Chapter 5

Facial Expression Recognition Using CNN-BiLSTM Architecture

5.1 Introduction

Facial expression recognition can be accomplished through the analysis of both static images and dynamic image sequences. Static images are valuable for extracting detailed geometric and appearance features of facial expressions [166]. However, they may not capture the complete spectrum of emotions due to their inability to depict dynamic changes associated with facial expressions. Facial expressions are essentially a complex interplay of muscle contractions and relaxations across the face, a dynamic process that static images alone cannot fully represent.

Deep learning has become increasingly prominent in the field of computer vision, particularly due to advancements in computational power such as GPU and TPU support for extensive data training. This has led to significant improvements in image recognition and detection. Among the deep learning architectures, the CNN has demonstrated remarkable capabilities in FER systems, owing to its robust feature extraction capabilities. CNNs are highly effective in processing static images by extracting features based on image appearances [167]. However, static images alone may not capture the entirety of emotions, as they lack dynamic sequences associated with facial expressions. To address this limitation, researchers have integrated various feature extraction approaches, such as combining CNNs with LSTM networks or CNN-RNN architectures [168] [169]. These techniques are designed to extract dynamic sequences of features from sequence of facial images, thereby enhancing the ability to recognize temporal aspects of facial expressions.

Inspired by different CNN-LSTM-based algorithms, this chapter presents fusion feature extraction techniques from a sequence of facial expression images. This study proposes (i) A CNN-LSTM fusion model for the analysis of the sequence of facial expressions, (ii) Data augmentation techniques used for generating various illumination, noise, and different angles of the facial image for improving the emotion recognition power of the system in the real-time scenario, (iii) A hyperparameter tweaked skeleton of VGG-19 used for extracting spatial feature, which overcomes the shortcoming of conventional feature method, (iv) The designed CNN-BiLSTM method is used to extract spatiotemporal features

which classify each emotion accuracy based on feature vector sequence and (v) The proposed system's performance is compared to the benchmark dataset and state-of-the-art methods. In addition, real-time facial expressions were collected in a controlled environment for this experiment.

The remaining section is organized as follows: Section 5.2 explains spatiotemporal features in facial expression recognition, Section 5.3 presents the proposed CNN- BiLSTM architecture and its parameter settings, Section 5.4 explains the real-time facial expression dataset collection and preprocessing method, Section 5.5 describes the results and analysis of the proposed model's performance and comparison results with a baseline and state-of-the-art techniques, Section 5.6 discusses the results and findings, and finally, Section 5.7 presents the chapter summary and insights of the proposed model.

5.2 Spatiotemporal Features

5.2.1 Spatial Features

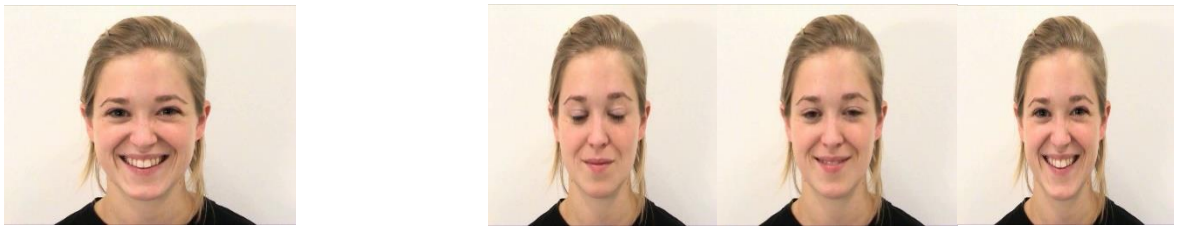
Spatial features refer to static facial characteristics such as the arrangement of facial components (e.g., eyes, nose, mouth) and their geometric relationships. Spatial features capture information about facial landmarks, texture, color, and shape. They are essential for recognizing basic facial expressions like happiness, sadness, anger, surprise, fear, disgust, and neutrality. In addition, this feature can be extracted using techniques like facial landmark detection, texture analysis, and deep learning-based feature extraction methods (e.g., CNN).

5.2.2 Temporal Features

Temporal features focus on the dynamic changes in facial expressions over time. They capture the motion and timing of facial muscle movements, which are crucial for understanding the intensity and duration of emotional expressions. Temporal features help differentiate between subtle variations of expressions and can enhance the accuracy of emotion recognition systems. In addition, these features can be captured using techniques such as optical flow analysis, facial action unit detection (based on the FACS), and RNNs or LSTM or GRU networks for sequence modeling.

5.2.3 Spatiotemporal Features

Spatiotemporal features combine spatial and temporal information to provide a comprehensive representation of facial expressions. Spatiotemporal features integrate both static facial attributes (spatial) and their dynamic changes over time (temporal). This combination enables the recognition of complex facial expressions, micro-expressions (brief facial movements lasting a fraction of a second), and continuous emotional states. In addition, these features often involve combining spatial and temporal information through techniques like spatiotemporal convolutional networks (STCNs), 3D convolutional neural networks (3D CNNs), or spatiotemporal feature fusion methods.



(a) Spatial Features (Static) (b) Spatiotemporal features (Dynamic)

Figure 5.1. Difference between Spatial and Spatiotemporal features

5.3 Proposed Framework for CNN-BiLSTM Model

The intended method seeks to unveil the connection between sequences of facial expressions and their respective labels. As highlighted earlier, facial expressions arise from the coordinated contraction and relaxation of one or more facial muscles.

5.3.1 Convolutional Neural Network

The CNN is a commonly employed architecture for extracting spatial information, exhibiting state-of-the-art performance across various computer vision tasks. Comprising four fundamental layers — convolution, pooling, ReLu, and fully connected — CNN's core architecture is illustrated in Figure 5.2 Proving effectiveness in diverse applications CNN excels in medical image analysis, image segmentation, object detection, and image classification [170][171]. The primary goal of CNN is to extract local descriptors from the top layer and transmit them to lower levels for intricate descriptor extraction. The convolution layer, containing filters determining each convolution block's feature map tensor, plays a crucial role in extracting unique attributes from input images. Kernels (filters) are

applied over input images with specified "stride(s)," resulting in a numeric matrix for the output volume size. The striding process reduces the output image dimension, necessitating padding to preserve the size of input images with low-level features by zero-padding the input volume.

The mathematical process of the convolution layer involves the application of filters over input images, utilizing striding, and necessitating padding to maintain input image size and capture low-level features effectively.

$$F(i,j) = (I * K)(i,j) = \sum I(i+m, j+n)K(m,n) \quad (5.1)$$

In equation (5.1), where 'k' represents the kernel with dimensions m x n, and 'F' signifies the output feature map. The notation $I * K$ denotes the operation between the convolution layer and the kernel. To introduce non-linearity in the output of CNN feature maps, the Rectified Linear Unit (ReLU) [172] layer is frequently applied.

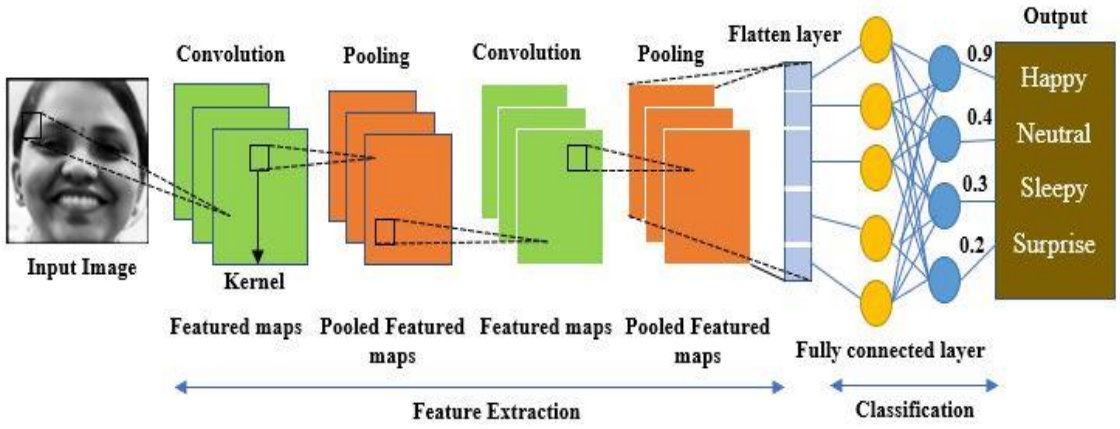


Figure 5.2. Basic Architecture and layers of CNN network

The pooling layer reduces the dimensionality of incoming data, aiming to minimize the feature map dimensions. This reduction not only facilitates more efficient feature learning but also decreases computation time for each block of the convolution operation. A commonly employed technique is max pooling, which selects the maximum value within a given input area. Following the pooling layer, a purposeful dropout layer [173] is introduced to generalize the network, preventing overfitting. Finally, the fully connected layer operates as a classifier, rendering judgments based on fundamental information gleaned from the convolution and pooling layers.

