Received: March 17, 2021. Revised: May 12, 2021.

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Convolutional Neural Network Based Facial Expression Recognition Using Image Filtering Techniques

Ritanshi Agarwal[1*] Neha Mittal[1] Hanmandlu Madasu[2]

[1]*Electronics and Communication Engineering, Meerut Institute of Engineering and Technology, Meerut, UP, India*
[2]*Computer Science and Engineering Department,*
Maturi Venkata Subba Rao Engineering College, Nadergul, Hyderabad, India
* Corresponding author's email: ritanshiagarwal31@gmail.com

Abstract: Facial expressions are indicative of one's mood that serves to communicate his/her the status of mind. Facial expression recognition poses problems when we attempt to use Convolutional Neural Network (CNN) architectures. The choice of a particular CNN architecture/model and the design of its internal architecture are two problems that need to be addressed. We have chosen a simple CNN architecture comprising two convolutional layers and two pooling layers. To improve its performance the input facial images displaying emotions are subject to two kinds of filtering: One by Gaussian filter and another by LoG. The classification of seven emotions by the CNN model on the filtered facial images from Extended Cohn-Kanade (CK+) dataset comprising 981 images has led to the attainment of 100% recognition accuracy thus vindicating the effectiveness of the proposed approach.

Keywords: Facial expression recognition, CNN, Gaussian, Laplacian of gaussian, CK+ dataset.

1. Introduction

The ability to identify the emotional state of a person through its body language is a part and parcel of the human endeavour. However, we rely more on facial expressions to infer the emotions of others. Facial expressions are a natural and spontaneous reaction of an individual to the situations in the social contexts and they serve as a medium to vent feelings without recourse to the native languages. To have better interpersonal relations, impressing others through facial expressions is necessary.

The emotion recognition along with security and safety features can be used to testify criminals, patients, etc. in a faster and more accurate manner. Recognizing facial emotion has always been a challenge for the past few decades. To detect emotions, the initial stage is to detect a face followed by facial expression recognition (FER). The most common face detector is due to Viola-Jones [1]. There are seven basic emotions such as anger, fear, surprise, sorrow, happiness, disgust and contempt which need to be recognized by machine learning using the facial expressions as input. To this end, Deep learning neural networks are highly favoured for the extraction of features from images followed by the classification using the inbuilt classifier called Softmax. They include: Convolutional neural networks (CNN), Deep CNN, HybridNet (HiNet) [2], etc. As we know that facial expressions are the result of the subtle movement of eyes, nose and mouth, we attempt to use these networks for capturing the changes in the face.

We will briefly review some approaches based on deep learning architectures for feature extraction. Reference [3] has proposed a part-based bidirectional hierarchical recurrent neural network (PHRNN) and multi-signal CNN (MSCNN) as spatio-temporal networks with a view to represent the dynamic variations of a face but these networks are unable to do so for emotions like fear and sad. One reason could be that the features from them have no ability to capture the perceptible changes in the faces. The Facial Expression Recognition (FER) must distinguish among the categories of emotions and to

accomplish this goal the system has to resolve the issue of different subjects displaying the same emotion and the same subject displaying different emotions. A novel identity-aware CNN (IACNN) is presented in [4], which learns the expressions and their identities. Xie and Hu have modified a CNN structure into two-branch framework termed as Deep comprehensive multi-patches aggregation CNN (DCMA-CNN) [5] to extract features from patches as well as from the whole face image and it shows good performance in recognizing fear and sad emotions of faces from CK+ dataset in spite of the fact that they possess the similar local patches. A weighted mixture deep neural network (WMDNN) is developed in [6] to recognize the facial expressions by using a weighted fusion of the partial VGG16 and shallow CNN. Reference [7] has implemented a shallow CNN without any pre-processing step on different publicly available datasets like CK+ (97.32%), JAFFE (77.27%) and YALE FACE and obtained the recognition rate of only 31.82%. A descriptive survey on deep FER can be found in [8].

Local Binary Patterns (LBPs) are the descriptors commonly used to elicit features from images. The techniques to generate LBP and their usage in FER can be found in [6, 7, 9, 10]. The Watermark Deep Neural Network [WMDNN} uses LBP in one of its fused CNN network [6]. The support vector machine (SVM) classifier with polynomial kernel function on LBP features of facial expressions from CK+ dataset produces an accuracy of 91.66% [7]. These patterns [9] and image based processes like coupled Gaussian [11] can be used for multi-view FER for the development of pose-invariant methods. Hierarchical deep neural network (DNN) also uses LBP images because of their simpler structure [12] but low accuracies are a curse of LBP features drawn from CK dataset in [13] whatever might be the classifier. There are many other image-based models like coupled Gaussian [11], facial feature points [14-16], features from salient facial patches [17, 18], etc. for recognizing the facial expressions These models aim at the maximum possible combination of facial action units (AUs) with a view to better the classification of emotion.

Image enhancement techniques like deblurring, contrast enhancement, sharpening, etc. bring out details from an image thereby helping us extract the effective features. The Gaussian processes for view-invariant facial expression recognition are presented in [11] and [19]. Facial emotions can be viewed as facial curves, so the edge detection could be one of the preferred approaches for pre-processing of images. Gaussian filter has several remarkable properties as mentioned in [20] hence it finds applications such as noise-reduction, edge-detection, etc. For example, Marr-Hilderith [21] operator called Laplacian of Gaussian is used for the edge-detection. In this paper, we have used the Gaussian and LoG filters at the pre-processing step on the facial expressions before inputting to the CNN architectures.

