An Improvised Facial Emotion Recognition System using the Optimized Convolutional Neural Network Model with Dropout

P V V S Srinivas[1], Pragnyaban Mishra[2]
Department of Computer Science and Engineering
Koneru Lakshmaiah Education Foundation (KLEF), Guntur, India

***Abstract*—Facial expression detection has long been regarded as both verbal and nonverbal communication. The muscular expression on a person's face reflects their physical and mental state. Using computer programming to integrate all face curves into a categorization class is significantly more important than doing so manually. Convolutional Neural Networks, an Artificial Intelligence approach, was recently developed to improve the task with more acceptance. Due to overfit during the learning step, the model performance may be lowered and regarded underperforming. There is a method dropout uses to reduce testing error. The influence of dropout is applied at convolutional layers and dense layers to classify face emotions into a distinct category of Happy, Angry, Sad, Surprise, Neutral, Disgust, and Fear and is represented as an improved convolutional neural network model. The experimental setup used the datasets namely JAFEE, CK48, FER2013, RVDSR, CREMAD and a self-prepared dataset of 36,153 facial images for observing train and test accuracy in presence and absence of dropout. Test accuracies of 92.33, 96.50, 97.78, 99.44, and 98.68 are obtained on Fer2013, RVDSR, CREMA-D, CK48, and JAFFE datasets are obtained in presence of dropout. The used features are countably large in the computation as a result the higher computation support of NVDIA with the capacity of GPU 16GB, CPU 13GB and memory 73.1 GB are used for the experimental purposes.**

Keywords*—Convolutional neural network (CNN); facial emotion recognition (FER); dropout; FER 2013; CREMAD; RVDSR; CK48; JAFFE

I. Introduction

The face expressions connect to emotions; they are essential identifiers for human sentiments. Most of the time, a person's facial expression is a nonverbal means of expressing emotion, and it may be used as tangible proof of human state. Human–computer interaction, psychiatric observations, intoxicated driver recognition, and lie detection are all viable uses facial emotion recognition.

The convolutional layers which act as a backbone for classification with artificial intelligence in several applications few of them are cancer detection [1], Brain Tumor Segmentation, Object detection [2], and crowd counting [3]. The CNN takes input facial image data, modify it using kernels, and then transmit the outputs to the next convolution layer. The final CNN layer's output is flattened and sent into the Feed Forward Neural Network for categorization into an emotion class. The learning stage entails training the model, while the evaluation stage entails putting it to the test and determining the acceptance percentage. Due to the impact of overfitting, it is more likely that the training phase is more fitted implies reduction of the test accuracy. By avoiding overfitting [4], the under described model expresses a research direction of reaching higher accuracy of facial emotion recognition.

The remaining of the paper is organized as follows: Section 2, Related Studies and Motivations; Section 3, Research Method; Section 4, Result and discussion; and Section 4, Conclusion.

II. Related Studies and Motivations

Image is a set of pixels represented by the function $f(x, y)$ such that the scalar quantity of an image's $x \in domain(x - axis)$ and $y \in range(y - axis)$ is equivalent to the amount of energy emitted from the location where the image was captured. Assume that $f(a, b)$ denotes a continuous variable picture that is transformed to a digital image in the form of $f(x, y)$ with $x \in \{0,1,2,..,\mathrm{M}-1\}$ and $y \in \{0,1,2,..,\mathrm{N}-1\}$ Here M, N are the length and breadth of the digital image [5].

$$f(x,y) = \begin{pmatrix} f(0,0) & \cdots & f(0,M-1) \\ \vdots & \ddots & \vdots \\ f(N-1,0) & \cdots & f(N-1,M-1) \end{pmatrix}$$

CNN is a Deep Learning method that can take an image as input, assign importance of learnable weights and biases to distinct aspects in the image, and distinguish one from the other. When compared to other classification methods, the amount of pre-processing required by a CNN is significantly less. While basic techniques need hand crafted of filters, CNN can learn these filters or characteristics with enough training [6] and successfully capture the Spatial and Temporal dependencies in a picture. Due to the reduced number of parameters involved and the reusability of weights, the architecture performs superior trained to better recognition of the image. The goal of the CNN is to compress the images into a structure that is easier to process while preserving important elements for a successful prediction. By retaining picture features, the CNN's flow from layer to layer minimises the dimension. Different kernels and pooling layers of CNN can accomplish this task. The residue left over after a few repetitions of the previous stages is fed into the dense layer for categorization according to the need and model specification.

Current techniques largely focus on face study while keeping the surrounding intact, resulting in a variety of redundant and erroneous features that hamper CNN training. Happy, Angry, Sad, Surprise, Neutral, Disgust, and Fear are the seven basic face emotion classes that the model learns during learning stage. Recently [7] [8], researchers have achieved remarkable progress in facial expression identification with higher number of classes [9], leading to advancements in neurology and applied mathematics, etc that are boosting research in the field of facial expression further. Moreover, advances in computer vision and machine learning have made emotion recognition more accurate tools of classification.

Dropout changed the idea of learning all the weights in the network in each training cycle to learning a proportion of the weights in the network. This problem is solved by addressing overfitting [10] in networks with many neurons, which are then associated with weights. Regularization was an important study topic prior to Dropout. Regularization approaches in neural networks, such as L1 and L2 weight penalties, are introduced. These regularizations, however, did not entirely alleviate the overfitting problem [11].

Co-adaptation is a key challenge when learning networks with high neurons. If all the weights are learnt at the same time in such a network, it is likely that certain connections will be more predictive than others, As the network is trained repeatedly in this case, the stronger connections are learned more, while the weaker ones are ignored. The traditional regularization, such as the L1 and L2, could not avoid this. The reason for this is that they also regularize depending on the connections' prediction abilities. As a result, they approach determinism in selecting and rejecting weights. As a result, the strong become stronger and the weak become weaker.

The researchers Nitish Srivastava et al. [10] extensively studied the impact of dropout and concluded that increasing the size of the neural network would not help. As a result, neural networks' size and accuracy have been limited. Dropout is a co-adaption strategy that can help to avoid these problems.

As seen below, there are numerous examples of major contributions in this field in the literature.

With the CK48 [12] and FER2013 [13] datasets, Mollahossei [14] proposed CNN for FER [13] obtained 93.2% and 61.1% accuracy, respectively. Using the CK48 dataset, [15] investigated the impact of data pre-processing prior to training the network on emotion categorization and found that it improved accuracy by 96.76%. Using the CK48 [12] dataset, [16] used the notion of action units and achieved 97.01% accuracy. [17] suggested a unique architecture based on Sparse Batch normalization, estimating the model's accuracy at 95.24% for JAFEE [18] and 96.87% for the CK48[12] dataset. Using the FER2013[13] database, Agrawal et Mittal [19] investigated the influence of adjusting CNN parameters on classification results and found a 65.23% average accuracy without dropout and 65.77% with dropout. The author in [20] used the datasets JAFEE [18] and CK48 [12] to train the model, achieving 95.23 % and 93.24 % accuracy, respectively.

