

Deep Learning Model for Vision-Based Dynamic Hand Gesture Recognition

Aisha Sakinah Saadon, Noramiza Hashim, Wan Noorshahida Mohd Isa

Faculty of Computing and Informatics (FCI), Multimedia University (MMU), Persiaran Multimedia,
63100 Cyberjaya, Selangor, Malaysia

noramiza.hashim@mmu.edu.my, wan.noorshahida.isa@mmu.edu.my

Abstract. Communication is essential in our daily life; however, senior citizens may experience communication difficulties due to declining abilities, making it difficult for them to request assistance when needed. The aim of this project is to develop a vision-based hand gesture recognition system to assist senior citizens, where the system will recognise a variety of gestures and translate them into their corresponding meanings for each gesture. A collection of hand gesture videos consisting of 14 gesture classes is used to build the models representing daily tasks. This solution is particularly valuable for bedridden senior citizens who face difficulty communicating with their caretaker. In this study, we compared the performance of three deep learning models: MobileNet + BiGRU, LSTM, and Transformer + DenseNet. Our findings revealed that LSTM outperformed the other two models, achieving a commendable accuracy rate of 94.33%. In comparison, MobileNet + BiGRU achieved an accuracy of 92.20%, and Transformer + DenseNet achieved an accuracy of 87.23%.

Keywords: Dynamic Hand Gesture Recognition, Computer Vision System, Deep Learning, Convolutional Neural Network, Elderly care

1. Introduction

While communication plays a crucial role for carrying out our daily activities, certain individuals face considerable difficulties in accomplishing this vital task. Seniors, who are typically aged 60 and above, often encounter obstacles in communication due to declining sensory, physical, and cognitive abilities. These challenges can be the result of age-related changes or specific verbal communication impairments. For elderly individuals to receive necessary assistance from caregivers with essential daily activities like eating, drinking, and medication management, it is crucial that they can communicate their needs effectively.

Static hand gestures refer to hand postures or positions that remain fixed or stationary for a certain period of time (Adithya & Rajesh, 2020). These gestures involve shaping the hands and fingers into specific configurations to convey a particular meaning or message. Examples of static hand gestures include holding up fingers to indicate a numerical value, making a "thumbs up" or "thumbs down" sign, or forming a "peace" sign with the fingers. On the other hand, dynamic hand gestures involve movement and changes in hand position or shape (Yu J. et al., 2022). These gestures are used to emphasise or enhance verbal communication and often accompany speech or other forms of expression. Dynamic hand gestures can involve gestures such as waving, pointing, sweeping motions, or gestures that depict the movement of an object or action. While static hand gestures are relatively fixed and convey a specific meaning on their own, dynamic hand gestures are more fluid and visually engaging. Therefore, this paper employs dynamic hand gestures for communication among the elderly.

Technological advancements have the potential to assist elderly individuals in communication, however further research is necessary to assess and enhance the system's performance. Gesture recognition technology leverages hand movement trajectories to identify both spatial and temporal movements. The computer analyses the detected hand gestures and translates them into messages. This project aims to implement this hand gesture recognition system that can accurately identify various types of gestures. We will compare the performance of several deep learning algorithms and find the best model among them.

2. Literature Review

Hand gesture recognition allows people to interact with complex machines using hand gestures from a distance without physically touching them. The system focuses on user comfort, efficiency, and cost reduction. The field is rapidly growing and commonly implemented in the field of elderly care as a means to enhance communication and interaction between elderly individuals and assistive technologies or caregiving systems. A recent study by Marzuki & Hashim (2022) focuses on the field of elderly care, specifically within the Muslim community. The authors aim to develop a vision-based hand gesture recognition system that can assist in providing care and support for the Muslim elderly. The paper experimented on deep learning algorithms that can accurately recognize and interpret dynamic hand gestures. It utilises a dataset consisting of ten gesture classes with 630 videos.

A paper by Oudah et al. (2020) showcased that the application of a hand gesture recognition system can assist elderly individuals with their daily activities. The study focuses on utilising depth data to recognize five different hand gestures, each associated with a specific request. By using the Kinect V2 depth sensor and implementing a CNN recognition model, the system was able to accurately track individual hand movements and distinguish different gestures effectively.

Vision-based hand gesture recognition uses cameras to capture hand gestures, which are then processed using algorithms to obtain movement and gesture information. These image processing algorithms extract important features such as hand shapes, orientations, contours, colour, and motion features (Singh et al., 2019). After performing feature extraction, the hand gestures are then categorised into different classes. A paper by Fahmid et al. (2022) presents a comprehensive review and analysis of recent advancements in vision-based gesture recognition methods. The performance outcomes of

vision-based gesture recognition have served as inspiration for advanced gesture-based technologies and applications development. These applications encompass various domains, including sign language recognition (Damaneh et al., 2023), hand gesture recognition for visually impaired individuals (Alashhab et al., 2022), gaze-aware hand gesture recognition for intelligent construction (X. Wang et al., 2023), and controlled robotic hand systems using vision-based hand gesture recognition (Hossain Gourab et al., 2021).

To prepare the video data input for analysis, various preprocessing techniques are employed to manipulate and refine the dataset by eliminating irrelevant features. One of the well-known preprocessing techniques is the hand pose estimation. In a recent study conducted by Sundar & Bagyammal (2022), they employed the MediaPipe Hands framework for hand pose estimation. This framework incorporates multiple machine learning models to accurately detect the 3D landmarks of a hand in real-time from a single frame. The process involves a palm detection model that provides a hand-bounding box with orientation based on the entire image. Additionally, a hand landmark model utilises the cropped image from the palm detector to determine precise 3D hand key points. The pipeline includes cropping the hand using landmarks from the previous frame, and palm detection is utilised only when the landmark model fails to locate the hand. The MediaPipe hand landmark detector can effectively track 21 hand landmarks, as depicted in Fig. 1.



Fig. 1: Hand landmarks representation using MediaPipe (MediaPipe, 2020)

Convolutional Neural Networks (CNN) are highly effective in image recognition and classification tasks. They extract features from input data and process them through multiple layers to perform classification or prediction. Recurrent Neural Networks (RNN), on the other hand, are designed to handle sequence data and are particularly adept at interpreting data with changing sequences. In a study conducted by Alashhab et al. (2022), 18 different CNN approaches were evaluated and compared. These included network architectures such as MobileNet (Rahman et al., 2023), EfficientNet (Tan and Le, 2019), Xception (Zhu et al., 2022), SqueezeNet (Tsivgoulis et al., 2022), and Darknet-53 (Redmon & Farhadi, 2018). (Lee et al., 2023) utilised ResNet50 specifically to identify individuals' faces and proposed a self-designed CNN model to recognise facial expressions. This model was developed with the aim of aiding visually impaired individuals in the recognition of facial expressions.