This paper is organized into five sections. Section 2 describes the proposed methodology in detail including a mathematical analysis of convolution and pooling operations in the CNN model. Section 3 contains the results of experimentation on the dataset and their analysis. Section 4 makes a comparison of the proposed method with the previous CNN based approaches and Section 5 gives the conclusions including a glimpse of the future scope.

2. Proposed methodology

This paper presents the facial expression recognition (FER) using pre-processing of images using Gaussian, and Laplacian of Gaussian (LoG) on different window sizes (3x3, 5x5, 7x7, 9x9). The feature extraction and classification are accomplished by the Convolutional Neural Network (CNN) on the pre-processed facial images. The architecture consists of convolutional layers with Rectified Linear Unit as the activation function, max-pooling layers, dense hidden layers and dense output layer with Softmax as the classifier. Figure 1 shows the steps of the proposed approach.

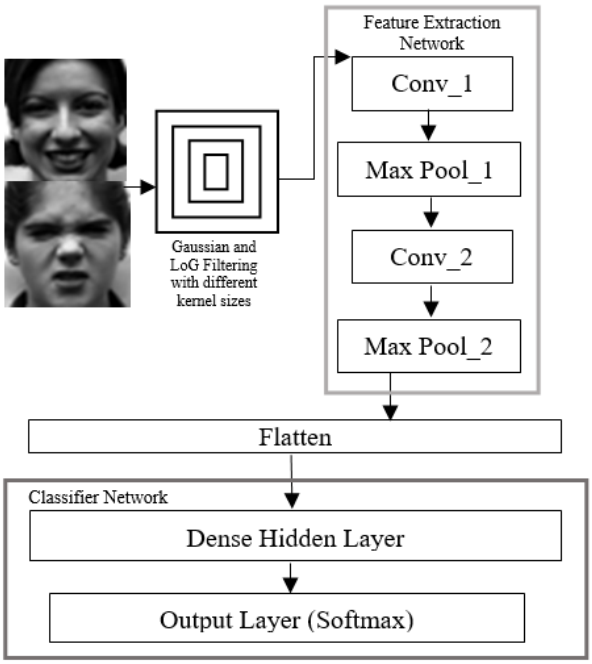


Figure. 1 Flowchart of the proposed approach

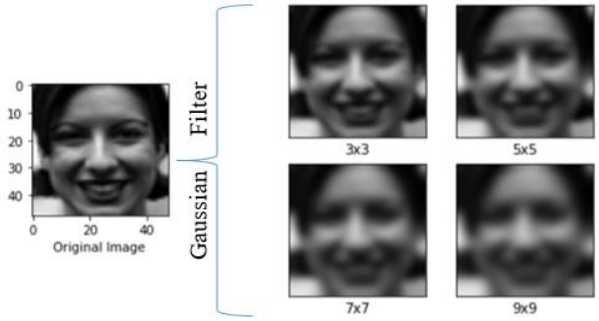


Figure. 2 Filtered images for different-sized Gaussian kernel

2.1 Pre-processing using digital filters

A filter is a 2-D mask which filters out the unwanted noise and preserves the desired signal. This mask is entrusted with the convolution operation on the image which can be performed either in overlapping or non-overlapping mode. For an understanding about how convolution works you may refer to [22].

2.1.1. Gaussian filtering

Gaussian filter smoothens/blurs the image and reduces the noise. We have tried 4 sizes for the Gaussian kernel to select a suitable size by experimentation. A filter performs a convolution operation to smoothen an image by making use of its separability property in the Cartesian plane for the ease of computation. The results for Gaussian kernels are depicted in Fig. 2**.**

2.1.2. Laplacian of Gaussian(LoG)

Laplacian filter is a second-order derivative linear filter mainly known for sharpening of images. An edge results from an abrupt change in the intensities at the pixel locations in the image. Laplacian identifies an edge pixel where the zero crossings of second derivative of image exist. LoG uses both Gaussian and Laplacian filters for reducing the noise content and sharpening of images respectively. In this work, firstly an image is filtered using Gaussian operator and secondly the zero-crossings of the second order derivatives of Laplacian filter are determined to yield edge pixels. The LoG filtered images are portrayed in Fig. 3.