Similar tests with datasets CREMAD [21] & RVDSR [22] were done by researchers [23] with accuracy notes of 65%, 58.33%, [24] 52.52%, 47.11 %, [25] 74.0 %, 67.5 % and [26] 62.84% only for CREMAD [21] dataset.

III. Research Method

A. Proposed CNN Model

The convolutional neural network proposed in this section is an optimal framework that contains layers namely an input, an interleaved Z grouped convolutional, a batch normalization, a polling, a fully connected, a dropout, a flatten and a dense that forward to an output layer at the end. The fully connected layer is resulted after slicing the convolutional, batch normalization and max pooling layers that are connected through a cross-layer connection as shown in the Fig. 1.

The input image to the proposed framework is a three-dimensional array with dimensions $h \times w \times n$ where h, w are spatial dimensions and n is the channel whose value is 1 as gray scale images are considered as input, $*$ It is used to perform the convolution operation and Act plays the role of activation function. The output of the convolutional layer is represented as follows.

$$hl_{2n,j}^{l} = conv_{n+1,j}^{l} = Act\left(u_{2n,j}^{l}\right)$$
$$= Act\left(\sum_i hl_{2n-1,i}^{l} * W_{i,j}^{2n}\right), \quad 1 \le n \le Z \tag{1}$$

Where as $W_{i,j}^{2n}$ represents the weight matrix between the feature space of $(2n-1)th$ hidden layer that belongs to i and feature space of $(2n)th$ hidden layer which belongs to j. $conv_{n+1,j}^{l}$ is j^{th} feature plane of $(n+1)th$, $hl_{2n,j}^{l}$ represents i^{th} feature space of the hidden layer $(2n-1)$ and $hl_{2n-1,i}^{l}$ represents the j^{th} feature space of hidden layer $(2n)$.

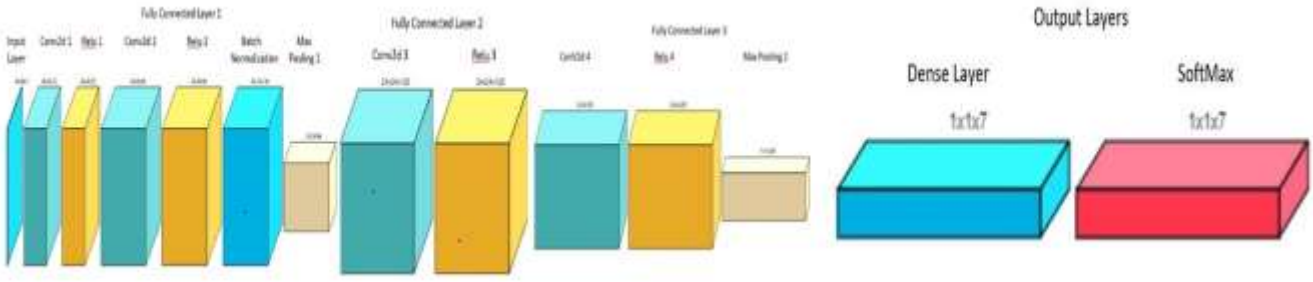


Fig. 1. Proposed CNN Architecture.

The output of the convolution layer obtained from Equation 1 is given to the Batch normalization layer where the batch of inputs is represented as $BT = \{hl^l_{1,2,3,\dots,2n}\}$ and the mean of the batch is given as.

$$E[hl^l_{BT}] = \frac{1}{m}\sum_{j=1}^{2n} hl^l_{2n} \quad (2)$$

And the variance of the batch is

$$Var[hl^l_{BT}] = \frac{1}{m}\sum_{j=i}^{n}(hl^l_{2n} - E[hl^l_{BT}])^2 \quad (3)$$

$$hl^{\wedge}_{BT} = \frac{hl^l_{2n} - E[hl^l_{BT}]}{\sqrt{var[hl^l_{BT}] + \varepsilon}} \quad (4)$$

Where $hl^{\wedge}_{BT}$ represents the normalized values

After calculating mean, variance, and normalized values. The output of the batch normalization layer is given as

$$hl^l_{2n+1,j} = Bat^l_{n+1,j} = \alpha * hl^{\wedge}_{BT}, 1 \le BT \le Z \quad (5)$$

$hl^l_{2n+1,j}$ represents j^{th} feature space of $(2n)th$ hidden layer, $Bat^l_{n+1,j}$ represents the batch normalization of layers j^{th}feature space of sample l

The output of the Batch normalization layer is fed as an input to the max polling layer in which fixed step sized feature space is used. The output of the polling layer is calculated by the following equation.

$$hl^l_{2n+2} = Max\ polling\{hl^l_{2n+1,j}\} \quad 1 \le n \le Z \quad (6)$$

In the proposed model a convolutional layer, a Batch normalization layer, and a Max Polling layer forms one fully connected layer. And the output of the fully connected layer is given to dropout layer where some of the neurons will be removed before going to the next fully connected layer. hl^l_{2n+2} is the output of a fully connected layer and will be given as input to the dropout layer. The output of the dropout layer is calculated as given below.

$$hl^l_{2n+3} = Act\left(\left(W^{[2n+2]} .* Drop^{[2n+2]}\right) hl^l_{2n+2}\right),$$

$$1 \le n \le Z \quad (7)$$

$W^{[2n+2]}$ is the weights associated to layer $2n+2$, $Drop^{[2n+2]}$ amount of dropout applied on layer $2n+2$, hl^l_{2n+2} is the output of fully connected layer and .* It is called Hadamard product.

The above process is repeated for all the fully connected layers of the proposed convolutional neural network model and the fully connected form of the proposed model which is also called flattening is given as mentioned below.

$$hl^l_{2Z+1} = (a_1 hl^l_1, a_2 hl^l_2, a_3 hl^l_3, \dots, a_{2Z} hl^l_{2Z}) \quad (8)$$

The binary string $AI = a_1, a_2, a_3, \dots, a_{2Z-1}$ represents a crossover indicator which indicates the cross-layer connection. For example $AI = 111, \dots, 1$ says that all $(2Z-1)$ convolution, batch normalization, polling, dropout which forms a fully connected layer are connected to the final fully connected layer which is also called as flatten layer, the representation $AI = 100, \dots, 0$ says that only the first one is the fully connected layer and the remaining are normal convolutional layers and the representation $AI = 000, \dots, 0$ says that there are no fully connected layers and all the layers in the network are just convolution layers. For the proposed model in this paper $AI = 1,0,1$ that says that we have a fully connected layer followed by a convolutional layer which was followed by a fully connected layer again. After performing flattening operation resulted from Equation 8 is fed as an input the dense layer in which Relu activation function is used followed by dropout layer which was again followed by a dense layer where the output is obtained. SoftMax function is used in the final dense layer to classify the output.

$$out^l_{Dense1} = Dense\left(Den_N, Act_{Relu}(hl^l_{2Z+1})\right) \quad (9)$$

out^l_{Dense1} denotes the output of the first hidden layer, Den_N represents the number of hidden neurons used in the dense layer, Act_{Relu} says the Relu Activation function is used here for activating the neurons and hl^l_{2Z+1} is nothing but the output of the flatten layer or final fully connected convolutional layer.

$$out^l_{Drop} = Act\left(\left(W^{[2Z+2]} .* Drop^{[2Z+2]}\right) hl^l_{2Z+2}\right) \quad (10)$$

out^l_{Drop} is the output of the dropout layer after the dense layer, Act is the activation function, $W^{[2Z+2]}$ and $Drop^{[2Z+2]}$ are respective weights assigned and the amount of dropout applied. And, finally the output of the last dense layer where the classified output is obtained as given below.

$$out^l_F = Dense\left(Den_C, Act_{Softmax}(out^l_{Drop})\right) \quad (11)$$

out^l_F is the final output of the proposed convolutional neural network, Den_C represents the total classes to classify, $Act_{softmax}$ is the SoftMax activation function.

