Chapter 6

Facial Expression Recognition Implementation with Deep Semi-supervised Convolutional Sparse Autoencoder

6.1 Introduction

It is clear from research on supervised FER techniques such as CNN, transfer learning, RNN, and LSTM, achieved state-of-the-art results on various FER datasets like CK+ [48], JAFFE [46], FER-2013 [52], and so on. In this context, the performance accuracy of the FER system model fully relies on a massive amount of labeled high-quality facial expression datasets. The problem with the existing FER datasets is that the total samples and expression counts on each emotion vary from dataset to dataset. This variation might be the reason for FER accuracy variations on different FER datasets. Additionally, combining different FER datasets into one needs careful consideration due to the ethnicity of the FER samples. Moreover, increasing the layer size while training FER with CNN is done to create an effective pattern for each expression on a smaller number of samples; however, this approach may lead to overfitting issues and poor classification results on some expressions. Nevertheless, it gives significant FER accuracy prediction results on training samples, even though it fails to perform well in a real-time environment. Thus, extracting more discriminative features from limited samples continues to be a valuable and challenging task.

6.1.1 Autoencoder

An autoencoder is a type of artificial neural network (ANN) that can learn a valuable feature representation either with or without supervision [191]. It employs a dimensionality reduction approach to decrease the dimension of features by eliminating fewer effective features to predict the target facial expression. The inclusion of irrelevant features can result in a reduction in the model's real-time performance. The autoencoder incorporates a bottleneck architecture, compressing input facial expression images into a concise form of feature representation [192]. It also utilizes techniques from unsupervised learning for tasks such as image compression, image denoising, anomaly detection, and feature representation [193].

The basic conceptual components of an autoencoder include three parts: (i) Encoder, (ii) Latent space / Code, and (iii) Decoder, as depicted in Figure 6.1. The encoder encodes the

input images using a hidden layer and outputs the best feature representation. It takes unlabeled variables for training, treating it as supervised learning to provide an output $\hat{x}$, which is reconstructed from the input images x. The level of information transmitted through the entire network is constrained in the latent space, leading to a learned compressed form of the input images to avoid redundancy.

The basic autoencoder of the encoder is defined as follows: $Code\ (c) = f(I) = \sigma(w.x + b), where\ w \in R^{mxn}, b \in R^n$, where I is an input function, σ is an activation function, w denotes weight, x denotes input facial expression, and b represents bias. The decoder reconstructs the output facial expression from the input image as defined as: $x' = f'(c) = \varphi(w'.c + b'), where\ w' \in R^{mxn}, b' \in R^n$. Here, the activation function σ same as φ. Finally, calculate the mean squared error of input and reconstructed output facial expression, where the cost function is represented as below equation (1):

$$Cost = min \sum_{i=1}^{n} |I - x'|^2 \quad (6.1)$$

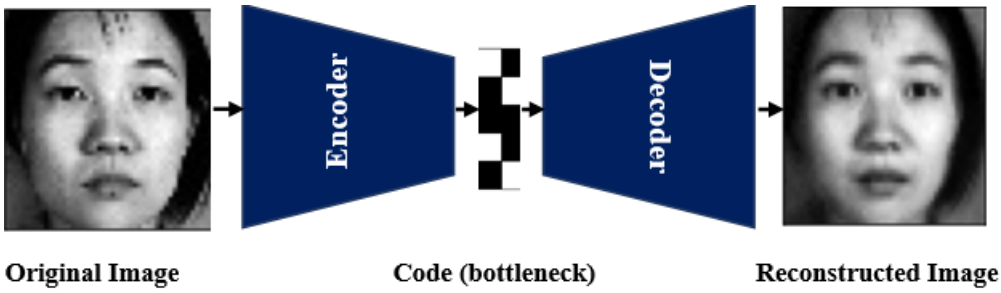


Figure 6.1 Skelton of Autoencoder

6.1.2 Type of Autoencoders

There are different kinds of autoencoders available such as

- **Vanilla Autoencoder:** This is the basic form of an autoencoder consisting of an encoder and a decoder. It aims to learn a compressed representation of the input data.
- **Sparse Autoencoder:** It introduces sparsity constraints during training, encouraging the model to learn a sparse representation of the input data, which can be useful for feature extraction.
- **Denoising Autoencoder:** Trained to reconstruct the original input from a corrupted version, denoising autoencoders are robust to noise and can learn meaningful features.

- **Variational Autoencoder (VAE):** VAEs learn a latent space representation of data and are generative models that can generate new samples similar to the training data. They are also used for tasks like data generation and anomaly detection.
- **Contractive Autoencoder**: It adds a regularization term to the loss function, penalizing the model for variations in the input space. This helps in learning a robust representation insensitive to small input variations.
- **Adversarial Autoencoder (AAE):** Combines autoencoders with adversarial networks, where the decoder tries to generate data that the discriminator cannot distinguish from real data.
- **Convolutional Autoencoder (CAE):** Designed for handling image data, CAEs use convolutional layers in the encoder and decoder to learn spatial hierarchies and patterns in images.
- **Stacked Autoencoder:** Consists of multiple layers of autoencoders stacked on top of each other, allowing the model to learn hierarchical representations of the input data.
- **Recurrent Autoencoder (RAE):** Utilizes RNNs in the encoder and decoder to process sequential data like time series or text.

Representation learning is an effective approach that can acquire higher-dimensional, robust feature representations from image samples. This method is unsupervised learning, extracting higher-level features from unlabeled input samples. The underlying idea of this approach is to extract higher-level features through unsupervised learning, while errors are calculated to assess the effectiveness of reconstructed output samples. In the next step, a classifier is incorporated and trained for classification tasks, as illustrated in Figure 6.2. The objective is to attain higher-level, efficient data representations using dimensionality reduction techniques. Building upon this, Mallick et al. [194] presented MRI image compression techniques utilizing a deep wavelet autoencoder, reducing the size of features and classifying them through deep neural networks. The authors achieved significant results with this approach compared to conventional supervised learning. Zhou et al. [195] introduced a semi-supervised stacked autoencoder feature and fine-tuned it for hyperspectral image classification. Similarly, Liang et al. [196] presented a stacked denoise autoencoder for remote sensing image classification, using a greedy layer-wise approach to train each layer for feature extraction and classification through backpropagation.

Furthermore, many researchers have attempted to use autoencoders in FER in different ways, as summarized by Mohana et al [197]. Some notable works, such as the one by Chatterjee et al., [198] proposed AFORET to overcome overlapping class problems in FER. The author employed a residual variational autoencoder to transform facial expressions into latent space, resulting in improved classifier performance. Lakshmi and Ponnusamy [199] presented a deep-stacked autoencoder for dimensionality feature reduction of facial expression samples and used an SVM for classification, incorporating conventional handcrafted feature extraction techniques such as HoG and LBP. However, these methods did not effectively extract features to enhance classification performance. Furthermore, researchers have directed their attention towards unsupervised techniques to enhance FER outcomes using unlabeled facial expression samples. This focus is due to the high cost associated with acquiring extensive labeled FER data, which poses challenges in expanding FER training datasets [200]. Additionally, the utilization of a GAN to generate new samples has been explored to address data imbalance issues in FER [200]. Consequently, a gap still exists in effectively developing a computationally efficient FER system with fewer samples and parameters as training large FER datasets with deep neural networks often requires a highly effective GPU processor.