5.3.2 Long-Short-Term Memory

Long Short-Term Memory is an enhanced iteration of the RNN algorithm, specifically addressing challenges associated with learning high-dimensional data. While conventional RNN architectures yield promising results for shorter image sequences, they encounter performance issues with longer sequences due to problems like vanishing/exploding gradients [174]. In contrast to standard RNN units, LSTM introduces a memory block to mitigate these challenges. This innovation allows for the retention and forgetting of past information over extended periods. The fundamental structure of LSTM, as illustrated in Figure 5.3, comprises memory units and three control gates: the forget gate, input gate, and the output gate. Here, x_t denotes the current input, and C_t and C_{t-1} represent the new and previous cell states, respectively. Additionally, h_t and h_{t-1} correspond to the new and previous outputs [175].

To overcome issues with longer sequences, LSTM employs various gates that aid in retaining relevant information based on the network's dependencies. Figure 5.3 depicts the basic structure of an LSTM, showcasing its input gate principle. The input gate (i_t) plays a crucial role in storing and updating new information about the current state.

$$i_t = \sigma(W_i.[h_{t-1}.x_t] + b_i) \tag{5.2}$$

$$\hat{C}_t = tanh(W_C.[h_{t-1}.x_t] + b_C) \tag{5.3}$$

$$C_t = f_t * C_{t-1} + i_t * \hat{C}_t \tag{5.4}$$

In the LSTM architecture, equation (5.2) involves passing h_{t-1} and x_t through a sigmoid activation layer to determine the relevant portion of the data to be added. Subsequently, in equation (5.3), the tanh layer is employed to obtain new knowledge after processing h_{t-1} and x_t. The transmission of information within LSTM units is influenced by the sigmoid function and the dot product, yielding values between 0 and 1. If the sigmoid's value is 1, information is deemed necessary to be transferred; otherwise, it may be withheld. The long-term memory information $\hat{C}_t$ and current moment information C_{t-1} are combined in equation (5.4), where W_C represents the sigmoid output and $\hat{C}_t$ denotes the tanh output. Additionally, W_C signifies the weight matrix and the input gate bias of the LSTM is denoted by b_i. The forget gate (f_t) comes into play in equation (5.5), enabling the system to decide

whether to retain or discard previous information based on network dependencies. Here, W_f represents the weight matrix, and b_f serves as the offset.

$$f_t = \sigma\left(W_f . [h_{t-1} . x_t] + b_f\right) \tag{5.5}$$

The outcomes are produced through the output gate (O_t). Employing equation (5.6), this gate discerns the states that need to be sustained by the inputs h_{t-1} and x_t (equation 5.7). The ultimate values are obtained by transmitting the new information, C_t, through the tanh layer using a state decision vector.

$$O_t = \sigma(W_O . [h_{t-1} . x_t] + b_O) \tag{5.6}$$

$$h_t = O_t * tanh(C_t) \tag{5.7}$$

where b_O and W_O are the weighted matrix and LSTM bias output gates, respectively.

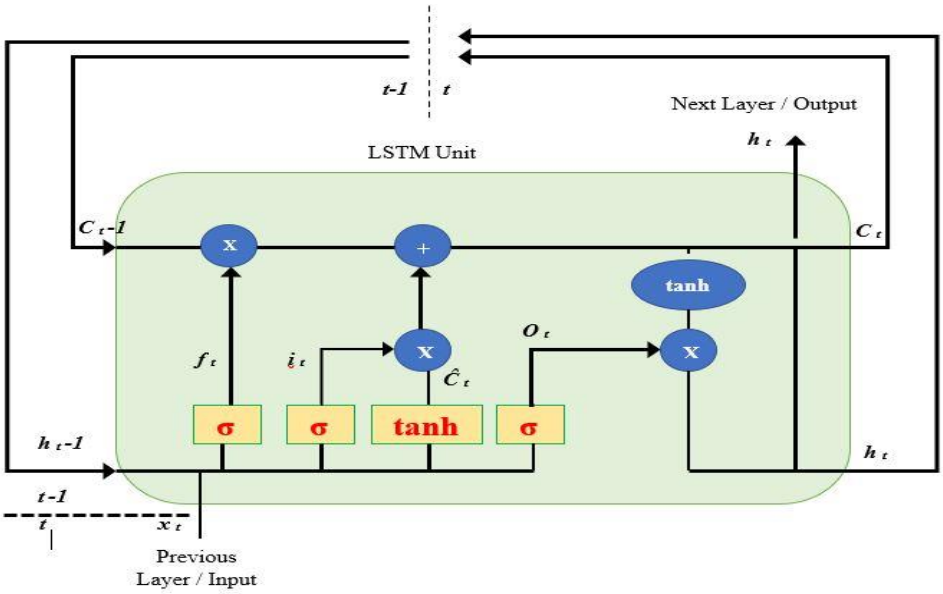


Figure 5.3. Basic structure of LSTM Network

5.3.3 Fusion of CNN-BiLSTM Model

Following this study, the integration of CNN and LSTM proves to be viable for extracting spatiotemporal features from a sequence of facial images. The subsequent step involves crafting a combined CNN-BiLSTM approach to enhance efficiency and fortify the network. To circumvent the intricacies and sensitivities associated with conventional feature extraction methods, a CNN with a time-distributed layer is employed to extract a sequence of

spatial-temporal features from corresponding labels based on the frame sequence. Subsequently, the extracted temporal dependencies of the feature vector sequence is input into the devised Bi-LSTM model, which incorporates both forward and backward passes to extract spatiotemporal features. The function of LSTM lies in acquiring information about cells preceding a given cell, yet it lacks access to data from the preceding cell. Consequently, the Bi-LSTM model is better suited for processing time series data. Ultimately, a softmax layer is utilized to classify each emotion.

$$TD\left(ReLU\left(Conv2D(X)\right)\right) \tag{5.8}$$

Equation (8) represents a TimeDistributed (TD) layer applied to the output of a convolutional layer (Conv2D) followed by a Rectified Linear Unit (ReLU) activation function.

$$Y_{enhanced} = TD\left(Enhanced\left(ReLU\left(Conv2D(X)\right)\right)\right) \tag{5.9}$$

Equation (5.9) represents TimeDistributed layer applied to the output of an enhanced convolutional layer (Enhanced), which includes ReLU activation, applied to the input X, which extracts spatiotemporal features. Also, Equation (5.10) describes the integration of a BiLSTM layer with the output for capture temporal dependencies across sequence of images of both forward and backward directions.

$$Y_{final} = BiLSTM(Y_{enhanced}) \tag{5.10}$$

$$ELU(x) = \begin{cases} x, & x > 0, \\ \alpha(e^x - 1) & x < 0. \end{cases} \tag{5.11}$$

Equation (5.11) denotes the ELU activation function; it has been used avoid dying problem in ReLU activation for improving training process and prediction results in this architecture.