For the sample l, the optimization model that was propose uses the formulas given below to calculate the activation of all Convolutional, Batch Normalization, Max Polling, Fully connected, Dropout, Dense layers.

$$hl^l_{2n,j} = conv^l_{n+1,j} = Act(u^l_{2n,j})$$

$$= Act\left(\sum_i hl^l_{2n-1,i} * W^{2n}_{i,j}\right), \quad 1 \le n \le Z$$

$$hl^{\wedge}_{BT} = \frac{hl^l_{2n} - E[hl^l_{BT}]}{\sqrt{var[hl^l_{BT}] + \varepsilon}}$$

$$hl^l_{2n+1,j} = Bat^l_{n+1,j} = \alpha * hl^{\wedge}_{BT}, \qquad 1 \le BT \le Z$$

$$hl^l_{2n+2} = Max\ polling\{hl^l_{2n+1,j}\}, 1 \le n \le Z$$

$$hl^l_{2n+3} = Act\left(\left(W^{[2n+2]} .* Drop^{[2n+2]}\right) hl^l_{2n+2}\right)$$

$$hl^l_{2Z+1} = (a_1 hl^l_1, a_2 hl^l_2, a_3 hl^l_3, \dots, a_{2Z} hl^l_{2Z})$$

$$out^l_{Dense1} = Dense\left(Den_N, Act_{Relu}(hl^l_{2Z+1})\right)$$

$$out^l_{Drop} = Act\left(\left(W^{[2Z+2]} .* Drop^{[2Z+2]}\right) hl^l_{2Z+2}\right)$$

$$out^l_F = Dense\left(Den_C, Act_{Softmax}(out^l_{Drop})\right)$$

B. The effect of Drop Out on FER Classification

Dropout is a process of dropping a %age of connection with probability (1-p). The Expectation with dropout $Ep(d)$ is represented as:

$$Ep(d) = \frac{1}{2}(t - \sum_{i=1}^{m} \delta_i w_i I)^2 \quad (12)$$

where δ is a dropout rate: $\delta \sim$ Bernoulli(p).

The Bernoulli(p) satisfy the properties listed below.

If $X$is a random variable with Bernoulli distribution, then:

$$P_r(X = 1) = p = 1 - P_r(X = 0) = 1 - q \quad (13)$$

The probability mass function f on this distribution, over possible outcomes k, is.

$$f(k;p) = \begin{cases} p & if\ k = 1 \\ q = 1 - p & if\ k = 0 \end{cases}$$

$$f(k;p) = p^k(1-p)^{1-k} \ \ \forall\, k \in \{0,1\}$$

$$f(k;p) = pk + (1-p)(1-k) \ \ \forall\, k \in \{0,1\} \quad (14)$$

Lemma 1: Dropping network connections increases generalization over non-dropout.

Let a weight $w_{pq}$is associated in between two consecutive hidden layers of p and q with absolute Mean Square Error E for a neural network. The projection for respective connections is represented as

$$\omega(p,q) = \frac{\partial^2 E}{\partial w_{pq}^2}\gamma^2(p,q) \quad (15)$$

$\gamma(p,q)$ is the parameter that control the Connection [27] with high projection improves the generalization of the Neural network. The generalization potential of the network with normalization is represented as

$$G_a = \prod_{\forall(a,b)} \omega'(p,q) \quad (16)$$

Where $\omega'$is the normalized view of the $\omega(p,q)$.

After few of the connection dropped the generalization of the neural network measured as

$$G_b = \prod_{\forall(a,b:\exists\omega'(\cdot)\to 0)} \omega'(p,q) \quad (17)$$

$$\forall Dout_{ab}: \exists\gamma(\cdot) \to 0 \ \Rightarrow \omega(\cdot) \to 0. \quad (18)$$

$$\therefore G_b > G_a \quad (19)$$

As can be seen from the equation above, dropping connections can result in a high degree of generalization.

Theorem 1: With every rise in hidden layer, the frequency of activation neuron falls.

To simplicity, the number of hidden layers is an even and it is n. An adaptive probability density function [28] for any hidden layer $l \ \ where\ 1 \le l \le n$ is defined as:

$$f(l) = 1 - (\sigma\sqrt{2\pi})^{-1} \int_{-\infty}^{l} exp\left(\frac{\left(l-\frac{n}{2}\right)^2}{2\sigma^2}\right) dl \quad (20)$$

The derivative of the function $f'(l) = (\sigma\sqrt{2\pi})^{-1} e^{(ln/2)^2/2\sigma^2}$

$\because f'(l) < 0 \ \Rightarrow f(l)\ is\ monotonically\ decreasing$

$\therefore f(n) < f(l) < f(1)$

$$\{f(l)\}_{mx} = 1 - (\sigma\sqrt{2\pi})^{-1} \int_{-\infty}^{1} exp\left(\frac{(l-\frac{n}{2})^2}{2\sigma^2}\right) dl < 1$$

$$\{f(l)\}_{mn} = 1 - (\sigma\sqrt{2\pi})^{-1} \int_{-\infty}^{n} exp\left(\frac{\left(l-\frac{n}{2}\right)^2}{2\sigma^2}\right) dl > 0 \quad (21)$$

it is cleared from above two equations that min and max ranges in between 0 and 1 is $0 < f(l) < 1$, Hence Proved.

IV. Experimentation and Results

A. About Datasets

For experimentation six different datasets are considered namely FER2013 [13], RVDSR [22], CREMAD [21], CK48 [12], JAFFE [18] and own dataset each of which consists of gray scale images of dimensions 48 x 48. For own dataset images and videos are collected from different web resources, all the images are converted to gray scale and resized to 48x48, the videos are converted to frames and preprocessed manually by considering only those frames which are of good. FER 2013[13] and own dataset consists of images belonging to seven different classes namely Angry, Disgust, Fear, Happy, Neutral, Sad and Surprise, whereas CREMAD [21] and RVDSR [23] datasets consist of images belonging to six different classes namely Angry, Disgust, Fear, Happy, Neutral, and Sad. FER 2013[13] consists of 35557, RVDSR [22] consists of 61,673, CREMAD [21] Consists of 61,309 CK48[12] and JAFFE [18] consists of 3540 and 3406, own dataset consists of 36,153 images, respectively. RVDSR [22] and CREMAD [21] datasets consist of videos of different expressions, all the videos are converted to frames. While converting the videos to frames, only the frame for every second is considered, after which the images are resized to 48x48 and taken for experimentation. The details of the images in the dataset are given in Table I.