Alashhab et al. (2022) also explored the role of RNNs, specifically Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). LSTM is effective for handling long sequences, while GRU is more suitable for short-term sequences and requires fewer training parameters and memory. Moreover, LSTM tends to exhibit higher accuracy on larger datasets. Previous research has convincingly shown the effectiveness of LSTM in tracking trajectories using solely skeleton information (M & Guddeti, 2022). Sundar & Bagyammal (2022) also implements MediaPipe hand landmark detection before applying LSTM. This implementation aligns well with the requirements of dynamic hand gesture recognition. By combining both techniques, this approach effectively harnesses the strengths of each method to precisely capture and classify dynamic hand gestures. This further strengthens the suitability of this combined approach for tasks related to dynamic hand gesture

recognition.

Transformer is another type of neural network architecture that is specifically designed to capture long-range dependencies and temporal relationships within a sequence of hand gesture video data. A study by (D'Eusanio et al., 2020) implements transformer-based neural networks for its hand gesture recognition system. The study primarily focuses on utilising data generated by active depth sensors, specifically depth data and infrared images. The transformer architecture employs an encoder-decoder structure with attention layers to effectively model the sequence. The study demonstrates the effectiveness of frame-level feature extraction and temporal aggregation using a transformer model.

In recent studies, it has been established that preprocessing techniques can enhance video representation and reduce video processing times (Sadiq et al., 2020; Wang J. et al., 2021). Deep learning models, such as CNN, RNN and Transformer have shown their effectiveness for hand gesture recognition (Singh D.K., 2021; Mahboob et al., 2023). Other studies have also demonstrated that hybrid approaches, such as CNN-RNN based framework (Nasir et al., 2021; Li & Langari, 2022) and Transformer-CNN based framework (Zeng & Kwong, 2023; Ullah et al., 2023), yield good performance results.

However, despite the success of these techniques and models, there is a gap in the literature regarding their implementation. Only a limited number of studies have explored the integration of preprocessing techniques and deep learning models for hand gesture recognition. Two notable studies in this domain include Bora et al. (2023) and Sundar & Bagyammal (2022). Sundar and Bagyammal (2022) employed MediaPipe to extract hand landmarks and utilised the LSTM method for their recognition. Bora et al. (2023) implemented MediaPipe hand landmark detection in combination with a Feed Forward Neural Network. Both of these studies specifically focused on sign language recognition.

Therefore, the aim of this paper is to bridge the existing gap by focusing on hand gesture recognition specifically designed to assist immobilised elderly individuals. One key aspect that differentiates our research is the implementation of MediaPipe hand landmark detection in combination with three different recognition models: Transformer + DenseNet, MobileNet + Bidirectional GRU (BiGRU), and Long-Short Term Memory (LSTM). By incorporating these diverse models, our research seeks to identify the best-performing model in terms of accuracy and processing time, surpassing the performance of existing approaches.

3. Research Methodology

In this study, we employed two preprocessing techniques which are the uniformly sampled key frame extraction method and the hand key points extraction method. For hand gesture recognition, we compared three different deep learning models, namely the Transformer + DenseNet, the MobileNet + Bidirectional GRU (BiGRU) and the Long-Short Term Memory (LSTM) model. This section explains the details on how these techniques were implemented.

3.1. Data Preprocessing

Data preprocessing techniques play a crucial role in facilitating the application of deep learning models by enhancing their effectiveness and efficiency. Data preprocessing involves two processes; uniformly sampled key frame extraction and hand key points extraction.

In this project, the uniformly sampled key frame extraction method serves as the first process. It is a technique used to select key frames at regular intervals from a sequence of frames. Every k th frame is selected from the original video, where the value of k is determined based on the duration of the video (Sadiq et al., 2020). Instead of processing every single frame from the video dataset, only a fraction of frames is uniformly selected. For instance, consider a 4-second video with 120 frames. By selecting every 10th frame, the video can be reduced to 12 frames.

This technique eliminates redundancy, effectively reducing the overall computational burden and processing time. Despite the reduced number of frames, the method still captures the essential information present in the original video. It successfully achieves a balance between computational efficiency and preserving meaningful temporal information. Fig. 2 demonstrates an example of the implementation results of this technique. A sequence of 15 frames was extracted from its original video, consisting of 120 frames. This example shows that the uniformly sampled set of frames exhibits fewer redundant frames and encompasses all relevant temporal features of the dynamic hand gesture.

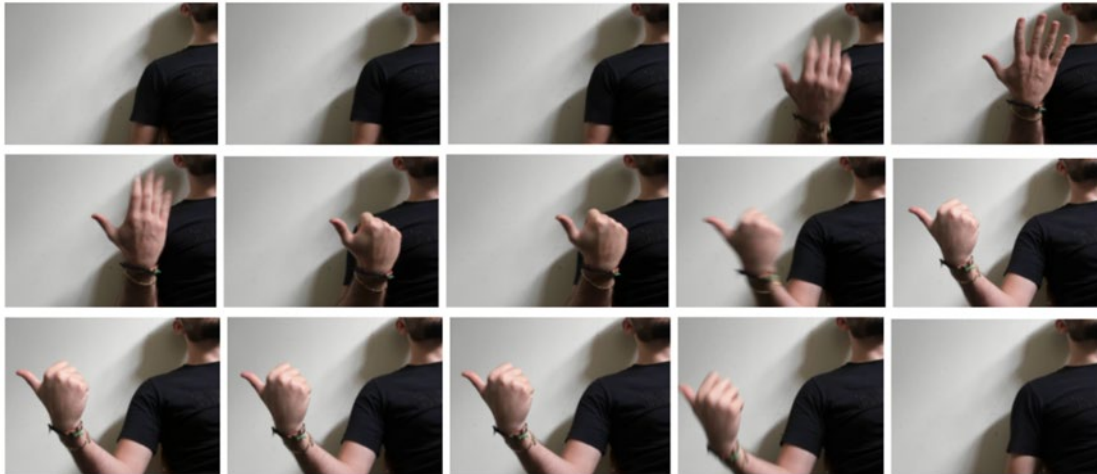


Fig. 2: Example of method implementation results

Consequently, the second process is the hand key points extraction. It involves capturing the essential key points of the hand detected in a frame. According to MediaPipe (2020), once the palm is detected in the entire image, a subsequent hand key points model utilises regression or direct coordinate prediction to accurately locate 21 3D hand-knuckle coordinates within the detected hand regions. This model effectively captures a reliable internal representation of hand posture, regardless of self-occlusions or partially visible hands. When we extract the hand key points from a single frame, we obtain the x , y , and z coordinates for each of the 21 individual hand key points. When the key points are extracted from a sequence of frames, the collection of hand key points data is compiled and stored in numpy arrays, in the form of x , y , and z coordinate values.

This hand key points extraction technique is implemented differently for the LSTM model, compared to the Transformer + DenseNet and MobileNet + BiGRU models. For LSTM, the extracted hand key points are directly utilised as a sequence input to the model. It will be able to learn the temporal patterns and dependencies from the sequential data. For the Transformer + DenseNet and MobileNet + BiGRU models, the extracted hand key points data is plotted on a black background image for each sequence of frames. These processed images serve as fixed-size representations of the hand key points. The images are then treated as inputs for both models.