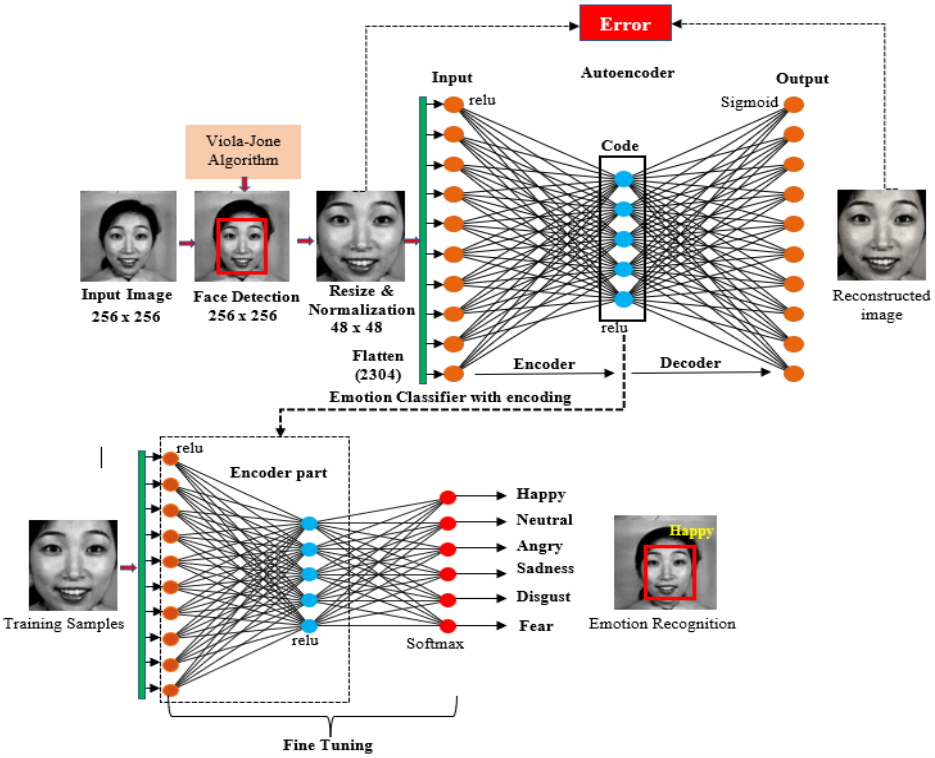


Figure 6.2 Overview of Autoencoders in FER

We introduced a new approach called a Deep Semi-supervised Convolutional Sparse Autoencoder (DSCSA) to enhance FER performance with fewer samples and designed with fewer parameters compared to conventional approaches. The face detection process utilizes the improved Viola-Jones algorithm with PSO[201], followed by normalization and resizing of pixel ranges for training purposes. The initially designed Deep Convolutional Sparse Autoencoder is trained with unlabeled FER samples. Sparsity is imposed on a hidden layer in the encoder to eliminate overfitting and irrelevant features. This mechanism leads to the successful reconstruction of the input as output. Subsequently, the SoftMax function is added on top of the encoder for training the model in a semi-supervised approach with label samples for classifying facial expressions. Thus, the techniques boost FER performance and increase the classification rate for each expression.

The summary of the primary contributions of the proposed autoencoder is as follows:

- Proposed an autoencoder that directly extracts facial features from raw images based on dimensionality reduction techniques, leveraging the robust power of convolutional kernels. Also, sparsity is introduced to eliminate redundant features while decoding.
- According to the literature review, this study is the first to employ FER with the proposed approach, aiming to enhance facial expression performance using fewer data samples. The results are compared with existing approaches.
- The unsupervised data-driven approach can be applied to extract features from various image classification datasets and across different application domains.
- The XAI methods like Grad-CAM and LIME Visualisation technique are used to explain the decision-making output process of DSCSA, which features significantly contributed to the classification of each facial expression.

The remaining section is organized as follows: Section 6.2 describes insight into the proposed methodology, dataset, pre-processing, and parameter setting. Section 6.3 explains the experimental setting, and results of DSCSA performance and comparison results with other state-of-the-art techniques in FER. Section 6.4 presents a thorough analysis and delves into the results, and finally, Section 6.5 summarises the proposed model performance and insights and future perspectives.

6.2 Proposed Methodology

This section describes the deep semi-supervised convolutional sparse autoencoder proposed design to focus on the most robust discriminate features to enhance FER

performance in a real-time environment. The model extracts spatial feature representations using a deep convolutional autoencoder, incorporating sparsity to eliminate redundant features during the facial feature reconstruction process carried out by the decoder. Subsequently, flattened layer and SoftMax functions are applied on top of the encoder to classify facial expressions by a semi-supervised approach.

6.2.1 Pre-processing

In this stage, the enhanced Viola-Jone with PSO [11] face detection approach is employed to detect faces in both JAFEE [202], CK+ [203] datasets, and the In-house dataset. The Viola-Jones algorithm comprises four steps, and the feature selection and threshold optimization in AdaBoost have been optimized using PSO [202]: (i) *Haar features* are used to extract facial feature structures using line, edge, and four rectangular kernels. (ii) *Integral images* help speed up the Haar feature calculation process. (iii) *AdaBoost* is then applied to create a robust classifier that distinguishes between face and non-face features. Finally, the (iv) *Cascade classifier* is used to remove non-face regions from images, enhancing the accuracy of face detection. The identified face regions, marked by rectangular boundaries, are then cropped, resized into 48 x 48 pixels, and stored in the appropriately labeled folder through automated programming. Additionally, histogram equalization [204] is employed to equalize the intensity levels of all facial expression images. Finally, min-max normalization is applied to the pre-processed FER image by dividing it by 255 to expedite the model learning process.

6.2.2 Deep Semi-Supervised Convolutional Spare Autoencoder (DSCSA)

Based on the superior performance of Convolutional Neural Networks on several image classification tasks, it was found to be efficient for extracting composite spatial features such as texture, shape, and edges [205][206][207]. In the DSCSA architecture, the Convolutional layer is integrated to extract spatial features from raw facial expressions, aiming for an efficient feature representation used for classification via the SoftMax layer, which assigns probability scores. This approach determines the number of parameters learned by the Convolutional layer in an unsupervised manner, incorporating sparsity to filter out irrelevant features in the latent space. Additionally, a small amount of labeled data samples has been provided for fine-tuning the entire DSCSA model for final emotion classification; this approach is called Semi-supervised learning [208][209].

The proposed model consists of numerous layers such as the Convolutional Layer, Batch Normalization, Max Pooling, and Upsampling layers, as shown in Figure 6.3. In this architecture, the output of one layer is connected to the input of the next layer, and error is calculated to determine the loss of reconstructed facial expressions before proceeding to the fine-tuning and classification process. During this process, two sets of data samples are included: one is unlabeled data $UD = \{x | x \in X\}$ and the other labelled dataset $LD = \{x, y | x \in X, y \in Y\}$. Specifically, the UD is used for training the convolutional sparse autoencoder while the LD is employed for supervised fine-tuning and training in facial expression classification.

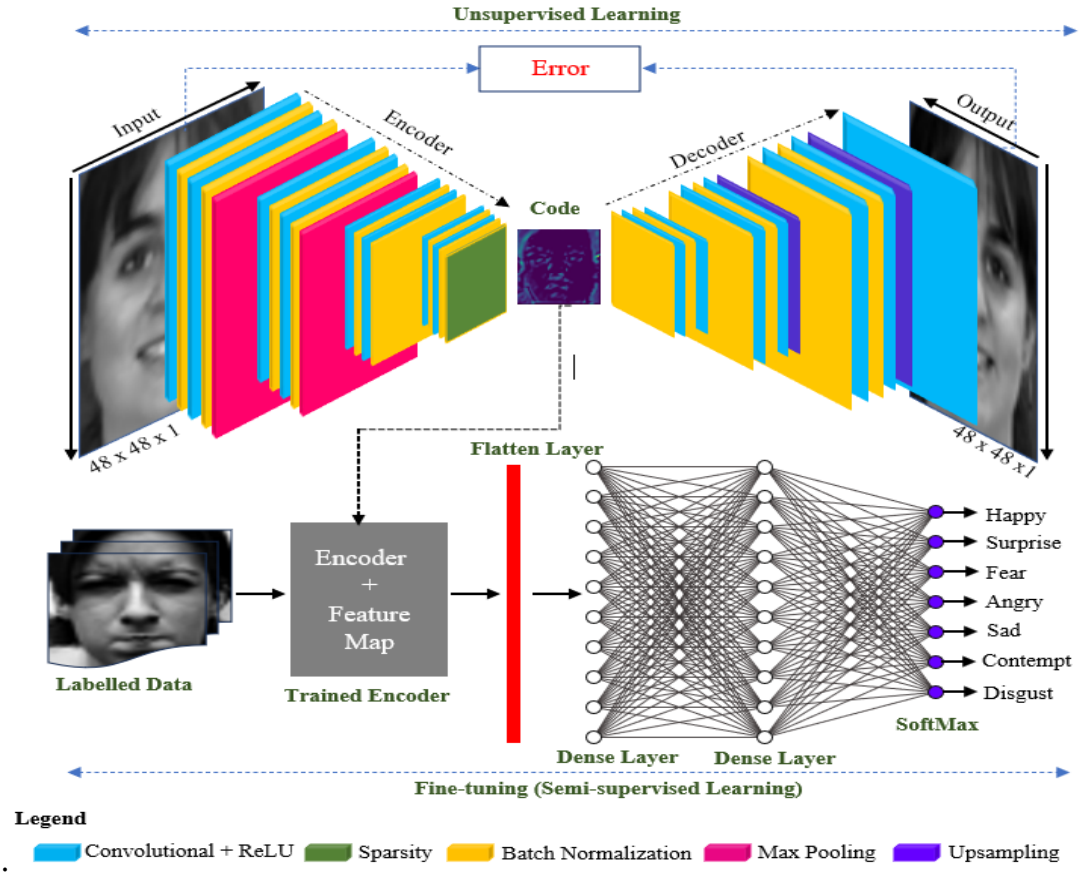


Figure 6.3. Deep Semi-Supervised Convolutional Sparse Autoencoder

The Convolutional Autoencoder is employed to compress facial expression features by combining local convolutional connections with an autoencoder. This approach simplifies the addition of reconstructed images for convolutional operations. The operation in this layer involves the linear multiplication of the filter mask and the input array images to produce a feature map. For example, let $f(x, y)$ represent the input facial expression, and $h(x, y)$ be the

filter mask. This operation is mathematically defined as $g(x,y) = f(x,y) * h(x,y)$. The convolutional layer outputs a feature map, and the process of input to output in this layer is referred to as a convolutional encoder. Subsequently, the output values are constructed through inverse convolutional operations, denoted as the convolutional decoder. Moreover, based on the standard autoencoder, the parameters of the encoder and decoder are calculated