TABLE I. Description of the Datasets used for Experimentation

No of Images Per Category	Name of the Dataset				
	Fer2013	CREMA-D	RVDSR	JAFFE	CK48
Happy	8989	9861	10887	448	621
Sad	6077	8835	9814	496	336
Angry	4953	10672	9612	486	540
Disgust	547	9298	11223	464	531
Fear	5121	11576	10112	512	336
Surprise	4022	NA	NA	472	996
Neutral	6198	11067	8914	528	216
Total Images	35887	61309	61673	3406	3540

B. About Experimentation Setup and Resources

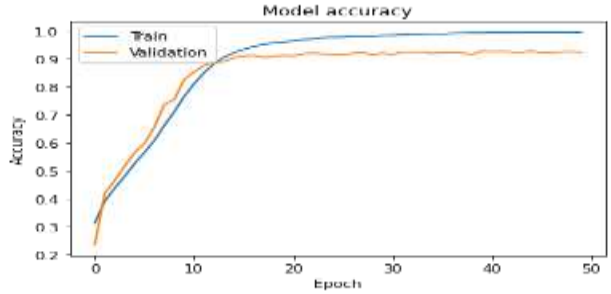
As the datasets consists of a huge number of images, implementation on the systems with general configuration will take more time. So, the support of Kaggle cloud platform was taken for performing the implementation that has the NVDIA GPU support of 16GB, CPU support of 13GB and memory support of 73.1GB, respectively.

B. Experimentation

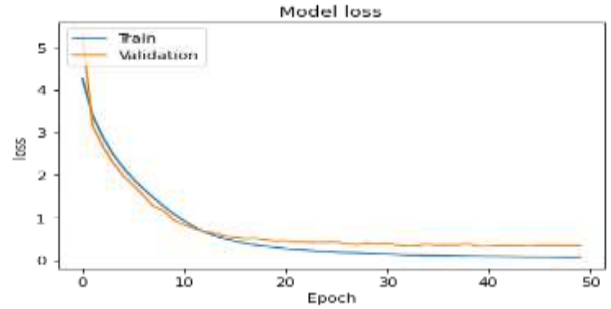
The implementation is done on the four datasets mentioned above in which three cases were considered. In Case 1, a Convolutional Neural Network model where the dropout layer was included in between the fully connected layers and after the flattening layer was considered. After the flatten layer, the dropout layer was added in between two dense layers. The amount of dropout percentage applied in between the fully connected layers is 0.25, whereas in between the dense layers is 0.5. Along with dropout l2. Kernel regularizer and l2. Bias regularizer of 0.01 and 0.01 were added to the convolutional and dense layers. Each fully connected layer consists of a convolutional layer, batch normalization layer and a max polling layer. Three fully connected layers, three dropout layers, one flatten layer and two dense layers are used to construct the model.

In Case 2, the same Convolutional Neural Network with slight modifications is used. In this case will have a dropout layer only between the denser layers and no dropout layers were used in between the fully connected layers. 0.5 is the dropout percentage applied in between the dense layers, l2. Kernel regularizer and l2. Bias regularizer are used in the same way as Case1. Whereas in Case 3 no Dropout layers were used, and the remaining considerations are the same as in Case 1 and Case 2. The total parameters used by the proposed convolutional network are 32,115,718 out of which 32,115,078 are trainable parameters and 640 are non-trainable parameters. The detailed description of the results obtained for all the three cases on the respective datasets is given in the tables and figures given below. An observation from the results after experimentation is overall performance of the model has been increased after using dropouts in between fully connected layers as well as dense layers and the over fitting and under fitting problems normally a CNN Model has been outshined by using dropouts, l2 kernel and bias regularization techniques.

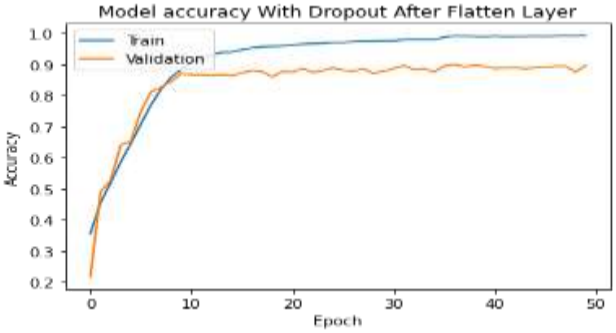
Fig. 2(a) to 2(f), 3(a to 3(f), 4(a) to 4(f) and 5(a) to 5(f) shows the Model Accuracy on FER 2013, RVDSR, CREMAD and own datasets in all the three cases which were explained above. Fig. 2(a) and 2(b), Fig. 3(a) and 3(b), Fig. 4(a) and 4(b), and Fig. 5(a) and 5(b) gives the Model Accuracy and loss in the case where dropout is used in between convolutional layers and in between dense layers, Fig. 2(c) and 2(d), Fig. 3(c) and 3(d), Fig. 4(c) and 4(d) and Fig. 5(c) and 5(d) gives model accuracy and loss in the case where dropout is used only after flattening the layer, i.e., between dense layers whereas Fig. 2(e) and 2(f), Fig. 3(e) and 3(f), Fig. 4(e) and 4(f) and Fig. 5(e) and 5(f) gives the model accuracy and loss of the case where dropout is not considered. It is observed that the case where dropouts are considered produced high accuracy and low loss when compared to other cases. The detailed description of Train and Test Accuracies, Train and Test Loss, Macro-average of Precision, Recall and f1-score, Weighted-average of Precision, Recall and f1-score are given in Table II to Table V. All the Experiments are done with 50 Epochs.



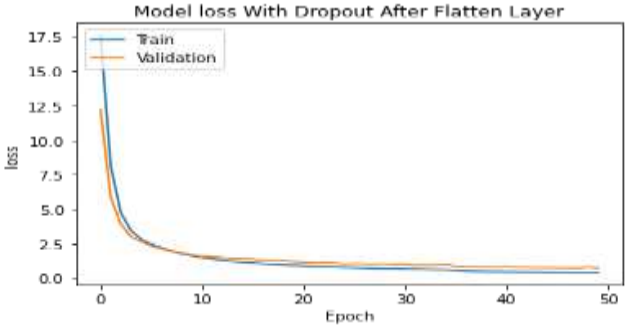
(a). Model Accuracy of the Proposed Model with Dropout on FER 2013 Dataset.



