Journal of Engineering Science and Technology Review 16 (4) (2023) 180 - 187

Research Article

JOURNAL OF
Engineering Science and Technology Review

www.jestr.org

Word Level Sign Language Translation using Deep Learning

Vighnesh Pathrikar[1], Tejas Podutwar[1], Akshay Siddannavar[1,*], Akash Mandana[1], K. Rajeswari[1], S.R. Vispute[1] and N.Vivekanandan[2]

[1]*Department of Computer Engineering, Pimpri Chinchwad College of Engineering, India*
[2]*Department of Mechanical Engineering, Pimpri Chinchwad College of Engineering, India*

Received 21 November 2022; Accepted 26 June 2023

Abstract

Sign language is an interactive language through which deaf-mute people can communicate with ordinary people. There are two ways to translate sign language: contact-based recognition and vision-based recognition. The contact-based method depends on external electrical devices, such as sensors, to identify the movements made by the person and translate them into text. The real-time motion of a person is caught via a web camera in the case of the vision-based technique and is subsequently converted to text using image processing and deep learning algorithms. In this paper, we try to compare and contrast various techniques for Sign Language Recognition and Translation. From our review, we came to a conclusion that the models that were trained on custom datasets were more accurate than the one's which were trained on datasets accumulated by researchers like WLASL. And most of the literature used CNN or a form of RNN, like GRU, LSTM and Transformers. From our study, we found that LSTM outperformed all the models with an average accuracy of 85.4% when data augmentation is utilized.

Keywords: Deep Learning, Sign language, American Sign Language, Sign Language Recognition, Classification.

1. Introduction

Speech is the most common form of communication; however, some people have trouble hearing or speaking. For those with these difficulties, communication presents a big challenge. In India, roughly 1 to 2 percent of the population constitutes deaf and dumb people. Communicating with the mute and deaf has always been a challenge for everyone unfamiliar with any type of Sign Language. The visual form of communication known as sign language uses structured hand motions to communicate thoughts. This is the only way to communicate with people with hearing and speaking impairments. Sign Language refers to a language that implies signs made with the hands and other movements, including facial expressions and postures of the body for communication in order to communicate with those people who can't interpret traditional languages of communication due to some disability of senses or other reasons [19].

Sign Language is mostly used by deaf community as they are unable to hear. This leads to a communication gap between deaf and normal people because most of the normal people community don't understand Sign Language.

According to the World Health Organization (WHO) nearly 500 million people worldwide suffer from hearing loss [11]. Eliminating barriers to communication that exist for both the general public and the hard-of-hearing group has gained more attention [12]. An assistive technology that automatically transforms an input gesture into its equivalent voice or text is called sign language recognition (SLR) [13]. As a result, the SLR system helps close the communication gap between the hearing and deaf-mute communities and opens up new opportunities for applications based on human-computer interaction [14–18].

2. Literature Survey

Previous researchers have used a wide range of techniques, these included hardware-based technique, which used some combinations of sensors to identify the gestures, used by the authors of [7] and [9], vision based deep learning on images, which used a collection of images to identify the sign, used by the authors of [1, 2, 5], and vision based deep learning on the pose of the sign, which extracts the landmarks of the gestures and uses that information to identify the sign, used by the authors of [2, 3, 4, 6, 8, 10]. Furthermore, authors of [2, 3] used the common dataset of word level American sign language, authors of [1, 6, 7, 8] created a custom dataset for sign language, authors of [4] used the publicly available Phoenix-2014T dataset, authors of [7] used a sentence level Chinese Language Dataset and authors of [5] used a custom dataset for alphabet level prediction as depicted in Fig. 1.

2.1. Hardware Based Approaches

Z. Wang T. Zhao et al. used multi-channel CNN to account for multiple features in sign language recognition in real-time [7]. They focused on reducing the communication between deaf-mute and ordinary people by sending the recognition results in a text format to a mobile as well as converting that text into speech by using a text-to-voice dictionary in Chinese. Their approach was highly accurate with an accuracy of 89.2%. They proved that their approach has good scalability and was effective in real-time. Their uniqueness was also in part because their model was trained to predict sentences with 4 words and not just restricted to single letters. Though this approach had the prerequisite that the person communicating in sign language had to wear two arm sensors, one for each arm. Their approach could be further improved and modified for longer sentences. This work resulted in the

*E-mail address: aeonic619@gmail.com
ISSN: 1791-2377
doi:10.25103/jestr.164.22

contribution of a sign language database with 20000 samples for further use by researchers.

A hardware glove containing sensors, an accelerometer, and a microcontroller was utilized by Lance Fernandes, Prathamesh Dalvi et al, to interpret the motions based on predetermined values of ranges for each gesture. With this gesture, the text that was provided to an Android application to be converted to speech is translated. The glove's accuracy was initially not very great because of various conflicts between the sensor values, which led it to forecast some values inaccurately. The photos from the dataset were then converted to grayscale images using a random appropriate threshold value, and then further converted to binary images using a software consisting of code to capture images. The accuracy of this model was found to be 99.8% as the dataset was made versatile [9].

2.2. Pose Based Approaches

Pose-based methods rely on localizing key points or joints in the human body from a single shot or video and analysing the posture trajectories. These methods typically use techniques such as pose estimation and keypoint detection to identify the location and movement of the body parts involved in a sign language gesture.