The reason why it has to be implemented differently is because LSTM is a RNN model commonly used for sequence data processing. In this case, the hand key points data can be treated as a temporal sequence, where the order of the points matters. On the other hand, both Transformer + DenseNet and MobileNet + BiGRU involve CNN architecture that are designed for image classification tasks. It operates on fixed-size image inputs and does not inherently handle temporal dependencies. Therefore, to utilise the models, the temporal sequence of hand key points must be converted into a fixed-size image representation.

3.2. Deep Learning Model

Our proposed models for hand gesture recognition are mainly based on either CNN or RNN alone or a combination of both. Transformer + DenseNet, MobileNet + BiGRU and LSTM are implemented and

compared. These methods have exhibited impressive performance in action recognition tasks (A. D'Eusonio et al., 2020; Rahman et al., 2023; Sundar & Bagyammal, 2022) and are well-suited for analysing time series data (Lai & Yanushkevich, 2018). Each model will be trained separately using the same experimental setup and dataset.

3.2.1. Transformers + DenseNet

The first proposed model is Transformer with CNN. The key idea behind the Transformer model is to replace traditional RNNs with an attention mechanism. RNNs process sequential data one step at a time, making them computationally expensive and difficult to parallelize. In contrast, the Transformer model allows for parallel computation and captures dependencies across the entire input sequence using self-attention. and learn expressive representations. Fig. 3 depicts the Transformer model architecture. The self-attention mechanism allows each position in the sequence to attend to all other positions, capturing the importance and relevance of different parts of the sequence. This allows the model to capture long-range dependencies more effectively compared to traditional sequential models (Valanarasu et al., 2021).

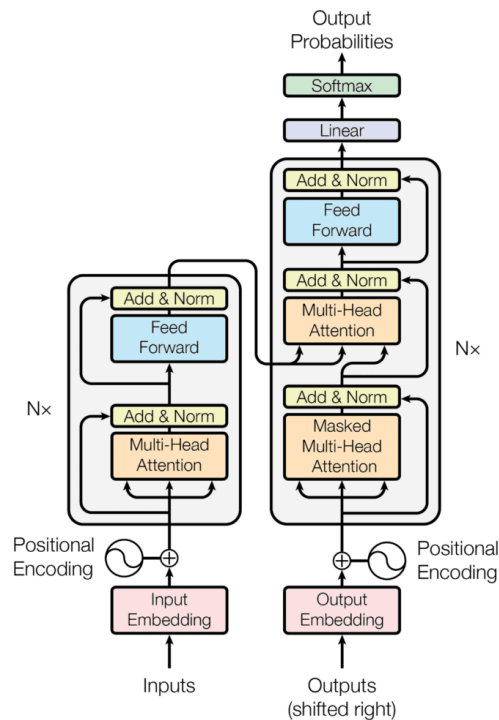


Fig. 3: Transformer model architecture (Vaswani et al., 2017)

DenseNet is a type of CNN that connects each layer to every other layer within the same block. This design encourages the sharing and reuse of features, leading to improved performance and reduced parameter count. DenseNet's structure allows for effective feature propagation and enhances the network's ability to generalise and learn complex patterns (Li et al., 2018). However, it also has noticeable drawbacks, including the absence of considering interdependencies between different channels and the lack of explicit modelling for interlayer feature map correlation (Zhang et al., 2019).

In this work, we focus on combining the strengths of both the Transformer and DenseNet architectures. In the combined implementation, the DenseNet architecture is used as a feature extractor to process the input data and extract relevant features. These features are then passed to the Transformer model, which captures the long-range dependencies and further refines the representations. By

leveraging the strengths of both architectures, the combined model benefits from the powerful feature extraction capabilities of DenseNet and the ability of the Transformer to capture long-range dependencies. Fig. 4 shows the Python script of DenseNet layers embedded together with the Transformer model.

```
def get_compiled_model():
    sequence_length = MAX_SEQ_LENGTH
    embed_dim = NUM_FEATURES
    dense_dim = 4
    num_heads = 1
    classes = len(label_processor.get_vocabulary())

    inputs = keras.Input(shape=(None, None))
    x = PositionalEmbedding(
        sequence_length, embed_dim, name="frame_position_embedding"
    )(inputs)
    x = TransformerEncoder(embed_dim, dense_dim, num_heads, name="transformer_layer")(x)
    x = layers.GlobalMaxPooling1D()(x)
    x = layers.Dropout(0.5)(x)
    outputs = layers.Dense(classes, activation="softmax")(x)
    model = keras.Model(inputs, outputs)

    model.compile(
        optimizer="adam", loss="sparse_categorical_crossentropy", metrics=["accuracy"]
    )
    print(model.summary())
    return model
```

Fig. 4: Python script of DenseNet layers embedded together with Transformer model

3.2.2. MobileNet + BiGRU

The second proposed model is the MobileNet + BiGRU model. MobileNet, a CNN architecture open-sourced by Google, is an excellent choice for training classifiers that require small and fast models, especially when computational resources are limited. In a study by F. Wang et al. (2021), MobileNet was utilised to extract gesture features. Fig. 5 shows the architecture of the MobileNet model.

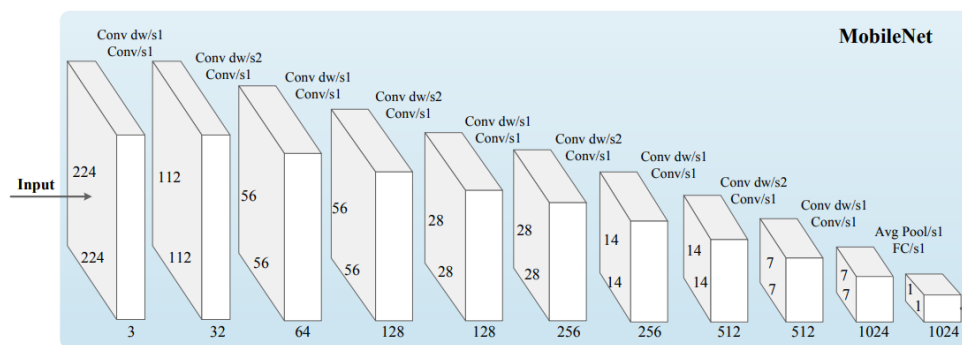


Fig. 5: MobileNet Architecture (F. Wang et al., 2021)

MobileNet decomposes the standard convolutional operation into two separate operations: depthwise separable convolution and pointwise convolution. The depthwise separable convolution applies a single convolutional filter per input channel, reducing the computational cost significantly. The pointwise convolution applies a 1x1 convolution to combine the output channels of the depthwise separable convolution. The MobileNet architecture employs a lightweight structure with fewer parameters, making it suitable for resource-constrained environments. Notably, MobileNet differs from other well-known CNN networks like AlexNet and ResNet in that it only connects the output of the (l-1)th layer as input to the lth layer, contributing to its efficiency and compactness.

The GRU (Gated Recurrent Unit) is a type of RNN cell that addresses the vanishing gradient

problem. It incorporates gating mechanisms, including the update gate and reset gate, to control the flow of information within the cell. These gates regulate the information update and retention in the cell's hidden state, allowing the network to capture long-term dependencies in sequential data. BiGRU, short for Bidirectional Gated Recurrent Unit, is a variant that captures contextual information from both past and future inputs (Qiao, Y. et al., 2023). It consists of two GRU layers, one processing the input sequence in the forward direction and the other in the backward direction. The outputs of both layers are concatenated to capture the dependencies in both temporal directions.