Initially, the facial expression input images x_i (where i range from 1 to p) and the wight w_j, of the convolutional kernel j is used for convolutional operation. Finally, the neuron values of the encoder are calculated for the output layer as shown in equation (6.2).

$$o_{ij} = f(x_i) = \sigma\left(w_j . x_i + b\right) \tag{6.2}$$

In equation (3), σ is a non-linear activation function. The activation function used for the output of the convolutional layer is a Rectified Linear Unit (ReLu) [172], followed by 3 x 3 convolutional filter.

$$ReLU\ (x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0 \end{cases} \tag{6.3}$$

Subsequently, batch normalization [210] is employed after the convolutional layer to overcome gradient exploding and speed up training by normalizing feature maps. Following that, max pooling is applied to reduce the number of parameters, which helps to reduce network computational complexity, defined as follows:

$$o_j^i = max\left(x_j^i\right) \tag{6.4}$$

where, each of the feature maps is divided into n overlapping features based on the size of the pooling region. x_j^i denotes the i^{th} region of j^{th} feature map, and o_j^i represents the i^{th} neuron of j^{th} output feature map. Although the number of feature maps matches the number of output feature maps, the number of neurons per feature map decreases following the pooling operation.

In addition, a sparsity penalty is imposed in the dense layer of the encoder in the proposed architecture, addressing the issues of trivial identity mapping, and overfitting and focusing on more interesting patterns of facial expression. In this paper, Kullback-Leibler (KL)-divergence [211][212] regularization has been applied to resolve the aforementioned issues, which measure the probability distribution between neurons, when the output is close

to 1 means active otherwise inactive it means output is equal to 0, it allows only a limited number of neurons to be triggered at same time.

$$J_{CSA}(W,b) = J(W,b) + \beta \sum_{j=1}^{sl} KL\left(\rho \middle\| \hat{\rho}_j\right) \quad (6.5)$$

where, ρ represents the desired level of sparsity, $\hat{\rho}_j$ defines the average of the hidden unit, which is added to the pooling layer. The level of sparsity has been determined through a hyperparameter tuning method, which in this case is set to 3. Moreover, the convolutional decoder is an encoder from the output o_{ij}, that x_i is reconnected $\hat{x}_i$, as defined in equation (6.6).

$$\hat{x}_i = f'(o_{ij}) = \varphi(w'_i . o_{ij} + b') \quad (6.6)$$

$$L_{CSA}[x_i, \hat{x}_i] = \|x_i - \hat{x}_i\|^2 \quad (6.7)$$

Additionally, the reconstruction error is calculated using equation (6.7). The convolutional sparse autoencoder (CSA) utilizes the Adam optimizer with a learning rate of 0.001 to minimize and optimize this error. Following the learning process with unlabeled data, the trained encoder component progresses to the subsequent stage of emotion classification.

6.2.3 Emotion Classification

After training the Convolutional Sparse Autoencoder, the next step involves calculating the reconstruction error of compressed facial expression images and the autoencoder's loss using the Structural Similarity Index (SSIM) and mean squared error function respectively. Subsequently, the autoencoder undergoes fine-tuning using labeled samples.

In this phase, the model incorporates a flattened layer and a dense layer with a SoftMax classifier. The flattening layer follows the final pooling layer in the encoder segment, where feature learning occurs. This layer represents features via 128 neurons linked to the SoftMax layer. These parameters undergo supervision and training via a fully connected layer alongside the SoftMax classifier. Furthermore, the network training loss is optimized using the Adam optimizer [213].

This model incorporates the SoftMax layer [214] for facial experience classification and recognition, as illustrated in Figure 6.3 and shown in Equation (6.8). In this paper, more

than six facial expressions are considered in each FER dataset, $\hat{y}_i$ represents the probability score of different facial expressions based on the trained model's prediction.

$$\hat{y}_i = \frac{e^{(o_i)}}{\sum_{k=1}^{7} e^{(o_k)}}, i = 0,1 \dots 7 \quad (6.8)$$

$$Loss_{DSCSA} = \frac{1}{N}\sum_{i=1}^{N} y_i . \log(y_i') \quad (6.9)$$

Furthermore, in semi-supervised learning, categorical cross-entropy serves as the cost function $Loss_{DSCSA}$, as shown in equation (6.9). Here y is the labels of sampled data and N is the number of class labels based on different FER samples.

6.2.4 Parameter setting and Layer details of DSCSA

Figure 4 presents layer details of the DSCSA architecture for FER, including convolutional layers, batch normalization, max pooling, UpSampling layer, flatten layer, and fully connected layer with the SoftMax function. The CSA consists of seven blocks with a 3 x 3 filter, and neuron sizes are 32, 64, 128, 256, and 1, as shown in the encoder and decoder parts, also including an UpSampling layer with 2 x 2 filters. After training the CSA with unlabeled samples, the trained encoder part is connected with the flattened layer, a fully connected layer with 128 neurons, and fine-tuned with labeled samples. Finally, it is connected with a SoftMax layer for emotion classification.

Furthermore, Table 1 shows the optimized parameters and their values, which were finally chosen after a lot of trial and error with a hyperparameter tuning approach in Kears [215].

Table 6.1. Optimised parameters for DSCSA

Parameter Name	Value
Input size	48 x 48 x 1
Activation Function	ReLu, Sigmoid
Kernel Size	3 x 3
Learning Rate	0.001
Loss Function	MSE, Binary Cross Entropy, Categorical Cross Entropy
Batch Size	32
Epoch	100
Optimizer	Adam

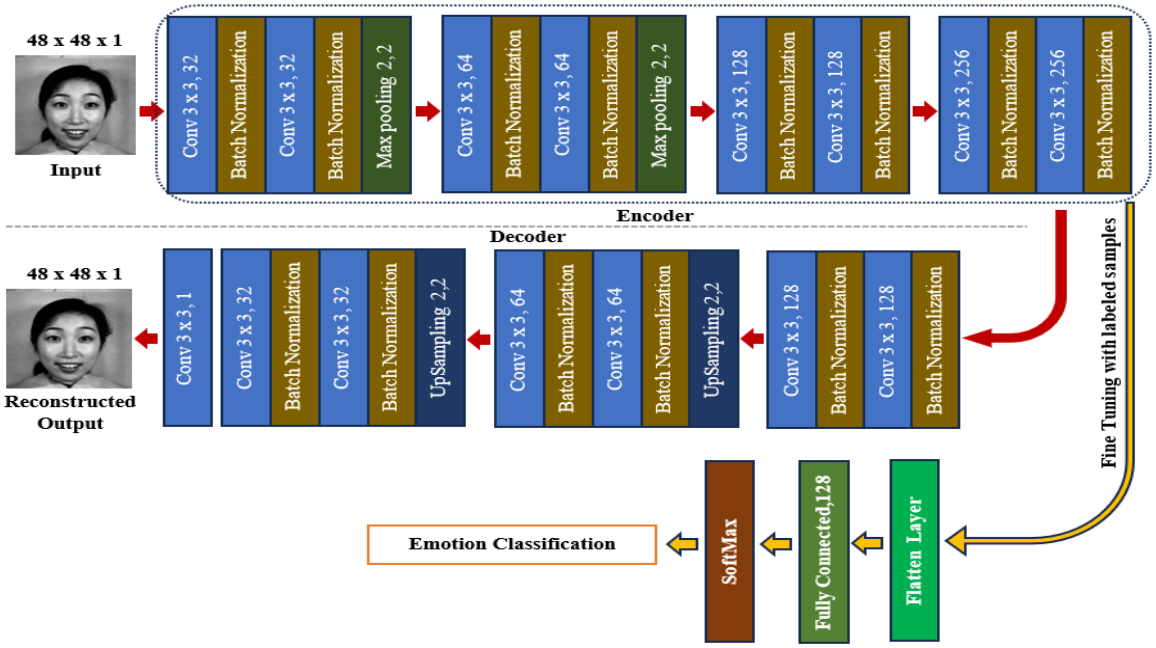


Figure 6.4. Network Architecture of DSCSA

6.3 Experimental Settings and Analysis

The experiment was conducted using the Keras framework with a TensorFlow backend [215] architecture and executed on an HP laptop running Windows 10 with a 64-bit OS. The laptop specifications include a 12th Gen Intel(R) Core (TM) i3-1215U processor at 1.20 GHz, 4GB RAM, and an Intel Core i3 processor, utilizing Jupyter Notebook for all experiments. For leveraging an external GPU, Google Colab with a T4 GPU processor was employed to train and analyze the results of all networks. Performance evaluation metrics such as accuracy, precision, recall, F1-Score, and Peak signal-to-noise ratio (PSNR) were computed to assess the model's effectiveness. Additionally, the dataset was divided into 80% training and 20% testing sets using hold-out cross-validation techniques across all datasets. A hyperparameter tuning mechanism was also implemented to identify the most optimized parameters for this experiment.