M. Boháek and M. Hrz developed a robust pose normalization scheme that took into account the signing space and processed the hand poses in a separate local coordinate system, independent of the body pose, and based recognition on estimation of the pose of the human body in the form of 2D landmark locations. They also included a number of enhancements to the body posture that increased its accuracy, such as a brand-new augmentation for sequential joint rotation. With 54 joint positions as the input, each picture produced a 108-dimensional posture vector. In the self-attention module, they employed transformers with 6 encoding layers and 9 heads. Once more, they employed 6 decoder layers and 9 heads for decoding. They were able to effectively identify 63.18% of sign recordings in the 100-gloss subset for Word Level American Sign Language (WLASL), an increase of 5% over the previous state of the art. They attained a recognition rate of 43.78% for the 300-gloss subgroup, a 3.8% relative improvement. They reported a test recognition accuracy of 100% using the LSA64 dataset [3].

The problems of low learning efficiency and short-term memory problems of large and complicated approaches [8] were addressed by B. Subramanian, B. Olimov, B. Naik, S.M., et al. They chose mediapipe and adjusted GRU as an approach. They improved the screening of irrelevant data by multiplying the GRU update gate with the reset gate. In comparison to straightforward RNN and other models built on top of RNN, namely, LSTM, regular GRU, BiGRU, and BiLSTM deep learning models, this approach demonstrated greater accuracy. Very low MAE and MSE values as well as strong R-squared values were present in their proposed GRU neural network model. The recommended technique had the advantages of improved prediction accuracy, very effective learning, and information processing capacity. Though the proposed model provides better accuracy it can't be said with certainty that it is always better than other approaches since the study was conducted on a limited dataset.

S. B. Abdullahi and K. Chamnongthai used FFV-Bi-LSTM (Fast Fisher Vector Bi-LSTM) to bring Spatio-temporal prosodic as well as angular features into consideration for prediction of the given gesture for distinguishing similar sign language gestures in a better manner [6]. The advantage that FFV-Bi-LSTM displayed was its high accuracy in the range of 91% to 98%. They considered rhythmic movement making their approach unique. The limitations were that choosing an FV (Fisher Vector) is a trial-and-error process, and FFV-Bi-LSTM was unable to detect small changes in hand trajectory. The result was that they found a new way to classify similar hand gestures. They concluded that sign language recognition should be dealt with as a multi-feature problem in the future. They also contributed to the creation of a large 3D hand-skeletal dataset.

Kayo Yin and Jesse Read, introduced a Spatial-Temporal Multi-Cue (STMC) Transformer which was used in the translation of sign language gloss notation to spoken language translations. Through their findings, they concluded that transformers outperformed recurrent networks in this setup and STMC-Transformer improved the state-of-the-art video-to-text translation by 7 BLEU. Additionally, they accomplished cutting-edge outcomes on several translation assignments using the PHOENIX-Weather 2014T and ASLG-PC12 datasets [4].

2.3. Appearance Based

Appearance-based methods involve analysing the visual appearance of sign language gestures using computer vision and deep learning techniques. This approach focuses on extracting the features of the hands and other body parts and recognizing sign language based on their texture, colour, or shape. Appearance-based methods utilize the visual information obtained from images or videos and often rely on feature extraction techniques.

Sakshi Sharma and Sukhwinder Singh proposed a convolutional neural network for sign language recognition using gestures. The proposed CNN model was termed G-CNN. In this work, a dataset consisting of 2150 images of the gestures of the Indian Sign Language (ISL) was collected using an RGB camera, and a publicly accessible dataset of ASL was used. In the data preparation phase, initially, the samples of the images were collected from the camera, and then they were resized to a size of 256x256. After the images were ready, they were labelled with the appropriate tags which provided a learning basis for the classifier. Once the data was labelled, it was trained against the proposed model. For the classification of sign language hand gestures, VGG-11 and VGG-16, two other architectures, had been explored and modified. The self-collected 43 unique ISL motions were tested to evaluate the performance of the work. For ISL and ASL, respectively, the G-CNN model obtains the highest accuracy of 94.83% and 100% respectively. The evaluation was done using the 10-fold cross-validation method. The suggested vision-based model removes the need for external technology and user dependence, making it easier to operate [1].

Shagun Katoch, Varsha Singh, et. al offered a method for creating an extensive, diversified, and trustworthy real-time Indo-Arabic digits (0-9) and English alphabets (A-Z) identification mechanism for ISL. Their main goal was to create a more universal recognition tool that would improve real-time recognition. A method was proposed that recognizes alphabets (A-Z) as well as Indo-Arabic digits (0-9) in a live video stream using the Bag of Visual Words model (BOVW), and outputs the anticipated labels as both text and speech. For classification, CNN and SVM are employed. With an accuracy of 99%, the system was successfully trained on all 36 ISL static alphabets and digits. SVM and CNN both produced high accuracy classifications of the photos, however

CNN outperformed SVM with fewer features. For superior results, skin colour segmentation and background subtraction were also used [5].

For the challenge of motion gesture recognition, Yanqiu Lao, Pengwen Xiong, and colleagues suggested the B3D ResNet model, which is built on 3D Residual ConvNet and Bi-directional LSTM networks. It is suggested that the B3D ResNet model be used to assess long-term temporal dynamic feature sequences and capture spatiotemporal feature information for video representation. Seventeen convolutional layers, two Bidirectional-LSTM layers, one fully connected layer, and one soft-max layer make up the majority of the B3D ResNet model architecture. This model incorporates video sequences to derive the spatiotemporal properties of the video sequence, which are then applied to the recognition of motion gestures. For the contrast test, two testing datasets, the DEVISIGN-D dataset and the SLR dataset were chosen in order to successfully demonstrate that the proposed network can recognize motion gestures. The B3D ResNet model excelled in extracting context-specific information from a sequence [10].