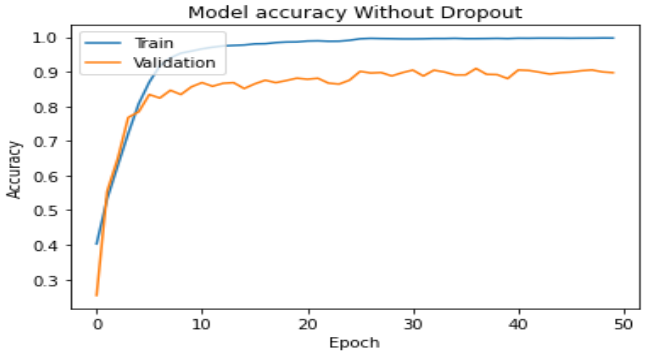
(b). Model Loss of the Proposed Model with Dropout on FER 2013 Dataset.



(c). Model Accuracy of the Proposed Model with Dropout only after Flatten Layer on FER 2013 Dataset.



(d). Model Loss of the Proposed Model with Dropout only after Flatten Layer on FER 2013 Dataset.



(e). Model Accuracy of the Proposed Model without Dropout on FER 2013 Dataset.

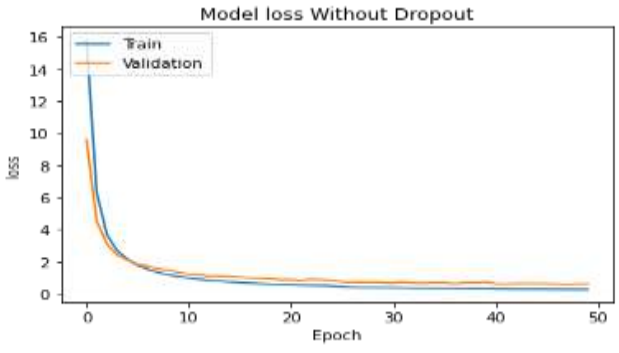
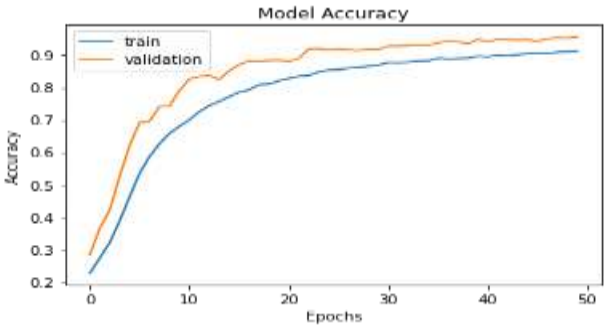
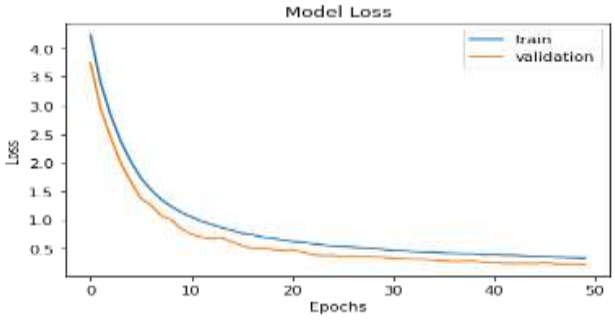


Fig. 2. (f). Model Loss of the Proposed Model without Dropout on FER 2013 Dataset.



(a) Model Accuracy of the Proposed Model with Dropout on RVDSR Dataset.



(b). Model Loss of the Proposed Model with Dropout on RVDSR Dataset.

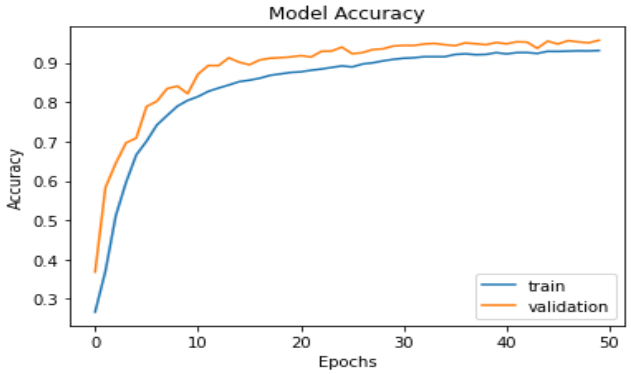
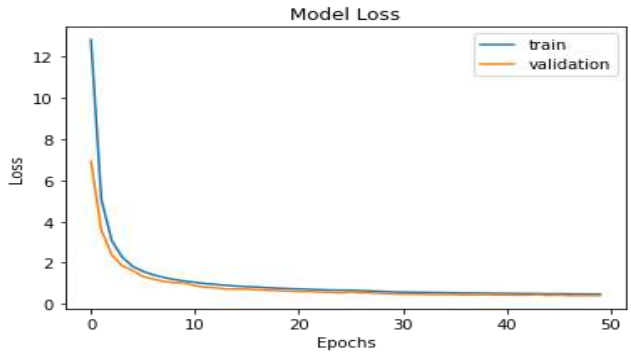
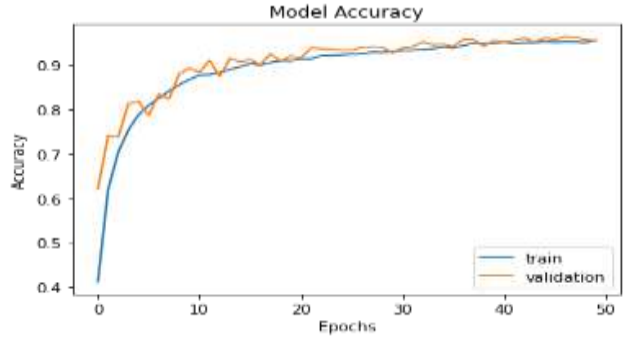


Fig 3(c). Model Accuracy of the Proposed Model with Dropout only after Flatten Layer on RVDSR Dataset.



(d). Model Loss of the Proposed Model with Dropout only after Flatten Layer on RVDSR Dataset.



(e). Model Accuracy of the Proposed Model without Dropout on RVDSR Dataset.