6.3.1 Dataset Description

Two publicly available standard FER datasets were used in this experiment to evaluate the model's performance as well as the In-house dataset as explained in the previous chapter. The first dataset is the CK+ [202] dataset, which includes eight expressions and a total of 593 video sequences from 123 subjects aged 18 to 50. The pixel resolution is 640 x 490, and it covers expressions like 276 happy, 180 angry, 112 sad, 332 surprised, 72

contempt, 236 disgust, 100 fear, and 327 neutral for evaluation. The second dataset is the JAFFE [203] dataset, which contains 213 facial expressions from 10 different female subjects. It has a resolution of 256 x 256 pixels and was validated by 60 Japanese subject experts. The dataset comprises seven expressions: 30 angry, 29 disgusted, 32 fearful, 31 happy, 30 neutral, 31 sad, and 30 surprised.

6.3.2 Performance Metrics

For this experiment, the DSCSA architecture has been evaluated using the following metrics: TP for True Positive, FN for False Negative, TN for True Negative, and FP for False Positive. The summary of these four metrics is referred to as the confusion matrix.

$$Accuracy = \frac{TP+FN}{TP+TN+FP+FN} \tag{6.10}$$

Accuracy is employed to assess the detection performance of the entire test set, evaluating positive instances as positive and negative instances as negative.

$$Precision = \frac{TP}{TP+FP} \tag{6.11}$$

Precision measures the proportion of true positive samples among the samples predicted as positive by the detection model.

$$Recall = \frac{TP}{TP+FN} \tag{6.12}$$

Recall is employed to measure the proportion of predicted positive cases out of all actual positive cases.

$$F1 - score = \frac{2*TP}{2*TP+FP+FN} \tag{6.13}$$

F1-Score is used to measure the harmonic mean of precision and recall rate and defines the discriminant ability of the designed model for each category.

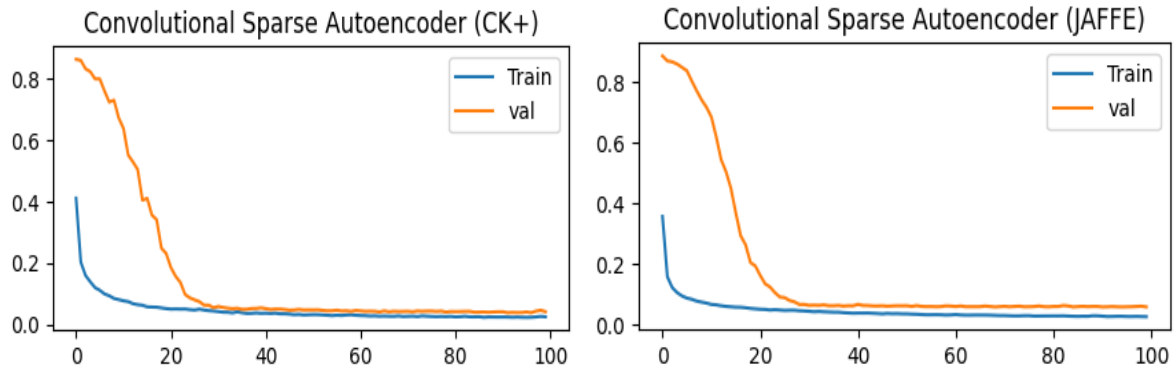
$$PSNR = 10\ log_{10}\ \frac{Max_I^2}{MSE} \tag{6.14}$$

The Mean Squared Error (MSE) is utilized to compute the autoencoder's reconstruction loss, while the PSNR is employed to compare the quality of the original and compressed reconstructed images.

6.3.3 Results and Analysis

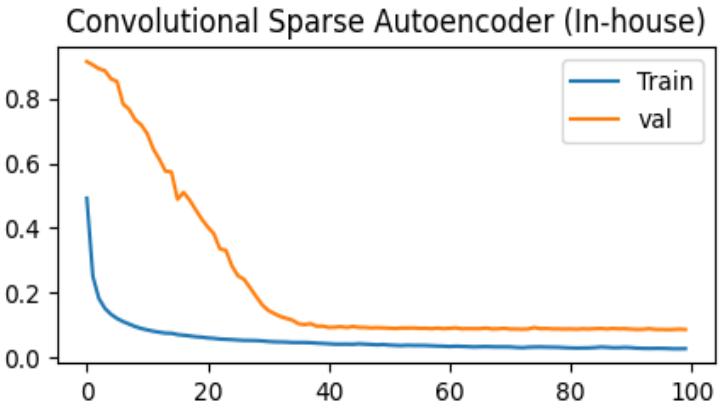
This section presents the experimental findings of DSCSA utilizing two distinct benchmark datasets. Initially, an unlabeled facial expression dataset underwent training with a convolutional sparse autoencoder to extract features effectively. The proposed model underwent training for 100 epochs with a batch size of 32. These hyperparameter values were selected after several tuning iterations to achieve better results and minimize the loss values. The loss of facial image reconstruction is calculated to validate the performance of the DSCSA autoencoder with a limited number of data samples. This proposed approach helps in learning an effective feature representation of facial expression samples in the latent space. Figure 6.5(a) displays the loss evaluation of CK+, Figure 6.5(b) shows the JAFFE dataset and Figure 6.5 (c) shows the In-house dataset with losses of 0.0415, 0.0517, and 0.2051 respectively. DSCSA initially showed signs of overfitting but stabilized after the 30th epoch on three different datasets, demonstrating that the model not only learns from the training data but also generalizes well to new data samples. This observation indicates that the proposed autoencoder layer has successfully learned to encode and decode facial expression input data, capturing its crucial features in the latent space.

On the other hand, the PSNR was calculated to validate the quality of the reconstructed facial expressions. PSNR is a widely used metric in image processing, where higher values signify lower reconstruction errors and, consequently, better image quality. This is particularly important in applications like facial expression reconstruction, where preserving fine details and subtle nuances is crucial for accurate interpretation. Based on this metric, the proposed autoencoder achieved PSNR values of 70.67 dB, 71.92 dB, and 67.04 dB on the CK+, JAFFE, and in-house datasets, respectively. In Figure 6.6(a), the reconstruction results of the CK+ dataset for different expressions are illustrated, while Figure 6.6(b) presents the results for the JAFFE dataset. Similarly, Figure 6.5(c) shows the results for the in-house dataset. Overall, these values and visual representations indicate that the proposed autoencoder not only effectively preserved the essential features of the facial expressions but also minimized reconstruction errors, ensuring a high-quality replication of the original images. This highlights the potential of the proposed method for accurate and reliable facial expression analysis.



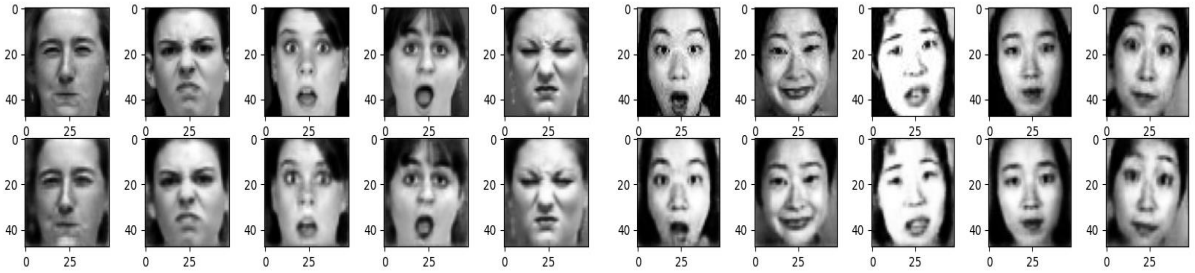
(a)

(b)

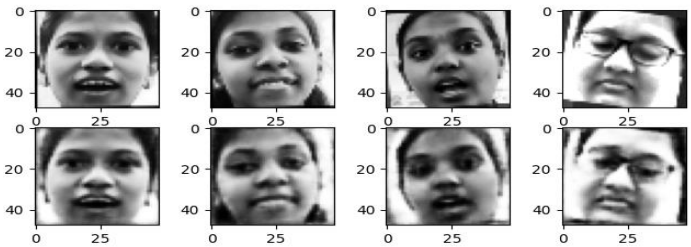


(c)