In this implementation, both MobileNet and BiGRU are applied together. The MobileNet architecture is used as a feature extractor to process input data, such as images or video frames. The output features from the MobileNet layers are then fed into the BiGRU layers to capture temporal dependencies and contextual information. Fig. 6 and Fig. 7 displays the Python script of the model, which visualises the architecture of MobileNet and BiGRU respectively.

```
def build_feature_extractor():
    feature_extractor = keras.applications.MobileNet(
        weights="imagenet",
        include_top=False,
        pooling="avg",
        input_shape=(IMG_SIZE, IMG_SIZE, 3),
    )
    preprocess_input = keras.applications.mobilenet.preprocess_input

    inputs = keras.Input((IMG_SIZE, IMG_SIZE, 3))
    preprocessed = preprocess_input(inputs)

    outputs = feature_extractor(preprocessed)
    return keras.Model(inputs, outputs, name="feature_extractor")

feature_extractor = build_feature_extractor()
```

Fig. 6: Python script of MobileNet model

```
# Utility for sequence model.
def get_sequence_model():
    class_vocab = label_processor.get_vocabulary()

    frame_features_input = keras.Input((MAX_SEQ_LENGTH, NUM_FEATURES))
    mask_input = keras.Input((MAX_SEQ_LENGTH,), dtype="bool")

    x = keras.layers.GRU(16, return_sequences=True)(
        frame_features_input, mask=mask_input
    )
    x = keras.layers.Bidirectional(keras.layers.GRU(8))(x)
    x = keras.layers.Dropout(0.4)(x)
    x = keras.layers.Dense(8, activation="relu")(x)
    output = keras.layers.Dense(len(class_vocab), activation="softmax")(x)

    rnn_model = keras.Model([frame_features_input, mask_input], output)

    rnn_model.compile(
        loss="sparse_categorical_crossentropy", optimizer="adam", metrics=["accuracy"]
    )
    return rnn_model
```

Fig. 7: Python script of BiGRU layers in Sequential model

3.2.3. LSTM

The third proposed model is the LSTM model. It is an extension of the conventional RNN architecture, designed to address the issue of short-term memory limitation in RNNs. LSTMs are particularly relevant to our project's objective of achieving real-time gesture recognition. One key advantage of using LSTMs for sequence classification is their ability to capture and retain information from the input data over longer temporal dependencies. This characteristic eliminates the requirement for manual feature engineering since LSTMs can learn directly from the raw time series data (Ünlü, R., 2021).

An LSTM neuron comprises three gates: the input gate, output gate, and forget gate. These gates play a crucial role in managing memory and controlling the flow of gradients within the recurrent network's memory. The input gate determines whether the information received from the previous cell should be retained or discarded. The output gate decides which information should be passed to the output neuron and whether it should be preserved in the cell's internal state matrix. The forget gate maintains information about the current context being processed by the cell. This information guides the cell's response when it receives new input during subsequent processing steps. By integrating these components, an LSTM neuron can acquire new knowledge while retaining previously learned information over time. The structure of the LSTM unit is visualised in Fig. 8.

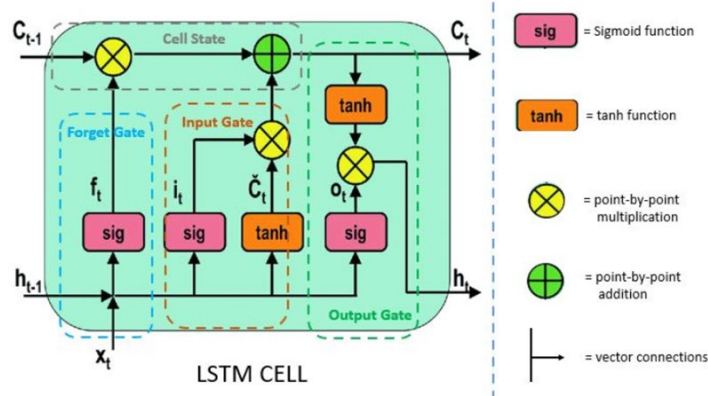


Fig. 8: Structure of LSTM unit (Smagulova & James, 2019)