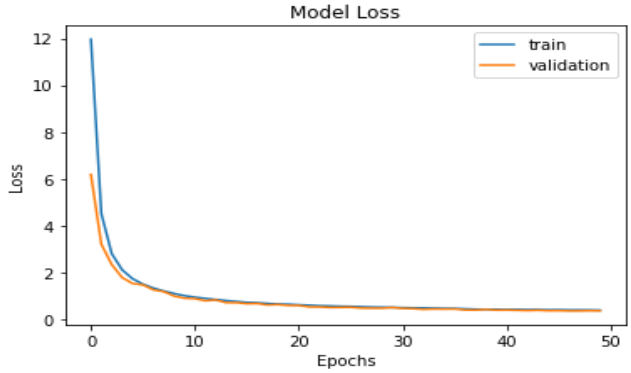
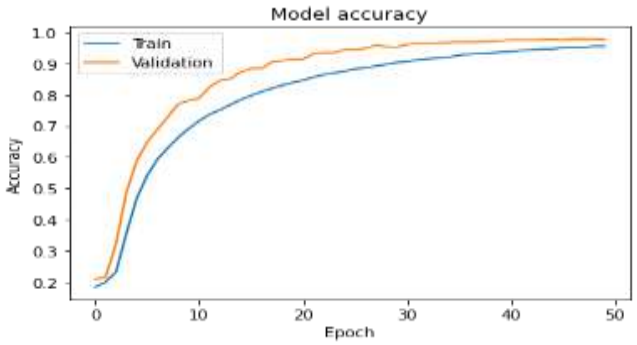
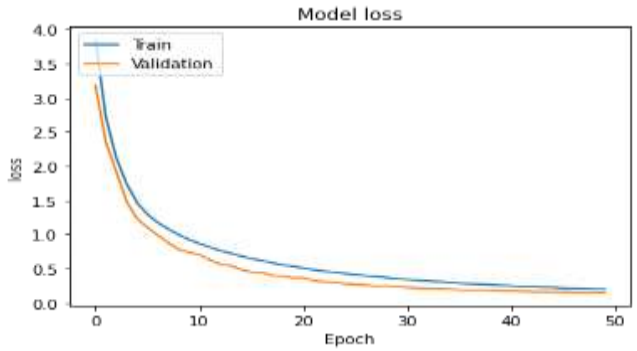


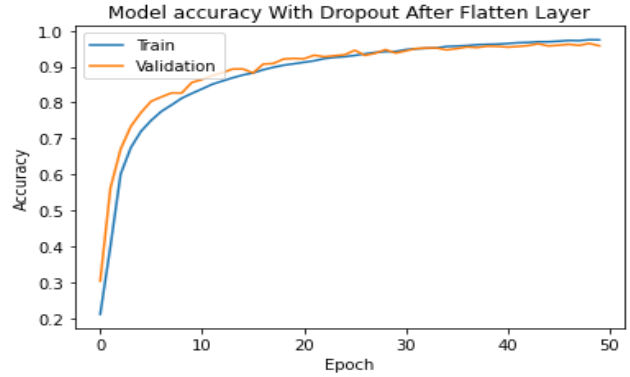
Fig. 3. (f). Model Loss of the Proposed Model Without Dropout on RVDSR Dataset.



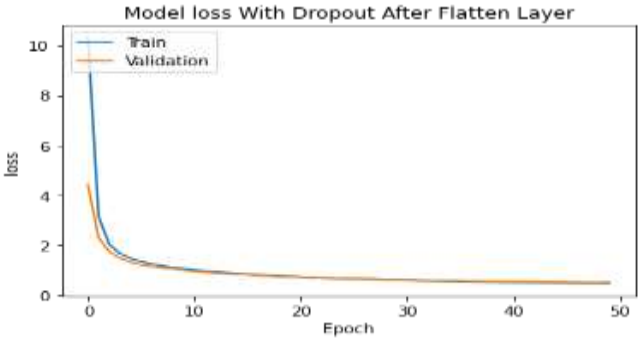
(a). Model Accuracy of the Proposed Model with Dropout on CREMA-D Dataset.



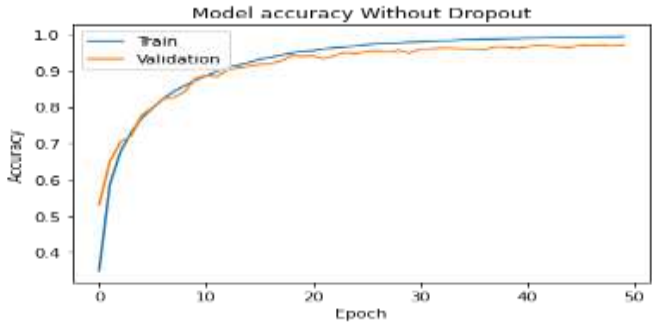
(b) Model Loss of the Proposed Model with Dropout on CREMA-D Dataset.



(c). Model Accuracy of the Proposed Model with Dropout only after Flatten Layer on CREMA-D Dataset.



(d). Model Loss of the Proposed Model with Dropout only after Flatten Layer on CREMA-D Dataset.



(e). Model Accuracy of the Proposed Model without Dropout on CREMA-D Dataset.

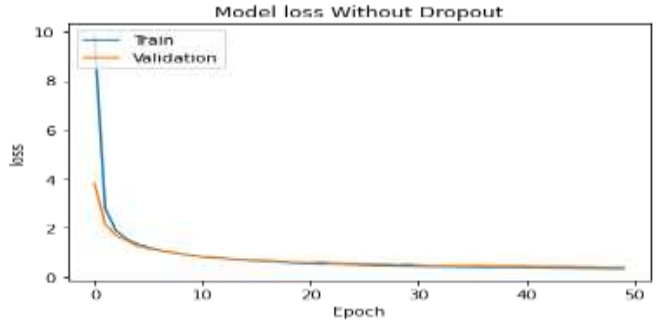
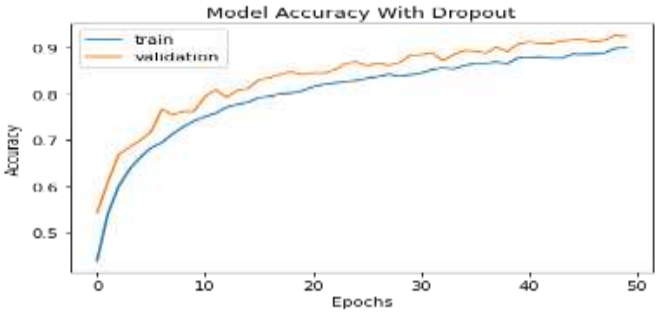
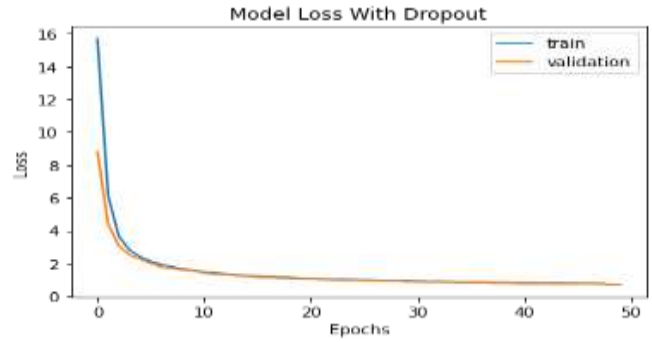


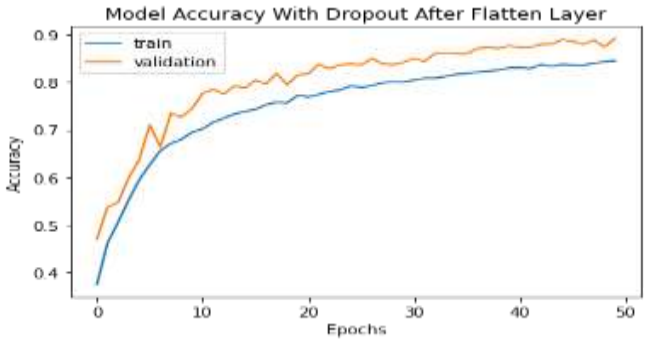
Fig. 4. (f). Model Loss of the Proposed Model without Dropout on CREMA-D Dataset.