Figure 6.5. Loss graph of DSCSA (a) CK+ (b) JAFFE (c) In-house dataset



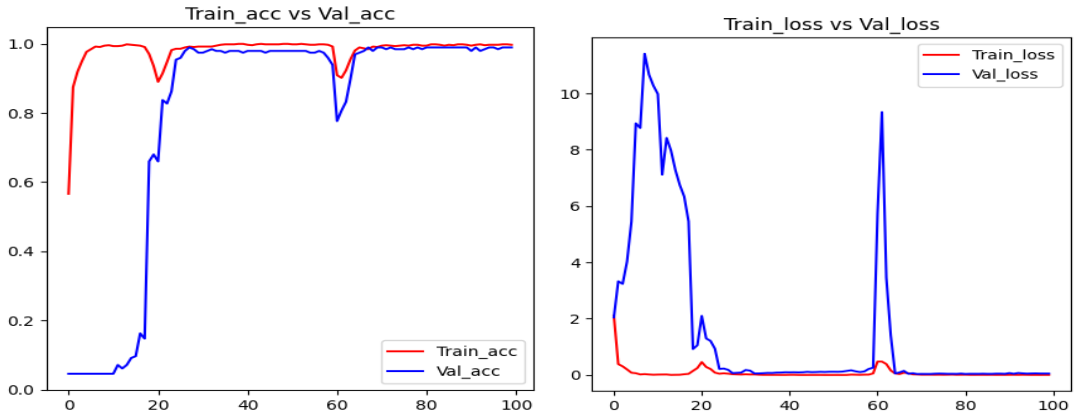
(a) CK+ (b) JAFFE



(C) In-house dataset

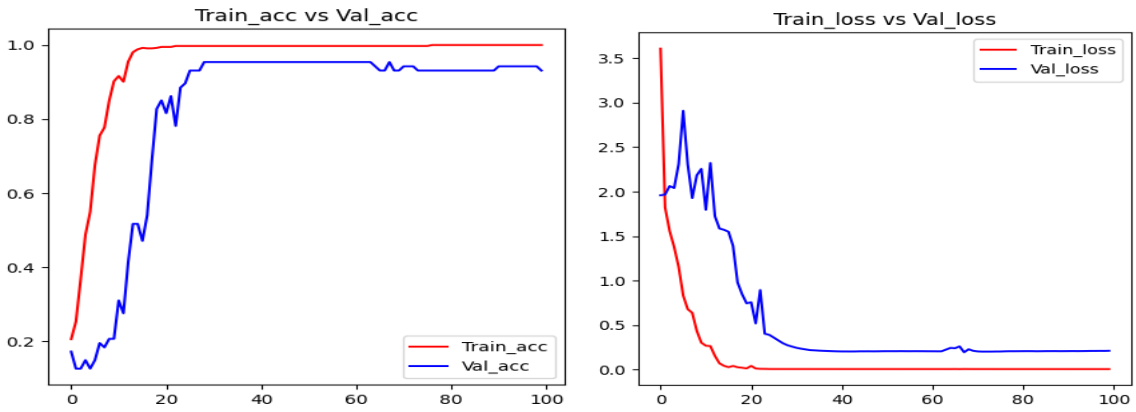
Figure 6.6. First row: Original Image Second row: Reconstructed Image

Figure 6.6. (a) illustrates the reconstruction results for the CK+ dataset, showcasing various facial expressions with notable accuracy. Similarly, Figure 6.6(b) presents the reconstruction results for the JAFFE dataset, demonstrating the autoencoder's capability to handle diverse and expressive grayscale images. Additionally, Figure 6.6(c) displays the reconstruction outcomes for the in-house dataset, further validating the robustness of the proposed model across different datasets. The obtained PSNR values and visual results collectively indicate that the proposed autoencoder effectively preserved essential features of facial expressions while minimizing reconstruction errors. This ensures that the output images closely resemble the original ones, maintaining the integrity of subtle details crucial for accurate facial expression analysis.



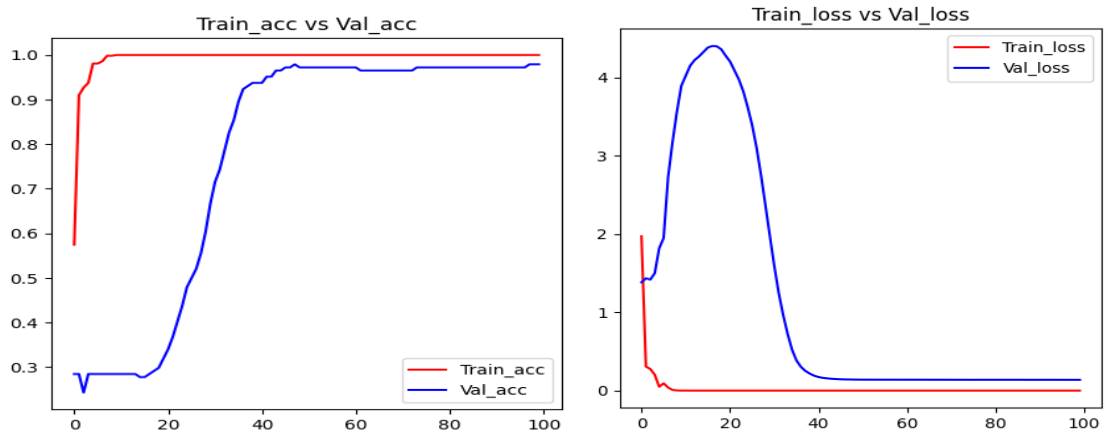
(a) (b)

Figure 6.7. Training and validation results on CK+ (a) Accuracy (b) Loss



(a) (b)

Figure 6.8. Training and validation results on JAFFE (a) Accuracy (b) Loss



(a) (b)

Figure 6.9. Training and validation results on In-house (a) Accuracy (b) Loss

In the subsequent phase of the model development, after training the CSA, additional layers—including a softmax layer, dense layer, and fully connected layers—were integrated above the encoder section. These layers were designed to utilize the learned feature vectors extracted from the unlabeled facial expression images during the CSA training process. This architecture enabled the model to effectively build on the unsupervised learning phase, transitioning seamlessly into supervised learning for classification purposes. To achieve this, labeled facial expression images were introduced to the latter part of the network, allowing the model to learn the mapping between the extracted features from the encoder and their corresponding class labels. The softmax layer, as part of the semi-supervised approach, facilitated the classification by assigning probabilities to each class during training. This approach ensured the model leveraged both labeled and unlabeled data, enhancing its robustness and generalization capabilities.

The training process was conducted over 100 epochs, with a batch size of 32, utilizing the Adam optimizer for efficient learning. The learning rate was set to 0.001, balancing the speed of convergence and stability of the training process. The results for the CK+ dataset are illustrated in Figure 6.7, which depicts the training and validation accuracy over the epochs. Initially, the model experienced significant fluctuations, but after the 20th epoch, it began to stabilize and converge, effectively identifying patterns associated with each class label. The model achieved an impressive accuracy of 0.9898 and a corresponding loss of 0.0515, reflecting its capability to learn and generalize effectively. Similarly, the performance on the

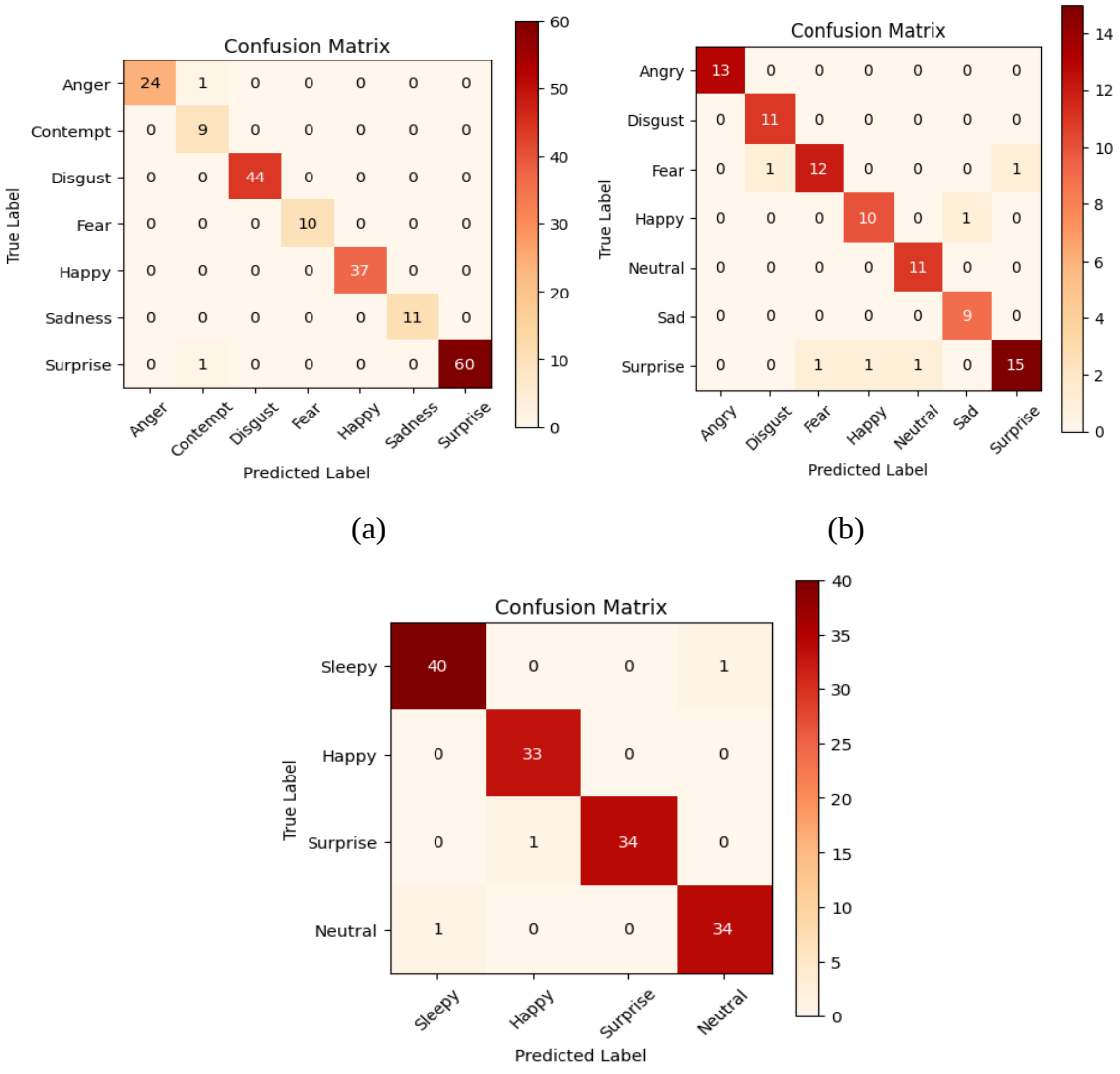
JAFFE dataset is displayed in Figure 6.8, showing the training and validation loss trends. While the training initially exhibited minor fluctuations, the model eventually stabilized, demonstrating its ability to generalize across the dataset. Finally, the performance of the in-house dataset is presented in Figure 6.9, which further validates the effectiveness of the proposed DSCSA model. These results collectively highlight the model's strong performance across diverse datasets, showcasing its ability to handle variations in data while maintaining high accuracy and low loss.