The implementation of the code involves importing important classes such as the LSTM layer and Dense layer from the Keras library. In Fig. 9, a Python script is provided to illustrate the structure of the sequential model. The model comprises three sets of LSTM layers. The first LSTM layer consists of 64 LSTM units and accepts the input shape of the number of frames and number of landmarks. For example, if the model receives an input of 30 frames by 63 landmarks, this layer will return the sequences to the next layer, as they are required for further processing. The second and third LSTM layers consist of 128 and 64 units, respectively. Subsequently, there are three Dense layers, which implement the concept of fully connected layers.

```
%pip install tensorflow==2.4.1
%pip install tensorflow-gpu==2.4.1

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense

model = Sequential()
model.add(LSTM(64, return_sequences=True, activation='relu',
              input_shape=(NUM_OF_FRAMES, NUM_OF_KEYPOINTS)))
model.add(LSTM(128, return_sequences=True, activation='relu'))
model.add(LSTM(64, return_sequences=False, activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(32, activation='relu'))
model.add(Dense(len(label_names), activation='softmax'))

model.compile(optimizer='Adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

Fig. 9: Python script of LSTM layers in Sequential model

The entire system employs the Rectified Linear Unit (ReLU) activation function. ReLU is a piecewise linear function that outputs the input directly if it is positive, and returns zero if the input is

negative. The Adam Optimizer technique is utilised as the optimizer, which iteratively updates the weights based on the training data (Sundar & Bagyammal, 2022). The final layer of the model utilises the Softmax activation function, which converts the outputs into a vector of probabilities ranging from 0 to 1. This enables a probabilistic interpretation of the output, which is highly valuable for our analysis. During the detection of a hand gesture, the model calculates the detection percentage for all 14 hand gestures. The model's prediction is determined as the gesture with the highest probability.

4. Experiment

4.1. Environmental Setup

The environmental setup for this project is as depicted in Fig. 10, the environment is designed for elderly individuals receiving care either in a hospital or in their own homes. The configuration involves placing a camera on a table at the foot of the bed. A typical bed has a length of 1.20 metres.

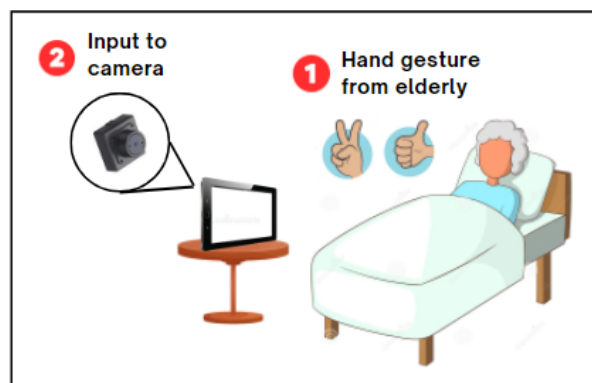


Fig. 10: Environmental setup for the elderly

4.2. Dataset

For this project, two distinct datasets were utilised; one for training and validation purposes, and another for testing. The training dataset was obtained from a publicly available source, while the testing dataset consists of our own data.









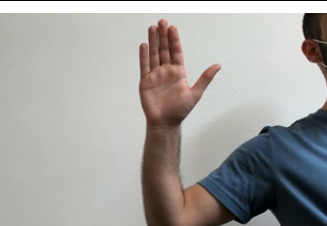
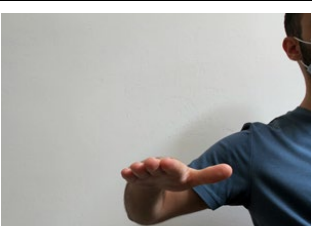
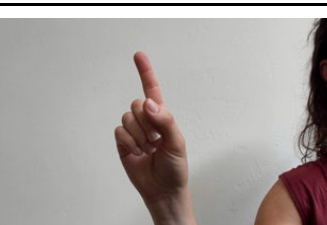
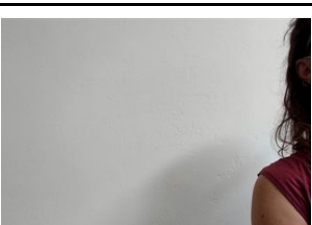
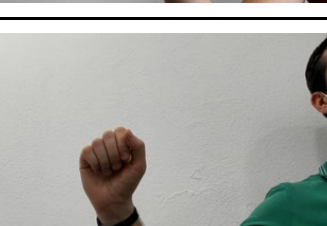
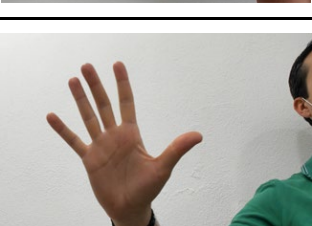
The training and validation dataset is sourced from the Dataset for Dynamic Hand Gesture Recognition Systems available on IEEE Dataport, as described by Fronteddu et al. (2021). This dataset encompasses 14 distinct hand gestures, each performed by 21 different users. Each user performed the same gesture three times. The motion was recorded using a 1920 x 1080p laptop camera at a frame rate of 30 frames per second. Each video has a duration of 4 seconds, comprising a total of 120 frames. In summary, the dataset consists of 882 videos with a combined total of 150,840 frames. Table 1 provides an overview of each dynamic gesture captured in our dataset. An interpretation was assigned to each gesture.

For the testing dataset, we employ our own self-recorded videos featuring a single user performing the same 14 hand gestures, with each gesture repeated three times. Each video consists of 120 frames, with a duration of 4 seconds. To maintain consistency with the existing dataset, we employed an empty background setup. The image frame contains the subject's face, hands, and upper body. In total, there are 42 pre-recorded videos. Sample screenshots of the videos are displayed in Table 2.

Table 1: Dynamic gestures (Fronteddu et al., 2021) with its message

Dynamic Gestures		Label & Message
Beginning	Ending	

		<p>Bath: I want to take a bath</p>
		<p>Call: I want to contact my family</p>
		<p>Change Clothes: I want to change my clothes</p>
		<p>Change Position: I want to adjust my body position</p>
		<p>Clean Room: Please clean my room</p>
		<p>Drink: I want to drink</p>

		<p>Eat: I want to eat</p>
		<p>Go Out: I want to go out</p>
		<p>Help: I need help</p>
		<p>Medicine: I need medicine</p>
		<p>Pray: I want to pray</p>
		<p>Quran: I want to recite or listen to the Quran</p>
		<p>Sick: I am sick</p>

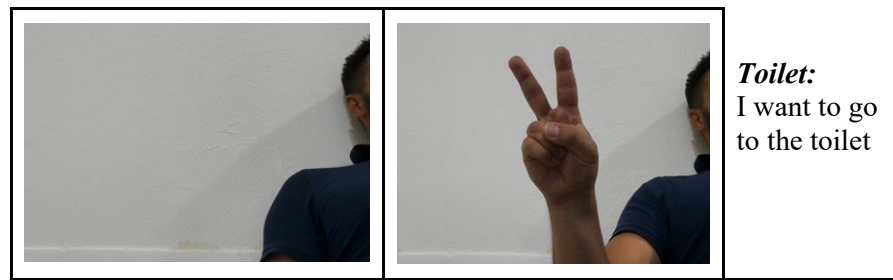
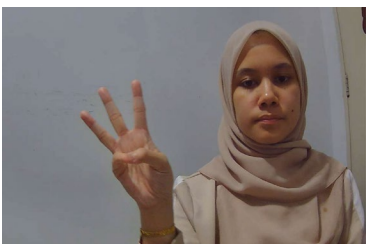

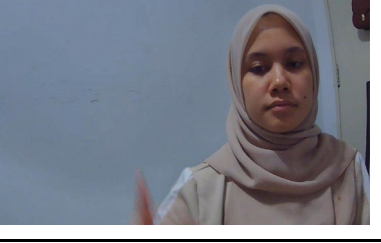
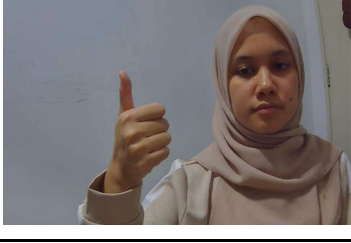
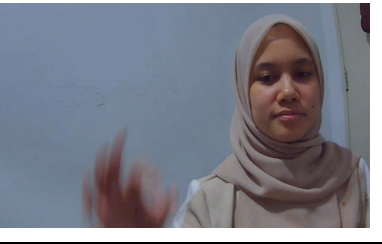
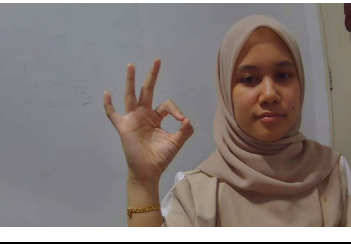


Table 2: Sample screenshots of the pre-recorded testing videos

Dynamic Gestures		Label
Beginning	Ending	
		Bath
		Call
		Change Clothes

4.3. Model Training and Testing

Before training the model, the dataset needs to be divided into a training set and a validation set. The 80:20 ratio is applied, resulting in 705 videos in the training set and 177 videos in the validation set. To extract the video frames, the OpenCV function Video-Capture is utilised. The preprocessing method called the uniformly sampled key frame extraction method is implemented to reduce frame redundancy and processing time. The number of frames was reduced from 120 frames to 15 frames. The images are resized to a suitable dimension of 512×512 pixels. The number of features for Transformer + DenseNet and MobileNet + BiGRU is 1024, while the number of hand key points for LSTM is 63. The models were trained for 100 epochs. Fig. 11, 12 and 13 are the summary for the Transformer + DenseNet, MobileNet + BiGRU and LSTM models respectively.

Layer (type)	Output Shape	Param #
input_4 (InputLayer)	[(None, None, None)]	0
frame_position_embedding (PositionalEmbedding)	(None, None, 1024)	15360
transformer_layer (TransformerEncoder)	(None, None, 1024)	4211716
global_max_pooling1d_1 (GlobalMaxPooling1D)	(None, 1024)	0
dropout_1 (Dropout)	(None, 1024)	0
dense_5 (Dense)	(None, 14)	14350

=====
 Total params: 4,241,426
 Trainable params: 4,241,426
 Non-trainable params: 0

Fig. 11: Transformer + DenseNet model summary

Layer (type)	Output Shape	Param #	Connected to
input_3 (InputLayer)	[(None, 10, 1024)]	0	[]
input_4 (InputLayer)	[(None, 10)]	0	[]
gru (GRU)	(None, 10, 16)	50016	['input_3[0][0]', 'input_4[0][0]']
bidirectional (Bidirectional)	(None, 16)	1248	['gru[0][0]']
dropout (Dropout)	(None, 16)	0	['bidirectional[0][0]']
dense (Dense)	(None, 8)	136	['dropout[0][0]']
dense_1 (Dense)	(None, 10)	90	['dense[0][0]']

=====
 Total params: 51,490
 Trainable params: 51,490
 Non-trainable params: 0

Fig. 12: MobileNet + BiGRU model summary

