Emergency sign language recognition from variant of convolutional neural network (CNN) and long short term memory (LSTM) models

Muhammad Amir As'ari [a,b,1,*], Nur Anis Jasmin Sufri [b,2], Guat Si Qi [b,3]

[a] Sport Innovation and Technology Center (SITC), Institute of Human Centered Engineering (IHCE), Universiti Teknologi Malaysia, Malaysia
[b] Department of Biomedical Engineering and Health Sciences, Faculty of Electrical Engineering, Universiti Teknologi Malaysia, Malaysia
[1] amir-asari@utm.my; [2] anisjasmin24@gmail.com; [3] si.qi-1998@graduate.utm.my
* corresponding author

ARTICLE INFO

Article history
Received June 15, 2023
Revised September 5, 2023
Accepted October 18, 2023
Available online February 29, 2024

Keywords
Sign language
Long short term memory
Convolutional neural networks
ConvLSTM

ABSTRACT

Sign language is the primary communication tool used by the deaf community and people with speaking difficulties, especially during emergencies. Numerous deep learning models have been proposed to solve the sign language recognition problem. Recently. Bidirectional LSTM (BLSTM) has been proposed and used in replacement of Long Short-Term Memory (LSTM) as it may improve learning long-team dependencies as well as increase the accuracy of the model. However, there needs to be more comparison for the performance of LSTM and BLSTM in LRCN model architecture in sign language interpretation applications. Therefore, this study focused on the dense analysis of the LRCN model, including 1) training the CNN from scratch and 2) modeling with pre-trained CNN, VGG-19, and ResNet50. Other than that, the ConvLSTM model, a special variant of LSTM designed for video input, has also been modeled and compared with the LRCN in representing emergency sign language recognition. Within LRCN variants, the performance of a small CNN network was compared with pre-trained VGG-19 and ResNet50V2. A dataset of emergency Indian Sign Language with eight classes is used to train the models. The model with the best performance is the VGG-19 + LSTM model, with a testing accuracy of 96.39%. Small LRCN networks, which are 5 CNN subunits + LSTM and 4 CNN subunits + BLSTM, have 95.18% testing accuracy. This performance is on par with our best-proposed model, VGG + LSTM. By incorporating bidirectional LSTM (BLSTM) into deep learning models, the ability to understand long-term dependencies can be improved. This can enhance accuracy in reading sign language, leading to more effective communication during emergencies.

1. Introduction

According to the statistics by United Nations in 2019, there are more than 70 million deaf people using sign language. Sign language is the main method of communication for the deaf people and hearing individuals who struggle to speak. Sign language is a visual language that utilizes hand or arm gestures and non-manual signs such as facial expressions and body movements to express semantic meaning [1].

One of the most challenging task for deaf people is to contact emergency services for help. Current methods available for deaf people to make emergency calls are by sending a Short Message Service (SMS) and Video Relay Service (VRS). The drawbacks of SMS are challenging to write and read in emergency situation, a short SMS convey insufficient information [2], and network temporary breakdown lead to

failure or delay of message [3]. On the other hand, VRS enables the deaf people to communicate with the sign language interpreter [4]. The interpreter can interpret remotely with live video feeds from a video phone on a screen and audio feeds from a head set [4]. However, there are many challenges to VRS. When the interpreter and the caller are physically separated, the cues that an interpreter can typically access are less visible [2]. According to study in [5], an average of 4.1 minutes is needed before taking the call and an average of 14.4 minutes is required to decode and interpret the call. A survey conducted on 355 VRS's interpreters was made by [6]. The outcomes were most of the video relay interpreters have reported to feel burnout when working in video relay settings. As a consequence, development of an automated sign language that could interpret the emergency sign is essential to overcome the drawbacks of the existing emergency assistance service methods. Studies of sign language classification are predominantly on machine learning and deep learning approaches. Machine learning is the application of Artificial Intelligence (AI) which gives the computer the ability to learn from experience without being explicitly programmed. Deep learning is a subclass of machine learning. Deep learning models are made up of multiple layers in which each layer is fed with inputs of previous layer and performs conversion and feature extraction upon the input [7]. As stated in [8], the training processes of both learning can either be supervised, unsupervised or reinforcement. But, deep learning only required minimal knowledge and human effort for extraction of key-features [9]. In developing a deep learning model, the model is required to be trained with a significant volume of data, ideally labelled data, to learn the weight and bias. However, it may be time consuming and costly to gather enough training data [10]. In semi-supervised approach, the training data is based on a small amount of labelled data and a large amount of unlabeled data, thus reducing the need of mass-labelled data. Yet, unlabeled data, which may be hard to gather, prevails in many applications. Transfer learning, or knowledge transfer, is a technique that transfers knowledge of one domain to another similar domain with the potential to solve the mentioned issue. Thus, the deep learning model can start with the pre-trained weights and be fine-tuned for optimal performance.

Transfer learning is defined by domains and tasks. Domain illustrates the distribution of the training data. It is defined by a feature space, χ and a marginal probability distribution $P(X)$, where $X = \{x_1, \dots, x_n\} \in \chi$. $\in\chi$ refers to the space of all possible feature vectors, whereas X is a particular learning sample. Given a certain domain, a task is defined by a label space, Y and a predictive function, $f(.)$ that predicts a corresponding label based on a given domain D. The task can be learned from the training data, which consists of collections of pairs of (x_i, y_i), where $x_i \in X$ and $y \in Y$. With source domain D_S, a source learning task T_S , a target domain D_T and a target learning task T_T in mind, transfer learning refers to the learning process that improves the predictive function f(.) in the target learning task T_T based on the knowledge obtained from D_s and T_s , where $D_S \neq$ D_T, or $T_S \neq T_T$. Fig. 1 shows the representation of transfer learning by definition of domain and task.

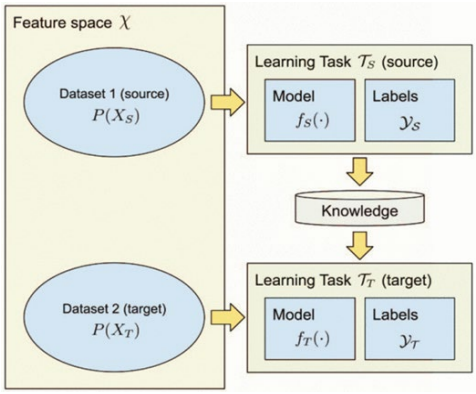

Fig. 1. Representation of transfer learning by the definition of domain and task [11]

It is necessary to be careful when utilizing transfer learning due to the possible unwanted effects. Certain features that are domain specific may have a higher frequency in one domain due to its relation to the domain topic, resulting in frequency feature bias where $P(X_s) \neq P(X_T)$. Another common bias is context feature bias, in which the same feature may have different significance or meaning in multiple

domains or context. Due to the context feature bias, the conditional distribution of a certain feature in the source and target domain may be different $P(Y_S|X_S) \neq P(Y_T|X_T)$ [12].