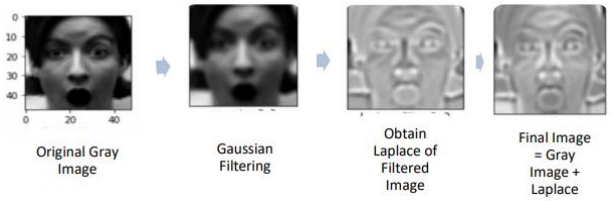


Figure. 3 The output images after applying LoG on an image

Table 1. CNN architecture

Input	Layer (Type)	Activation	Output
48x48	Convolution (3x3, 16 filters)	ReLU	46x46x16
	Pooling (2x2)	-	23x23x16
23x23	Convolution (3x3, 28 filters)	ReLU	21x21x28
	Pooling (2x2)	-	10x10x28
10x10x28	Flatten	None	2800
2800	Dense (Hidden)	ReLU	128
	Dropout (20%)	-	
	Dense (Output)	Softmax	7

2.1.3. The CNN architecture

The feature extraction and classification are the two main tasks performed on the pre-processed images by the chosen CNN architecture comprising two convolutional layers each of which is followed by a pooling layer of size 2x2. We have used both max pooling and average pooling to reduce the sizes of the convolved images called the feature maps. These are fed into the 2-layer classifier called Softmax consisting of a hidden dense layer of 128 nodes followed by 20% dropouts and an output layer of 7 nodes for 7 classes. The details of CNN architecture are given in Table 1. We have opted OpenCV and Tensor flow in Python to implement.

2.2 Feature extraction and classification

2.2.1. Convolutional layers

The enhanced images after the filtration are transferred to CNN architecture. As the number of feature maps generated is equal to the number of filters used; we have chosen 16 and 28 filters in the first and the second convolutional layers respectively to limit the computational cost.

2.2.2. Rectified linear unit

We have selected an activation function called Rectified Linear Unit (ReLU) as it can suppress the

negative content or noise by setting it to zero, expressed as:

$$ReLU(x) = max(x, 0) \quad (1)$$

2.2.3. Pooling layer

Pooling reduces the size of a feature map by taking either the max or an average of all the values in a mask of 2x2 with a stride of 2. The mask is moved row-wise on the feature map until all rows are covered.

2.2.4. Fully connected layers

The reduced feature map from the pooling layer is flattened into 1D array/list and entered into Dense layer of 128 nodes followed by the activation layer employing ReLU. There is a chance of overfitting that happens during the training of the network when the training and validation accuracies begin to differ. In this situation, the trained network cannot classify the test images correctly. One of the reasons for overfitting is that the neighbouring nodes in the layer may land up in the similar weights. To avoid this situation, we have used 20% dropout which means that 20% of the nodes are discarded randomly in the penultimate layer of the model. There are 7 classes of facial expressions in CK+ dataset, so the output dense layer consists of 7 nodes with Softmax as the inbuilt classifier in CNN. The Softmax calculates the probability for each class based on the trained features. The class with the highest probability receives a label, i.e. is identified with the class of the test image.

2.3 Mathematical footing

As mentioned above that at the preprocessing phase, we can choose either Gaussian filter or LoG filter to operate on the facial images to make them amenable for feature extraction by CNN. The equations for the Gaussian and LoG filters are as follows:

$$G(x, y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\left\{\frac{x^2+y^2}{\sigma^2}\right\}} \quad (2)$$

Omitting the normalization constant and taking the second order derivative w.r.t. both x and y, we get LoG as:

$$LoG(x, y) = \left[\frac{x^2 + y^2 - 2\sigma^2}{\sigma^4}\right] e^{-\left\{\frac{x^2+y^2}{\sigma^2}\right\}} \quad (3)$$

Where (x, y) is a pixel location and σ^2 is the variance. For implementing these filters, we make use of their kernels/masks of different sizes, kxk; with k=3, 5, 7,9. Then K=k*k is the number of pixel intensities, in the sub image centred at location (x, y) and w_j denotes the weights in the kernel. Representing both z_j and w_j as vectors, the filtering operation yields the modified pixel intensity at (x, y) position as:

$$F(x, y) = \sum_{j=1}^{K} z_j \, w_j \quad (4)$$

Thus convolving the kernel centered at every pixel location leads to the filtered image F. The image size is reduced on convolving resulting in "valid" size. By padding zeros on both column-wise and row-wise on the original image, we can maintain the "same" size during the convolution operation. This image is input to the CNN model. As we have two convolutional layers along with ReLU functions and two pooling layers in the CNN architecture, we will see what happens to the filtered image. In the first convolutional layer along with ReLU, the pixel intensity, $F(x, y)$ at (x,y) is modified to the following:

$$F_{c1}(x, y) = Re\,L\,U\left[\sum_{j=1}^{K1} z_{c1,j} \, w_{c1,j}\right] \quad (5)$$

Where $F_{c1}(x, y)$ is the modified pixel intensity because of convolving a kernel with weights, $w_{c1,j}$ on the sub image $z_{c1}(x, y)$ of size K1=k1xk1 in the filtered image $F(x, y)$ centred at (x, y). As ReLU(z)=max(0, z), it only eliminates the negative values. If kernel function gives a negative value for a sub image, it is set to zero by ReLU. As a membership function cannot be negative, Eqns. (5) represents the membership function value for a sub image. At the end of convolution operation on every pixel in the filtered image, we get the feature map denoted by F_{c1}. Computation of $F_{c1}(x, y)$ involves a k1xk1 window. We will generate several feature maps by applying different filters at the convolutional layer. Next we apply the pooling operation which can either be max or average. For this, each feature map F_{c1} is divided into blocks of size b1xb1 such that each element of this block is derived from a window size of k1xk1. Let us consider each time a block and then select the max value using max pooling or compute an average value by the average pooling. These operations are expressed as follows:

$$P_{c1}(x,y) = Max[F_{c1}]_{k1xk1} \quad (6)$$

$$P_{c1}(x,y) = \frac{1}{K1}\sum_{i=1}^{k1}\sum_{j=1}^{k1} F_{c1,ij} \quad (7)$$

It may be noted that Eqs. (6) and (7) provide the max membership function and an average membership function respectively from a block. After pooling we will have a feature map P_{c1}comprising either the max or average membership function values as per our choice.