```

Model: "sequential_1"
-----
Layer (type)                Output Shape                Param #
-----
lstm_3 (LSTM)                (None, 15, 64)             32768
lstm_4 (LSTM)                (None, 15, 128)            98816
lstm_5 (LSTM)                (None, 64)                  49408
dense_3 (Dense)              (None, 64)                  4160
dense_4 (Dense)              (None, 32)                  2080
dense_5 (Dense)              (None, 14)                  462
-----
Total params: 187,694
Trainable params: 187,694
Non-trainable params: 0

```

Fig. 13: LSTM model summary

During the testing phase, each video was processed to extract its frames. The frames were preprocessed and then analysed using the three proposed models, namely Transformer + DenseNet, MobileNet + BiGRU, and LSTM. For each frame, the models calculated the probability percentage associated with each gesture. Subsequently, the results from the videos were averaged to obtain a more representative and stable evaluation metric.

5. Result and Discussion

Table 3 compares the results of the three proposed models. The best results are stressed in bold. It can be observed that there is a huge difference in the amount of processing and training time. The LSTM model had the shortest processing and training time at 35 minutes, followed by the Transformer + DenseNet model at 49 minutes and MobileNet + BiGRU at 55 minutes. The difference in time is due to the architectural differences and complexities of the models. LSTM generally has a simpler architecture compared to Transformer + DenseNet and MobileNet + BiGRU.

Table 3: Model training and validation results

Recognition model	Processing time and training time	Validation accuracy	Validation loss
Transformer + DenseNet	49 mins	87.23%	71.42%
MobileNet + BiGRU	55 mins	92.20%	34.31%
LSTM	35 mins	94.33%	21.54%

In terms of accuracy, the LSTM model achieved the highest validation accuracy of 94.33%, followed by MobileNet + BiGRU at 92.20%, and Transformer + DenseNet at 87.23%. The superior accuracy of the LSTM model can be attributed to its ability to capture temporal dependencies in the input data. LSTM is specifically designed to model sequential data, which provides an inherent advantage for gesture recognition tasks that involve temporal patterns. Additionally, as mentioned earlier, the hand key point extraction technique is applied differently for the LSTM model. This

technique directly extracts the x, y, and z coordinates of the hand key points, which are then used as input for the LSTM model.

The LSTM model also had the lowest validation loss of 21.54%, indicating better generalisation and a closer fit to the ground truth compared to the other models. The lower loss value suggests that the LSTM model effectively learned the underlying patterns and relationships within the input data, leading to improved accuracy. The results for the testing phase were recorded as well. Table 4 displays the comparison of the testing results between the proposed models.

Table 4: Comparison of the testing results between the proposed models

Gesture label/class	Average Probability Percentage (%)		
	Transformer + DenseNet	MobileNet + BiGRU	LSTM
Bath	99.99	38.40	62.95
Call	99.99	71.97	98.99
Change Clothes	0.66	98.39	99.36
Change Position	99.99	96.57	47.51
Clean Room	0.00	35.79	34.57
Drink	4.07	92.45	15.73
Eat	33.55	86.25	56.07
Go Out	81.99	54.68	90.03
Help	66.68	43.59	99.98
Medicine	98.87	85.28	78.35
Pray	75.66	98.87	99.21
Quran	97.16	96.49	90.94
Sick	98.53	95.65	79.58
Toilet	5.74	92.01	80.79

The Transformer + DenseNet model exhibits inconsistent probability results. While it can yield high probability values of up to 99.99% for certain gestures, it may also encounter cases where the probability drops to 0.00% for other gestures. MobileNet + BiGRU and LSTM show more consistent results. For MobileNet + BiGRU, the lowest probability value is 35.79% and the highest probability is 98.87%. Meanwhile, for LSTM, the lowest probability value is 15.73% and the highest probability is 99.21%. Based on the results, it is observed that the architecture of MobileNet + BiGRU and LSTM are better suited for the gesture recognition task, resulting in more consistent probability values. Furthermore, the Transformer + DenseNet model is prone to overfitting. The model becomes too specialised to the training data and fails to generalise well to unseen data. The recognition results show that it assigns excessively high probabilities to certain gestures but perform poorly on other gestures, resulting in low probabilities.