For the purposes of categorizing sign language, Dongxu Li, Cristian Rodriguez Opazo, et al. employed two standard approaches. The first kind used purposefully chosen characteristics to describe the spatial-temporal information from picture frames and combine them into a higher dimensional code. The first hidden layer was created with 64 neurons and the subsequent layers with 96,128 and 256 neurons respectively and GRU's number of layers was set to 2. They started with a pre-trained VGG16 model for this method and then fine-tuned it using a stacked GRU model. They also used the I3D, three-dimensional convolutional network as another method for appearance-based techniques. The second method was pose-based techniques, which were aimed at localizing the key points or human body joints from a single shot or video. For this method, they used OpenPose to extract 55 body and hand 2D key points from a frame. Which included 13 upper-body joints and 21 joints for both the left and right hands. This was fed to a stack of 2 GRUs with hidden sizes of 64, 64, 128, and 128 for the four subsets respectively. They also proposed their own Temporal Graph Convolution Networks (TGCN), which stacks many residual graph convolutional blocks and uses the average pooling result throughout the temporal dimension as the feature representation of posture trajectories. 3D Convolutional Networks (I3D) trained on appearance-based images had the highest accuracy of 86.98%. And their proposed Temporal Graph Convolutional Network trained on pose-based frames had the 2nd highest accuracy of 79.64% [2].

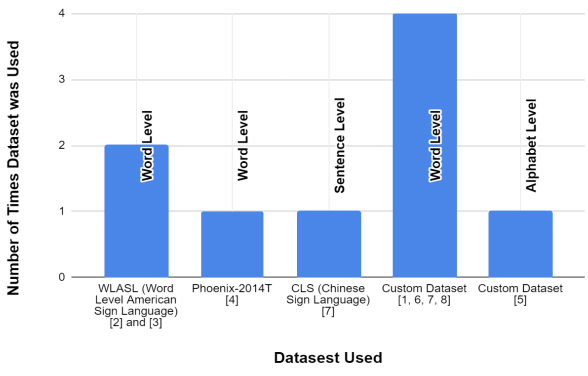

Fig. 1. Comparative Analysis of Datasets chosen.

Table 1. Analysis of various sign language detection algorithms.

Reference No.	Models Compared	Best Model
[2]	Pose-GRU, Pose-TGCN, VGG-GRU, I3D	I3D
[3]	I3D, TK-3D ConvNet, Fusion-3, GCN-BERT, Pose- TGCN, Pose-GRU, SPOTER	TK-3D ConvNet
[8]	standard GRU, BiGRU, simple RNN, LSTM, and BiLSTM	MOPGRU
[1]	G-CNN, VGG-11, VGG-16	G-CNN
[5]	SVM and CNN	CNN
[10]	Bi-LSTM, HMM-DTC, DNN, C3D and B3D ResNet	B3D ResNet

Researchers had used various deep learning algorithms for sign language detection. Some of which include CNN, RNN, LSTM, GRU etc. Other algorithms that are used by the researchers are the modifications to these base algorithms. For example, Pose-GRU and VGG-GRU are the models derived from the GRU model. Similarly, G-CNN, VGG-11, I3D, HMM-DTC, C3D are the other algorithms.

Table 2. Analysis of the accuracy of the best models.

Reference No.	Model	Accuracy
[1]	G-CNN	94.83%
[2]	I3D for WLASL300	86.98%
[3]	TK-3D ConvNet for WLASL300	68.75%
[5]	CNN	99.64%
[6]	FFV-Bi-LSTM	91% to 98%
[7]	Multi-Channel CNN	89.2%
[8]	MOPGRU	95%
[9]	CNN+hardware glove	99.98%
[10]	B3D ResNet	86.9% to 89.8%

The above table shows the accuracies of the best models in each study. Some of the researchers used the custom dataset for training their models. In such studies, the accuracy achieved by the models was high, but they were not able to perform well in real-time sign language detection. While the models trained on public datasets are less biased and tend to provide better results. Amongst the given models, CNN used with the hardware gloves gave the highest accuracy of 99.98%.

3. Methodology

3.1. Feed Forward Neural Networks (FNN)

FNN are indeed the least complex neural networks, which is composed of one or more layers of neurons [27]. Which performs some mathematical functions on a given set of inputs and produces a given set of outputs. The output from each layer is forwarded to the next layer which again performs their own set of mathematical operations on it. The initial layer of neurons is called the input layer, the final layer is named the output layer and any intermediary layers are termed as the hidden layers. In feedforward networks, the information only moves ahead from the input layer, through the hidden layers and to the output layer [26]. The downside

of the Feed forward layers is that it can only deal with data, which is sequential, it only takes into account the current input and it is impaired of the capacity to entirely memorize the information of the past. This is solved by using Recurrent Neural Networks [23].

3.2. Recurrent Neural Networks (RNN)

RNN are an extension of traditional neural networks, having the capability of storing the outcome of a specific layer and passing it back as the input in order to foretell the outcome of the layer. Feed FNNs have a disadvantage, that is, the output can only be propagated in the forward direction.

Thus, RNNs solve this problem by using its "memory", which captures information about what has been calculated so far [24]. Whose equation is given by.

$$z^{(t)} = b + W_1(h^{(t-1)}) + W_2 x^{(t)} \quad (1)$$

$$h^{(t)} = \tanh(z^{(t)}) \quad (2)$$

$$r^{(t)} = c + V(h^{(t)}) \quad (3)$$

Where $h^{(t)}$ represents the hidden state, which depends on the current input state $x^{(t)}$, and the previous hidden state $h^{(t-1)}$. And the output, $r^{(t)}$, is given by a combination of the hidden state and the input state.