Furthermore, Convolutional Neural Network (CNN) is one of deep learning algorithm with fully connected networks that assigns weight to various features of the input and differentiates one from another. The study in [13] point out that CNN is optimize for image segmentation, classification, detection, and retrieval related tasks. The study in [14] listed the outstanding CNN models such as LeNet, AlexNet, VGG Net, NiN, and recent models include DenseNet, FractalNet, GoogLeNet with Inception units, and Residual Networks. Apart from that, Recurrent Neural Network (RNN) is a sequential network, which is capable for interpreting and predicting sequential information tasks such as recognition of handwriting [15] and speech [16]. Long Short-Term Memory (LSTM) is an extension of RNN, function to handle vanishing and exploding gradient problem of RCNN by creating memory cell blocks [17]. LSTM is suitable for applications with unknown long lags between important events such as language modelling, speech-to-text transcription, machine translation, and other applications [18]. Bidirectional Long Short-Term Memory (BLSTM) is constructed by having two side by side LSTM layers that first layer in forward direction and second layer in backward direction [19]. BLSTM uses both future and past data to make current prediction. Lately, BLSTM has been applied in real-time violence detection in football stadium [20], cryptocurrency price prediction [21], transportation mode recognition [22] and favorite video classification [23].

Long-term Recurrent Convolutional Neural Networks (LRCN) is one of the common CNN+LSTM models that formulate the CNN layers is used for feature extraction, whereas LSTM is used for sequence prediction. Another popular CNN+LSTM model is Convolutional LSTM (ConvLSTM). The difference between ConvLSTM and LSTM is that ConvLSTM takes in 3D data instead of 1D data as its input and produce 3D output vectors. ConvLSTM has been applied in change detection of hyperspectral images [24], video salient object detection [25], and Controllable Space-Time Video Super-Resolution [26]. Both LRCN and ConvLSTM are deeply explain in Section 2.5.

In context of development of an automated sign language, several methods have been proposed earlier. For example, work in [27] developed sign language by using Artificial Neural Network (ANN) that fed with shape, motion and colour based features while study in [28] proposed the time domain based features from surface electromyography. The implemented of CNN in recognizing the sign language were based on color image [29], depth [30] and combination of both color image and depth [31]. Since automated vision-based sign language involve video or image sequences as input, the LRCN model has been proposed by [32] for recognizing 40 daily vocabularies. However, only [33] attempted to formulate the LRCN model for emergency sign language with limited performance evaluation of the existing LRCN model. However, only a few pre-trained LRCN models i.e. GoogLeNet + LSTM and VGG-16 + LSTM were modelled in representing the emergency sign language.

Recently, Bidirectional LSTM (BLSTM) has been proposed and used in replacement of LSTM as it may improve learning long-term dependencies as well as increasing the accuracy of the model [34]. However, there is little comparison for performance of LSTM and BLSTM in LRCN model architecture in sign language interpretation application.

Therefore, the study has been focused on dense analysis of CNN+LSTM variants mainly the LRCN model including training the CNN from scratch as well as modelled with pre-trained CNN such as VGG-19 and ResNet50-V2. Other than that, ConvLSTM model, which is a special variant of LSTM designed for video input, also has been modelled and compared with the LRCN in representing the emergency sign language recognition.

2. Method

2.1. Research Flow

Fig. 2 shows research implementation methodology that is organized as follows: (1) Data Acquisition, (2) Dataset Pre-processing, (3) Dataset Splitting into Training and Test Sets, (4) Deep learning model

architecture implementation, and (5) Performance evaluation. The project is implemented in Python. The details of the five steps will be explained in the following subsections.

Fig. 2. Research workflow

2.2. Research workflow

The dataset to be used is a video dataset of the hand gestures of Indian sign language (ISL) words used in emergencies provided by [33]. The video dataset contains eight ISL words commonly used in emergency situation: 'accident', 'call', 'doctor', 'help', 'hot', 'lose', 'pain', and 'thief'. The video dataset consists of two samples of the signing of each ISL word by 26 people (12 males and 14 females) in the age group of 22 up to 26 years old in an indoor environment with normal lighting conditions. Fig. 3 shows the key frame sequences of the hand gestures of the ISL words in the 'Cropped_Data' dataset