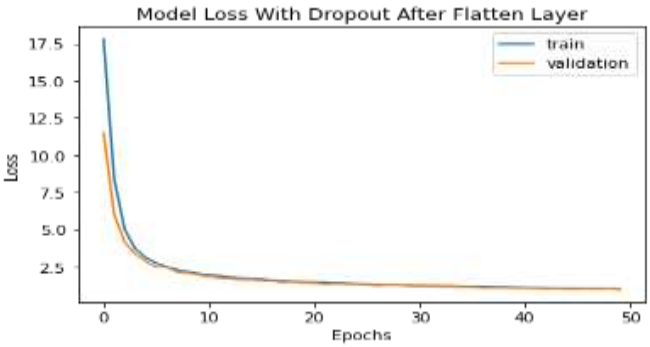
(a). Model Accuracy of the Proposed Model with Dropout on Own Dataset.



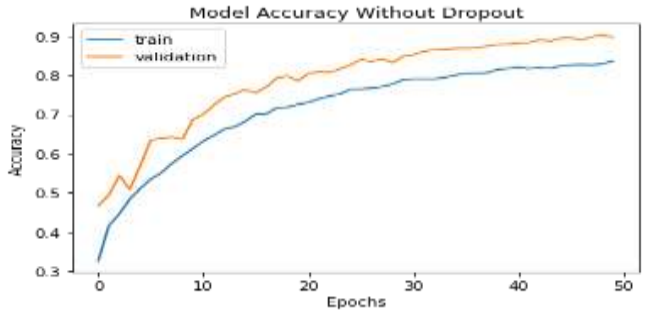
(b). Model Loss of the Proposed Model with Dropout on Own Dataset.



(c). Model Accuracy of the Proposed Model with Dropout only after Flatten Layer on Own Dataset.



(d). Model Loss of the Proposed Model with Dropout only after Flatten Layer on Own Dataset.



(e). Model Accuracy of the Proposed Model without Dropout on Own Dataset.

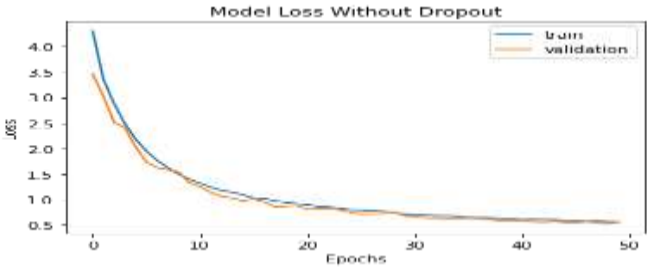


Fig. 5. (f). Model Loss of the Proposed Model without Dropout on Own Dataset.

Table VI gives the performance comparison of proposed CNN model over the existing work done and found that the proposed model obtained a better performance when the datasets Ck48, JAFFE, FER 2013, RVDSR, CREMAD and Own dataset are considered for evaluation.

TABLE II. PERFORMANCE COMPARISON OF THE MENTIONED THREE CASES ON FER2013 DATASET

Performance Measure	Case 1	Case 2	Case 3
Train Accuracy	99.32	99.38	99.74
Test Accuracy	92.33	89.61	89.24
Train Loss	0.0728	0.3908	0.2572
Test Loss	0.3248	0.7305	0.6737
Macro Avg of Precision	0.94	0.91	0.91
Macro Avg of Recall	0.93	0.90	0.91
Macro Avg of f1-Score	0.93	0.90	0.90
Weighted Avg of Precision	0.93	0.90	0.90
Weighted Avg of Recall	0.92	0.90	0.90
Weighted Avg of f1-Score	0.92	0.89	0.89
Time Taken Per Epoch	9 Sec	10 Sec	10 Sec

TABLE III. PERFORMANCE COMPARISON OF THE MENTIONED THREE CASES ON RVDSR DATASET

Performance Measure	Case 1	Case 2	Case 3
Train Accuracy	95.58	95.08	95.39
Test Accuracy	96.50	95.29	95.16
Train Loss	0.23	0.39	0.43
Test Loss	0.24	0.37	0.41
Macro Avg of Precision	0.96	0.95	0.94
Macro Avg of Recall	0.95	0.94	0.94
Macro Avg of f1-Score	0.95	0.94	0.93
Weighted Avg of Precision	0.95	0.94	0.93
Weighted Avg of Recall	0.94	0.93	0.93
Weighted Avg of f1-Score	0.94	0.93	0.92
Time Taken Per Epoch	55 Sec	56 Sec	57 Sec

TABLE IV. PERFORMANCE COMPARISON OF THE MENTIONED THREE CASES ON CREMAD DATASET

Performance Measure	Case 1	Case 2	Case 3
Train Accuracy	95.53	97.73	99.03
Test Accuracy	97.78	95.81	96.09
Train Loss	0.1978	0.4428	0.3676
Test Loss	0.1508	0.4995	0.4418
Macro Avg of Precision	0.97	0.96	0.96
Macro Avg of Recall	0.97	0.96	0.96
Macro Avg of f1-Score	0.97	0.96	0.96
Weighted Avg of Precision	0.97	0.96	0.96
Weighted Avg of Recall	0.97	0.96	0.96
Weighted Avg of f1-Score	0.97	0.96	0.96
Time Taken Per Epoch	123 Sec	125 Sec	126 Sec

TABLE V. PERFORMANCE COMPARISON OF THE MENTIONED THREE CASES ON OWN DATASET

Performance Measure	Case 1	Case 2	Case 3
Train Accuracy	90.52	86.86	89.91
Test Accuracy	92.45	89.68	89.19
Train Loss	0.6938	0.9604	0.4292
Test Loss	0.7083	0.9764	0.5534
Macro Avg of Precision	92	89	89
Macro Avg of Recall	92	89	89
Macro Avg of f1-Score	91	89	88
Weighted Avg of Precision	92	89	88
Weighted Avg of Recall	92	89	88
Weighted Avg of f1-Score	91	88	87
Time Taken Per Epoch	43 Sec	45 Sec	47 Sec

TABLE VI. PERFORMANCE COMPARISON OF PROPOSED CNN MODEL WITH EXISTING WORK DONE

Name of The Author	Datasets Used	Percentage of Test Accuracy	Proposed Model Test Accuracy
Mollahosseini et al. [14]	CK48	93.2	FER2013: 92.33 CK48: 99.44 JAFFE: 98.68 CREMAD: 97.78 RVDSR: 96.50
Mollahosseini et al. [14]	FER 2013	61.1	
Lopes et al. [15]	CK48	96.76	
Mohammadpour et al. 16]	CK48	97.01	
Cai et al. [17]	JAFFE	95.04	
Cai et al. [17]	CK48	96.87	
Agarwal et al. [19]	FER 2013	65%	
Deepak jain et al. [20]	JAFFE	95.23	
Deepak jain et al. [20]	CK48	93.24	
Rory Beard et al. [23]	CREMAD	65	
Rory Beard et al. [23]	RVDSR	58.33	