The significance of these equations lies in their ability to process sequential data. The intermediate state z(t) combines information from the previous hidden state and the current input, providing a context for the current time step. The hidden state h(t) captures the current representation of the input sequence, incorporating information from previous time steps through the recursive nature of the RNN. The output r(t) is derived from the hidden state and can be used for various tasks such as classification, prediction, or further processing.

The downside of using RNNs is the Vanishing Gradient Problem, that is, the gradient, which carries information becomes smaller and smaller, thus its updates keep becoming more insignificant [25]. This is solved by using LSTM [23].

Recurrent Neural Networks (RNNs) are a type of neural network specifically designed for processing sequential data by maintaining internal memory. RNNs are capable of capturing temporal dependencies and have several use cases where they outperform other neural networks. RNNs are particularly useful for tasks involving sequential data, where capturing and modelling temporal dependencies is critical. RNNs enable the modelling of sequential context, making them valuable in various applications across different domains.

3.3. GRU

Gated Recurrent Neural Networks were designed to solve the issue regarding long-term dependency in the traditional RNN. The gated recurrent neural network is similar to the long-short term memory network with a slight variation in the internal architecture. GRU does not maintain the internal cell state like the LSTM and has the hidden state only. Due to this change, the training of the GRU network is faster compared to the LSTM units. The architecture of the GRU cell composes two gates namely: reset gate and the update gate. The reset gate can also be called the short-term gate while the update gate can be called a long-term gate.

The importance of the previous cell state in order to calculate the current cell state is given by the equation:

$$r_{(t)} = \sigma\left(W_{(r)} \cdot [h_{(t-1)}, x_{(t)}]\right) \quad (4)$$

where $r_{(t)}$ denotes the output value of the reset gate, $W_{(r)}$ denotes the corresponding weight associated with the cell, $h_{(t-1)}$ denotes the cell state at the previous step and $x_{(t)}$ denotes the current input given to the model.

The update gate decides how much part of the updated cell state should be considered to form the new cell state for the current cell state and it is given as:

$$z_{(t)} = \sigma\left(W_{(z)} \cdot [h_{(t-1)}, x_{(t)}]\right) \quad (5)$$

Now, when the outputs of the reset gate and the update gate are calculated, they are used to calculate the current cell state of the GRU cell, which is the output of the cell. The current cell state is calculated as follows:

$$q_{(t)} = \tanh\left(W \cdot [r_{(t)} * h_{(t-1)}, x_{(t)}]\right) \quad (6)$$

$$h_{(t)} = (1 - z_{(t)}) * h_{(t-1)} + z_{(t)} * q_{(t)} \quad (7)$$

where $h_{(t)}$ denotes the output of the GRU cell which is calculated using the $z_{(t)}$ and $r_{(t)}$.

GRU is less CPU intensive compared to the LSTM model providing approximately similar accuracy compared to the LSTM model.

Gated Recurrent Unit (GRU) networks are a type of recurrent neural network (RNN) that, like LSTM networks, can effectively capture and model sequential dependencies. GRUs have a simplified architecture compared to LSTMs, making them computationally more efficient and easier to train. GRU networks are used in video analysis tasks, such as action recognition, gesture recognition, and video captioning. GRUs can process sequential frames of a video and capture the temporal dependencies between different frames, enabling accurate recognition of actions or generation of descriptive captions. GRUs offer a balance between computational efficiency and modelling capacity, making them particularly useful in scenarios where training time or model complexity needs to be optimized.

3.4. LSTM

RNNs bring with them the issue of long-term reliance. LSTM is built to overcome the same. Results from the previous LSTM cell are passed onto the succeeding cell at each iteration [22]. Issue of vanishing gradient is addressed by LSTM [21]. Components of an LSTM cell [23]:

1. Output gate
2. Forget gate
3. Cell state
4. Input gate

Forget gate inputs are previous hidden state & current input, it is later on refined by an activation function as per configuration. If this function's outcome is in proximity of one the information is stored or else it is discarded. The input gate's function is to contribute information to the cell's state. Responsibility of the forget gate is to bring the cell's state up to date. The output gate specifies the value of the following hidden state[23]. The equation for forget gate of LSTM is given as:

$$u_t = \sigma\left(m_i\,[l_{t-1}, h_t] + r_i\right) \quad (8)$$

$$c_t = tanh\ (m_i\ [l_{t-1}\ ,\ h_t] + r_i) \quad (9)$$

Input gate has the responsibility of bringing the cell state up to date.

$$v_t = \sigma\ (m_f\ [l_{t-1}\ ,\ h_t] + r_f) \quad (10)$$

The forget gate decides which information from the l_{t-1} and h_t is to be forgotten by supplying it to the sigmoid (σ) activation function.

$$w_t = \sigma\ (m_o\ [l_{t-1}\ ,\ h_t] + r_o) \quad (11)$$

The next hidden cell state's value is specified by the output (w_t). Here u_t illustrates forget gate, v_t illustrates input gate, w_t illustrates output gate, σ illustrates the sigmoid activation function, m_x illustrates weight associated with neurons in respective gates, l_{t-1} is the output of the time step ($t-1$), input supplied for the current time step is depicted by h_t , and r_x is the value of bias for the gates.

The significance of these equations lies in the LSTM's ability to selectively update, forget, and output information, thus enabling the network to capture and retain long-term dependencies in sequential data.

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network (RNN) that are specifically designed to address the vanishing gradient problem, allowing them to effectively capture and remember long-term dependencies in sequential data. LSTM networks are well-suited for time series forecasting, anomaly detection, and pattern recognition tasks. They can capture temporal dependencies and learn complex patterns in the data. Applications of LSTM in this domain include stock market prediction, weather forecasting, energy load forecasting, and predicting customer behaviour.