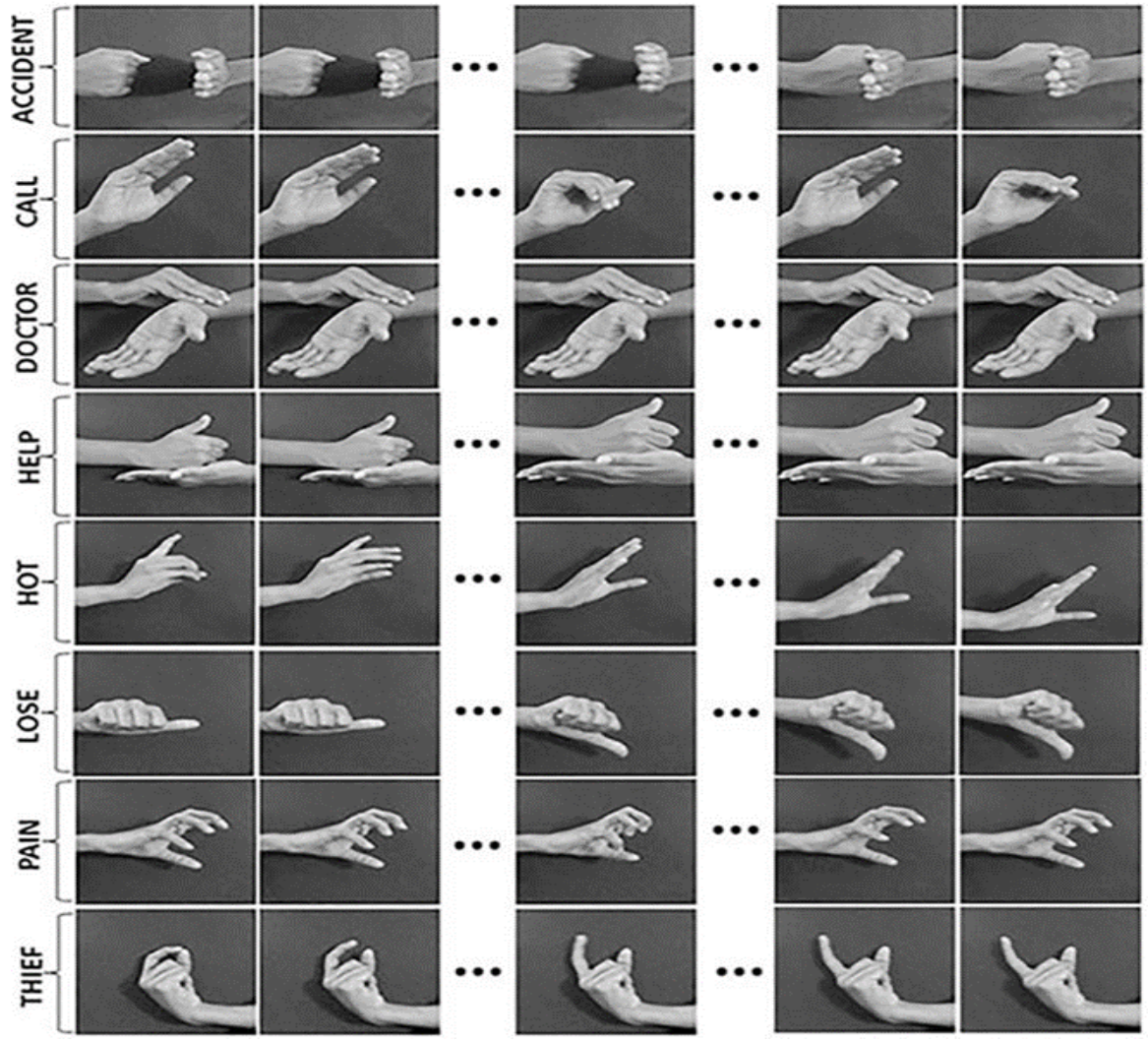

Fig. 3. The keyframe sequences of the hand gestures of the ISL words included in the `Cropped_Data' set [33]

Two folders are provided: one which contains the raw data of the video, and the other contains cropped data in which surroundings of the hand gesture are mostly cropped out. In this project, the cropped data is used to reduce the amount of data pre-processing. The comparison of cropped data with raw data of the videos is shown in Fig. 4.

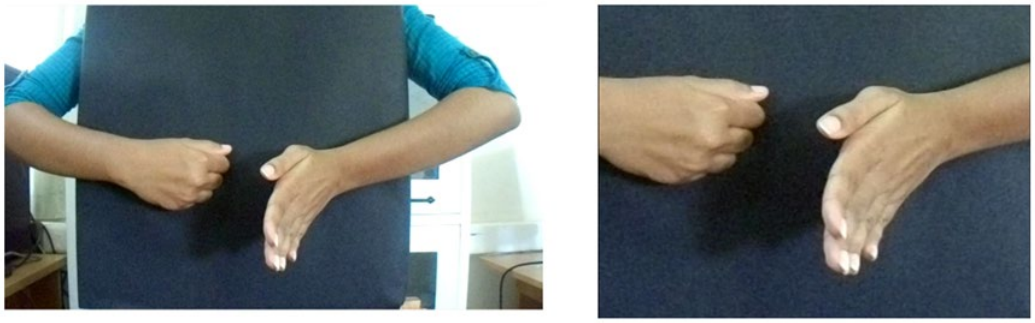

Fig. 4. Comparison of cropped data with raw data of the videos

2.3. Dataset Preprocessing

Before deep learning model implementation, the data requires further pre-processing with the help of OpenCV library with python. A list of labels for the data is first obtained: Accident, Call, Doctor, Help, Hot, Lose, Pain, and Thief. Twenty video frames were extracted from each video and re-sized to scale factor of 64 for both image width and height. The video frames were then normalized to range of 0 - 255 pixels. A dataset of the Indian Sign Language was created with each data (video) having its own feature (video frames), label, and video file path in the Google Drive.

The labels were converted to one-hot encoded labels using Keras library in which they are represented as binary vectors as shown below. This is to avoid possible bias of the deep learning model when simple indexing (0 - 7) of the categorical label is used. Fig. 5 illustrates the process of one-hot encoding of categorical labels of the classes.

Index	Sign Language
0	Accident
1	Call
2	Doctor
3	Help
4	Hot
5	Lose
6	Pain
7	Thief

One-hot Encoding

Index	Accident	Call	Doctor	Help	Hot	Lose	Pain	Thief
0	1	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0
3	0	0	0	1	0	0	0	0
4	0	0	0	0	1	0	0	0
5	0	0	0	0	0	1	0	0
6	0	0	0	0	0	0	1	0
7	0	0	0	0	0	0	0	1

Fig. 5. One-hot encoding of categorical labels

2.4. Dataset Splitting

The data in the dataset with their respective features and one-hot encoded labels were shuffled to avoid potential bias and split into two parts: 80% for training set and 20% for test set. The training set was used to fit the parameters of the model whereas the test set was used to evaluate the performance of the deep learning model.

2.5. Deep Learning Model Architecture Implementation

In this study, TensorFlow and Keras libraries with python were utilized to build our deep learning models, named as LRCN and ConvLSTM models. LRCN and ConvLSTM models were chosen for classifying sign language video dataset due to their ability of processing both spatial and temporal features.

The general flow of processing in the LRCN architecture is shown in the Fig. 6. Two models: CNN and LSTM models were built separately. Video frames of the training set were fed to the CNN model for training on the spatial features. The output of the CNN model was inputted into the LSTM model for training on temporal sequence modelling and predicting the sign language. LSTM layer was also swapped with BLSTM layer to compare their performance.

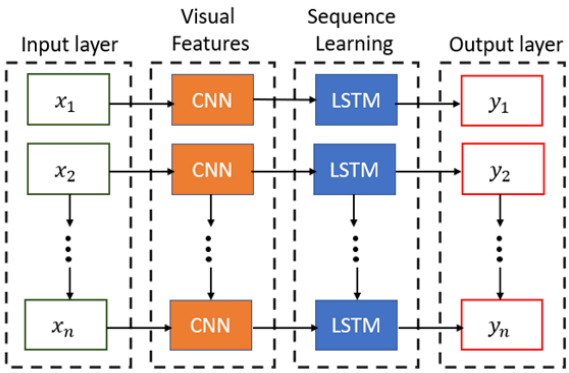

Fig. 6. LRCN general architecture

LRCN detailed architecture is made up of two parts: feature extraction layers and classification layers. Conv2D, MaxPooling3D and DropOut layers are considered as one 'subunit' of CNN. In this study, 3, 4, and 5 subunits of CNN in LRCN architecture were modelled. Also, LSTM layer was also swapped with BLSTM layer for performance comparison. Additionally, transfer learning of LRCN model was established by swapping the feature extraction layers with VGG-19 and ResNet50-V2 respectively while trying out combination of them with LSTM or BLSTM. The example of LRCN with 4 subunits can be shown in Fig. 7.