V. CONCLUSION AND FUTURE WORK

It is demonstrated in the proposed method that the use of dropout handled overfitting, resulting in a 3.09 percent gain in test accuracy in the category of CNN Model that uses dropout in fully connected convolutional layers and a 0.37 percent gain in test accuracy in the other category of CNN Model that uses dropout in dense layers using FER2013 dataset. Also, it has been noticed that there is a significant improvement of test accuracy for other datasets namely JAFEE, CK48, RVDSR, CREMAD and a self-prepared. Some FER recent developments in the literature were compared to the proposed model and found to be greater due to the implementation of dropout, as shown in Table VI.

The future scope of this paper will be to investigate the trade-off between overfitting and underfitting for FER by CNN models and developing mathematical models for managing a percentage of overfitting and or underfitting to achieve higher accuracy.

REFERENCES

[1] Bodavarapu, Pavan Nageswar Reddy, et al. "Optimized Deep Neural Model for Cancer Detection and Classification Over ResNet." *Smart Technologies in Data Science and Communication*. Springer, Singapore, 2021. 267-280.

[2] Mandhala, Venkata Naresh, et al. "Object detection using machine learning for visually impaired people." *International Journal of Current Research and Review* 12.20 (2020): 157-167.

[3] Mishra, et al. "Crowd Counting Using CSR-Net Architecture," *International Journal of Advanced Trends in Computer Science and Engineering*, Vol 8, no. 6, p. 2762-2767, 2019, 10.30534/ijatcse/2019/13862019.

[4] NarasingaRao, M. R., et al. "A survey on prevention of overfitting in convolution neural networks using machine learning techniques." *Int. J. Eng. Technol* 7.2.32 (2018): 177-180.

[5] RC Gonzalez, et al., "Digital Image Processing, Global Edition.", 2017, ch. 4, p. 203–303.

[6] Pan, Xin, and Jian Zhao. "A central-point-enhanced convolutional neural network for high-resolution remote-sensing image classification." International Journal of Remote Sensing 38.23 (2017): 6554-6581.

[7] Ashok Kumar, P. M., Jeevan Babu Maddala, and K. Martin Sagayam. "Enhanced Facial Emotion Recognition by Optimal Descriptor Selection with Neural Network." *IETE Journal of Research* (2021): 1-20.

[8] Videla, Lakshmi Sarvani, and PM Ashok Kumar. "Facial Expression Classification Using Vanilla Convolution Neural Network." 2020 7th International Conference on Smart Structures and Systems (ICSSS). IEEE, 2020.

[9] Srinivas, P. V. V. S., and Pragnyaban Mishra. "Facial Expression Detection Model of Seven Expression Types Using Hybrid Feature Selection and Deep CNN." *International Conference on Intelligent and Smart Computing in Data Analytics: ISCDA 2020*. Springer Singapore, 2021.

[10] Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." *The journal of machine learning research* 15.1 (2014): 1929-1958.

[11] Hinton, Geoffrey E., et al. "Improving neural networks by preventing co-adaptation of feature detectors." *arXiv preprint arXiv:1207.0580* (2012).

[12] Lucey, Patrick, Jeffrey F. Cohn, Takeo Kanade, Jason Saragih, Zara Ambadar, and Iain Matthews. "The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression." In *2010 ieee computer society conference on computer vision and pattern recognition-workshops*, pp. 94-101. IEEE, 2010.

[13] Wolfram Research, "FER-2013" from the Wolfram Data Repository, 2018.

[14] Mollahosseini, Ali, David Chan, and Mohammad H. Mahoor. "Going deeper in facial expression recognition using deep neural networks." *2016 IEEE Winter conference on applications of computer vision (WACV)*. IEEE, 2016.

[15] Lopes, André Teixeira, et al. "Facial expression recognition with convolutional neural networks: coping with few data and the training sample order." *Pattern Recognition* 61 (2017): 610-628.

[16] Mohammadpour, Mostafa, et al. "Facial emotion recognition using deep convolutional networks." *2017 IEEE 4th international conference on knowledge-based engineering and innovation (KBEI)*. IEEE, 2017.

[17] Cai, Jun, et al. "Facial expression recognition method based on sparse batch normalization CNN." *2018 37th Chinese Control Conference (CCC)*. IEEE, 2018.

[18] Lyons, Michael, Miyuki Kamachi, and Jiro Gyoba. "Japanese female facial expression (JAFFE) database.", 2017.

[19] Agrawal, Abhinav, and Namita Mittal. "Using CNN for facial expression recognition: a study of the effects of kernel size and number of filters on accuracy." *The Visual Computer* 36.2 (2020): 405-412.

[20] Jain, Deepak Kumar, Pourya Shamsolmoali, and Paramjit Sehdev. "Extended deep neural network for facial emotion recognition." *Pattern Recognition Letters* 120 (2019): 69-74.

[21] Cao, Houwei, David G. Cooper, Michael K. Keutmann, Ruben C. Gur, Ani Nenkova, and Ragini Verma. "Crema-d: Crowd-sourced emotional multimodal actors dataset." *IEEE transactions on affective computing* 5, no. 4 (2014): 377-390.

[22] Livingstone, Steven R., and Frank A. Russo. "The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English." *PloS one* 13, no. 5 (2018): e0196391.

[23] Beard, Rory, et al. "Multi-modal sequence fusion via recursive attention for emotion recognition." *Proceedings of the 22nd Conference on Computational Natural Language Learning*. 2018.

[24] Athanasiadis, Christos, Enrique Hortal, and Stylianos Asteriadis. "Audio–visual domain adaptation using conditional semi-supervised Generative Adversarial Networks." *Neurocomputing* 397 (2020): 331-344.

[25] Ghaleb, Esam, Mirela Popa, and Stylianos Asteriadis. "Multimodal and temporal perception of audio-visual cues for emotion recognition." *2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII)*. IEEE, 2019.

[26] Ristea, Nicolae-Cătălin, Liviu Cristian Duțu, and Anamaria Radoi. "Emotion recognition system from speech and visual information based on convolutional neural networks." *2019 International Conference on Speech Technology and Human-Computer Dialogue (SpeD)*. IEEE, 2019.

[27] LeCun, Yann, John S. Denker, and Sara A. Solla. "Optimal brain damage." *Advances in neural information processing systems*. 1990.

[28] An, Feng-Ping, and Jun-E. Liu. "Medical Image Segmentation Algorithm Based on Optimized Convolutional Neural Network-Adaptive Dropout Depth Calculation." *Complexity* 2020 (2020).