We will now move to the second convolutional layer and repeat Eq. (5) at this layer to get the outcome (i.e. the modified pixel intensity) arising out of convolution with the kernel centred at (x, y) as given by:

$$F_{c2}(x,y) = Re\,L\,U\left[\sum_{j=1}^{K2} z_{c2,j}\, w_{c2,j}\right] \quad (8)$$

where $z_{c2,j}$ are the pixel intensities in the sub image of P_{c1} with size K2=k2xk2 and $w_{c2,j}$are the weights. Note that the computation of $F_{c2}(x,y)$ involves a kernel of size k2xk2 that operates on the sub image of P_{c1} to yield a single value from Eq. (8). The number of operations performed is K2 such that each operation involves a window of size B1=b1xb1 in the first pooled image, P_{c1} with the number of windows covered in P_{c1} is therefore K2xB1 and the number of windows covered in F_{c1} is K2xB1xK1. As a result, increasing the number of convolutional layers will result in covering most of the original input image, $F_{c1}$to CNN in each operation. To make it more clear we note that the membership function values from the second convolutional layer cover more image area (windows) of F_{c1} than that covered by the membership function values from the first convolutional layer in F_{c1}. One drawback of this increased covering is that the local information gives rise to the global information.

The two pooling operations are given by.

$$P_{c2}(x,y) = Max[F_{c2}]_{k2xk2} \quad (9)$$

$$P_{c2}(x,y) = \frac{1}{K2}\sum_{i=1}^{k2}\sum_{j=1}^{k2} F_{c2,ij} \quad (10)$$

The pooling helps select either max or average from each block of the feature map thereby modifying the membership functions. Whatever may be the CNN model, the final membership functions are a mix of pixel intensities and weights. This mathematical footing will be used in the results' section to offer justification for the results of human emotion classification.

3. Results of experimentation

3.1 Dataset

This work is carried out on Extended Cohn-Kanade dataset [23] (Commonly called CK+ dataset) consisting of 118 subjects with 7 classes, i.e., Anger (AN), Contempt (CO), Disgust (DI), Fear (FE), Happy (HA), Sadness (SA) and Surprise (SU). We have considered 327 sequences of subjects and taken the last three frames of each sequence to get a total of 981 images each of which is a gray scale image of size 48x48. The count of samples for each emotion is as follows: 135 of AN, 54 of CO, 177 of DI, 75 of FE, 207 of HA, 84 of SA and 249 of SU. The database is divided into the training set of 70% samples, validation set of 15% and testing set of 15%. Fig. 4 shows typical images of all emotions from the dataset.

3.2 Image augmentation

A smaller dataset falls prey to overfitting while training the CNN (architecture) model. To mitigate this problem, the image augmentation is the last resort. The operations such as zoom, shear, horizontal and vertical flip; rescale, rotation, etc. help increase the dataset for training and/or validation. In Python, the image augmentation can be achieved without affecting the original data using the function called ImageDataGenerator from tensorflow library. We have used only the horizontal flip in the training data because the frontal view of a human face has symmetry.

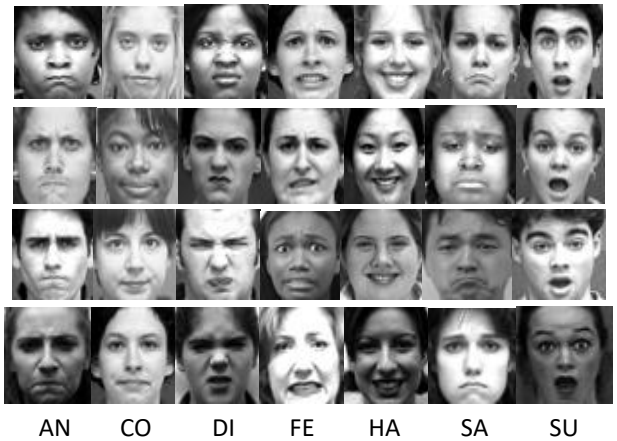


Figure. 4 Images from CK+ dataset

3.3 Results

This paper utilizes Gaussian and LoG filters of different kernel sizes at the pre-processing stage and they affect the average recognition rates of CNN model as shown in Figs. 5 and 6 for the case of max pooling and in Figs. 7 and 8 for the case of average pooling. Note that Figs. 5 and 7 correspond to Gaussian filter whereas Figs. 6 and 8 to LoG filter. It is observed that the filtering methods have helped improve the recognition accuracies achievable with the CNN model. The Gaussian kernel of size 3x3 in the Gaussian filter leads to the high recognition rate of 99.32% whereas the Gaussian kernel of sizes 7x7 in LoG is found to give the best accuracy of 100% on CK+ dataset under max pooling in the CNN model. The Gaussian kernels of sizes 3x3 and 5x5 in Gaussian filter yield the maximum accuracy of 97.959% while the Gaussian kernels of sizes 3x3, 5x5 and 7x7 in LoG bestow the best accuracy of 100% on the same dataset under average pooling. As Gaussian filter has the smoothening property and Laplacian operator has the edge detection property, their combination in the form of LoG [21] imbibes both these properties which not only smooth the faces but also yield edges from the facial curves that are hallmarks of emotions. Hence the best performance of CNN model is attributed to the filtered facial images using LoG.