Furthermore, Figure 6.10 provides a detailed analysis of the classification accuracy for individual class labels within the CK+, JAFFE, and In-house datasets, presented in the form of confusion matrices. These matrices effectively illustrate the strengths and weaknesses of the model in distinguishing between different facial expressions. Figure 6.10(a) depicts the outcomes for the CK+ dataset, which includes seven class labels. While the model performs well overall, slight misclassifications are observed. For instance, anger and surprise are occasionally misclassified as contempt, with respective error rates of 0.04% and 0.02%. These minor errors highlight areas where the model struggles to differentiate between subtle overlaps in expressions, likely due to shared features such as eyebrow positioning or mouth shape. Similarly, Figure 6.10(b) presents the confusion matrix for the JAFFE dataset, which also consists of seven class labels. Here, more nuanced misclassifications are evident. For example, fear is sometimes misclassified as disgust or surprise, while happiness is occasionally confused with sadness. Additionally, surprise is misclassified into multiple categories, such as fear, happiness, and neutral, reflecting the complexity of distinguishing expressions that share similar facial dynamics. These errors indicate that some expressions, such as surprise, exhibit features that overlap significantly with other emotions.

Figure 6.10(c) showcases the performance of the model on the In-house dataset, which includes four expressions. Despite the limited number of classes, some degree of misclassification persists. For instance, sleepy is occasionally misclassified as neutral, and surprise is sometimes identified as happy. These errors can be attributed to the inherent blending of facial features in certain expressions, particularly in naturalistic settings where mixed emotions may be more pronounced.

Overall, the analysis reveals a recurring challenge in facial expression recognition: the mixture of features across expressions. Facial expressions often share subtle similarities in

shape, movement, and appearance across different individuals, which can blur the boundaries between distinct emotions. This overlap contributes to the observed misclassifications and underscores the importance of further refining the model to better capture these nuances for improved accuracy.



(a) (b)

(c)

Figure 6.10. Confusion matrix results (a) CK+, (b) JAFFE, (c) In-house

Additionally, the ROC curve illustrates the model's classification performance for each facial expression across varying threshold values. Figures 6.11(a), (b), and (c) provide insights into the performance of the CK+, JAFFE, and In-house datasets for each facial expression. The results show that the proposed model has the ability to recognize these expressions in real-time.

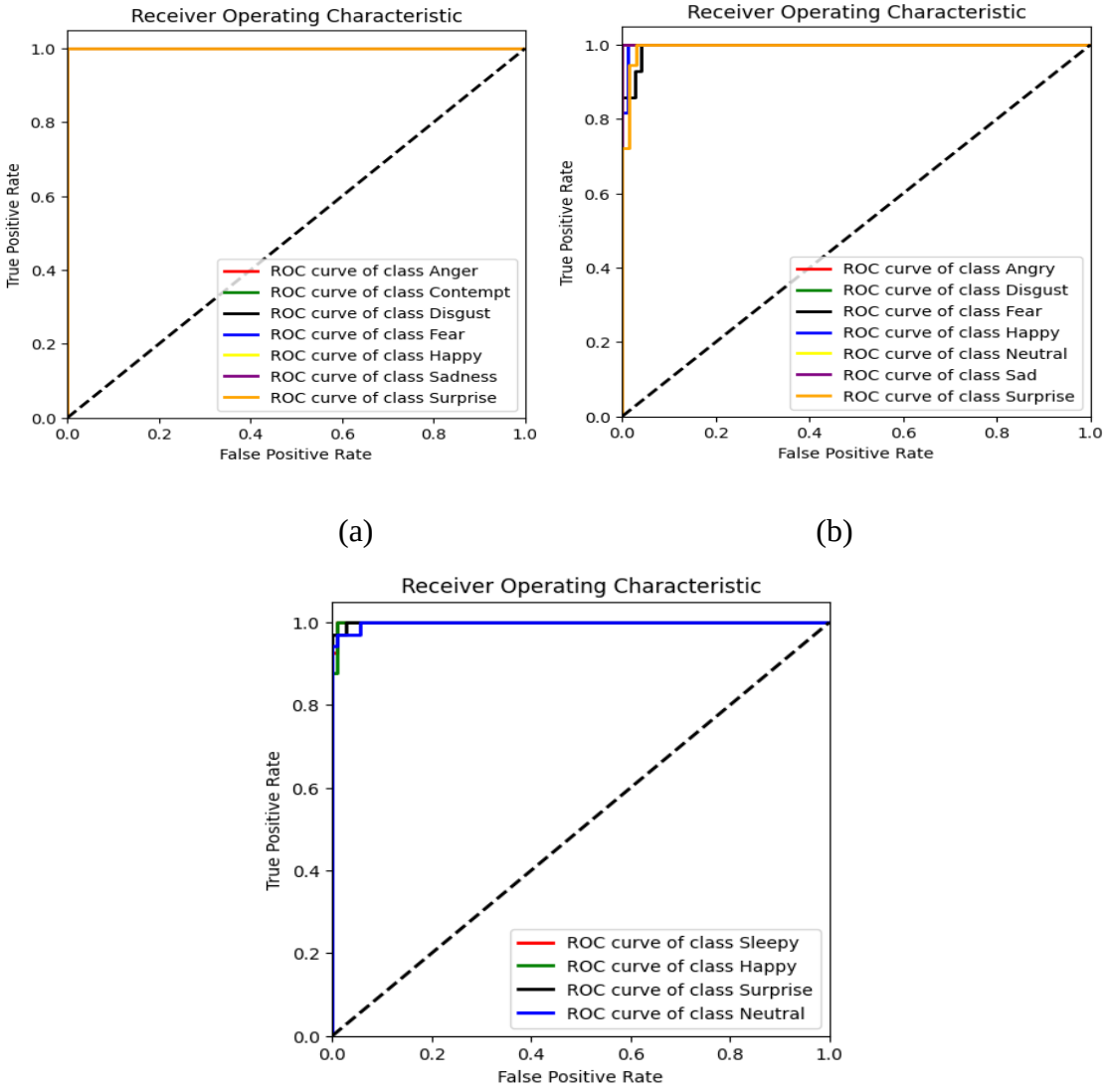


Figure 6.11. Receiver Operating Characteristic (a) CK+ (b) JAFFE (c) In-house

6.3.4 Performance comparison with baseline model and existing approach

Additionally, various baseline models, including CNN [216], DenseNet 121 [217] utilizing ImageNet features, and Convolutional Autoencoder (CAE), have been utilized to assess and contrast the performance of the proposed model. These methodologies are commonly employed in existing literature to develop FER systems for a wide range of applications. Despite this, specific hyperparameters such as the ReLU activation function, 100 epochs, 32 batch size, and Adam optimizer with a learning rate of 0.001 are kept consistent across all baseline models. This uniformity is crucial for comparing the

computational efficiency, particularly in terms of training and testing durations, with those of the proposed model. Tables 6.2, 6.3, and 6.4 display the precision, recall, F1-score, accuracy, training, and testing times for each model's performance on the CK+, JAFFE, and In-house datasets. The results demonstrate the effectiveness of the proposed model across different baseline models, despite the imbalanced nature of the dataset and the lower number of samples for each expression, particularly achieving notable performance on both CK+, JAFFE, and In-house datasets. Nevertheless, the baseline models did not perform well and failed to yield significant results under these conditions, likely due to challenges in identifying significant facial feature patterns with a limited number of data samples. Furthermore, the proposed autoencoder with sparsity helped to ignore irrelevant features and focus on efficient features during decoding, resulting in improved accuracy compared to the baseline AE performance.