The gesture labels for Quran and Sick consistently yield high results. However, the Clean Room gesture label shows poor results, while the Drink gesture label exhibits inconsistent outcomes. This is due to the complexity of the gesture. The Quran and Sick gesture are more distinctive and do not involve complex movement. Fig. 14 and 15 visualises the frame screenshots of Clean Room and Drink gesture labels, respectively.

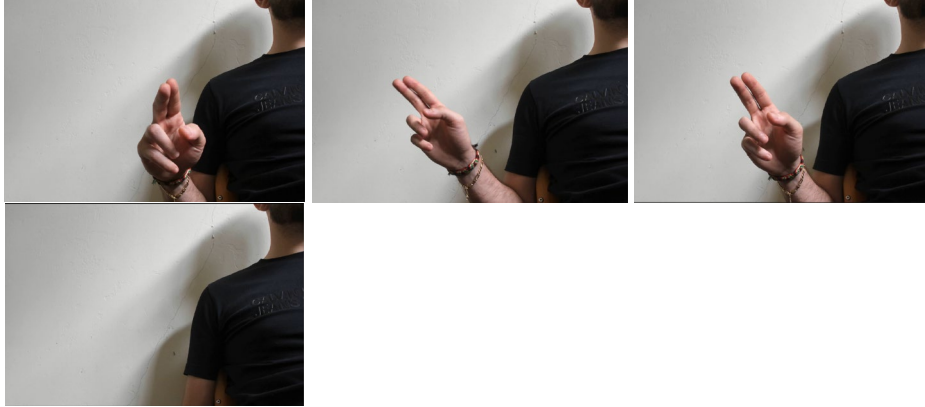


Fig. 14: Frame screenshots of Clean Room gesture

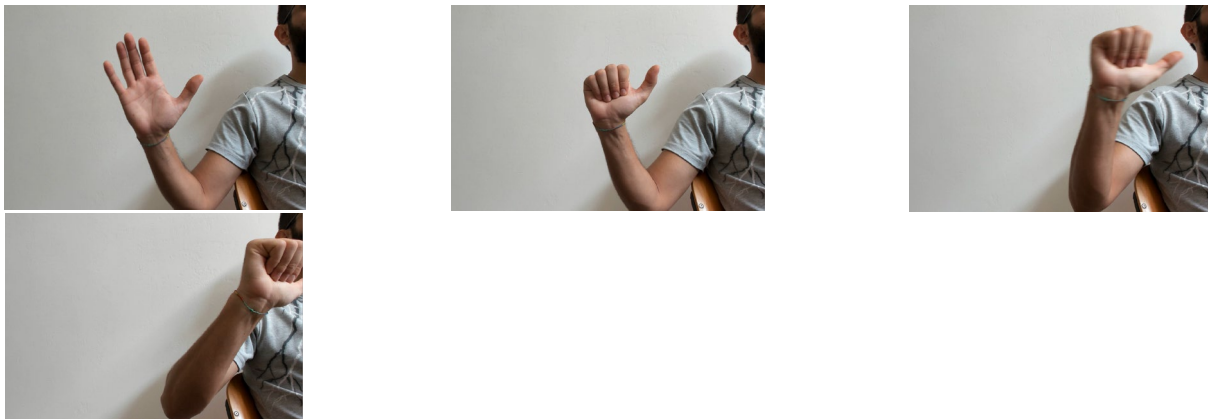


Fig. 15: Frame screenshots of Drink gesture

While the Clean Room gesture involves subtle variations that are challenging for the model to capture, the Drink gesture occasionally has high probabilities, reaching as high as 92.45%, but it can also produce low probabilities as low as 4.07%. This inconsistency indicates that the distinguishing characteristics of the Drink gesture are subtle or easily confused with other similar gestures in the dataset. This issue can be addressed by replacing the current gesture associated with the Drink label with a more distinct and easily recognizable gesture.

Fig. 16 illustrates the learning curves for all three proposed models. The learning curve of Transformer + DenseNet exhibits an overfitting pattern. The training loss consistently decreases, while the validation loss initially decreases but eventually starts to increase again. This suggests that the model has learned the training dataset too well, resulting in an increase in generalisation error. The learning curves of both MobileNet + BiGRU and LSTM demonstrate a good fit graph.

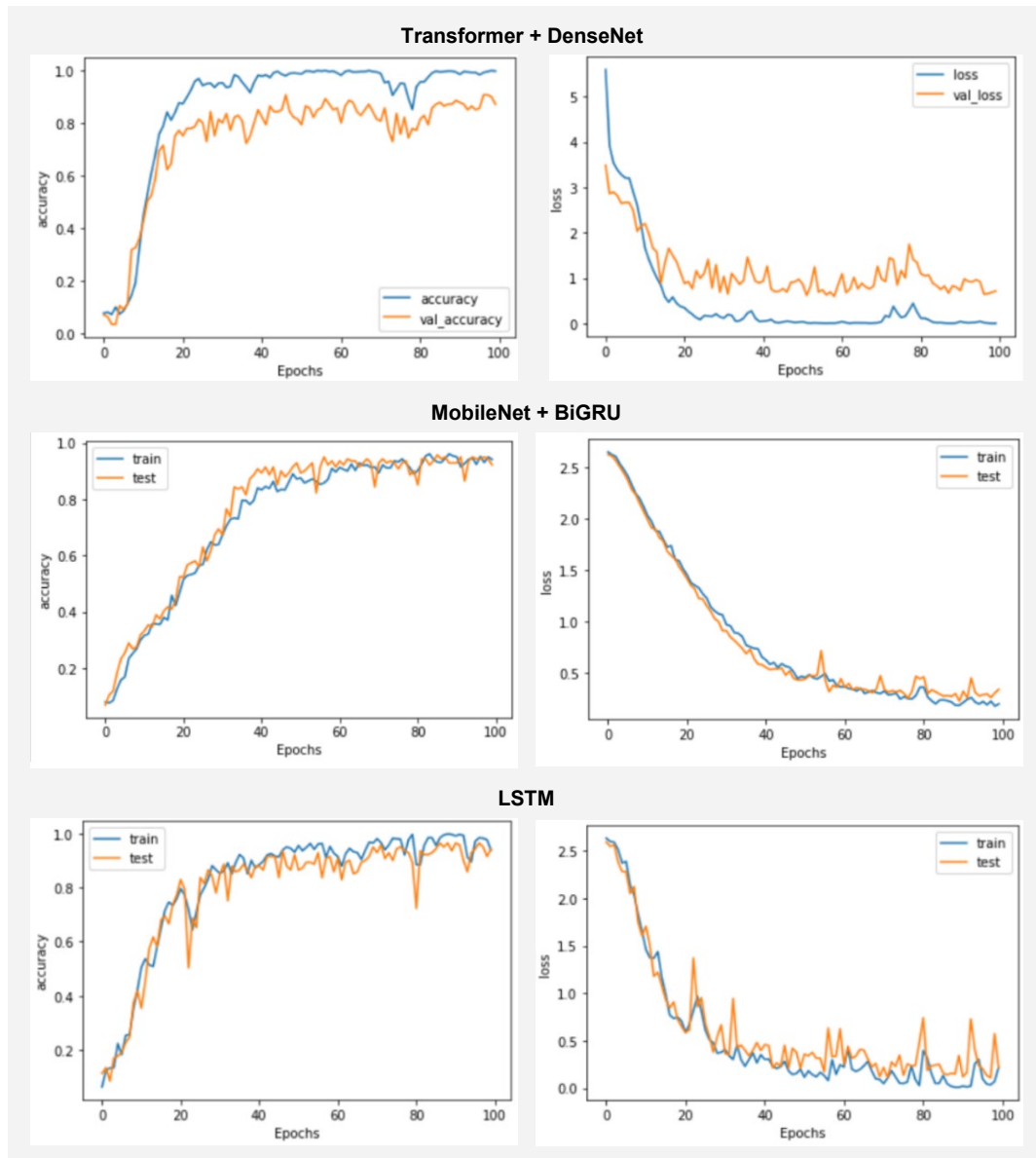


Fig. 16: Learning curves of all three proposed models

6. Conclusion

In this study, we compared the performance of three deep learning models for dynamic hand gesture recognition. During the training and validation process, LSTM exhibited the best performance, achieving the highest accuracy of 94.33% and the lowest loss value of 21.54%. LSTM demonstrated stability and consistency across the 14 classes and has outperformed the other two proposed models. This paper also made significant contributions by implementing preprocessing techniques, namely the uniformly sampled key frame extraction and hand key points extraction. These techniques had a notable impact on reducing processing time and enhancing the overall performance of the deep learning models. In future work, we will focus on resolving the problem of the few insignificant gestures that produce inconsistent results. Additionally, there is a goal to incorporate real-time testing for the model, ensuring accurate recognition outcomes. This research is driven by the motivation to explore and implement advanced deep learning techniques that can provide optimal performance for hand gesture recognition.

Acknowledgements

This work was supported by FRGDS grant via Multimedia University, Cyberjaya, Selangor, Malaysia

under the grant number MMUE/210029.

References

- Adithya V., & Rajesh R. (2020). A Deep Convolutional Neural Network Approach for Static Hand Gesture Recognition. *Procedia Computer Science*, 171, 2353–2361.
- Bora, J., Saine Dehingia, Abhijit Boruah, Anuraag Anuj Chetia, & Dikhit Gogoi. (2023). Real-time Assamese Sign Language Recognition using MediaPipe and Deep Learning. 218, 1384–1393.
- Damaneh, M. M., Mohanna, F., & Jafari, P. (2023). Static hand gesture recognition in sign language based on convolutional neural network with feature extraction method using ORB descriptor and Gabor filter. *Expert Systems with Applications*, 211, 118559.
- D'Eusano, A., Simoni, A., Pini, S., Borghi, G., Vezzani, R., & Cucchiara, R. (2020, November 1). *A Transformer-Based Network for Dynamic Hand Gesture Recognition*. IEEE Xplore.
- Fahmid, A., Noramiza, H., Junaidi, A., Md Roman, B., Wan Noor Shahida, M., Jia, U., Mohammad, A., & Mohd, N. (2019). Vision Based Gesture Recognition from RGB Video frames Using Morphological Image Processing Techniques. *International Journal of Advanced Science and Technology*, 28(13), 321–332.
- Fronteddu G., Porcu S., Floris A., Atzori L.. (2021). Dataset for Dynamic Hand Gesture Recognition Systems. IEEE Dataport.
- Hossain Gourab, J., Raxit, S., & Hasan, A. (2021, July 1). *A Robotic Hand: Controlled With Vision Based Hand Gesture Recognition System*. IEEE Xplore.