Without pre-processing, the performance of CNN model is inferior. Tables 2 and 3 show the accuracies obtained with this model for Gaussian kernels of different sizes in both Gaussian and LoG filters using max pooling and average pooling respectively.

3.4 Performance analysis

There are several measures such as sensitivity, specificity and recognition accuracy/rate to evaluate the performance of a model. These measures depend on how the images are classified correctly or incorrectly. If the number of images belonging to a class is classified correctly then it is termed as true positive (TP) but if they are falsely classified then they come under false positive (FP). On the other hand, if the number of images belonging to a different class is classified correctly then it is termed as true negative (TN), but if they are classified falsely then they come under false negative (FN). We also need the weighted average of all classes. Using the above terms, we now define the performance measures. Sensitivity refers to the true positive rate or recall that tells how much the model is able to recognize a particular class correctly. Specificity refers to the true negative rate that tells how much the model is able to recognize other classes correctly. The recognition rate or accuracy is the ability of the model to recognize how many classes correctly.

The performance measures are expressed as:

$$Sensitivity = \frac{TP}{TP + FN} \tag{11}$$

Tables 4 and 5 show the sensitivity and specificity of the results of CNN model under the max pooling

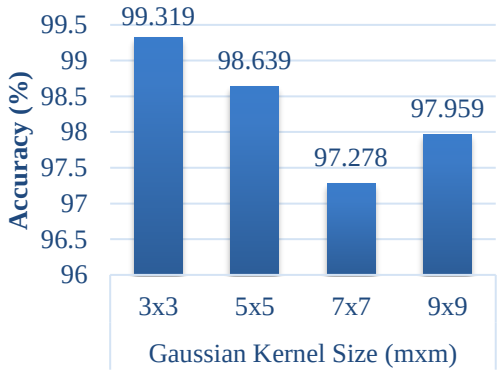


Figure. 5 Accuracy (%) of different Gaussian Kernels on CK+ dataset using Max Pooling

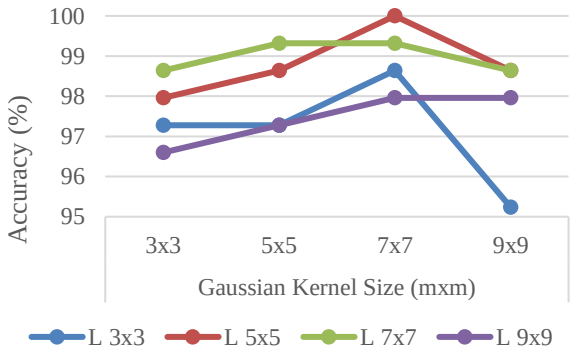


Figure. 6 Accuracy (%) of LoG kernels on CK+ dataset using Max Pooling

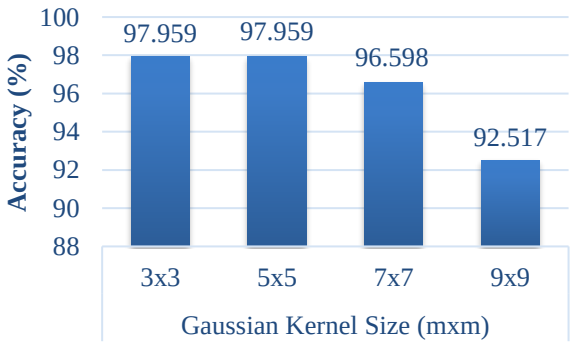


Figure. 7 Accuracy (%) of different Gaussian Kernels on CK+ dataset using average pooling

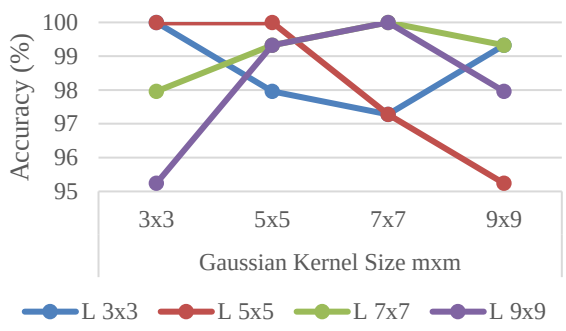


Figure. 8 Accuracy (%) of LoG kernels on CK+ dataset using average pooling

Table 2. Accuracy (%) obtained on CK+ dataset (Max Pooling)

Kernel Size	**Gaussian + CNN**	**LoG + CNN**			
		3x3	**5x5**	**7x7**	**9x9**
	Accuracy(%)	**Accuracy(%)**			
3x3	**99.32**	97.28	97.96	98.64	96.60
5x5	98.64	97.28	98.64	99.32	97.28
7x7	97.28	98.64	**100.00**	99.32	97.96
9x9	97.96	95.24	98.64	98.64	97.96