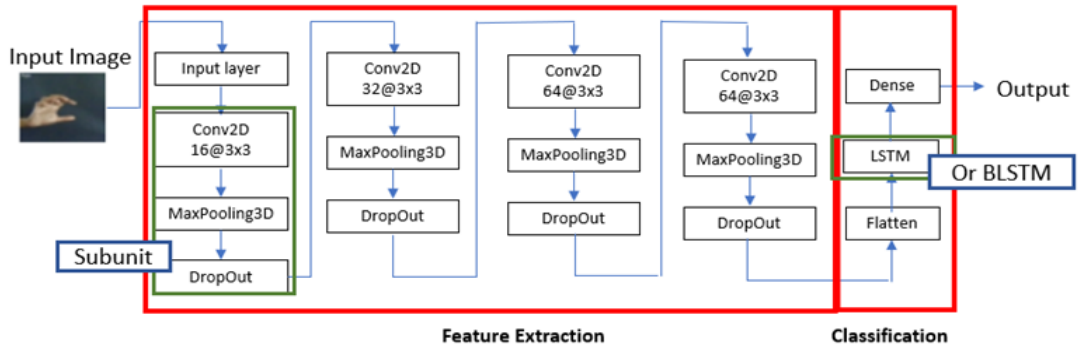

Fig. 7. Detailed architecture of LRCN with 4 subunits

ConvLSTM detailed architecture consists of feature extraction layers and classification layers. For simplicity, ConvLSTM, MaxPooling3D and DropOut layers are considered as one 'subunit' of ConvLSTM. In this study, we have tested 3, 4, and 5 subunits of ConvLSTM. The example of ConvLSTM with 4 subunits can be referred in Fig. 8.

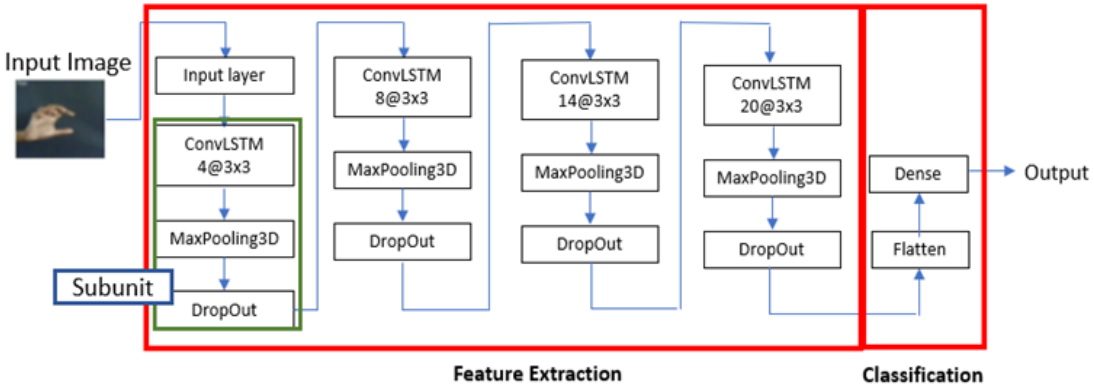

Fig. 8. Detailed architecture of ConvLSTM with 4 subunits

The list of the deep learning models deployed in this study is listed in Table 1.

Table 1. Deep Learning Models

Approach	ConvLSTM	LRCN
Models	ConvLSTM (3)	LRCN (3 CNN + LSTM)
	ConvLSTM (4)	LRCN (3 CNN + BLSTM)
	ConvLSTM (5)	LRCN (4 CNN + LSTM)
		LRCN (4 CNN + BLSTM)
		LRCN (5 CNN + LSTM)
		LRCN (5 CNN + BLSTM)
		VGG-16 + LSTM
		VGG-16 + BLSTM
		ResNet50-V2 + LSTM
		ResNet50-V2 + BLSTM

The settings of deep learning model training for both LRCN and ConvLSTM is shown in Table 2. The loss function categorical cross-entropy was used to calculate the loss of the model with multi-class classification problem. The optimizer Adam was used for adaptive learning rate as it has better convergence in the training process graphs for our application compared to other optimizers. The metric accuracy was used to judge the performance of the model. Shuffling of data was applied to prevent potential bias of the architecture. 20% of the training set was set aside as validation set for evaluation and tuning the model hyperparameter, whereas the remaining 80% of the training set was used for training. Validation loss was monitored during the training such that when for 20 epochs the decrement of validation loss has stopped. Early stopping callback is applied and the model training was stopped before reaching max epoch of 150. The model weight based on best value of validation loss was restored.

Table 2. Setting for both ConvLSTM and LRCN

Setting	Details
Loss	Categorical cross-entropy loss function
Optimizer	Adam
Metrics	Accuracy
Input dimension	4
Output dimension	1
Max epoch	150
Batch size	25
Shuffle	True
Validation split	0.2
Callbacks	Earlystopping
Monitor	Validation loss
Patience	20
Mode	Min
Restore_best_weight	True

2.6. Performance Evaluation

The performance of the deep learning models was evaluated using confusion matrix, accuracy, categorical cross-entropy loss, recall, precision and F1-Score. The confusion matrix is a M × M matrix, where M is the number of classes. The confusion matrix shows four possible outputs: True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). Metrics for evaluating model performance such as accuracy, recall, precision and F1-score [35] are calculated using equation (1) until equation (4) respectively.

$$Accuracy \ \frac{TP + TN}{TP + TN + FN + FP} \quad (1)$$

$$Recall \ \frac{TP}{TP + FN} \quad (2)$$

$$Precission \ \frac{TP}{TP + FP} \quad (3)$$

$$F1 \ Score \ \frac{2 \times Recall \times Precision}{Recall + Precision} \quad (4)$$

3. Results and Discussion

3.1. Training Progress Results

From the training progress, two training progress graphs were generated for each deep learning model trained, in which one compares the training accuracy with the validation accuracy, while the other compares the training loss and the validation loss. The rough estimate of the number of epochs ran may be obtained from the graphs. In this study, the max number of epochs is set as 150 where epoch is defined as the number of times the deep learning model transit through the entire training dataset [36]. However, if the conditions of early stopping callback, namely 20 continuous epochs with little validation loss difference, is fulfilled, the model may terminate its training progress earlier. The batch size was set as 25, meaning 25 samples are processed before the next model parameters update [36]. For each loss graph, y-axis represents the loss while x-axis represents the number of epochs. For accuracy graph, y-axis denotes the accuracy of the model while x-axis shows the number of epochs. The blue line denotes the results from the training progress, whereas the red line denotes the results from the validation process. Fig. 9 shows the loss graph (left) and accuracy graph (right) of ConvLSTM with 3 subunits. The deep learning model took 87 min 37 sec to complete 53 training epochs with early stopping callback. The deep learning model has achieved final training accuracy of 100%, final validation accuracy of 96.97%, final loss of 0.04%, and final validation loss of 7.89%.

