (Alvin *et al.* 2021) with 24 static classes. (Halder and Tayade 2021) has got better accuracy however the dataset is of single hand signs.

6. Conclusions

Sign language recognition is an area of active research and researchers are using several new techniques to meet the real-time recognition of signs of different sign language families. The research is fuelled using artificial intelligence and different machine learning models. Computer vision techniques help in building inexpensive vision-based recognition system for easier user access. The deep learning models have proved to outperform the other classification model to handle the real-time dynamic signs. The most used algorithm for image classification is the CNN model, however for the sequential data like the videos or continuous frames the recurrent neural networks like LSTM, GRU and their extended models are used because of the ability to remember the required information.

In this work, the feasible method of data collection and latest techniques of feature extraction is adopted for BSL fingerspelling and selected signs. The data collection is a challenging step in the process and collected from real-time capturing of frames using OpenCv. To effectively handle the storage capacity of the collected data, features are directly extracted during real time data collection using MediaPipe. This method has helped to handle larger number of frames in the form of NumPy array with the hand feature coordinates. This technique has also helped in reducing the complex pre-processing steps usually involved in an image classification problem. GRU and LSTM are adopted as the two suitable models as it can handle the series data. For the created dataset LSTM outperformed GRU with an overall f1 score of 0.94 and test data accuracy of 94.23 % which were the key measures of success. Critical evaluation helped to understand the performance of system in real-time and achieved 86% accuracy by testing with five signers. Streamlit web application helped to provide an easy and simple GUI.

This work has been able to cover the objectives of achieving the aim of developing the application for BSL sign language recognition with the proposed methods and methodology.

7. Limitations & Future Works

7.1 Limitations

MediaPipe solution has helped in identification of BSL signs trained using the machine learning models, however flickering of the landmarks are observed in real-time.

The collected data has an orientation limitation for the palm signs as L, M, N are trained in an orientation with dorsal side of non-dominant palm facing the camera, whereas the R, V are trained for the palmer side of non-dominant side facing the camera. Current work scope is limited to hands as area of interest for fingerspelling and words.

The detection is not impacted by cluttered background or objects but seems to be impacted when multiple people and hands appear on screen during real-time detection.

7.2 Future Works

Current work has focussed on the static and dynamic hand sign and finger spelling detections for left- hand and right-hand users. The work can be extended for continuous sign language detection of sentences along with body postures and face expressions for larger vocabulary of words. This will help in building a fully trained BSL recognition system for real-time application.

8. Reference list

- ADEYANJU, I.A., O.O. BELLO and M.A. ADEGBOYE, 2021. Machine learning methods for sign language recognition: A critical review and analysis. *Intelligent Systems with Applications*, **12**, 200056
- ADHIKARY, S., A.K. TALUKDAR and K. KUMAR SARMA, 2021. A Vision-based System for Recognition of Words used in Indian Sign Language Using MediaPipe. *IEEE Xplore* [online], **6**, 390–394 [viewed 26 Aug 2022]. Available from:
https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9702551&casa_token=ibN_ai7s6U8AAAAA:t99DjR0gsZoec_MYuIK-nZeNTBKWcY2HPb_WbIPg4rHXTxWnrauSsInkZb53v6wSbo4MfbzxXS4c
- ALVIN, A. *et al.*, 2021. Hand Gesture Detection for Sign Language using Neural Network with Mediapipe. *Ultima Computing : Jurnal Sistem Komputer*, **13**(2), 57–62
- ANON., n.d. *Holistic* [online] Available from:
<https://google.github.io/mediapipe/solutions/holistic.html>
- ANON., 2017. *British Deaf Association* [online] Available from: <https://bda.org.uk/help-resources/#BSL>
- ANON., n.d. *Hands* [online] Available from:
<https://google.github.io/mediapipe/solutions/hands.html>
- ANON., 2022. *You Searched for Bsl Bill Passes Third Reading* [online] [viewed 7 Jul 2022]. Available from: <https://bda.org.uk/?s=bsl+bill+passes+third+reading>
- ANON., n.d. *Sklearn.metrics.multilabel_confusion_matrix* [online] Available from:
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.multilabel_confusion_matrix.html
- ANON., n.d. *Streamlit Docs* [online] Available from: <https://docs.streamlit.io/>
- ANON., 2019. *Which Countries Recognize Sign Language as an Official Language?* [online] [viewed 13 Jun 2022]. Available from:
<https://www.worldatlas.com/articles/which-countries-recognize-sign-language-as->

an-official-language.html.