3.5. Bi-LSTM

The Bidirectional-LSTM RNN is mostly employed in natural language processing. Unlike traditional LSTM, the input travels in both directions, and it may use information from both sides. It's also an effective tool for simulating the sequential relationships between words and sentences in both directions.

To summarize, BiLSTM adds an additional LSTM layer that reverses the direction of information flow. In a nutshell, the input sequence flows backwards in the extra LSTM layer. The outputs of both LSTM layers are then combined in a variety of methods, including average, sum, multiplication, and concatenation. The equation for forward input gate of Bi-LSTM is given as:

$$\vec{v}_t = \sigma\ (m_f\ [l_{t-1}\ ,\ h_t] + r_f) \quad (12)$$

The equation for backward input gate of Bi-LSTM is given as:

$$\vec{v}_t = \sigma\ (m_f\ [l_{t+1}\ ,\ h_t] + r_f) \quad (13)$$

where v_t illustrates input gate, σ illustrates the sigmoid activation function, m_x illustrates weight associated with neurons in respective gates, l_{t-1} is the output of the time step ($t-1$), input supplied for the current time step is depicted by h_t , and r_x is the value of bias for the gates.

Bidirectional Long Short-Term Memory (Bi-LSTM) networks are a type of recurrent neural network (RNN) that have the ability to process sequential data in both forward and backward directions. This bidirectional nature makes them suitable for a variety of applications where the context from both past and future inputs is important. Natural Language Processing is one of the most important applications of Bi-LSTM. The bidirectional nature of Bi-LSTMs allows them to capture dependencies in both directions, making them effective for tasks where the meaning of a word or phrase depends on its surrounding context.

3.6. Mediapipe

A tool called Mediapipe is used to build machine learning pipelines for time series data, including audio and video, and so on [8]. This multi-platform framework is compatible with Android, Desktop Server, embedded devices, and iOS such as the Raspberry Pi and Jetson Nano.

A specific pre-trained TensorFlow or TFLite model serves as the foundation for solutions, which are open-source pre-built examples. The Framework serves as the foundation for MediaPipe Solutions. To find the critical spots, we used the Face, Hand, and Pose solutions from the sixteen solutions it currently offers.

Since MediaPipe provides pre-trained models and APIs for pose estimation, which can accurately detect and track landmarks on human bodies, facial expressions, hand gestures and it is designed to efficiently process video and image data in real-time, it makes it the best framework to go with.

4. Implementation

4.1. Data Collection

The WLASL dataset contains over 2,000 unique ASL gestures corresponding to a vocabulary of 2,000 words. These gestures were recorded from diverse signers in a controlled environment with consistent lighting and background conditions. The dataset provides a wide range of variations in sign language gestures, allowing for robust training and evaluation of models [2].

Each video in the dataset is labelled with the corresponding word it represents, enabling the development of algorithms for automatic recognition and classification of ASL gestures [2].

The data is in MP4 video format, consisting of a collection of videos capturing ASL gestures. The videos have durations ranging from 0.3 to 0.8 seconds, providing concise representations of the sign language gestures. The dataset includes a JSON file that serves as a mapping reference, associating words with their corresponding video IDs. In some cases, multiple video IDs may be linked to a single word. This JSON mapping provides a convenient way to access and organize the videos based on the specific words they represent [2].

This data was collected by previous researchers from multiple sign language educational websites and YouTube tutorials [2].

Using RNN-based neural networks (RNN, LSTM, GRU, Bi-LSTM) is highly beneficial for video data processing in sign language prediction due to their ability to capture temporal dependencies and sequential patterns in the data.

Using CNNs for video data processing in sign language prediction offers several advantages. CNNs excel in capturing spatial patterns making them suitable for analysing individual frames of sign language videos. CNNs can detect features, such as hand shapes, facial expressions, and body movements,

which are crucial for interpreting sign language gestures.

Since we wanted to build a system which satisfies a real-world use case, we decided to use the WLASL dataset, instead of creating our own. Fig. 3 showcases an overview of the proposed architecture, we capture a frame from the video, extract the landmark coordinates using mediapipe, then using a collection of such coordinates for a video, we label it. In this way the labelled dataset was prepared, which was further expanded using scaling. This data was then used to train four different models, RNN, LSTM, GRU and Bi-LSTM, since these models were common in the reviewed literature.

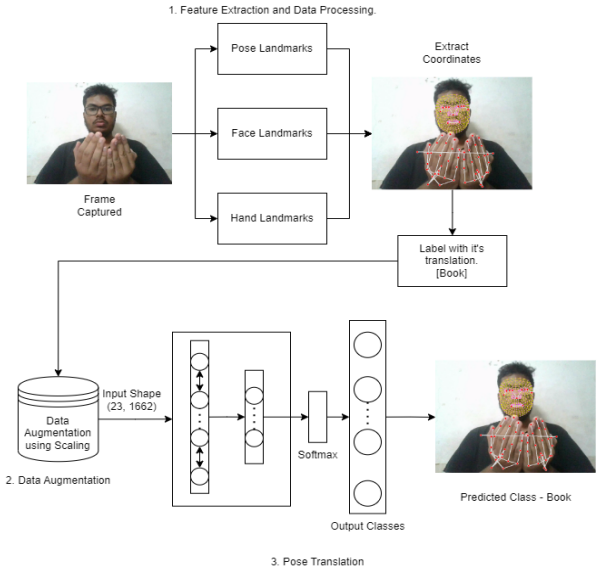

Fig. 3. Overview of proposed architecture.

4.2. Data Pre-processing

We used Word Level American Sign Language, WLASL, the largest video dataset for Word-Level American Sign Language (ASL) recognition, which features 2,000 common different words in ASL [2].