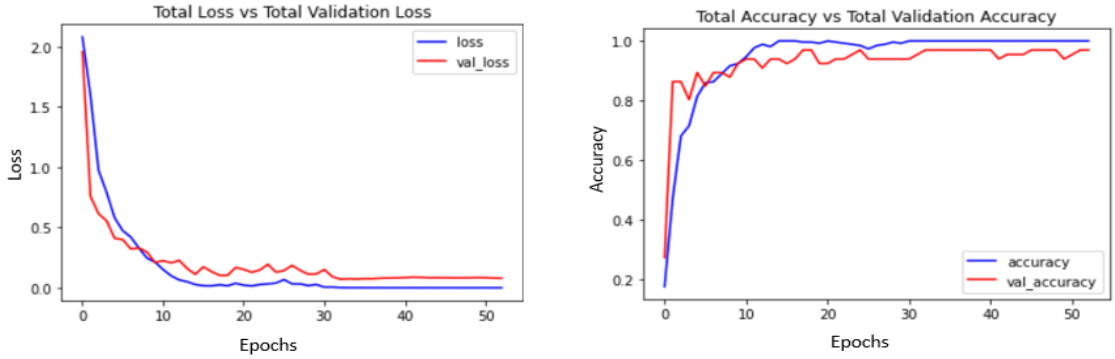

Fig. 9. Loss and Accuracy graphs of ConvLSTM (3 subunits) model

Fig. 10 presents the loss graph (left) and accuracy graph (right) of CNN-LSTM with 3 subunits of CNN. The training process took 15 min 12 sec to run 81 training epochs with early stopping callback. The deep learning model has achieved final training accuracy of 100%, final validation accuracy of 96.97%, final loss of 1.26%, and final validation loss of 13.14%.

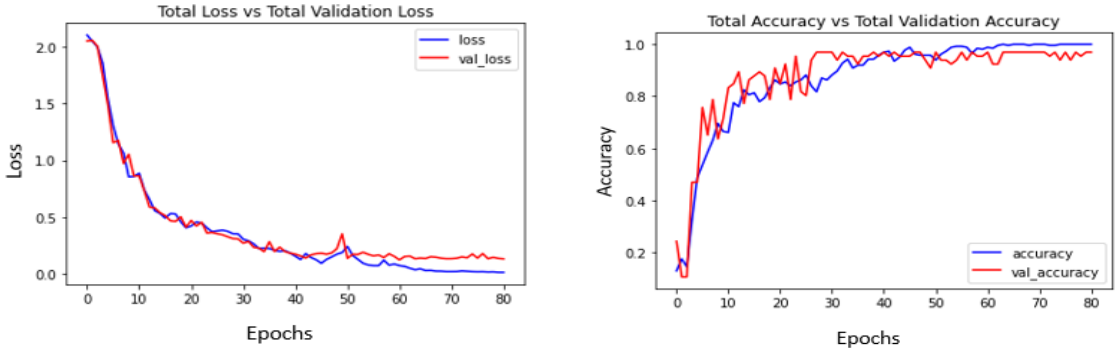

Fig. 10. Loss and Accuracy graphs of CNN (3 subunits) + LSTM model

For CNN + BLSTM with 3 subunits of CNN, the loss graph (left) and accuracy graph(right) are shown in Fig. 11. The model has completed 80 training epochs in 14 min 54 sec with early stopping callback. With respect to the training performance of the model, final training accuracy of 99.62%, final validation accuracy of 98.48%, final loss of 3.00%, and final validation loss of 9.40% are achieved.

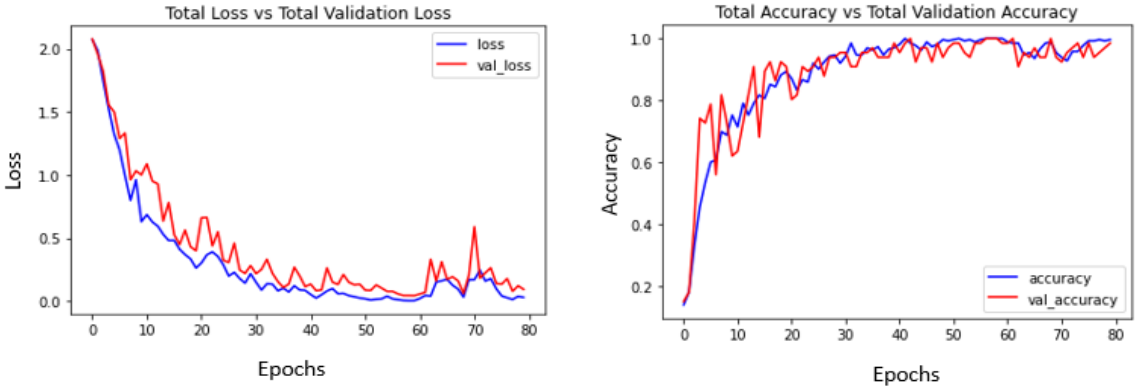

Fig. 11. Loss and Accuracy graphs of CNN (3 subunits) + BLSTM model

The training performance graphs (left: loss; right: accuracy) for the model VGG-19 + LSTM are shown in Fig. 12. The period of training progress of the model is 143 min 4 sec. It took 25 training epochs with early stopping callback for the model to complete its training. From the training progress of the model, we obtain a final training accuracy of 100.00%, final validation accuracy of 95.45%, final loss of 0.01%, and final validation loss of 25.68%.

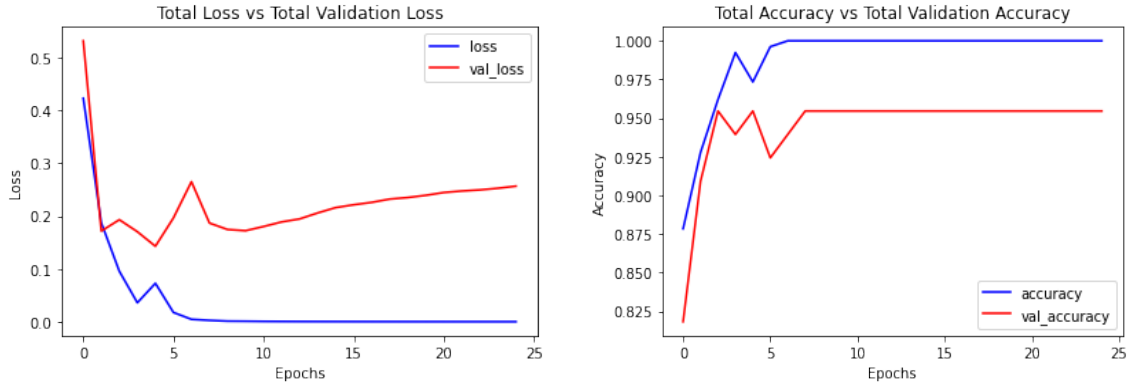

Fig. 12. Loss and Accuracy graphs of VGG-19 + LSTM model

Table 3 summarizes the accuracy and loss of all the deep learning models used. Overall, all models achieved training and validation accuracy above 95% with LRCN (4 CNN subunits with BLSTM) achieving the highest validation accuracy, which is 99.62%. This might be due to the advantages of BLSTM using both future and past data to make current prediction.