- Lai, K., & Yanushkevich, S. N. (2018, August 1). CNN+RNN Depth and Skeleton based Dynamic Hand Gesture Recognition. IEEE Xplore.
- Lee, J.R., Ng, K.-W., & Yoong, Y.-J. (2023). Face and Facial Expressions Recognition System for Blind People Using ResNet50 Architecture and CNN. *Journal of Informatics and Web Engineering*, 2(2), 284–298.
- Li, Q., & Langari, R. (2022). EMG-based HCI Using CNN-LSTM Neural Network for Dynamic Hand Gestures Recognition. *IFAC-PapersOnLine*, 55(37), 426–431.
- Li, S., Deng, M., Lee, J., Sinha, A., & Barbastathis, G. (2018). Imaging through glass diffusers using densely connected convolutional networks. *Optica*, 5(7), 803.
- M, R., & Guddeti, R. M. R. (2022). Human identification system using 3D skeleton-based gait features and LSTM model. *Journal of Visual Communication and Image Representation*, 82, 103416.
- Mahboob, T., Chung, M. Y., & Choi, K. W. (2023). EMG-based 3D hand gesture prediction using transformer–encoder classification. *ICT Express*.
- Marzuki H. A., & Hashim, N. (2022). Dynamic Hand Gesture Recognition Based on Deep Learning for Muslim Elderly Care. 544–557.
- MediaPipe. (2020). Hand Landmark Model. Mediapipe Github.
- Oudah, M., Al-Naji, A., & Chahl, J. (2020). Elderly Care Based on Hand Gestures Using Kinect Sensor. *Computers*, 10(1), 5.
- Qiao, Y., Xu, H.-M., Zhou, W.-J., Peng, B., Hu, B., & Guo, X. (2023). A BiGRU joint optimised attention network for recognition of drilling conditions.

- Rahman, M. H., Jannat, M. K. A., Islam, M. S., Grossi, G., Bursic, S., & Aktaruzzaman, M. (2023). Real-time face mask position recognition system based on MobileNet model. *Smart Health*, 100382.
- Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. *ArXiv:1804.02767 [Cs]*.
- Sadiq, B., Muhammad, B., Abdullahi, M., Onuh, G., Ali, A., Babatunde, A. (2020). Keyframe Extraction Techniques: A Review. *ELEKTRIKA- Journal of Electrical Engineering*, 19, 54-60.
- Singh, D. K. (2021). 3D-CNN based Dynamic Gesture Recognition for Indian Sign Language Modeling. *Procedia Computer Science*, 189, 76–83.
- Singh, S., Gupta, A. K., & Singh, T. (2019). Computer Vision based Hand Gesture Recognition A Survey. *International Journal of Computer Sciences and Engineering*, 7(5), 507–515.
- Sundar, B., & Bagyammal, T. (2022). American Sign Language Recognition for Alphabets Using MediaPipe and LSTM. *Procedia Computer Science*, 215, 642–651.
- Tan, M., & Le, Q. (2019, May 24). *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. Proceedings.mlr.press; PMLR.
- Tsivgoulis, M., Papastergiou, T., & Megalooikonomou, V. (2022). An improved SqueezeNet model for the diagnosis of lung cancer in CT scans. *Machine Learning with Applications*, 100399.
- Ullah, W., Hussain, T., Ullah, M., Mi Young Lee, & Sung Wook Baik. (2023). TransCNN: Hybrid CNN and transformer mechanism for surveillance anomaly detection. 123, 106173–106173.
- Ünlü, R. (2021). Cost-oriented LSTM methods for possible expansion of control charting signals. *Computers & Industrial Engineering*, 154, 107163.
- Valanarasu, J. M. J., Oza, P., Hacihaliloglu, I., & Patel, V. M. (2021, July 6). *Medical Transformer: Gated Axial-Attention for Medical Image Segmentation*. ArXiv.org.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). *Attention Is All You Need*. ArXiv.org.
- Wang, F., Hu, R., & Jin, Y. (2021). Research on gesture image recognition method based on transfer learning. *Procedia Computer Science*, 187, 140–145.
- Wang, J., Zeng, C., Wang, Z.-S., & Jiang, K. (2021). An improved smart key frame extraction algorithm for vehicle target recognition. 97, 107540–107540.
- Wang, X., Veeramani, D., & Zhu, Z. (2023). Gaze-aware hand gesture recognition for intelligent construction. *Engineering Applications of Artificial Intelligence*, 123, 106179.
- Yu, J., Qin, M., & Zhou, S. (2022). Dynamic gesture recognition based on 2D convolutional neural network and feature fusion. *Scientific Reports*, 12(1).
- Zeng, C., & Kwong, S. (2023). Combining CNN and Transformers for Full-Reference and No-Reference Image Quality Assessment. 549, 126437–126437.
- Zhang, K., Guo, Y., Wang, X., Yuan, J., & Ding, Q. (2019). Multiple Feature Reweight DenseNet for Image Classification. *IEEE Access*, 7, 9872–9880.
- Zhu, Y., JiaYI, H., Li, Y., & Li, W. (2022). Image identification of cashmere and wool fibres based on the improved Xception network. *Journal of King Saud University - Computer and Information Sciences*, 34(10, Part B), 9301–9310.