Table 3. Accuracy (%) obtained on CK+ dataset (Average Pooling)

Kernel Size	**Gaussian + CNN**	**LoG + CNN**			
		3x3	**5x5**	**7x7**	**9x9**
	Accuracy(%)	**Accuracy(%)**			
3x3	97.959	**100**	**100**	97.96	95.24
5x5	97.959	97.96	**100**	99.32	99.32
7x7	96.598	97.28	97.28	**100**	**100**
9x9	92.517	99.32	95.24	99.32	97.96

and average pooling respectively for different kernel sizes in both Gaussian and LoG filters.The confusion matrix is a tabular presentation of the performance of a model. Fig. 9 shows the confusion matrix of the model when its input image (facial expression) is pre-processed by Gaussian kernel of size 3x3. In this, the sensitivity falls down because of misclassification of SU.

$$Specificity = \frac{TN}{TP + FP} \tag{12}$$

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} \tag{13}$$

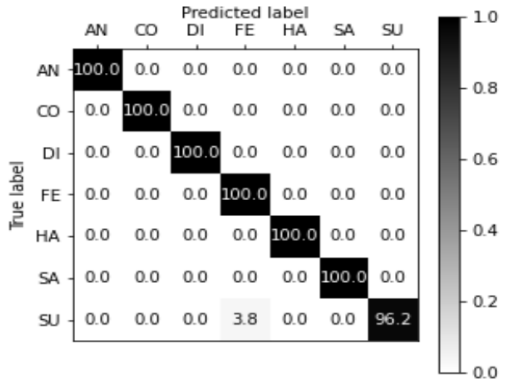
Figs. 10-12 show the confusion matrices of the CNN model when its inputs are the filtered images using LoG. In this the Gaussian kernels are varied to have sizes of 3x3, 7x7 and 9x9 but with the fixed Laplacian kernel of size 7x7. It can be seen that CO, FE, HA, SA and SU emotions are well classified except for DI that is misclassified as CO for some of the combinations of Gaussian and Laplacian kernels. This is because the facial expressions of AN, DI and CO have similarities which can be noticed from the visual inspection of the dataset. These similarities are reflected in the reduced values of sensitivity and specificity. The CNN model in this case achieves 100% accuracy for all emotion classes.

3.5 Mathematical justification for the results

As noted in the mathematical footing that the use of small kernel sizes in one or two convolutional layers preserves the local information is vindicated by the best performance obtained with 3x3 and 7x7 kernels. But increasing the kernel size to 9x9 deteriorates the performance of CNN model. This is also true for both filters that work better with small kernel sizes.

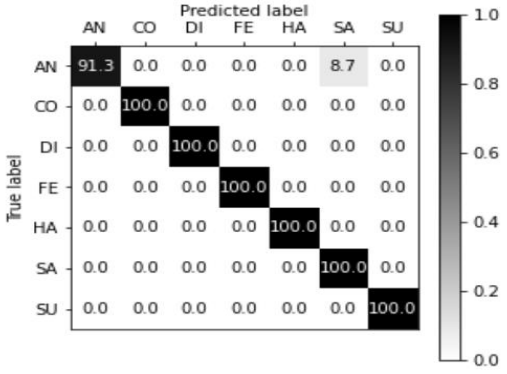
4. A comparison with the previous models

There are several CNN models that have evolved over the three decades. We have chosen some of these models for comparison. Table 6 shows a comparison of the performance of the proposed CNN model with that of other Deep learning models on the same CK+ dataset. It can be seen that with the use of LoG filters at the pre-processing stage improves the average recognition rate to 100%. Moreover, these filters require very less computational time. The existing methods have a limitation that a small change in the facial curves would bring about a substantial degradation in the recognition accuracies. The proposed approach involving the filtered input is robust to such changes in the facial curves.



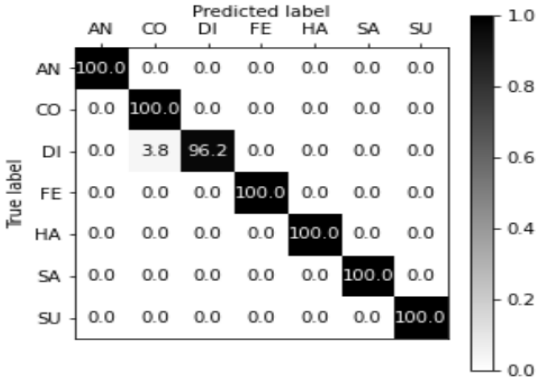
Sensitivity = 99.35%
Specificity = 99.92%

Figure. 9 Confusion matrix of model on CK+ dataset when pre-processed with 3x3 Gaussian kernel



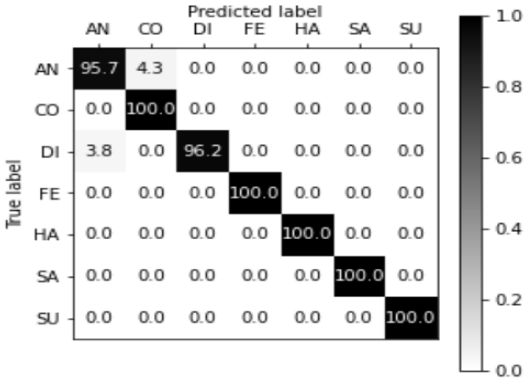
Sensitivity = 98.76%
Specificity = 99.84%