Table 3. Accuracy and Loss of Deep Learning Models

Model	Final Training Accuracy	Final Validation Accuracy	Final Loss	Final Validation Loss
ConvLSTM (3)	1.0000	0.9697	0.0004	0.0789
ConvLSTM (4)	1.0000	0.9242	0.0044	0.2183
ConvLSTM (5)	0.9848	0.9545	0.0380	0.1552
LRCN (3 CNN + LSTM)	1.0000	0.9697	0.0126	0.1314
LRCN (3 CNN + BLSTM)	0.9962	0.9848	0.0300	0.0940
LRCN (4 CNN + LSTM)	1.0000	0.9545	0.0080	0.1301
LRCN (4 CNN + BLSTM)	0.9962	0.9962	0.0184	0.1103
LRCN (5 CNN + LSTM)	0.9962	0.9848	0.0305	0.0921
LRCN (5 CNN + BLSTM)	1.0000	0.9697	0.0002	0.0678
VGG-19 + LSTM	1.0000	0.9545	0.0001	0.2568
VGG-19 + BLSTM	1.0000	0.9848	0.0001	0.0812
ResNet50V2 + LSTM	1.0000	0.9394	0.0001	0.1804
ResNet50V2 + BLSTM	1.0000	0.9697	0.0000	0.1532

3.2. Performance Evaluation of Trained Model

After the testing stage, the performance of the deep learning models was evaluated in confusion metrics as well as metrics in testing accuracy, testing loss, precision, recall, and F1-score. Table 4 summarizes the testing accuracy, testing loss, precision, recall, and F1-score obtained by the trained models.

Table 4. Performance evaluation of trained model

Models	Testing Accuracy	Testing Loss	Precision	Recall	F1-score
ConvLSTM (3)	0.8434	0.6747	0.8434	0.8434	0.8434
ConvLSTM (4)	0.9036	0.6729	0.9036	0.9036	0.9036
ConvLSTM (5)	0.8813	0.7252	0.8313	0.8313	0.8313
LRCN (3 CNN + LSTM)	0.9157	0.3315	0.9157	0.9157	0.9157
LRCN (3 CNN + BLSTM)	0.9398	0.4492	0.9398	0.9398	0.9398
LRCN (4 CNN + LSTM)	0.9398	0.3053	0.9398	0.9398	0.9398
LRCN (4 CNN + BLSTM)	0.9518	0.5855	0.9518	0.9518	0.9518
LRCN (5 CNN + LSTM)	0.9518	0.2552	0.9518	0.9518	0.9518
LRCN (5 CNN + BLSTM)	0.9277	0.2950	0.9277	0.9277	0.9277
VGG-19 + LSTM	0.9639	0.1263	0.9639	0.9639	0.9639
VGG-19 + BLSTM	0.9518	0.3413	0.9518	0.9518	0.9518
ResNet50-V2 + LSTM	0.9277	0.3630	0.9277	0.9277	0.9277
ResNet50-V2 + BLSTM	0.8795	0.7775	0.8795	0.8795	0.8795

In general, all LRCN models performs significantly better in all performance metrics than the ConvLSTM model with the exception of ResNet50V2 + BLSTM model which has a testing accuracy, precision, recall, and F1-score of 0.8795. Thus, this implies that LRCN model is more suitable for sign language recognition application with video input than ConvLSTM. Although there is an increase of testing accuracy for LRCN 3 CNN + BLSTM and 4 CNN + BLSTM compared to the LSTM counterparts, a decrease of testing accuracy, precision, recall, and F1-score are observed for other LRCN models with BLSTM compared to the LSTM counterparts. Therefore, it is inconclusive whether BLSTM would increase the testing accuracy of LRCN model. Also, there is a significant increase in testing loss for LRCN with LSTM model compared to LRCN with BLSTM model. Further tuning of the model may be required to adjust the testing loss off LRCN with BLSTM model. Out of all models, VGG-19 + LSTM model has outperformed all testing model with testing accuracy of 96.39%, testing loss of 12.63%, precision of 96.39%, recall of 96.39%, and F1-score of 96.39%.

Table 5 illustrates the comparison of the testing accuracy of trained models with prior models using the same dataset. ConvLSTM models has lower testing accuracy than multi-class SVM model, thus it can be concluded that ConvLSTM may not be suitable for sign language interpretation with video-based input. Out of all trained models, the performance of proposed LRCN with VGG-19 and LSTM model is on par with the existing study which manage to exceed the LRCN with pre-trained GoogLeNet + LSTM model in testing accuracy, even though the proposed model is still lower than the model with pre-trained VGG-16 + LSTM.

This suggests that VGG-16 may be more suitable for sign language interpretation. Moreover, the proposed LRCN model especially the developed model from scratch such as LRCN with 3 CNN + BLSTM, 4 CNN + LSTM, and 4 CNN + BLSTM are on par with the existing pre-trained GoogLeNet + LSTM, even though these proposed models are smaller (in size).

Table 5. Trained model compared to previous works on the same dataset

Author	Method	Testing Accuracy (%)
Adithya and Rajesh [3]	Multi-Class SVM	90
	Pre-trained GoogleNet + LSTM	96
Areeb and Nadeem [4]	Pre-trained VGG-16 + LSTM	98
Tested models	ConvLSTM (3)	0.8434
	ConvLSTM (4)	0.9036
	ConvLSTM (5)	0.8813
	LRCN (3 CNN + LSTM)	0.9157
	LRCN (3 CNN + BLSTM)	0.9398
	LRCN (4 CNN + LSTM)	0.9398
	LRCN (4 CNN + BLSTM)	0.9518
	LRCN (5 CNN + LSTM)	0.9518
	LRCN (5 CNN + BLSTM)	0.9277
	VGG-19 + LSTM	0.9639
	VGG-19 + BLSTM	0.9518
	ResNet50-V2 + LSTM	0.9277
	ResNet50-V2 + BLSTM	0.8795

Fig. 13 shows the confusion matrix of the best performing model in this study: VGG-19 + LSTM. A misclassification rate of 3.61% is obtained. Six models are correctly classified by the model, which are accident, doctor, lose, thief, help, and pain.