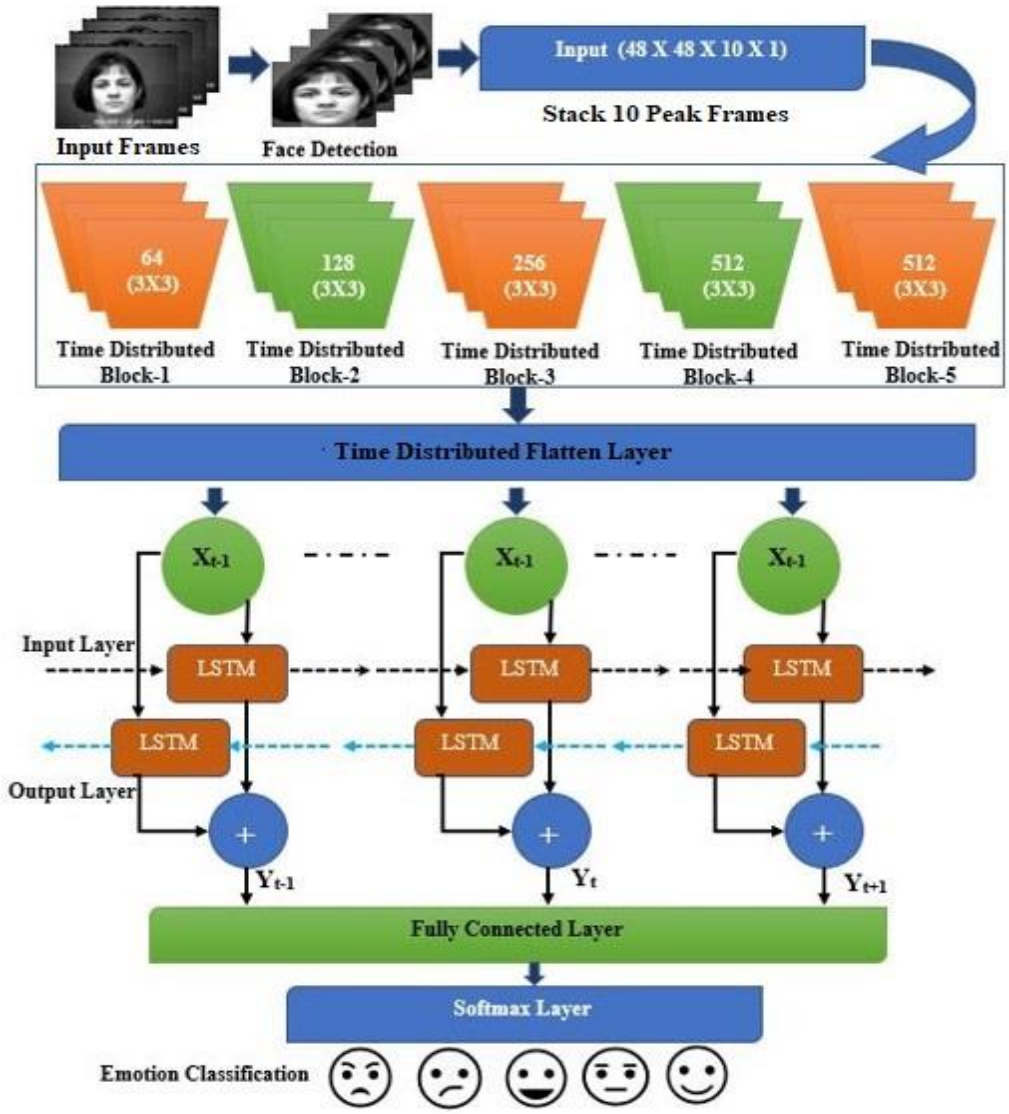


Figure 5.4. Proposed CNN – BiLSTM Architecture

Figure 5.4 illustrates the proposed CNN-BiLSTM network. Initially, to enhance network efficiency, the VGG-19 architecture is employed with an increased number of layers for extracting spatial features from a sequence of images. However, VGG-19's deep structure and feature extraction capability led to reduced facial expression recognition and overfitting issues, exacerbated by the layer's size and the vanishing gradient problem. Subsequently, fine-tuning is performed based on the dataset and image sequence dimensions. The proposed network comprises 21 layers, including 8 TimeDistributed layers, 2 dropout layers, 2 batch normalization layers, 1 flatten layer, 4 pooling layers, 1 FC layer, and 2 LSTM layers with a softmax function for categorical classification. Each convolution block is followed by either a

dropout or batch normalization layer. A 3 x 3 kernel convolutional layer with Elu activation function is utilized for feature extraction. A 2 x 2 max-pooling layer reduces the input dimension size. Batch normalization is employed instead of dropout to normalize the activation map output, further enhancing network performance. The feature map vector is then directed to the BiLSTM layer, which combines forward and backward passes to extract temporal information. The first LSTM calculates the sequence of features from the current input points, while the second LSTM reads and adds reverse sequence features. This bidirectional approach improves spatiotemporal feature extraction by enhancing interaction between neurons in both states. The input shape for the LSTM layer is (4691903), and Table 5.1 provides a summary of the proposed techniques. The softmax layer utilizes labels and learned feature maps to classify and predict each expression efficiently. Softmax, with its non-linear function for multi-class classification, enhances model generality as shown in equation (5.12). To address overfitting issues, regularization techniques such as dropout, early stopping, and kernel_initializer are applied.

$$P(x) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \tag{5.12}$$

5.3.4 Parameter Setting

The overall framework of the proposed system, depicting various stages of FER, is illustrated in Figure 5.5. The pre-processing pipeline operates on the raw sequence of frames from facial images. Within this pipeline, operations such as histogram equalization, bilateral filtering, flipping, rotating, and normalization are executed to enhance features and augment the dataset size. The pre-processed dataset is divided into three segments: training, validation, and testing, each containing distinct samples. Subsequently, the CNN-BiLSTM network is trained and applied to extract spatiotemporal features from the dynamic sequence for classifying facial expressions. Hold-out cross-validation techniques are employed to calculate the model's accuracy and loss for each epoch. The proposed approach's performance is evaluated using various metrics, including the confusion matrix, accuracy, precision, recall, and F1-score. Additionally, the proposed model is benchmarked against state-of-the-art techniques to assess its effectiveness.