Figure. 10 Confusion matrix of model on CK+ dataset when pre-processed with 7x7 Laplacian after 3x3 Gaussian kernel



Sensitivity = 99.35%
Specificity = 99.96%

Figure. 11 Confusion matrix of model on CK+ dataset when pre-processed with 7x7 Laplacian after 7x7 Gaussian kernel



Sensitivity = 98.67%
Specificity = 99.83%

Figure. 12 Confusion matrix of model on CK+ dataset when pre-processed with 7x7 Laplacian after 9x9 Gaussian kernel

Table 4. Sensitivity and Specificity Obtained On CK+ Dataset (Max Pooling)

Kernel Size	**Sensitivity(%)**					**Specificity(%)**				
	Gaussian + CNN	**LoG + CNN**				**Gaussian + CNN**	**LoG + CNN**			
		3x3	**5x5**	**7x7**	**9x9**		**3x3**	**5x5**	**7x7**	**9x9**
3x3	99.35	97.47	98.10	98.76	96.62	99.92	99.47	99.84	99.84	99.30
5x5	98.76	97.75	98.76	99.35	97.48	99.90	99.52	99.69	99.96	99.61
7x7	97.58	98.76	**100**	99.35	98.10	99.42	99.84	**100**	99.96	99.79
9x9	98.09	95.54	98.76	98.67	98.10	99.51	99.49	99.84	99.83	99.79

Table 5. Sensitivity and Specificity Obtained On CK+ Dataset (Average Pooling)

Kernel Size	**Sensitivity(%)**					**Specificity(%)**				
	Gaussian + CNN	**LoG + CNN**				**Gaussian + CNN**	**LoG + CNN**			
		3x3	**5x5**	**7x7**	**9x9**		**3x3**	**5x5**	**7x7**	**9x9**
3x3	98.1	**100**	**100**	98.04	95.7	99.84	**100**	**100**	99.72	99.29
5x5	98.19	98.1	**100**	99.35	99.35	99.77	99.79	**100**	99.96	99.96
7x7	96.87	97.75	97.75	**100**	**100**	99.39	99.7	99.74	**100**	**100**
9x9	93.3	99.35	95.95	99.35	98.1	98.74	99.96	99.17	99.96	99.78

Table 6. A comparison with deep learning models on CK+ dataset

Method Used	Year	Dataset Used		Accuracy (%)
		Total Images	No. of Classes	
PHRNN-MSCNN [3]	2015	593	7	98.50
IACNN [4]	2017	981	7	95.37
WMDNN [6]	2017	80-120 for each class	6	97.02
DCMA-CNN [5]	2019	927	6	93.46
Hierarchical DNN [12]	2019	927	6	96.46
HiNet [2]	2019	-	6	97.80
Shallow CNN [7]	2020	981	7	97.32
Gaussian - CNN		**981**	**7**	**99.32**
LoG –CNN		**981**	**7**	**100**

5. Conclusion and future scope

5.1 Conclusions

This paper demonstrates the power of the Gaussian and LoG filters in altering the facial images in such a manner that the chosen CNN architecture with two convolutional layers and two pooling layers gives very high recognition accuracies of 99.32% with Gaussian-CNN combination and 100% with LoG-CNN combination. This is possible because of the enhancement of facial images which has taken place due to the use of either of Gaussian filter and LoG filter. It may be noted that without using the filtered images the performance of CNN model degrades to a great extent. The major contribution of this paper is that it demonstrates how the filtered facial images influence the performance of a CNN model. The first observation from experiments is that LoG makes a significant change in the facial images by delineating the facial curves unlike the Gaussian filter leading to the highest accuracy of 100% and the second observation is that the average pooling is more effective than the max pooling because there are many LoG-CNN combinations that provide 100% accuracy.

Thus LoG at the preprocessing stage and the average pooling during the training of the CNN model go a long way in improving the performance of the CNN model in terms of recognition accuracy.

The performance of the combination of filter and CNN model is corroborated by the mathematical justification. This kind of mathematical analysis of the filter-CNN model combination is the first of its kind in the literature. It is hoped that this analysis would help select an appropriate architecture for a CNN model.

5.2 Future scope

Out of an extensive experimentation it has been observed that emotions like angry (AN), disgust (DI) and contempt (CO) have similarities in their respective facial images. This type of similarities hampers the recognition accuracies of these emotions. One way of overcoming this problem is to find facial patches and determine the facial action units (AUs).

In this work we have used CK+ dataset but there other datasets like FER2013, JAFFE, YALEFACE, etc. are in vogue in the literature. So our future work is to see the efficacy of the proposed approach on these datasets.

Conflicts of Interest

The authors declare no conflict of interest.

Author Contributions

The contribution of authors are as follows: Conceptualization and methodology, Neha Mittal, software, Ritanshi Agarwal; validation, Neha Mittal, Ritanshi Agarwal; formal analysis, M. Hanmandlu; investigation, M. Hanmandlu; resources, Neha Mittal; writing-original draft preparation, RitanshiAgarwal; writing-review and editing, M Hanmandlu; visualization, and supervision, Neha Mittal and M. Hanmandlu.