ANON., 2022. *Our Work* [online] [viewed 7 Jul 2022]. Available from:

<http://wfdeaf.org/our-work/#:~:text=70%20million%20deaf%20people.%20200%2B%20sign%20languages.%20Unlimited>

ANON., n.d. *British Sign Language Bill Set to Clear Final Stage before Becoming Law*

[online] Available from: <https://www.gov.uk/government/news/british-sign-language-bill-set-to-clear-final-stage-before-becoming-law>

BIRD, J.J., A. EKÁRT and D.R. FARIA, 2020. British Sign Language Recognition via Late Fusion of Computer Vision and Leap Motion with Transfer Learning to American Sign Language. *Sensors*, **20**(18), 5151

BISHT, D. *et al.*, 2022. *Smart Communication System Using Sign Language*

Interpretation [online] [viewed 25 Aug 2022]. Available from: <https://ieeexplore.ieee.org/document/9770914>

BRITISH SIGN LANGUAGE, 2015. *Learn British Sign Language* [online] Available from: <https://www.british-sign.co.uk/>

BROWNLEE, J., 2017. *Stacked Long Short-Term Memory Networks* [online] Available from: <https://machinelearningmastery.com/stacked-long-short-term-memory-networks/>

BUCKLEY, N., L. SHERRETT and E. LINDO SECCO, 2021. A CNN sign language recognition system with single & double-handed gestures. *IEEE Xplore* [online], 1250–1253 [viewed 3 Sep 2022]. Available from:

https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9529449&casa_token=S81QE0KbobwAAAAA:ZI7e_vD8ybngLfYsSFyGBrOBGVJbGGdE58-IGKO2wg0xtdpt-oeWSHMrfjVYRgUrXLKy6bf_U8bc&tag=1

CHO, K. *et al.*, 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* [online] Available from: <https://arxiv.org/pdf/1406.1078v3.pdf>

COOPER, H., B. HOLT and R. BOWDEN, 2011. Sign Language Recognition. *Visual*

- Analysis of Humans* [online], 539–562 Available from:
https://link.springer.com/chapter/10.1007/978-0-85729-997-0_27
- DEY, R. and F.M. SALEM, 2017. Gate-variants of Gated Recurrent Unit (GRU) neural networks. *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pp.1597-1600
- DOBILAS, S., 2022. *LSTM Recurrent Neural Networks — How to Teach a Network to Remember the Past* [online] Available from: <https://towardsdatascience.com/lstm-recurrent-neural-networks-how-to-teach-a-network-to-remember-the-past-55e54c2ff22e>
- DUY KHUAT, B. *et al.*, 2021. Vietnamese sign language detection using Mediapipe. *2021 10th International Conference on Software and Computer Applications*, 162–165
- EBRAHIM AL-AHDAL, M. and M.T. NOORITAWATI, 2012. Review in Sign Language Recognition Systems. *IEEE Xplore* [online], 52–57 Available from: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6222666&casa_token=EtLUMqaf9-AAAAAA:LIAq7P7H-TLweyyynYavmZhWEswypqAYfiTD9x1c1jaRPd-qTB9ETaWta9gEOen_VcIez1nVbk2q
- GRUBER, N. and A. JOCKISCH, 2020. Are GRU Cells More Specific and LSTM Cells More Sensitive in Motive Classification of Text? *Frontiers in Artificial Intelligence*, **3**
- GUARDIAN, T., 2014. *Deaf Couple Angry with Hospital over Lack of Interpreter during Birth of Son* [online] [viewed 7 Jul 2022]. Available from: <https://www.theguardian.com/society/2014/jan/19/deaf-couple-lack-interpreter-birth-university-college-hospital-london>
- HALDER, A. and A. TAYADE, 2021. Real-time Vernacular Sign Language Recognition using MediaPipe and Machine Learning. *International Journal of Research Publication and Reviews*, **2**(5), 9–17
- HAMEED, H. *et al.*, 2022. Privacy-Preserving British Sign Language Recognition Using Deep Learning. *www.techrxiv.org* [online] Available from:

https://www.techrxiv.org/articles/preprint/Privacy-Preserving_British_Sign_Language_Recognition_Using_Deep_Learning/19170257