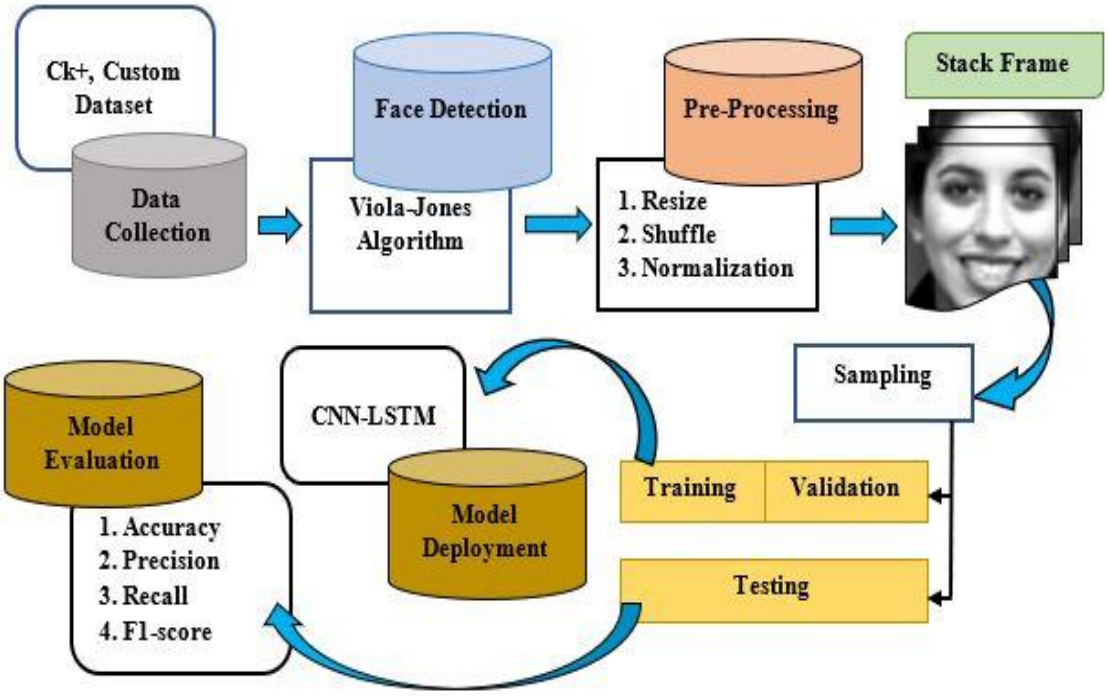


Figure 5.5. Overall design of the proposed CNN-BiLSTM FER system

Tables 5.1 and 5.2 present the parameter settings for the proposed CNN with the Bi-LSTM network. The dataset is partitioned into a training set and a test set in an 8:2 ratio to extract features from the sequence of images. During each training round, essential model parameters such as batch size, input size, hidden units, and dropout are closely monitored. The effectiveness of facial expression recognition is notably impacted by the number of hidden units; hence, early stopping, batch normalization [176] and dropout are employed to prevent overfitting and enhance the generalization of the proposed network. Given the constrained computing capacity, the proposed model undergoes training with 100 iterations and a batch size of 10. Additionally, the Adam optimizer [177] incorporating features from the AdaGrad and RMSProp algorithms to address sparse gradient and noise challenges, is chosen to train the network, with a learning rate set at 0.001. These parameters collectively contribute to optimizing the performance of the network.

Table 5.1 Summary of CNN-BiLSTM network layers

#	Type	Kernel Size	Stride	Kernel/size	kernel_initializer	Parameter
	Input layer					10x48x48x1
1	TimeDistributed	3 x 3	1	64	he_normal	640
2	TimeDistributed	3 x 3	1	64	he_normal	36926
3	Pool	2 x 2	2	-		0
4	Dropout	-	-	0.45		0
5	TimeDistributed	3 x 3	1	128	he_normal	73856
6	TimeDistributed	3 x 3	1	128	he_normal	147584
7	Pool	2 x 2	2	-		0
8	BatchNormalization	-	-	-		128
9	TimeDistributed	3 x 3	1	256	he_normal	2195168
10	TimeDistributed	3 x 3	1	256	he_normal	590048
11	Pool	2 x 2	2	-		0
12	Dropout	-	-	0.45		0
13	TimeDistributed	3 x 3	1	512	he_normal	1180160
14	TimeDistributed	3 x 3	1	512	he_normal	2359808
15	Pool	2 x 2	2	-		0
16	BatchNormalization	-	-	-		2048
17	TimeDistributed	-	-	-		4691903
18	LSTM	-	-	512		656384
19	LSTM			64		164352
20	FC	-	-	128		16512
21	Output	-	-	5		645

Table 5.2 Optimized parameter for proposed Method

Parameter Name	Value
Input Size	48 x 48 x 10 x 1
Activation Function	Elu
Learning Rate	0.001
Batch-Size	10
Dropout Rate	0.4
Iteration	100
Optimizer	Adam
Early Stopping	10

5.4 Dataset Preparation and Preprocessing

5.4.1 Dataset Description

5.4.1.1 Benchmark Dataset

The CK+ [178] dataset consists of 593 image sequences featuring 123 different individuals. Among these images, 327 have been labeled to represent seven facial expressions: anger, happiness, sadness, surprise, fear, contempt, and disgust. Figure 5.6 illustrates sample facial expressions from the CK+ dataset. Each expression in this database begins with a neutral state and culminates in a peak expression. For this study, the last three frames were selected to incorporate spatiotemporal features. Out of the seven emotions, only five are considered in this research, with less emphasis on contempt and disgust for training and testing purposes. The training set comprises 80% of the data, while the remaining 20% is allocated for testing, where peak images are validated for the evaluation process.

Figure 5.6. Sample CK+ facial expression images [178]: (a) Surprise, (b)Disgust, (c) Happy

5.4.1.2 In-house dataset

For the analysis, the proposed system uses a sequence of facial expressions captured in-house within a lab setting to assess spatiotemporal features. In this context, Raspberry Pi and Red Green Blue (RGB) camera modules are employed to capture real-time, spontaneous facial expressions from subjects in an unconstrained environment. The Raspberry Pi, a credit card-sized computer typically connected to a TV or monitor, utilizes its camera port for image and video processing in the field of computer vision. Specifically, a 5MP camera with Raspberry Pi 3/4 Model B is utilized for frame capturing in these experiments.

The primary objective of developing this dataset is to differentiate subjects' emotions during learning, encompassing happiness, surprise, sleepiness, and neutrality. These emotions are recorded in a controlled laboratory setting, allowing for free hand and head movements. Before the study, subjects provide consent through a consent form, and after recording their facial expressions, they are asked to make subjective judgments about their feelings—a technique often referred to as self-annotation.

This in-house dataset comprises 40 female subjects aged 21 to 26 years, exhibiting variations in lighting, occlusion, and positions. The dataset includes a sequence of facial expressions from individuals who have given consent for research purposes. The experimental setup is depicted in Figure 5.7, and the entire process occurs in the Centre for Machine Learning and Intelligence laboratory.

Facial expression acquisition poses challenges, and the frame sequence proves more useful for emotion recognition compared to static facial expressions lacking temporal information. In a total of 1600 image sequences, four emotions are captured: happiness representing pride and delight during the study, neutrality indicating an inactive state of learning, sleepiness signaling low energy during learning, and surprise denoting an extreme sense of enjoyment. Each clip spans 10-15 seconds, with a frame rate of 10 frames per second. Additionally, the frames encompass three peak levels. The algorithm for the facial expression dataset is compiled using the following steps.

- **Step 1:** Initiate the Raspberry Camera to capture the subject's spontaneous facial expression.

- **Step 2:** Employ Viola-Jones with PSO face detection algorithms for identifying the subject's facial expression. This method incorporates four techniques: Haar features, Integral images, Ada-boost, and Cascade classifier. Haar features extract facial information through line, edge, and four rectangular kernels. Integral images expedite the Haar feature extraction process, and an Ada-boosting classifier constructs a robust feature for face and non-face detection in frames. Finally, a Cascade classifier is applied to eliminate non-face regions from the video frame.

- **Step 3:** Preprocess the captured video frames by converting them to grayscale and scaling them to 48 × 48 pixels. During this image preprocessing, the probability density function of the resized frames is determined.

$$P(G_M) = \frac{N_M}{N} \quad (5.13)$$

In the equation, G_M represents the quantity of greyscale video frames for a specific emotion, N_M indicates the number of frames occurring, and N is the total pixel count in a single frame.

- **Step 4:** Conclusively, ten video frames for each of the four emotions are stored in the designated folder.

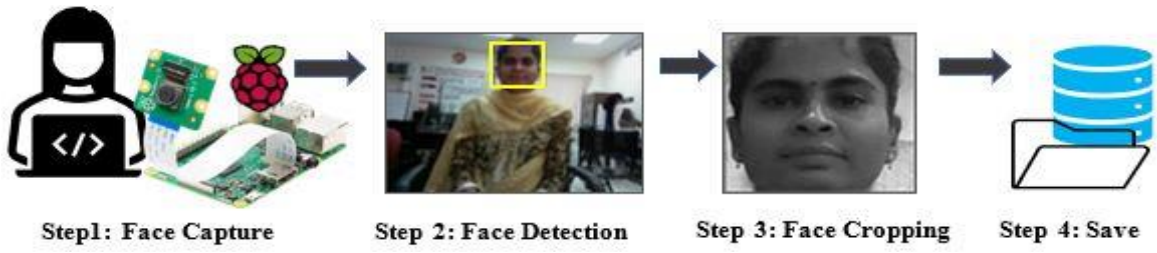
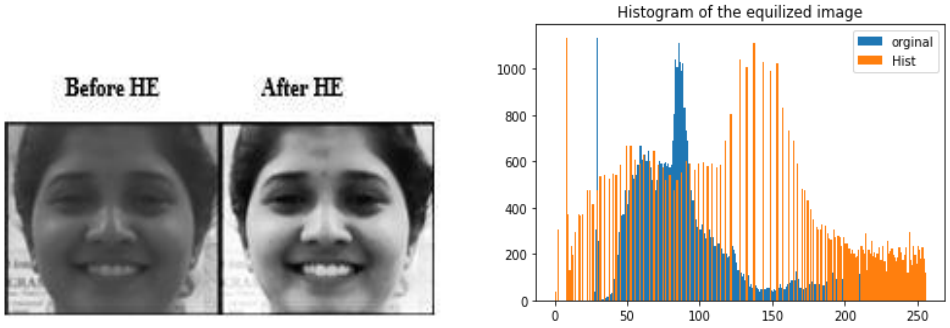