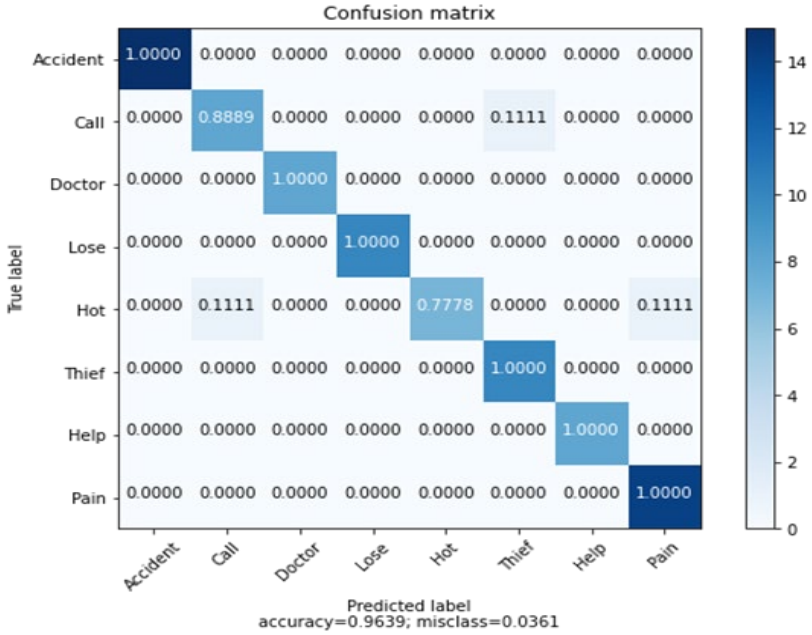

Fig. 13. Confusion matrix of VGG-19 + LSTM model on testing dataset

3.3. Demonstration of Proposed Emergency Sign Language Recognition based on LRCN (VGG-19 + LSTM)

Fig. 14 shows the example of tested video using the trained model (LRCN with VGG-19 and LSTM) consisting of 8 samples from the test dataset. It can be seen from this figure that the model managed to predict the sample successfully based on actual label.

Fig. 14. Classification result of VGG-19 LSTM model with labels

4. Conclusion

The primary aim of the project is to develop deep learning model for classifying emergency sign language in order to assist sign language user in making emergency calls. While SMS and VRS have been adopted in a number of countries, mostly EU and US, some countries have yet to adopt VRS, including Malaysia. SMS and VRS presents a number of complications in making emergency call, with the most notable one as length call process. Therefore, a deep learning model based on CNN-LSTM has been developed to classify various emergency signs in sign language. CNN-BLSTM and ConvLSTM model architectures are developed as well to compare their performance with CNN-LSTM. Transfer learning is implemented in this study, namely VGG-19 and ResNetV50, to compare the performance of pre-trained networks with the developed neural networks. LRCN approach is foreseen to have better performance than ConvLSTM approach, and we seek to investigate the performance difference between LSTM and BLSTM in LRCN model architecture. The outcome of the study has shown that LRCN approach has a generally better performance than ConvLSTM model. The comparison between LSTM and BLSTM in LRCN model architecture remain inconclusive. From the study, the best performance model is VGG-19 + LSTM model with the best testing metrics among all trained models. The testing accuracy, recall, precision, and F1-score of VGG-19 + LSTM model are 96.39%, 12.63%, 96.39%, and 96.39% respectively. In conclusion, the study has successfully met the objectives of developing CNN-LSTM model for emergency sign language classification, evaluating performance of developed models, as well as comparing several model architectures (CNN-LSTM, CNN-BLSTM, ConvLSTM).

Acknowledgment

The authors would like to express their appreciation to Universiti Teknologi Malaysia (UTM) for endowing this research and the Ministry of Higher Education (MOHE) Malaysia for supporting this research work under Fundamental Research Grant Scheme (FRGS/1/2023/ICT02/UTM/02/1).

Declarations

Author contribution. All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.
Funding statement. This research was funded by Ministry of Higher Education (MOHE) Malaysia for supporting this research work under Fundamental Research Grant Scheme (FRGS/1/2023/ICT02/UTM/02/1).
Conflict of interest. The authors declare no conflict of interest.
Additional information. No additional information is available for this paper.

References

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

[2] C. Warnicke, "Equal Access to Make Emergency Calls: A Case for Equal Rights for Deaf Citizens in Norway and Sweden," *Soc. Incl.*, vol. 7, no. 1, pp. 173–179, Jan. 2019, doi: 10.17645/si.v7i1.1594.

[3] Y. Wang, J. Li, X. Zhao, G. Feng, and X. Luo, "Using Mobile Phone Data for Emergency Management: a Systematic Literature Review," *Inf. Syst. Front.*, vol. 22, no. 6, pp. 1539–1559, Dec. 2020, doi: 10.1007/s10796-020-10057-w.

[4] C. Warnicke and C. Plejert, "The headset as an interactional resource in a video relay interpreting (VRI) setting," *Interpret. Int. J. Res. Pract. Interpret.*, vol. 20, no. 2, pp. 285–308, Sep. 2018, doi: 10.1075/intp.00013.war.

[5] J. Napier, R. Skinner, and G. Turner, "'It's good for them but not so for me': Inside the sign language interpreting call centre," *Int. J. Transl. Interpret. Res.*, vol. 9, no. 2, pp. 1–23, Jul. 2017, doi: 10.12807/ti.109202.2017.a01.

[6] S. Chang and D. Russell, "Coming Apart at the Screens: Canadian Video Relay Interpreters and Stress," *Journal of Interpretation.*, vol. 3, no. 9, pp. 19, Nov. 29, 2022. Online: Available at: https://digitalcommons.unf.edu/joi/vol30/iss1/6/.

[7] S. Dargan, M. Kumar, M. R. Ayyagari, and G. Kumar, "A Survey of Deep Learning and Its Applications: A New Paradigm to Machine Learning," *Arch. Comput. Methods Eng.*, vol. 27, no. 4, pp. 1071–1092, Sep. 2020, doi: 10.1007/s11831-019-09344-w.

[8] S. Malik, A. K. Tyagi, and S. Mahajan, "Architecture, Generative Model, and Deep Reinforcement Learning for IoT Applications: Deep Learning Perspective," in *Internet of Things*, Springer Science and Business Media Deutschland GmbH, 2022, pp. 243–265, doi: 10.1007/978-3-030-87059-1_9.

[9] C. Janiesch, P. Zschech, and K. Heinrich, "Machine learning and deep learning," *Electron. Mark.*, vol. 31, no. 3, pp. 685–695, Sep. 2021, doi: 10.1007/s12525-021-00475-2.

[10] F. Zhuang *et al.*, "A Comprehensive Survey on Transfer Learning," *Proc. IEEE*, vol. 109, no. 1, pp. 43–76, Jan. 2021, doi: 10.1109/JPROC.2020.3004555.

[11] R. Ribani and M. Marengoni, "A Survey of Transfer Learning for Convolutional Neural Networks," in *2019 32nd SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T)*, Oct. 2019, pp. 47–57, doi: 10.1109/SIBGRAPI-T.2019.00010.