- HARSHANKUMARHRK, 2022. *What Is Streamlit?* [online] Available from: <https://medium.com/featurepreneur/what-is-streamlit-e106c6d9719d>
- HEYDARIAN, M., T.E. DOYLE and R. SAMAVI, 2022. MLCM: Multi-Label Confusion Matrix. *IEEE Access*, **10**, 19083–19095
- HOCHREITER, S. and J. SCHMIDHUBER, 1997. Long Short-Term Memory. *Neural Computation*, **9**(8), 1735–1780
- IBM CLOUD EDUCATION, 2020. *What Is Artificial Intelligence (AI)?* [online] Available from: <https://www.ibm.com/cloud/learn/what-is-artificial-intelligence>
- JAY, M., 2018. *History of Sign Language - Deaf History - Start ASL* [online] [viewed 6 Sep 2022]. Available from: <https://www.startasl.com/history-of-sign-language>
- KOSTADINOV, S., 2019. *Understanding GRU Networks* [online] Available from: <https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be>
- KUMAR, A., 2020. *Keras - Categorical Cross Entropy Loss Function* [online] [viewed 6 Sep 2022]. Available from: https://vitalflux.com/keras-categorical-cross-entropy-loss-function/#:~:text=categorical_crossentropy%3A%20Used%20as%20a%20loss%20function%20for%20multi-class
- KUMAR, K., 2022. DEAF-BSL: Deep LEarning Framework for British Sign Language recognition. *ACM Transactions on Asian and Low-Resource Language Information Processing*, **21**(5), 1–14
- LIWICKI, S. and M. EVERINGHAM, 2009. Automatic recognition of fingerspelled words in British Sign Language. *IEEE Xplore* [online], pp.50-57 [viewed 14 Nov 2021]. Available from: https://ieeexplore.ieee.org/abstract/document/5204291?casa_token=1vaTR1OTQ8sAAAAA:vRUd0wB7S3sKD_nCbkWXaL48-3cNM6plotaj6tNi5N2T_UX_aUx04QemDKKXWWnNoopmzXb_vIA
- NIMISHA, K. and A. JACOB, 2020. A Brief Review of the Recent Trends in Sign

- Language Recognition. *IEEE Xplore* [online], 186–190 [viewed Aug 2022].
Available from: <https://ieeexplore.ieee.org/document/9182351>
- NISAR, R., 2020. *(Visually) Interpreting the Confusion-Matrix*: [online] [viewed 27 Aug 2022]. Available from: <https://medium.com/analytics-vidhya/visually-interpreting-the-confusion-matrix-787a70b65678>
- OLAH, C., 2015. *Understanding LSTM Networks -- Colah's Blog* [online] Available from: <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>
- OLSZEWSKA, J.I. and M. QUINN, 2019. British Sign Language Recognition In The Wild Based On Multi-Class SVM. *Proceedings of the 2019 Federated Conference on Computer Science and Information Systems*, pp.81-86
- OPENCV, 2018. *About* [online] Available from: <https://opencv.org/about/>
- PATEL, P. and N. PATEL, 2019. Vision Based Real-time Recognition of Hand Gestures for Indian Sign Language using Histogram of Oriented Gradients Features. *International Journal of Next-Generation Computing* [online], 92–102 [viewed 2 Sep 2022]. Available from: <https://ijngc.perpetualinnovation.net/index.php/ijngc/article/view/158>
- RAJ, R.D. and A. JASUJA, 2018. British Sign Language Recognition using HOG. *IEEE Xplore* [online], 1–4 [viewed 27 Aug 2022]. Available from: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8546967&casa_token=o-1c9jjOgfIAAAAA:Sk00XHo8nJsF6RPBRmXKEKB4PrsLX-Zl38p_rtg93vKNovEj7dJs54hvENbizke30NtX_HiMUK0Q&tag=1
- RAMBHAU, P.P., 2013. Recognition of Two Hand Gestures of word in British Sign Language (BSL). *International Journal of Scientific and Research Publications* [online], **3**(10) Available from: <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.396.129&rep=rep1&type=pdf>
- RASTGOO, R., K. KIANI and S. ESCALERA, 2021. Sign Language Recognition: A Deep Survey. *Expert Systems with Applications*, **164**, 113794
- RAVAL, J.J. and R. GAJJAR, 2021. Real-time Sign Language Recognition using Computer Vision. *2021 3rd International Conference on Signal Processing and*