Figure 5.7 Framework for Facial Expression Dataset Collection using IoT Kit

5.4.2 Pre-processing

The acquired frames capturing facial expressions exhibit diverse lighting conditions and image noise. Furthermore, grayscale images are preferred over color scales due to their reduced information on facial expressions. As a result, employing RGB photos becomes unnecessary. To enhance visual contrast, Histogram Equalization [179] is applied across the grayscale video frames, as illustrated in Figure 5.8 (a). The original images are represented in blue, while the results of post-histogram Equalization are in orange. To eliminate noise from the video frames, a Bilateral Filter [180] is employed. This non-linear, edge-preserving, and noise-reducing image-smoothing filter proves more effective in noise reduction compared to alternatives like the median filter and Gaussian blur filter. The outcomes of the Bilateral filter are depicted in Figure. 5.8 (a), showcasing the final facial expression after completing all processing steps. Ultimately, each pixel is divided by 255 for normalization.



(a) Histogram equalization

(b) In-house database

Figure 5.8. Collected emotion databases are happy (row 1), neutral (row 2), sleepy (row 3), surprise (row 4)

5.4.3 Data Augmentation

Data augmentation serves as an effective technique in deep learning to enhance dataset volume, consequently boosting model performance. Recent research in image processing, particularly in deep learning-based augmentation, has focused on addressing overfitting concerns arising from limited training data while creating diverse real-world scenarios. Facial images were initially captured with a straight neck and an upward direction as shown in Figure. 5.9, are considered by the FER system, acknowledging that facial positions may vary in real-time based on camera positioning or an individual's posture. Various augmentation techniques, including flipping, rotation, color space manipulation, cropping, translation, and noise injection, are available. The application of these techniques is dataset-specific, lacking universal rules. For this study, the training dataset undergoes a specific augmentation process. Firstly, video frames of left and right faces are horizontally

flipped. Subsequently, each emotion's video frame undergoes rotation from -20 degrees to +20 degrees, a safe range for facial images, followed by horizontal flipping of the rotated frames [181]. The augmentation process involves generating synthetic frames through rotation and horizontal flipping, effectively expanding the data size and enhancing the proposed model's effectiveness. The outcomes of these enhancement procedures are depicted in Figure 5.9.

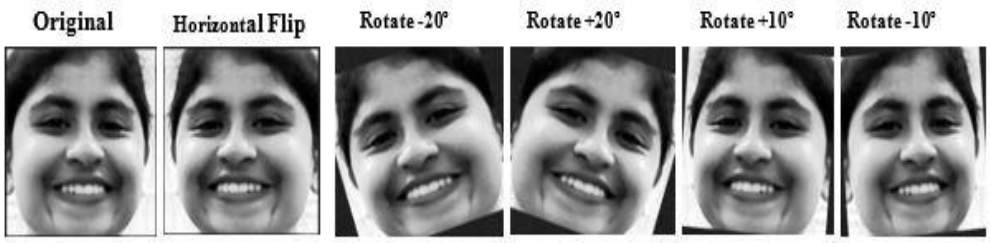


Figure 5.9. After Data Augmentation

5.5 Experimental Results and Analysis

5.5.1 Experimental Setting

The FER datasets are divided into training (80%) and testing (20%) sets for this experiment. Performance data is obtained using the hold-out cross-validation method. The utilized network, as detailed in Table 5.1, comprises nine TimeDistributed convolution layers with learning rates set at 0.001 and trained for 100 epochs. To address overfitting, an early stopping method is implemented. The evaluation involves CNN, LSTM, CNN with LSTM networks, and the proposed networks, conducted on Google Colab environments using Python and the Keras package with TensorFlow backends. Additionally, the experiments utilize graphics processing units (GPUs) installed in a Dell desktop featuring an Intel(R) Core (TM) i5-8400 CPU @ 2.80GHz 2.81 GHz.

5.5.2 Performance Metrics

The proposed system's effectiveness is assessed using the following performance metrics. True Positive (TP) represents accurately predicted emotions, False Positive (FP) indicates misclassified classes, True Negative (TN) signifies correctly recognized emotions, and False Negative (FN) denotes misclassified emotions by the FER system. The accuracy, precision, recall, and F1-measure are calculated using the following equations:

$$Accuracy = (TP + TN)/(FP + FN + TP + TN) \quad (5.14)$$

$$Precision = \ TP/(TP + FP) \quad (5.15)$$

$$Recall = \ TP/(TP + FN) \quad (5.16)$$

$$F1 - Measure = 2\ X\ (Precision\ X\ Recall)/(Precision + Recall) \quad (5.17)$$

In this context, accuracy is characterized as the proportion of correctly classified test frames to the total frames. Precision is described as the ratio of correctly identified frames for a specific emotion in the test set to the total correct frames in the emotion recognition results. Recall represents the ratio of correctly identified frames for a particular emotion to the total number of frames in that emotion. The F1 score, in turn, offers an average measure of accuracy and recall for the proposed system.

5.5.3 Experimental Results

Figures 5.10 and 5.11 illustrate the accuracy and cross-entropy (loss) performance assessments of the proposed model on the CK+ and In-House datasets throughout the training and testing phases. For the CK+ dataset, at epoch 100, the training and testing accuracies stand at 0.91% and 0.84%, respectively. Similarly, on the In-House dataset, the training and testing accuracies are 0.99% and 0.92% at epoch 50. Notably, the initial stage of training exhibits a slower convergence speed, followed by a more accelerated convergence around epochs 90–100 (refer to Figure 5.10. (a)), underscoring the model's satisfactory learning efficiency. The training curve demonstrates a reduced slope during model training. As shown in Figure. 5.11. (a), training accuracy gradually increases with slight fluctuations. Moreover, the CK+ and In-House datasets register training and testing losses of 0.84 and 0.61, respectively.

Figure 5.12 depicts the confusion matrix of the proposed model on both datasets. In Figure 5.12(a), showcasing the CK+ dataset's performance, among 750 image sequences, 21 were misclassified due to facial appearance similarities across five emotions. Happy and surprise exhibited superior performance, benefiting from facial features and time series information, while anger, sadness, and fear were occasionally misclassified. For instance, anger was falsely classified as sad, sad as happy or angry, and fear as sad, angry, or happy. Precision, recall, and F1 scores for the four emotions ranged from 0.83 to 1.0, with an average precision and recall of 0.92 and 0.91, respectively.

In Figure 5.12(b), the confusion matrix for the In-House dataset illustrates that happy and sleepy achieved higher performance compared to neutral and surprise. Notably, neutral overlapped with sleepy expressions, and surprise slightly overlapped with neutral expressions. Precision, recall, and F1 scores for the five emotions varied from 0.38 to 1.0, with an average precision and recall of 0.85 and 0.84, respectively. Given the inherent challenges of facial expression recognition, including similarities in facial features and individual variations of the same expression, the proposed CNN with BiLSTM demonstrates notable results. It shows reliable true positive and true negative values, fewer false negatives and false positives, and consistent performance in categorizing emotion sequences.