On this dataset, we used Mediapipe, and marked 468 face landmarks, 21 hand landmarks, per hand and 33 full body landmarks [20].

For the words chosen, we chose the first 15 words with the greatest number of videos, i.e., "book", "drink", "computer", "before", "chair", "go", "clothes", "who", "candy", "cousin", "deaf","fine","help","no" and "thin".

Since LSTM requires input sizes to be the same, we took the least number of frames, 23 in our case, from the chosen videos, and uniformly chose the same number of frames, 23, from all the videos.

4.3. Data Transformation

For Evaluating the Models, we decided to Augment the 23 frames using Scaling, this increases the training data and helps us in getting better results.

We also found out that face landmarks make most of the input data, but only some of the key points contribute towards the prediction of the sign, hence we decided to also test the models without the face key points as an input. Fig. 4 depicts the two classes of the chosen landmarks and the points which are marked by them.

This left us with four permutations on which we will train our models:

1. With face landmarks and without data augmentation.
2. With face landmarks and with data augmentation.
3. Without face landmarks and without data augmentation.
4. Without face landmarks and with data augmentation.

After deciding which data to use, we fixed the test size to 0.2 and proceeded with our implementation.

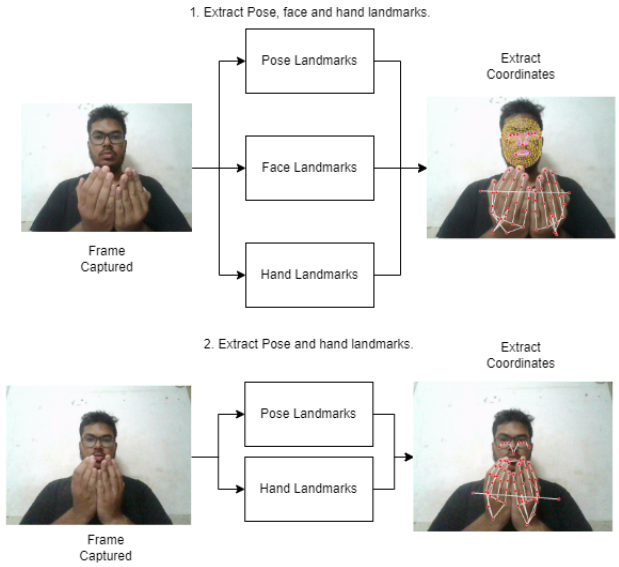

Fig. 4. Comparison between chosen landmarks.

4.4. Modelling and Results

4.4.1. RNN

For RNN, we used a hidden layer with size 128, followed by a dense layer of size 10, followed by a dense layer of size 15. The results for the permutation of the inputs were as follows.

1. For just face landmarks, an accuracy of 22.22% was attained after 450 epochs of training the model on the filtered key points.
2. For face landmarks with data augmentation, an accuracy of 83.33% was attained after 450 epochs of training the model on the filtered key points.
3. For just hand and pose landmarks, an accuracy of 18.15% was attained after 180 epochs of training the model on the filtered key points.
4. For hand and pose landmarks with data augmentation, an accuracy of 83.33% was attained after 150 epochs of training the model on the filtered key points.

4.4.2. GRU

For GRU, we used a three hidden layer with size of 64, 128 and 64 units respectively followed by two dense layers of size 64 and 32 units respectively. The results for the permutation of the inputs were as follows.

1. For just face landmarks, an accuracy of 18.51% was attained after 176 epochs of training the model on the filtered key points.
2. For face landmarks with data augmentation, an accuracy of 81.481% was attained after 163 epochs of training the model on the filtered key points.
3. For just hand and pose landmarks, an accuracy of 18.52% was attained after 186 epochs of training the model on the filtered key points.
4. For hand and pose landmarks with data augmentation, an accuracy of 87.04% was attained after 97 epochs of training the model on the filtered key points.

4.4.3. LSTM

For LSTM, we used three hidden layers with size 32, 128 and 64 units respectively followed by two dense layers of size 64 and 32 units followed by a dense layer of size 15 units. The results for the permutation of the inputs were as follows.

1. For just face landmarks, an accuracy of 22.22% was attained after 375 epochs of training the model on the filtered key points.
2. For face landmarks with data augmentation, an accuracy of 83.33% was attained after 324 epochs of training the model on the filtered key points.
3. For just hand and pose landmarks, an accuracy of 25% was attained after 150 epochs of training the model on the filtered key points.
4. For hand and pose landmarks with data augmentation, an accuracy of 87.5% was attained after 100 epochs of training the model on the filtered key points.

4.4.4. Bi-LSTM

For Bi-LSTM, we used one hidden layer with size 128, followed by two dense layers of size 32 and 15. The results for the permutation of the inputs were as follows.

1. For just face landmarks, an accuracy of 22.22% was attained after 400 epochs of training the model on the filtered key points.
2. For face landmarks with data augmentation, an accuracy of 72.22% was attained after 500 epochs of training the model on the filtered key points.
3. For just hand and pose landmarks, an accuracy of 25.95% was attained after 125 epochs of training the model on the filtered key points.
4. For hand and pose landmarks with data augmentation, an accuracy of 83.33% was attained after 105 epochs of training the model on the filtered key points.

4.4.5. Results

From the experiments, it is observed that the addition of face landmarks usually led to a decrease in the accuracy of the models. Furthermore, data augmentation played an important role in increasing each model's accuracy as depicted in Fig. 5.