Communication (ICPSC)

- RENOTTE, N., 2021. sign language and artificial intelligence - Search. *www.bing.com* [online] [viewed 9 Sep 2022]. Available from: <https://www.bing.com/videos/search?q=sign+language+and+artificial+intelligence&&view=detail&mid=DE9C2C9FC00DFDDDD19BDE9C2C9FC00DFDDDD19B&&FORM=VRDGAR&ru=%2Fvideos%2Fsearch%3Fq%3Dsign%2Blanguage%2Band%2Baritificial%2Bintelligence%26FORM%3DHDRSC4>
- RENOTTE, N., 2022. *Nicknochnack/ActionDetectionforSignLanguage* [online] Available from: <https://github.com/nicknochnack/ActionDetectionforSignLanguage>
- RUSSELL-PULERI, S., 2020. *Gated Recurrent Units Explained Using Matrices: Part 1* [online] [viewed 28 Aug 2022]. Available from: <https://towardsdatascience.com/gate-recurrent-units-explained-using-matrices-part-1-3c781469fc18>
- SHANGEETHA., R.K., V. VALLIAMMAI. and S. PADMAVATHI., 2012. Computer vision based approach for Indian Sign Language character recognition. *IEEE Xplore* [online], 181–184 Available from: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6428790&casa_token=72ZSDJpMS8YAAAAA:fsLN-g4viH4_dXI2axPmTH_kMnN5rMxVyPYz4e1LgZfblnJAsr3T1w0JaoafOYtDUXT9FYLJADHW&tag=1
- SHARMA, P., 2020. *Keras Optimizers Explained with Examples for Beginners* [online] [viewed 6 Sep 2022]. Available from: <https://machinelearningknowledge.ai/keras-optimizers-explained-with-examples-for-beginners/#:~:text=Keras%20Adam%20Optimizer%20%28Adaptive%20Moment%20Estimation%29%20The%20adam>
- SHARMA, S., S. SHARMA and A. ATHAIYA, 2020. ACTIVATION FUNCTIONS IN NEURAL NETWORKS. *International Journal of Engineering Applied Sciences and Technology*, **04**(12), 310–316
- SOKOLOVA, M. and G. LAPALME, 2009. A systematic analysis of performance measures for classification tasks. *Information Processing & Management* [online],

- 45(4), 427–437 [viewed 29 Aug 2019]. Available from:
<https://www.sciencedirect.com/science/article/pii/S0306457309000259>
- SUBRAMANIAN, B. *et al.*, 2022. An integrated mediapipe-optimized GRU model for Indian sign language recognition. *Scientific Reports* [online], **12**(1), 11964 Available from: <https://www.nature.com/articles/s41598-022-15998-7>
- SUHARJITO *et al.*, 2018. Feature Extraction Methods in Sign Language Recognition System: A Literature Review. *IEEE Xplore* [online], 11–15 [viewed 2 Sep 2022]. Available from:
https://ieeexplore.ieee.org/abstract/document/8626857?casa_token=3q-mZVGpGUQAAAAA:0vrQ924AF4Eb9AiFPTi2mX3wMEu9uMWw7mZzMGZNjPU407MYPM5TNxp9t-YaAGv_VO8dtSf1jJo
- SURESH, A., 2020. *What Is a Confusion Matrix?* [online] Available from:
<https://medium.com/analytics-vidhya/what-is-a-confusion-matrix-d1c0f8feda5>
- TANVIR, M. *et al.*, 2021. Real-Time Recognition of Bangla Sign Language Characters: A Computer Vision Based Approach Using Convolutional Neural Network. *IEEE Xplore* [online], 177–180 [viewed 4 Sep 2022]. Available from:
https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9718800&casa_token=vcLDr3STb20AAAAA:WXHC5SVyCgHezjKDIPU-S4F8JSAXugojYBAvFhHY6rwzTVpN6zjdrbXoeTP5OvNtiV3-aafiQs3
- TEAM, K., n.d. *Keras Documentation: Callbacks API* [online] Available from:
<https://keras.io/api/callbacks/>
- TEAM, K., n.d. *Keras Documentation: Adam* [online] Available from:
<https://keras.io/api/optimizers/adam/>
- TELEPHONE: 0115 965 8781, P.O.C.H.T.S.L.S. 9NA, 2022. *British Sign Language Bill Set to Clear Final Stage before Becoming Law* [online] [viewed 7 Jul 2022]. Available from: <https://www.gov.uk/government/news/british-sign-language-bill-set-to-clear-final-stage-before-becoming-law>
- TRAYNOR, R., 2017. *International Sign Languages – Robert Traynor* [online] [viewed 24 Aug 2022]. Available from:
<https://hearinghealthmatters.org/hearinginternational/2017/international-sign->