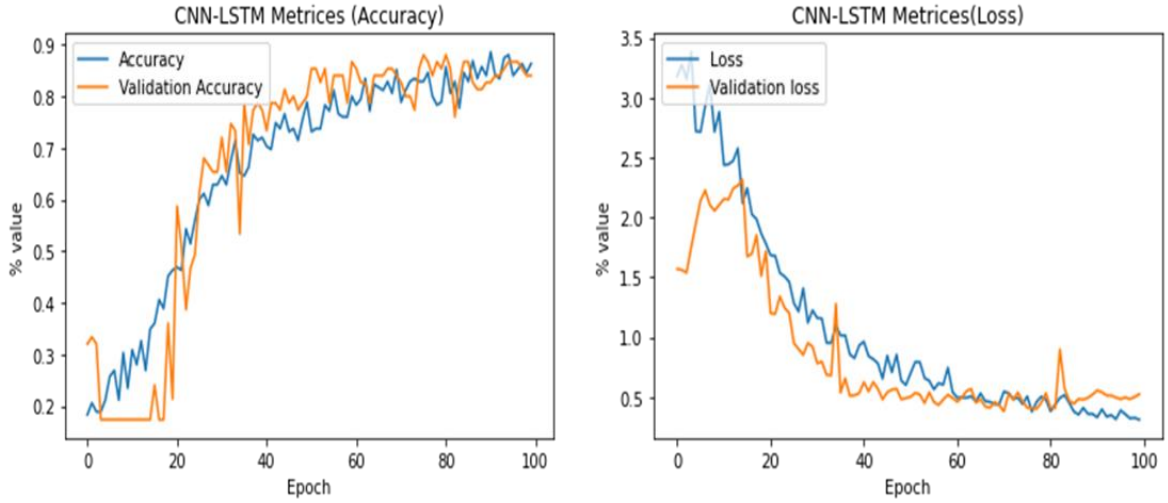


Figure 5.10. CNN-BiLSTM in CK+ dataset (a) Accuracy (b) Loss

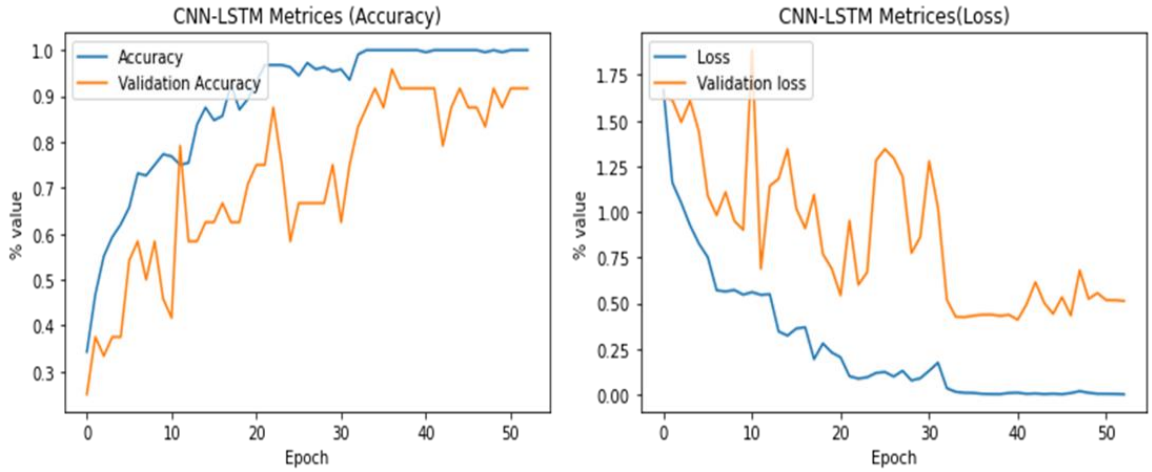


Figure 5. 11. CNN-BiLSTM in In-house dataset (a) Accuracy (b) Loss

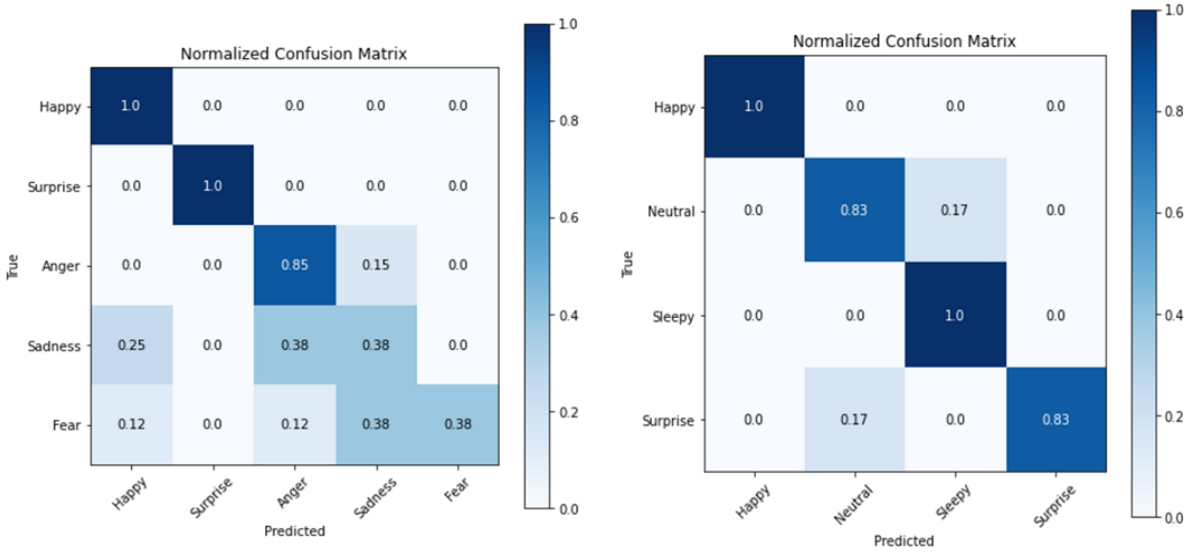


Figure 5.12. (a) Confusion matrix of CK+ (b) Confusion matrix of In-house dataset

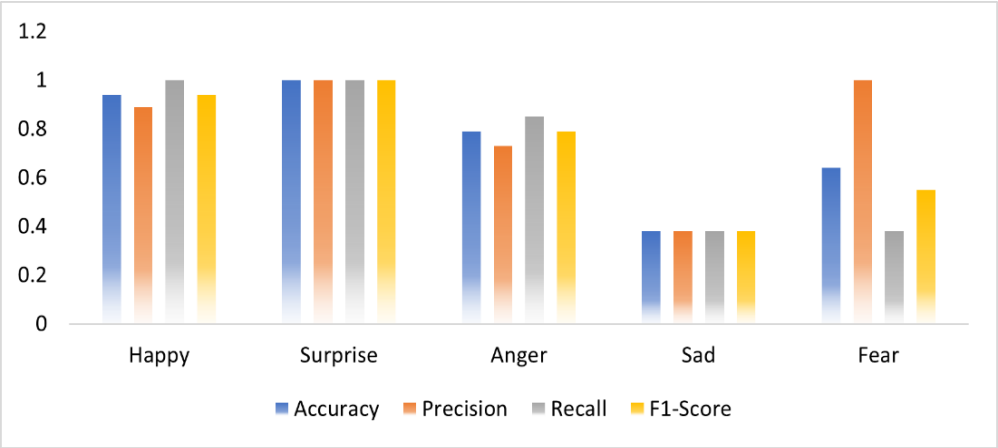


Figure 5.13. Recognition accuracy of the proposed model on CK+ dataset.