Among the models, LSTM performed the best when data augmentation was in play, with an average accuracy of 85.41%, followed by GRU at 84.25%, RNN at 83.33% and finally Bi-LSTM at 77.77%. When data augmentation was not utilized, Bi-LSTM performed the best with an average accuracy of 24.08%, followed by LSTM at 23.61%, RNN at 20.18% and finally GRU at 18.155%.

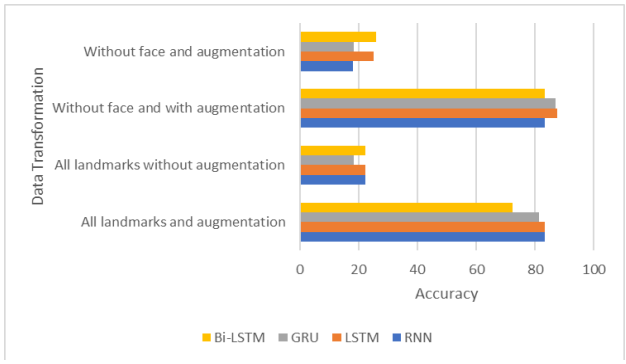

Fig. 5. Comparison of each model for the permutation of the input data.

5. Conclusion

In this study, the results of various approaches for Sign language recognition are compared.

This research discovered the following limitations after reviewing the relevant literature. First, the dataset used by the previous literature wasn't the same, making it difficult to conclude from the literature. Second, the sign languages chosen for prediction were different. Third, some papers rely on datasets built by the researchers themselves either directly or with the help of volunteers, hence the data itself in the dataset might be improper due to the lack of expertise of the signer and may introduce bias depending on different features used for SLR. Furthermore, from the previous literature, we conclude that the models that were trained on custom datasets were more accurate than the ones which were trained on accumulated datasets from professional signers, like WLASL. And most of the literature used CNN or a form of RNN, like GRU, LSTM, and Transformers.

Thus, we opted to utilize a general dataset WLASL. We first extracted the coordinates for pose, hand, and face from the WLASL dataset for a few chosen words. These were then used to model RNN, LSTM, GRU, and Bi-LSTM, in which LSTM on average outperformed the rest of the models. The proposed architecture can be effectively used to translate a signers gesture in real-time.

Further iterations of this work should use a publicly available dataset for the benefit of society, so that the results may be used by a wide range of individuals in everyday life. They should also consider a better image processing method to map the key features of the face and the body. Also, developing a low-latency architecture solution for common use of sign language detection is an aspect which requires extensive research and experimentation.

References

[1] S. Sharma and S. Singh, "Vision-based hand gesture recognition using deep learning for the interpretation of sign language," *Exp Sys Appl*, vol. 182, p. 115657, Nov. 2021, doi: 10.1016/j.eswa.2021.115657.

[2]D. Li, C. R. Opazo, X. Yu, and H. Li, "Word-level Deep Sign Language Recognition from Video: A New Large-scale Dataset and Methods Comparison," 2019, doi: 10.48550/ARXIV.1910.11006.

[3] M. Bohacek and M. Hruz, "Sign Pose-based Transformer for Word-level Sign Language Recognition," in *2022 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)*, Waikoloa, HI, USA: IEEE, Jan. 2022, pp. 182–191. doi: 10.1109/WACVW54805.2022.00024.

[4] Y. Kayo and R. Jesse, "Better Sign Language Translation with STMC-Transformer," in *Proceedings of the 28th International Conference on Computational Linguistics*, Barcelona, Spain: International Committee on Computational Linguistics, Dec. 2020, pp. 5975–5989. [Online]. Available: https://aclanthology.org/2020.coling-main

[5]S. Katoch, V. Singh, and U. S. Tiwary, "Indian Sign Language recognition system using SURF with SVM and CNN," *Array*, vol. 14, p. 100141, Jul. 2022, doi: 10.1016/j.array.2022.100141.

[6] S. B. Abdullahi and K. Chamnongthai, "American Sign Language Words Recognition Using Spatio-Temporal Prosodic and Angle Features: A Sequential Learning Approach," *IEEE Access*, vol. 10, pp. 15911–15923, 2022, doi: 10.1109/ACCESS.2022.3148132.

[7] Z. Wang et al., "Hear Sign Language: A Real-Time End-to-End Sign Language Recognition System," in *IEEE Trans. on Mobile Comput*, vol. 21, no. 7, pp. 2398-2410, 1 July 2022, doi: 10.1109/TMC.2020.3038303.

[8] B. Subramanian, B. Olimov, S. M. Naik, S. Kim, K.-H. Park, and J. Kim, "An integrated mediapipe-optimized GRU model for Indian sign language recognition," *Sci Rep*, vol. 12, no. 1, p. 11964, Jul. 2022, doi: 10.1038/s41598-022-15998-7.

[9] L. Fernandes, P. Dalvi, A. Junnarkar, and M. Bansode, "Convolutional Neural Network based Bidirectional Sign Language Translation System," in *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, Tirunelveli, India: IEEE, Aug. 2020, pp. 769–775. doi: 10.1109/ICSSIT48917.2020.9214272.

[10] Y. Liao, P. Xiong, W. Min, W. Min, and J. Lu, "Dynamic Sign Language Recognition Based on Video Sequence With BLSTM-3D Residual Networks," *IEEE Access*, vol. 7, pp. 38044–38054, 2019, doi: 10.1109/ACCESS.2019.2904749.

[11]N. B. Ibrahim, H. H. Zayed, and M. M. Selim, "Advances, Challenges and Opportunities in Continuous Sign Language Recognition," *J. Eng. Appl. Sci.*, vol. 15, no. 5, pp. 1205–1227, Dec. 2019, doi: 10.36478/jeasci.2020.1205.1227.