languages/

UCL, 2019. *BSL Timeline* [online] Available from: <https://www.ucl.ac.uk/british-sign-language-history/bsl-timeline>

VIJAY, U., 2020. *Early Stopping to Avoid Overfitting in Neural Network- Keras* [online] Available from: <https://medium.com/zero-equals-false/early-stopping-to-avoid-overfitting-in-neural-network-keras-b68c96ed05d9>

YANG, S., X. YU and Y. ZHOU, 2020. LSTM and GRU Neural Network Performance Comparison Study: Taking Yelp Review Dataset as an Example. 2020 *International Workshop on Electronic Communication and Artificial Intelligence (IWECAI)*, pp. 98-101

9. Appendices

9.1 Appendix A: Ethics Application



Project status

Status
Approved

Actions

Date	Who	Action	Comments
09:32:00 08 July 2022	Femi Isiaq	Supervisor approved	
20:28:00 07 July 2022	Shany Stephen	Principal investigator submitted	
20:18:00 07 July 2022	Shany Stephen	Principal investigator saved	

[Get Help](#)

Ethics release checklist (ERC)

Project details

Project name:

Principal investigator:

Faculty:

Level:

Course:

Unit code:

Supervisor name:

Supervisor search:

Other investigators:

Figure 34-Ethics 1

Checklist

Question	Yes	No
Q1. Will the project involve human participants other than the investigator(s)?	<input type="radio"/>	<input checked="" type="radio"/>

Q1a. Will the project involve **vulnerable participants** such as children, young people, disabled people, the elderly, people with declared mental health issues, prisoners, people in health or social care settings, addicts, or those with learning difficulties or cognitive impairment either contacted directly or via a **gatekeeper** (for example a professional who runs an organisation through which participants are accessed; a service provider; a care-giver; a relative or a guardian)?

Q1b. Will the project involve the use of **control groups** or the **use of deception**?

Q1c. Will the project involve any **risk to the participants' health** (e.g. intrusive intervention such as the administration of drugs or other substances, or vigorous physical exercise), or involve psychological stress, anxiety, humiliation, physical pain or discomfort to the investigator(s) and/or the participants?

Q1d. Will the project involve **financial inducement** offered to participants other than reasonable expenses and compensation for time?

Q1e. Will the project be carried out by individuals unconnected with the University but who wish to use staff and/or students of the University as participants?

Q2. Will the project involve sensitive materials or topics that might be considered offensive, distressing, politically or socially sensitive, deeply personal or in breach of the law (for example criminal activities, sexual behaviour, ethnic status, personal appearance, experience of violence, addiction, religion, or financial circumstances)?

Q3. Will the project have detrimental impact on the environment, habitat or species?

Q4. Will the project involve living animal subjects?