Table 6.2. Experimental and comparison Results on CK+

Performance	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (%)	Training Time	Testing Time	PSNR (dB)
CNN	0.67	0.63	0.63	0.77	80s	6s	-
DenseNet 121	0.95	0.94	0.94	0.94	74s	4s	-
AE	0.95	0.96	0.96	0.96	60s	2s	68.65
DSCSA	**0.97**	**0.99**	**0.98**	**0.98**	**55s**	**2s**	**70.67**

Table 6.3. Experimental and comparison Results on JAFFE

Performance	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (%)	Training Time	Testing Time	PSNR (dB)
CNN	0.77	0.81	0.77	0.78	75s	4.3s	-
DenseNet 121	0.86	0.87	0.86	0.86	62s	3.5s	-
AE	0.90	0.91	0.90	0.90	58s	3.5s	69.90
DSCSA	**0.93**	**0.94**	**0.93**	**0.93**	**52s**	**2.5s**	**71.92**

Table 6.4. Experimental and comparison results on In-house

Performance	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (%)	Training Time	Testing Time	PSNR (dB)
CNN	0.66	0.67	0.66	0.66	84s	4s	-
DenseNet 121	0.72	0.73	0.73	0.73	72s	3.2s	-
AE	0.83	0.84	0.83	0.83	60s	3s	64.09
DSCSA	**0.98**	**0.98**	**0.98**	**0.98**	**53s**	**2.8s**	**67.04**

Table 6.5. Comparison of results achieved by proposed approach with existing methods

Methods	Dataset	Accuracy (%)
CNN (AlexNet) [218]	CK+	94.40
RBM [219]		96.80
CNN [220]		93.20
PNN [221]		92.00
DBN [222]		96.70
DAE (DSAE) [223]		95.79
CNN [224]		94.39
GAN (cGAN) [225]		97.30
Hierarchical Deep Neural Network [226]		89.81
Proposed		**98.98**
Boosted-LBP + SVM [227]	JAFFE	81.00
Boosted Deep Belief Network [225]		68.00
Radial encoding of local Gabor features and classifier [228]		55.87
Local Directional Pattern [229]		82.60
DBN [230]		90.95
Proposed		**93.10**
CNN-BiLSTM (Previous Proposed)	In-house	92.84
Proposed		**98.00**

Furthermore, Table 6.5 illustrates the comparison of DSCSA performance with state-of-the-art techniques. The table's first section compares CK+ prediction accuracy across different existing approaches, showcasing a 1.68% increase in accuracy compared to other methods. Similarly, the second section highlights a significant improvement of 2.15% in prediction accuracy on the JAFFE dataset compared to existing approaches. This improvement underscores the efficacy of the proposed approach, which integrates dimensionality reduction techniques to emphasize pertinent features for constructing robust facial expression patterns and enhancing classification accuracy across various expressions.

6.3.5 Time Complexity analysis of proposed architecture

Subsequently, the time complexity has been analysed for this DSCSA approach. Complexity in a deep learning model is assessed based on the number of operations required for one forward pass, size of the image, the filter size, the number of layers, and neuron size, which also depends on system configuration. In this model, the encoder comprises a convolutional 2D layer representing $O(m^2 * d * k^2)$, here m is the input image size, d is the depth of the filter and k is the filer size. Bach normalization takes $O(m^2 * d)$, max pooling takes $O(1)$, and finally, the encoder has a time complexity of $O(m^2 * d * k^2)$. The same analysis applies to the decoder part. The time complexity for the overall network is $O(((m^2 * d) + w) + i * e)$, where e represents the number of epochs, i denotes the input length and w represents the number of weights.

6.3.6 Model Prediction Analysis using XAI

This section presents the importance of XAI [231], which not only provides heatmaps but also facilitates a deep understanding and visualization of emotion across different facial expressions through variations in colors. In the context of images, XAI refers to the creation and use of AI models and methodologies that can offer clear and understandable justifications for their judgments and forecasts in tasks involving images. This is especially crucial because many contemporary AI models—particularly deep neural networks—are viewed as 'black boxes' because of their intricate architectures and the unclear processes by which they derive particular results. In these experiments, Grad-CAM [232] and Image-LIME [233] have been used to explain the performance of the proposed model and its feature contributions for prediction. The Grad-CAM algorithm calculates the gradients between the last convolutional layer of a CNN's feature maps and its predicted class score. The regions impacting the

model's prediction are then visualized by applying a ReLU activation, calculating a weighted sum of these feature maps, and superimposing the resulting spatial map on the input image. This process highlights important facial features in image areas for better interpretability.

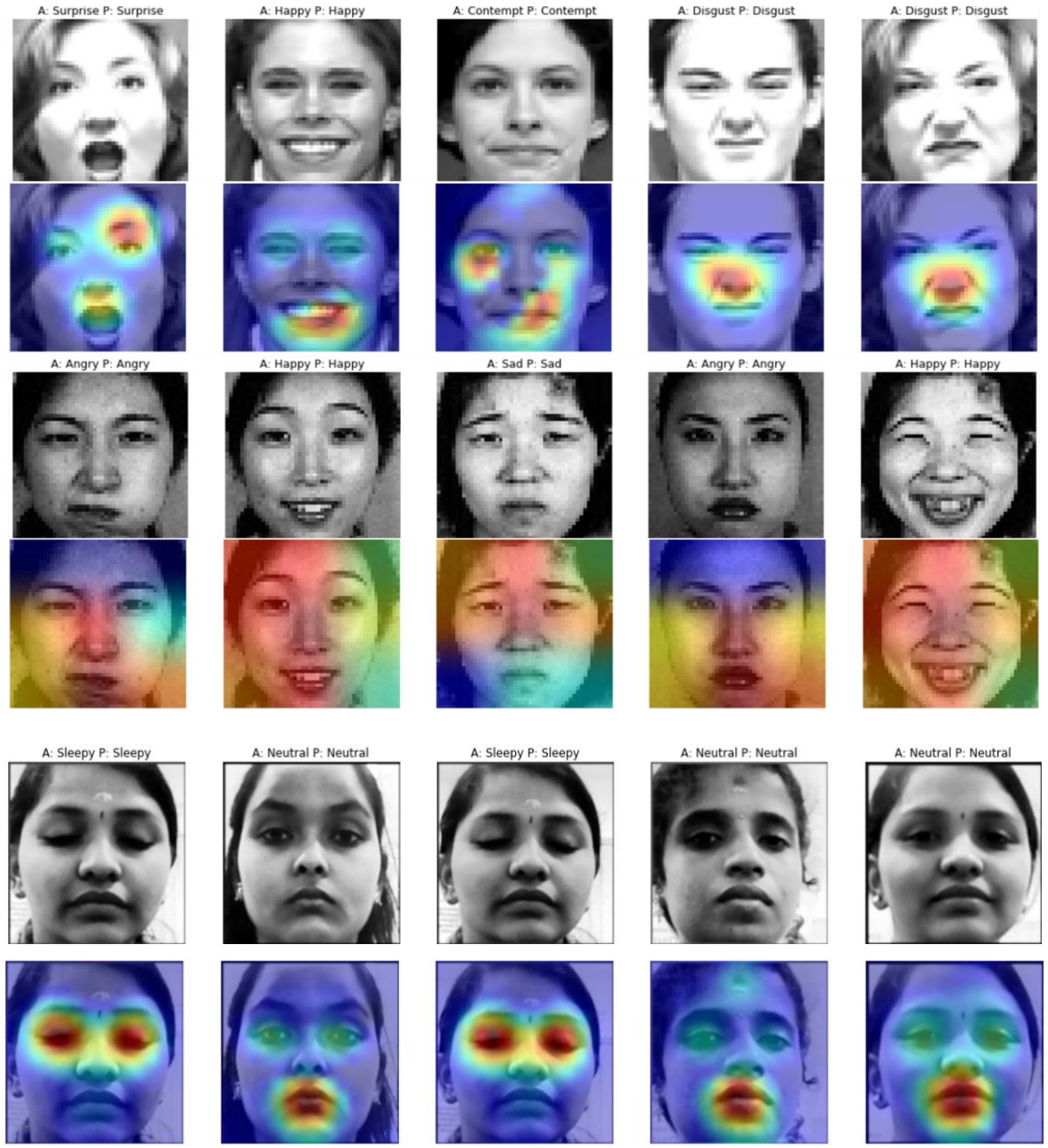
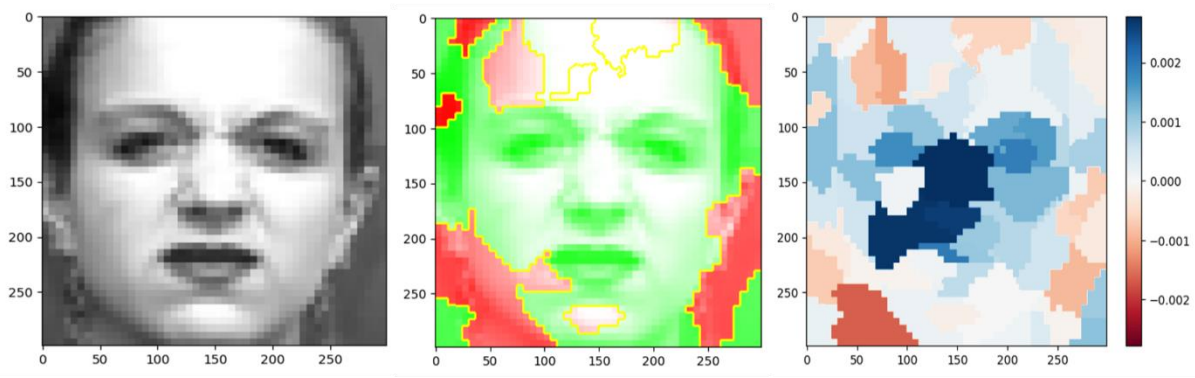