[12] O. Koller, "Quantitative Survey of the State of the Art in Sign Language Recognition," 2020, doi: 10.48550/ARXIV.2008.09918.

[13]A. Mittal, P. Kumar, P. P. Roy, R. Balasubramanian, and B. B. Chaudhuri, "A Modified LSTM Model for Continuous Sign Language Recognition Using Leap Motion," *IEEE Sensors J.*, vol. 19, no. 16, pp. 7056–7063, Aug. 2019, doi: 10.1109/JSEN.2019.2909837.

[14] A. Wadhawan and P. Kumar, "Sign Language Recognition Systems: A Decade Systematic Literature Review," *Arch Computat Methods Eng*, vol. 28, no. 3, pp. 785–813, May 2021, doi: 10.1007/s11831-019-09384-2.

[15] Z. Sun, "A Survey on Dynamic Sign Language Recognition," in *Adv Comp, Comm Comp Sci*, S. K. Bhatia, S. Tiwari, S. Ruidan, M. C. Trivedi, and K. K. Mishra, Eds., in Advances in Intelligent Systems and Computing, vol. 1158. Singapore: Springer Singapore, 2021, pp. 1015–1022. doi: 10.1007/978-981-15-4409-5_89.

[16] J. S. Raj, A. M. Iliyasu, R. Bestak, and Z. A. Baig, Eds., *Innov Dat Comm Technolog Applic: Proceedings of ICIDCA 2020*, vol. 59. in Lecture Notes on Data Engineering and Communications Technologies, vol. 59. Singapore: Springer Singapore, 2021. doi: 10.1007/978-981-15-9651-3.

[17] E. Kiran Kumar, P. V. V. Kishore, A. S. C. S. Sastry, and D. Anil Kumar, "3D Motion Capture for Indian Sign Language Recognition (SLR)," in *Sm Comp Infor*, S. C. Satapathy, V. Bhateja, and S. Das, Eds., in Smart Innovation, Systems and Technologies, vol. 78. Singapore: Springer Singapore, 2018, pp. 21–29. doi: 10.1007/978-981-10-5547-8_3.

[18] R. Itkarkar Rajeshri, A. K. V. Nandi, and V. B. Mungurwadi, "Indian Sign Language Recognition Using Combined Feature Extraction," in *Adv Med Phys Healthc Eng*, M. Mukherjee, J. K. Mandal, S. Bhattacharyya, C. Huck, and S. Biswas, Eds., in Lecture Notes in Bioengineering. Singapore: Springer Singapore, 2021, pp. 1–7. doi: 10.1007/978-981-33-6915-3_1.

[19] K. Vaibhav, "India Must Empower Millions with Hearing Disability says Deaf and Mute Entrepreneur Vaibhav Kothari," *The Times of India*, Aug. 17, 2020. https://timesofindia.indiatimes.com/india-must-empower-millions-with-hearing-disability-says-deaf-and-mute-entrepreneur-vaibhav-kothari/articleshow/77591952.cms (accessed Oct. 10, 2022).

[20] "On-device machine learning for everyone," *Mediapipe*. https://developers.google.com/mediapipe (accessed Jul. 28, 2022).

[21] C. Sanket, K. Rajeswari, and S. Vispute, "A Review on using Long-Short Term Memory for Prediction of Stock Price," *Inter J Eng Res Technol (IJERT)*, vol. 10, no. 11, pp. 251–256, Nov. 2021, doi: 10.17577/IJERTV10IS110117.

[22] V. Pathrikar, T. Podutwar, A. Siddannavar, A. Mandana, K. Rajeswari, and S. R. Vispute, "Research on Various Time Forecasting Algorithms for Predicting Covid-19 Cases," *Inter J Eng Res Technol (IJERT),* vol. 10, no. 12, pp. 451–460, Dec. 2021, doi: 10.17577/IJERTV10IS120203.

[23] V. Pathrikar, T. Podutwar, S. R. Vispute, A. Siddannavar, A. Mandana, and K. Rajeswari, "Forecasting Diurnal Covid-19 Cases for Top-5 Countries Using Various Time-series Forecasting Algorithms," in *2022 International Conference on Emerging Smart Computing and Informatics (ESCI)*, Pune, India: IEEE, Mar. 2022, pp. 1–6. doi: 10.1109/ESCI53509.2022.9758373.

[24] H. T. Rauf *et al.*, "Time series forecasting of COVID-19 transmission in Asia Pacific countries using deep neural networks," *Pers Ubiquit Comput*, vol. 27, no. 3, pp. 733–750, Jun. 2023, doi: 10.1007/s00779-020-01494-0.

[25] S. Shastri, K. Singh, S. Kumar, P. Kour, and V. Mansotra, "Time series forecasting of Covid-19 using deep learning models: India-USA comparative case study," *Ch, Sol & Fr*, vol. 140, p. 110227, Nov. 2020, doi: 10.1016/j.chaos.2020.110227.

[26] S. Chaturvedi, R. N. Titre, and N. Sondhiya, "Review of Handwritten Pattern Recognition of Digits and Special Characters Using Feed Forward Neural Network and Izhikevich Neural Model," in *2014 International Conference on Electronic Systems, Signal Processing and Computing Technologies*, Nagpur, India: IEEE, Jan. 2014, pp. 425–428. doi: 10.1109/ICESC.2014.83.

[27] S. Himavathi, A. J. Dhanaseely, and E. Srinivasan, "Performance comparison of cascade and feed forward neural network for face recognition system," in *International Conference on Software Engineering and Mobile Application Modelling and Development (ICSEMA 2012)*, Chennai, India: Institution of Engineering and Technology, 2012, pp. 21–21. doi: 10.1049/ic.2012.0154.