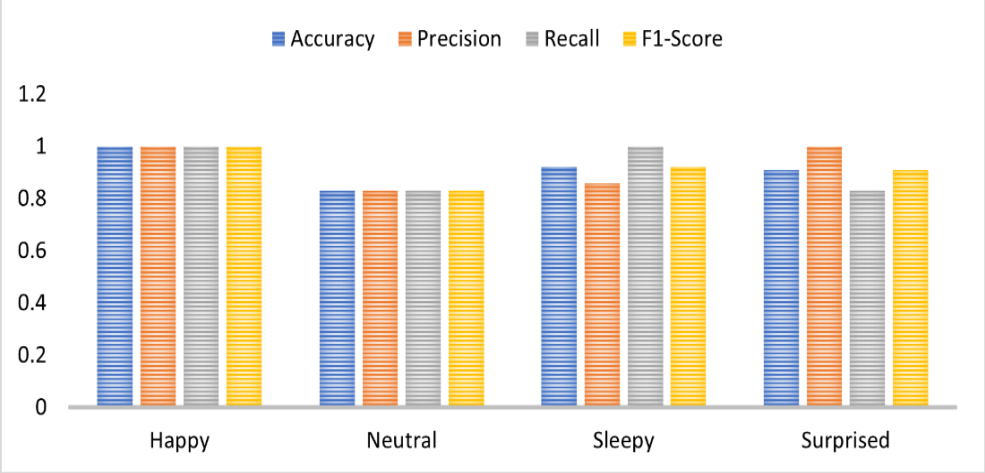


Figure 5.14. Recognition accuracy of the proposed model on In-house dataset.

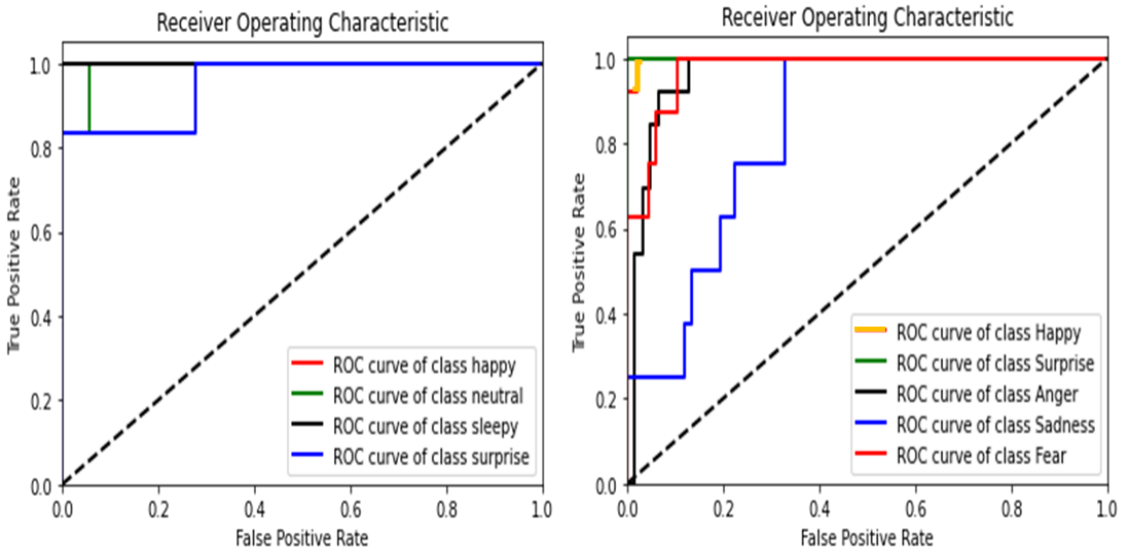


Figure 5.15. ROC curve of the proposed model (a) In-house dataset (b) CK+ dataset

Furthermore, Figures 5.13 and 5.14 present the accuracy, precision, recall, and F1-score of both datasets. Also, Figure 5.15 presents ROC curves, illustrating the relationship between the true positive rate and false positive rate to evaluate the overall performance. The ROC curve distinctly indicates that the suggested model achieves a performance of 0.92 on the In-House dataset and 0.84 on the CK+ dataset. The central aim of this research is to understand the connection between sequences of images depicting human facial expressions and their corresponding labels. Additionally, the objective is to extract spatiotemporal features utilizing a CNN architecture with Bi-LSTM.

In Figure. 5.15 (a), Happy and sleepy are accurately identified, while neutral and surprise exhibit slight misclassifications at 0.17%. In Figure. 5.15 (b), a higher classification rate is observed for happy, surprise, fear, and anger yielding satisfactory results, and sadness shows slightly lower results in the CK+ dataset. Furthermore, the primary goal of this research is to achieve satisfactory results in recognizing facial emotions from dynamic sequences of facial expressions. The experimental results demonstrate that the proposed method surpasses competitive state-of-the-art methods, as evidenced in Tables 5.4 and 5.5.

5.5.4 Comparison with baseline approaches and state-art-of-the-art techniques

To assess the performance and generalization capability of the proposed model, hold-out cross-validation techniques are employed to compare it against other baseline networks such as CNN, LSTM, and CNN-LSTM. A model with an identical structure to the proposed

one has been formulated and validated for this purpose. Table 5.3 provides a summary of the training parameters for each model. All models share the same learning rate, loss function, epoch count, batch size, optimizer, and activation function, which significantly impact model performance during training. The calculation of gradients is heavily influenced by the choice of loss function and optimizer, while the learning rate governs the adjustment of parameters concerning the gradient. Typically, the ReLU activation function is utilized in CNN and LSTM networks. However, in this study, the Elu activation function is employed instead of ReLU to mitigate the issue of dying ReLU. Moreover, Elu has shown superior performance compared to LeakyReLU. When compared to the other architectures listed in Table 5.3, the proposed model exhibits the lowest parameter count. Both the proposed model and all other models are trained and tested using the CK+ dataset and an In-house dataset, as depicted in Table 5.4.

Table 5.3 Parameter settings of each baseline model

Methods	Learning Rate	Epoch	Batch Size	Loss Function	Optimizers	Parameters
CNN	0.001	100	10	Cross entropy	Adam	76 M
LSTM	0.001	100	10	Cross entropy	Adam	57 M
CNN+LSTM	0.001	100	10	Cross entropy	Adam	49 M
Proposed Model	0.001	100	10	Cross entropy	Adam	46 M

Table 5.4 Comparison between different models on CK+ and In-house dataset

Methods	Accuracy CK+ (%)	Time	Accuracy In-house dataset (%)	Time
CNN	69.84	8 min	74.56	12 min
LSTM	50.78	10 min	79.38	16 min
CNN+LSTM	78.34	7 min	84.32	10 min
Proposed Model	84.87	5.5 min	92.84	7.5 min

The proposed model demonstrates a notable improvement in performance, escalating from 69.85% to 84.87% on the CK+ dataset and from 74.56% to 92.84% on the In-house dataset, owing to the incorporation of time series features. From Table 4, it is evident that the

CNN architecture exhibits subpar performance when considering time sequence information. Conversely, the CNN-BiLSTM model achieves slightly superior results compared to both the general CNN (69.84% on CK+, 74.56% on In-house) and LSTM (50.78% on CK+, 79.38% on In-house) networks, with scores of 78.34% on CK+ and 84.32% on the In-house dataset. Additionally, the computational time for each model is calculated, revealing that the proposed model requires less time for training compared to other baseline models. The size of the dataset and the complexity of image sequences inevitably influence both training time and accuracy.

Moreover, the proposed model is evaluated using current state-of-the-art methodologies, as detailed in Table 5.5. Analysis reveals that several existing systems attained slightly lower accuracy, ranging from 54.47% to 76.74%. Notably, moderate accuracies of 77.29%, 78.04%, 80.00%, 80.30%, 81.40%, and 82.68% were achieved in studies 33, 35, 5, 41, 35, and 49, respectively. Compared to these state-of-the-art methods, the proposed model demonstrates a 2.19% increase in accuracy, further affirming its effectiveness.