Figure 6.12. Grad-CAM Visualization of CK+, JAFFE, and In-house

Figure 6.12 displays the Grad-CAM visualization of CK+, JAFFE, and In-house samples with various facial expressions, clearly illustrating how the model accurately predicted emotion labels and presented features contributing to that decision. Noteworthy features, such

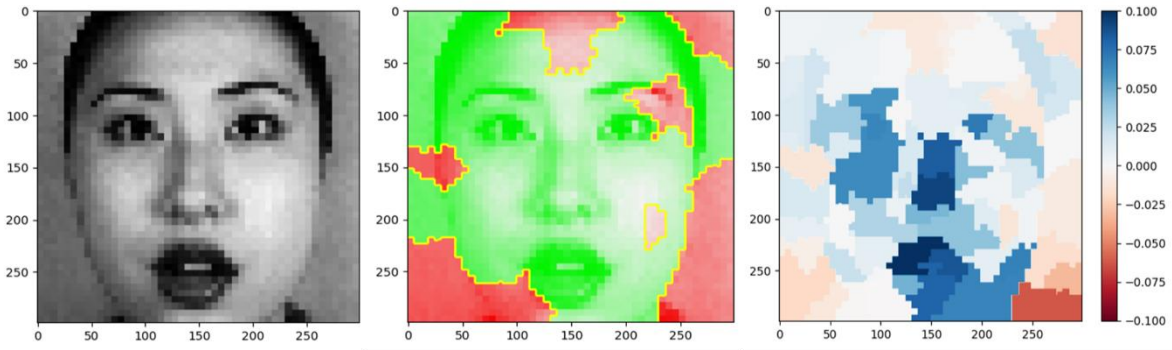
as an open mouth during surprise, pulled mouth corners when smiling, tightened lips during contempt, and nose wrinkles during disgust, are significant for identifying these expressions. These features are also described in FAC for classifying facial expressions.

On the other hand, Image-LIME [233] helps to understand how the model works internally to make final decisions. Figures 6.13, 6.14, and 6.15 show the LIME predictions of CK+, JAFFE, and In-house samples for disgust, surprise, and happy expressions, respectively. In these figures, the first image displays the original test samples, the second shows the super-pixels of LIME on facial expression, where the green color indicates positively contributed features, and red color indicates negatively contributed features. The third image illustrates the color intensity of each feature region, where blue color has a higher contribution than red color for prediction. It highlights the superpixels near the nose as highly contributing to predicting the expression of disgust. Similarly, Figure 6.14, shows the surprising facial expressions in JAFFE test samples.



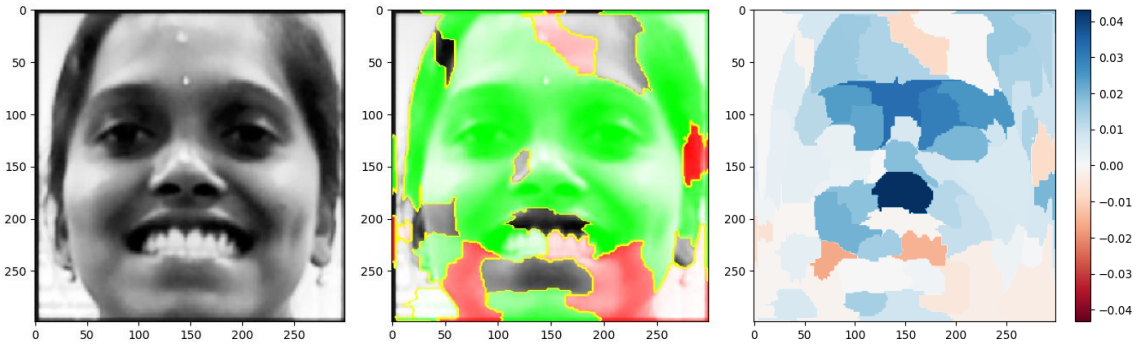
(a) Original images (b) LIME Features (c) Plot weights as Heat-map

Figure 6.13. LIME Visualization of DSCSA on Disgust expression in CK+



(a) Original images (b) LIME Features (c) Plot weights as Heat-map

Figure 6.14. LIME Visualization of DSCSA on surprise expression in JAFFE



(a) Original images (b) LIME Features (c) Plot weights as Heat-map

Figure 6.15. LIME Visualization of DSCSA on surprise expression in In-house

6.4 Discussion

This study presents a FER system utilizing a deep convolutional autoencoder, incorporating sparsity, and trained through a semi-supervised approach. The system is evaluated on two standard benchmark datasets commonly employed in FER research. In a conventional fully connected network, the CNN layer is highly effective in extracting spatial information from labeled samples with huge data samples. However, increasing the layer size to construct effective patterns for accurate image classification based on a supervised approach often leads to a rise in computational time due to an increase in layers and parameters.

In addition, various transfer learning techniques are commonly employed to enhance the prediction accuracy of deep learning models when data samples are limited and address overfitting issues, including those used in FER systems. Many pre-trained models are trained on large image datasets that include non-human subjects. While these models prove effective for general image classification tasks, they may not yield significantly higher accuracy results when applied to human facial expression data. Although the pre-trained weights of these models contribute to the generalization of FER models on the training time, they may not yield significant prediction results in real-time environments. Moreover, these models encounter challenges when predicting emotions in real-time, particularly in complex environments with factors such as high illumination, occlusion, and pose variations. This issue has been explicitly addressed by numerous researchers in their comparative studies and research of transfer learning in FER systems. To address this limitation, this study proposed the replacing conventional convolutional layer with a convolutional autoencoder with

minimum layers to focus on effective feature representation using an unsupervised approach and training on unlabeled data samples. Subsequently, the trained encoder part and feature vector are added to the fully connected layer, incorporating a SoftMax function for facial expression classification. The second phase of the model is trained using a semi-supervised approach, fine-tuning the network with labeled facial expression samples. As a result, significant improvements have been achieved on benchmark datasets, which also encompass variations in illumination, albeit to a lesser extent.

Additionally, the proposed approach has been compared with baseline models like CNN, transfer learning methods such as DenseNet-121, and CAE, leveraging both the CK+ and JAFFE datasets. The performance outcomes, including precision, recall, F1-score, accuracy, training, and testing metrics for each model, are detailed in Tables 6.2 and 6.3. To ensure consistent performance evaluation, the same set of hyperparameter values was applied to each model. Based on the experiments, the DSCSA model achieved significant results and outperformed existing state-of-the-art techniques. Moreover, the model's performance has been elucidated using XAI techniques. In general, FER on static images may not fully capture the real emotions of the user. Therefore, in the future, a multimodal-based FER system will be proposed to analyze dynamic user emotion prediction and analysis.

6.5 Chapter Summary

This chapter presents a computationally effective autoencoder-based FER system with a limited number of layers and parameters. The system is experimented with a smaller dataset containing CK+, JAFFE, and In-house samples to address challenges in conventional techniques, such as overfitting and the need to increase layer size for accuracy improvement, which leads to increased computational costs. The proposed system, named DSCSA, consists of two parts. After pre-processing, the first part involves training on unlabeled FER samples using a convolutional sparse autoencoder. Loss and PSNR values demonstrate the effectiveness of this part in learning the most discriminative feature representation. Subsequently, the learned encoder part, along with the feature vector, is connected to a fully connected layer and SoftMax for significant facial expression classification through a semi-supervised learning approach.

The experimental findings demonstrate that DSCSA attains substantial performance improvements over baseline methods and state-of-the-art approaches. These improvements

are validated through metrics commonly utilized in FER systems and deep learning methodologies. Furthermore, XAI techniques, specifically in these experiments, such as Grad-CAM and image LIME, have been employed to demonstrate the model's performance across various facial expressions in both datasets.