Q5. Will the project involve the development for export of 'controlled' goods regulated by the Export Control Organisation (ECO)? (This specifically means military goods, so called dual-use goods (which are civilian goods but with a potential military use or application), products used for torture and repression, radioactive sources.) [Further information from the Export Control Organisation \[https://www.gov.uk/government/organisations/export-control-organisation\]](https://www.gov.uk/government/organisations/export-control-organisation)

Q6. Does your research involve: the storage of records on a computer, electronic transmissions, or visits to websites, which are associated with terrorist or extreme groups or other security sensitive material? [Further information from the Information Commissioners Office \[https://ico.org.uk/for-organisations/guide-to-data-protection/\]](https://ico.org.uk/for-organisations/guide-to-data-protection/)

Get Help

Declarations

I/we, the investigator(s), confirm that:

- The information contained in this checklist is correct.
- I/we have assessed the ethical considerations in relation to the project in line with the

Figure 35-Ethics 2

University but who wish to use staff and/or students of the University as participants?

- Q2.** Will the project involve sensitive materials or topics that might be considered offensive, distressing, politically or socially sensitive, deeply personal or in breach of the law (for example criminal activities, sexual behaviour, ethnic status, personal appearance, experience of violence, addiction, religion, or financial circumstances)?
- Q3.** Will the project have detrimental impact on the environment, habitat or species?
- Q4.** Will the project involve living animal subjects?
- Q5.** Will the project involve the development for export of 'controlled' goods regulated by the Export Control Organisation (ECO)? (This specifically means military goods, so called dual-use goods (which are civilian goods but with a potential military use or application), products used for torture and repression, radioactive sources.) [Further information from the Export Control Organisation \[https://www.gov.uk/government/organisations/export-control-organisation\]](https://www.gov.uk/government/organisations/export-control-organisation)
- Q6.** Does your research involve: the storage of records on a computer, electronic transmissions, or visits to websites, which are associated with terrorist or extreme groups or other security sensitive material? [Further information from the Information Commissioner's Office \[https://ico.org.uk/for-organisations/guide-to-data-protection/\]](https://ico.org.uk/for-organisations/guide-to-data-protection/)

Declarations

I/we, the investigator(s), confirm that:

- The information contained in this checklist is correct.
- I/we have assessed the ethical considerations in relation to the project in line with the University Ethics Policy.
- I/we understand that the ethical considerations of the project will need to be re-assessed if there are any changes to it.
- I/we will endeavour to preserve the reputation of the University and protect the health and







safety of all those involved when conducting this research/enterprise project.

- If personal data is to be collected as part of my project, I confirm that my project and I, as Principal Investigator, will adhere to the General Data Protection Regulation (GDPR) and the Data Protection Act 2018. I also confirm that I will seek advice on the DPA, as necessary, by referring to the [Information Commissioner's Office further guidance on DPA \[https://ico.org.uk/for-organisations/guide-to-data-protection-404/\]](https://ico.org.uk/for-organisations/guide-to-data-protection-404/) and/or by contacting information.rights@solent.ac.uk []. By Personal data, I understand any data that I will collect as part of my project that can identify an individual, whether in personal or family life, business or profession.
- I/we have read the [prevent agenda \[https://www.gov.uk/government/publications/prevent-duty-guidance/prevent-duty-guidance-for-higher-education-institutions-in-england-and-wales\]](https://www.gov.uk/government/publications/prevent-duty-guidance/prevent-duty-guidance-for-higher-education-institutions-in-england-and-wales).

Get Help

Figure 36-Ethics3

9.2 Appendix B: Development Codes

Implementation	Code in the form of PDF
Data Collection for Fingerspelling	 Data_collection_Alph.pdf
Training for fingerspelling	 Alpha_Data_Training.pdf
Data collection for 6 BSL Words	 Word_Data_collection.pdf
Training for BSL words	 Word_Data_Training.pdf
Critical Evaluation	 Critical_Validation.pdf
GUI	 Streamlit_Gui.pdf

9.3 Appendix C: BSL Words Results

Six words trained with LSTM with same architecture, data trained for 48 epochs and the weights got restored by early stopping, training accuracy is 99.42% and validation accuracy is 98.84%.



Figure 37-Accuracy Plot of BSL words

Accuracy for test data is 99.07 and overall F1 scores =0.9



Figure 38-F1_Scores of words

Recall is plotted to see the number of FN and TN , the class 'Tea' has recall 0.93 , rest five classes have recall of 1.



Figure 39-Recall of words

The model trained and tested for six words with 180 videos per class had greater accuracy, this shows with less classes the performance in terms of accuracy is more.