Table 5.5. Performance comparisons with the existing approach

Authors	Approach	Dataset	Accuracy (%)
Bilkhu et al., [182]	SVM+NN	CK+	80.00
Peter et al., [183]	PCA+KNN	CK+	77.29
Shan et al., [184]	CNN+KNN	CK+	80.30
		JAFFE	76.74
Buciu and Pitas., [185]	PCA + NMF + LNMF	CK+	81.40
Pranav et al., [186]	CNN	In-house	78.04
Wang and Yin, [187]	Topographic Context + LDA	CK+	82.68
Tan et al., [188]	HOSVD	CK+ / JAFFE	73.30
Sert and Aksoy,[189]	AAM + AUs + SVM	CK+	54.47
	Proposed Method	**CK+**	**84.87**
		In-house	**92.84**

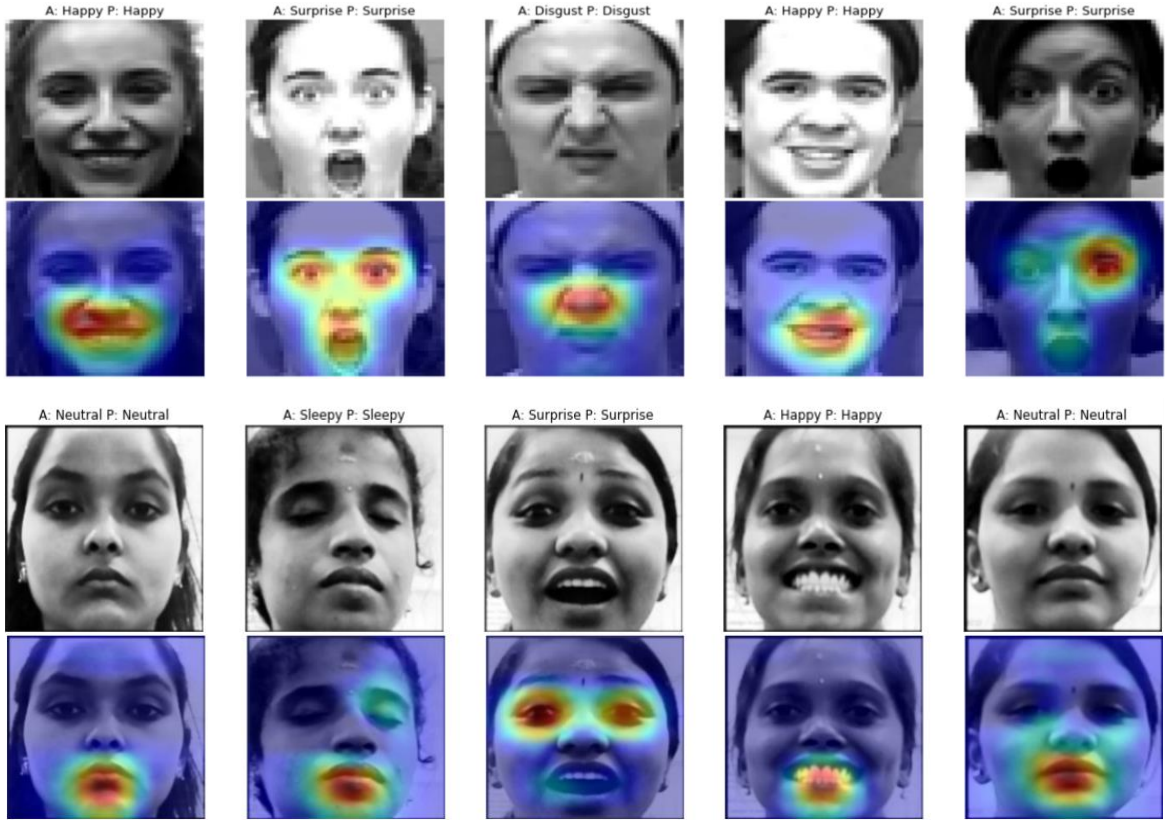


Figure 5.16 Grad-CAM visualizations of correctly identified FER samples: the first two rows correspond to the CK+ dataset, while the last two rows correspond to the In-house dataset

Figure 5.16 shows Grad-CAM (Gradient-weighted Class Activation Map) [190] visualization of the CNN-Bi-LSTM model's prediction results from the last CNN layer with correctly identified labels. Grad-CAM is a deep learning technique used to visualize and understand the decisions made by the CNN model. Grad-CAM computes the gradient of the predicted class score with respect to the feature maps of the last convolutional layer, determining the importance of each feature map for a specific class by analyzing these gradients. In Figure 5.16, the CK+ dataset on the top depicts the most highly concentrated features on facial expressions for emotion classification, such as the raised lip corners contributing to the classification of happy emotions, nose wrinkles contributing to disgust emotions, and open mouth with raised upper lids for surprise expressions. Similarly, the proposed model has been shown to predict sleepy expressions with closed eyelids using the in-house dataset.

5.6 Discussion

The proposed model has been juxtaposed with baseline models such as CNN, LSTM, and CNN with BiLSTM (VGG-19) network. The in-house datasets collected present various challenging conditions including illumination variations, partial occlusion, and noise. The hyperparameter-tuned CNN with LSTM exhibited efficient performance on this dataset as well as on the benchmark CK+ dataset. Before integration, this combined model underwent experimental evaluation on CNN and LSTM networks individually to extract spatiotemporal features from a sequence of images. Typically, facial expression datasets are compiled using high-quality cameras under controlled settings. However, in this study, an emotion dataset was captured using a low-quality 5MP camera module attached to a Raspberry Pi to analyze the FER system's performance with challenging images. Initially, a CNN was trained for FER but yielded unsatisfactory testing results as CNN struggles to extract spatiotemporal properties from image sequences. Despite its effectiveness with 2D images, it proved inadequate for this task. Subsequently, the LSTM model was employed to evaluate emotion recognition performance from feature vectors extracted from the image sequence. However, due to the massive size of the features when flattening the image sequence (48x48x3x1), LSTM couldn't effectively handle them, rendering its performance insufficient for developing a FER system model. Finally, features were extracted using the VGG-19 architecture and fed into the LSTM network. However, due to the number of layers, the FER model encountered the vanishing gradient problem. Consequently, this model was optimized through adjustments in neuron size, layer count, batch normalization, dropout, and learning rate.

Moreover, the proposed model undergoes examination with pre-processing, resulting in a boosted recognition rate of 0.92, whereas without pre-processing, the recognition rate drops to 0.80. When CNN and LSTM models are evaluated individually, performance suffers due to overfitting and misclassification, particularly with sleepy and neutral emotions. Similarly, the VGG-19 architecture model is utilized to extract spatiotemporal features from the sequence of images without adjusting layer size and activation functions before entering the LSTM network. Despite this, the combined network yields better results compared to general CNN and LSTM networks. However, hyperparameter-tuned CNN with BiLSTM models demonstrate improved recognition accuracy and classification outcomes compared to the VGG-19 architecture. Tables 5.4 and 5.5 depict the effectiveness of the proposed system

with different baseline architectures. Additionally, the proposed model marginally suffers in generalization during training due to an insufficient sample size. Moreover, it lacks the ability to recognize unseen emotions that were not part of the training data.

5.7 Chapter Summary

This chapter introduces fusion feature extraction techniques applied to a sequence of facial expression images. The study presents the following contributions: (i) A CNN-BiLSTM fusion model designed for analyzing sequences of facial expressions. (ii) Utilization of data augmentation techniques to generate varied illumination, noise, and facial image angles, enhancing the system's emotion recognition capabilities in real-time scenarios. (iii) Integration of a hyperparameter-tuned VGG-19 skeleton for extracting spatial features, addressing limitations of traditional feature methods. (iv) Implementation of a CNN-BiLSTM for extracting spatiotemporal features, enabling accurate emotion classification based on feature vector sequences. For this experiment, the in-house dataset was collected using a small 5MP camera attached to a Raspberry Pi, presenting challenges such as high illumination and noise. To mitigate these challenges, preprocessing techniques were employed before inputting data into the CNN-BiLSTM model. Furthermore, the evaluation of benchmark datasets showcased the proposed model's performance, achieving 0.84% and 0.92% accuracy on the in-house and CK+ datasets respectively. Additionally, the efficiency of baseline models on the FER dataset was compared with the proposed CNN-BiLSTM, which effectively learns the sequential relationships in facial expressions to identify emotions.