[12] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *J. Big Data*, vol. 3, no. 1, p. 9, Dec. 2016, doi: 10.1186/s40537-016-0043-6.

[13] L. Alzubaidi *et al.*, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," *J. Big Data*, vol. 8, no. 1, p. 53, Mar. 2021, doi: 10.1186/s40537-021-00444-8.

[14] M. Z. Alom *et al.*, "A State-of-the-Art Survey on Deep Learning Theory and Architectures," *Electronics*, vol. 8, no. 3, p. 292, Mar. 2019, doi: 10.3390/electronics8030292.

[15] R. Ghosh, C. Vamshi, and P. Kumar, "RNN based online handwritten word recognition in Devanagari and Bengali scripts using horizontal zoning," *Pattern Recognit.*, vol. 92, pp. 203–218, Aug. 2019, doi: 10.1016/j.patcog.2019.03.030.

[16] G. Saon, Z. Tuske, D. Bolanos, and B. Kingsbury, "Advancing RNN Transducer Technology for Speech Recognition," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Jun. 2021, vol. 2021-June, pp. 5654–5658, doi: 10.1109/ICASSP39728.2021.9414716.

[17] Y. Yu, X. Si, C. Hu, and J. Zhang, "A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures," *Neural Comput.*, vol. 31, no. 7, pp. 1235–1270, Jul. 2019, doi: 10.1162/neco_a_01199.

[18] A. Sherstinsky, "Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network," *Phys. D Nonlinear Phenom.*, vol. 404, p. 132306, Mar. 2020, doi: 10.1016/j.physd.2019.132306.

[19] Z. C. Lipton, J. Berkowitz, and C. Elkan, "A Critical Review of Recurrent Neural Networks for Sequence Learning," *arxiv*, p. 38, May 29, 2015. [Online]. Available at: https://arxiv.org/abs/1506.00019v4.

[20] D. J. Samuel R. *et al.*, "Real time violence detection framework for football stadium comprising of big data analysis and deep learning through bidirectional LSTM," *Comput. Networks*, vol. 151, pp. 191–200, Mar. 2019, doi: 10.1016/j.comnet.2019.01.028.

[21] M. J. Hamayel and A. Y. Owda, "A Novel Cryptocurrency Price Prediction Model Using GRU, LSTM and bi-LSTM Machine Learning Algorithms," *AI*, vol. 2, no. 4, pp. 477–496, Oct. 2021, doi: 10.3390/ai2040030.

[22] A. Das Antar, M. Ahmed, and M. A. R. Ahad, "Recognition of human locomotion on various transportations fusing smartphone sensors," *Pattern Recognit. Lett.*, vol. 148, pp. 146–153, Aug. 2021, doi: 10.1016/j.patrec.2021.04.015.

[23] A. K. Agirman and K. Tasdemir, "BLSTM based night-time wildfire detection from video," *PLoS One*, vol. 17, no. 6, p. e0269161, Jun. 2022, doi: 10.1371/journal.pone.0269161.

[24] W.-S. Hu, H.-C. Li, L. Pan, W. Li, R. Tao, and Q. Du, "Spatial–Spectral Feature Extraction via Deep ConvLSTM Neural Networks for Hyperspectral Image Classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 6, pp. 4237–4250, Jun. 2020, doi: 10.1109/TGRS.2019.2961947.

[25] H. Huang, C. Liu, L. Tian, J. Mu, and X. Jing, "A novel FCNs-ConvLSTM network for video salient object detection," *Int. J. Circuit Theory Appl.*, vol. 49, no. 4, pp. 1050–1060, Apr. 2021, doi: 10.1002/cta.2924.

[26] G. Xu, J. Xu, Z. Li, L. Wang, X. Sun, and M.-M. Cheng, "Temporal Modulation Network for Controllable Space-Time Video Super-Resolution," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2021, pp. 6384–6393, doi: 10.1109/CVPR46437.2021.00632.

[27] N. K. Tamiru, M. Tekeba, and A. O. Salau, "Recognition of Amharic sign language with Amharic alphabet signs using ANN and SVM," *Vis. Comput.*, vol. 38, no. 5, pp. 1703–1718, May 2022, doi: 10.1007/s00371-021-02099-1.

[28] Zhang, Yang, Qian, and Zhang, "Real-Time Surface EMG Pattern Recognition for Hand Gestures Based on an Artificial Neural Network," *Sensors*, vol. 19, no. 14, p. 3170, Jul. 2019, doi: 10.3390/s19143170.

[29] A. A. Barbhuiya, R. K. Karsh, and R. Jain, "CNN based feature extraction and classification for sign language," *Multimed. Tools Appl.*, vol. 80, no. 2, pp. 3051–3069, Jan. 2021, doi: 10.1007/s11042-020-09829-y.

[30] W. Aly, S. Aly, and S. Almotairi, "User-Independent American Sign Language Alphabet Recognition Based on Depth Image and PCANet Features," *IEEE Access*, vol. 7, pp. 123138–123150, 2019, doi: 10.1109/ACCESS.2019.2938829.

[31] Q. Gao, U. E. Ogenyi, J. Liu, Z. Ju, and H. Liu, "A Two-Stream CNN Framework for American Sign Language Recognition Based on Multimodal Data Fusion," in *Advances in Intelligent Systems and Computing*, vol. 1043, Springer Verlag, 2020, pp. 107–118, doi: 10.1007/978-3-030-29933-0_9.

[32] S. Yang and Q. Zhu, "Continuous Chinese sign language recognition with CNN-LSTM," in *https://doi.org/10.1117/12.2281671*, Jul. 2017, vol. 10420, p. 104200F, doi: 10.1117/12.2281671.

[33] V. Adithya and R. Rajesh, "Hand gestures for emergency situations: A video dataset based on words from Indian sign language," *Data Br.*, vol. 31, p. 106016, Aug. 2020, doi: 10.1016/j.dib.2020.106016.

[34] M. Jia, J. Huang, L. Pang, and Q. Zhao, "Analysis and Research on Stock Price of LSTM and Bidirectional LSTM Neural Network," in *Proceedings of the 3rd International Conference on Computer Engineering,*

Information Science & Application Technology (ICCIA 2019), Jul. 2019, pp. 467–473, doi: 10.2991/iccia-19.2019.72.

[35] A. Tharwat, "Classification assessment methods," *Appl. Comput. Informatics*, vol. 17, no. 1, pp. 168–192, Jan. 2021, doi: 10.1016/j.aci.2018.08.003.

[36] J. Brownlee, D. Learning, D. Between, N. Network, and G. Cook, "What is the Difference Between a Batch and an Epoch in a Neural Network ?," pp. 1-5, 2018. [Online]. Available at: https://deeplearning.lipingyang.org/wp-content/uploads/2018/07/What-is-the-Difference-Between-a-Batch-and-an-Epoch-in-a-Neural-Network_.pdf.