Acknowledgments

The authors gratefully acknowledge the support of the Vision club of Department of Electronics and Communication Engineering at Meerut Institute of Engineering and Technology, Meerut for undertaking this research work.

References

[1] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features”, In: *Proc. of the 2001 IEEE computer society conference on computer vision and pattern recognition (CVPR)*, Kauai, HI, USA, pp. I-I, 2001.

[2] M. Verma, S. K. Vipparthi, and G. Singh, “HiNet: Hybrid inherited feature learning

network for facial expression recognition", *IEEE Letters of the Computer Society*, Vol. 2, No. 4, pp. 36-39, 2019.

[3] K. Zhang, Y. Huang, Y. Du, and L. Wang, "Facial expression recognition based on deep evolutional spatial-temporal networks", *IEEE Transactions on Image Processing*, Vol. 26, No. 9, pp. 4193-4203, 2017.

[4] Z. Meng, P. Liu, J. Cai, S. Han, and Y. Tong, "Identity-aware convolutional neural network for facial expression recognition", In: *Proc. of 12th IEEE International Conference on Automatic Face & Gesture Recognition*, Washington, DC, pp. 558-565, 2017.

[5] S. Xie and H. Hu, "Facial expression recognition using hierarchical features with deep comprehensive multipatches aggregation convolutional neural networks", *IEEE Transactions on Multimedia*, Vol. 21, No. 1, pp. 211-220, 2019.

[6] B. Yang, J. Cao, R. Ni, and Y. Zhang, "Facial expression recognition using weighted mixture deep neural network based on double-channel facial images", *IEEE Access*, Vol. 6, pp. 4630-4640, 2017.

[7] R. Ravi, S. V. Yadhukrishna, and R. Prithviraj, "A face expression recognition using CNN & LBP", In: *Proc. of 4th International Conference on Computing Methodologies and Communication (ICCMC)*, Erode, India, pp. 684-689, 2020.

[8] S. Li and W. Deng, "Deep facial expression recognition: A survey", *IEEE Transactions on Affective Computing*, 2020.

[9] S. Moore and R. Bowden, "Local binary patterns for multi-view facial expression recognition", *Computer Vision and Image Understanding*", Vol. 115, No. 4, pp. 541-558, 2011.

[10] M. F. Valstar, M. Mehu, B. Jiang, M. Pantic and K. Scherer, "Meta-analysis of the first facial expression recognition challenge", *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, Vol. 42, No. 4, pp. 966-979, 2012.

[11] O. Rudovic, M. Pantic, and I. Patras, "Coupled Gaussian processes for pose-invariant facial expression recognition", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 35, No. 6, pp. 1357-1369, 2013.

[12] J. Kim, B. Kim, P. P. Roy, and D. Jeong, "Efficient facial expression recognition algorithm based on hierarchical deep neural network structure", *IEEE Access*, Vol. 7, pp. 41273-41285, 2019.

[13] P. Suja and S. Tripathi, "Analysis of emotion recognition from facial expressions using spatial and transform domain methods", *International Journal of Advanced Intelligence Paradigms*, Vol. 7, No. 1, pp. 57-73, 2015.

[14] M. Pantic, L. J. M. Rothkrantz, "Facial action recognition for facial expression analysis from static face images", *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, Vol. 34, No. 3, pp. 1449-1461, 2004.

[15] Y. Li, S. Wang, Y. Zhao, and Q. Ji, "Simultaneous facial feature tracking and facial expression recognition", *IEEE Transactions on Image Processing*, Vol. 22, No. 7, pp. 2559-2573, 2013.

[16] W. Zheng, Y. Zong, X. Zhou, and M. Xin, "Cross-Domain color facial expression recognition using transductive transfer subspace learning", *IEEE Transactions on Affective Computing*, Vol. 9, No. 1, pp. 21-37, 2016.

[17] S. L. Happy and A. Routray, "Automatic facial expression recognition using features of salient facial patches", *IEEE Transactions on Affective Computing*, Vol. 6, No. 1, pp. 1-12, 2014.

[18] L. Zhong, Q. Liu, P. Yang, J. Huang, and D. N. Metaxas, "Learning multiscale active facial patches for expression analysis", *IEEE Transactions on Cybernetics*, Vol. 45, No. 8, pp. 1499-1510, 2015.

[19] S. Eleftheriadis, O. Rudovic, and M. Pantic, "Discriminative shared Gaussian processes for multiview and view-invariant facial expression recognition", *IEEE Transactions on Image Processing*, Vol. 24, No. 1, pp. 189-204, 2015.

[20] M. Basu, "Gaussian-based edge-detection methods-A survey", *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, Vol. 32, No. 3, pp. 252-260, 2002.

[21] D. Marr and E. Hildreth, "Theory of edge detection", In: *Proc. of the Royal Society of London, Series B, Mathematics and Physical Sciences*, Vol. 207, pp. 187-217, 1980.

[22] R. C. Gonzalez and R. E. Woods, *Digital image processing*, Third Edition, Pearson Education, India, 2008.

[23] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews, "The extended Cohn-Kanade dataset (CK+): A complete dataset for action unit and emotion-specified expression", In: *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, San Francisco, CA, pp. 94-101, 2010.