Journal of Theoretical and Applied Information Technology
30th September 2020. Vol.98. No 18
© 2005 – ongoing JATIT & LLS

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

BENCHMARKING OF CONVOLUTIONAL NEURAL NETWORKS FOR FACIAL EXPRESSIONS RECOGNITION

[1]AHMED HESHAM MOSTAFA, [2] HALA ABDEL-GALIL EL-SAYED, [3] MOHAMED ABDEL-FATTAH BELAL

[1] Computer Science dept. Faculty of Computer Science and Artificial Intelligence, Helwan University Cairo, Egypt

[2] Prof of Computer Science dept. Faculty of Computer Science and Artificial Intelligence, Helwan University, Egypt

[3] Prof of Computer Science dept. Faculty of Computer Science and Artificial Intelligence, Helwan University, Egypt

E-mail: [1] ahmed.hisham@fci.helwan.edu.eg, [2] hala.nagy@fci.helwan.edu.eg, [3] belal@fci.helwan.edu.eg

ABSTRACT

Facial Expression Recognition (FER) is one of the most interesting problems in computer science due to its potential applications in AI, many studies were proposed for the FER, but it based on traditional machine learning techniques and these techniques do not have the generalizability to classify expressions from unseen images or those that are captured from the wild, so it is still a difficult and a complex problem. Recently, trends of research in various fields have begun to transfer to deep learning techniques, since it can learn and capture features automatically, robustness to natural variations in the data and generalizability. This work presents a comparative analysis of the popular convolutional neural network (CNN) models based on modular CNN architectures such as ResNet, DenseNet, MobileNet, NASNetMobile, Inception and Xception, applied on FER problem. The purpose of the paper is benchmarking the best architecture models, in order to help researchers to explore and investigate the best architectures for future research in FER based CNN models. For the comparative analysis multiple metrics were used such as Accuracy, Loss, precision, recall, number of parameters and model size, to conduct experiments facial expressions dataset was used from the AffectNet dataset with 287,651 images for training and 4000 images for validation represent eight facial expressions.

Keywords: *Convolutional Neural Network, Deep learning, Emotion Recognition, FER, Facial Expression Recognition*

1. INTRODUCTION

The Aim of artificial intelligence is to make the interaction and communication between human and AI systems more natural. AI systems to interact with human, it should understand verbal and non-verbal communication.

One of the most important non-verbal communication is the facial expression which forms a language of emotions that can express a wide range of human emotional and feeling states, although facial expressions can be easily recognized by human beings, facial expression recognition by machine is still a great challenge [1,2].

Moreover, FER is an interesting and challenging problem because of it is useful for many tasks and the application in AI such as human-computer interaction (HCI), virtual reality (VR), augment reality (AR), advanced driver assistant systems (ADASs) and video games, etc. So, there has been considerable research interest in AI systems to recognize facial expression [1,2].

Many techniques have been proposed for detect emotions based on automatic facial expression recognition, but these techniques based on traditional machine learning techniques such as support vector machines, Bayesian classifiers, etc. These techniques have been successful when classifying facial expressions in a controlled environment, but these techniques do not have the generalizability to classify expressions from unseen images or those that are captured from the wild [1,2].

Recently, trends of research in various fields have begun to transfer to deep learning techniques, Due to it has many benefits such as it can learn and capture features automatically, robustness to natural variations in the data is automatically learned and generalizability [3] and there are many architectures used in deep learning such as Convolutional Neural Network (CNN), Deep Belief Network (DBN), Deep Autoencoder (DAE) and Recurrent Neural Network (RNN)[1,2].

Many works have been done in Deep-learning-based FER and it shows that DL-FER techniques reduced the dependence on face physics-based models and other pre-processing techniques by allowing end to end learning [2].

In this paper, we present a review and a comparative analysis of the common architectures in deep convolutional neural networks applied on facial expression recognition problem to recognize eight facial expressions: Neutral, Happiness, Sadness, Surprise, Fear, Disgust, Anger and Contempt.

This study used the pretrained models provided by Keras framework [4] and following CNN models are used in the study: ResNet [5], ResNetV2[6], DenseNet [7], MobileNet [8], MobileNetV2[9], NASNetMobile [10], Inception [11,12] and Xception [13].

Comparative performance measure based on appropriate metrics such as Accuracy, Loss, precision, recall, number of parameters and model size using facial expression dataset from the AffectNet [14] dataset with 287,651 images for training and 4000 images for validation

2. RELATED WORK

Many studies used and inspired the convolutional neural networks for FER problem whether with fine tuning or modifing the architcture or cobining with other architctures such as Savoiu, et al.[15] used ResNet50 with VGG16, also Pramerdorfer, et al.[16] introduced comparison between inception ,vgg19 and ResNet, Hasani, et al. in [17,18] proposed model based on Inception-ResNet, also Hasani, et al. in [19] modified the Resnet-110 by applying a differentiable function with a bounded gradient instead of shortcut connection between the input and the output of the ResNet module, Also Tran, et al. [20] used models based on combination of Inception-ResNet. While Peng, Min, et al. [21] proposed a finetuned version of Resnet10, while Li, Junnan, et al. [22] proposed model based on Xception model, While Giannopoulos, et al. [23] introduced comparative study applied on GoogleNet (Inception) and AlexNet. Xia, Xiao-Ling, et al. [24] proposed model based on Inception-v3, Sang, et al. [25] and Zhang, et al. [26], proposed model based on DenseNet.

3. CNN ARCHITECTURES

Nearly all CNN models have common design basis which are based on applying convolutional and pooling operations to the input, then periodically down sampling the heights and width while the number of feature maps are being increased [1].

The convolutional layer has a set of filters to convolve through the input image and produce various types of feature maps. The convolution operation has three main benefits: local connectivity, which learns correlations among neighboring pixels; weight sharing in the same feature map, which reduces the number of the parameters to be learned; and shift-invariance to the location of the object [15].

The pooling layer follows the convolutional layer and is used to reduce the spatial size of the feature maps and the computational cost of the network. Average pooling and max pooling are the two most commonly used nonlinear down-sampling strategies for translation invariance [1,15].

The fully connected layer (FC) is usually included at the end of the network where all cells in FC layer are fully connected to activations in the previous layer and to allow the 2D feature maps to be converted into 1D feature maps for further feature representation and classification [1,27].

The CNN architectures can be divided into classic and modern architecture [28], Classic architecture such as LeNet-5[29], AlexNet [30], VGG16[31] and the modern architecture such as ResNet [5], DenseNet [7], MobileNet [8], Inception [11,12] and Xception [13].

3.1 Classic CNN architectures [16]

Classic CNN architectures were simply of stacked convolutional layers, with classic network architectures the first intuition that the increasing the number of layers lead to get a higher accuracy. But as the number of layers are increased there are two problems that are faced:

a) Vanishing Gradients: During backpropagation, the gradient flow from the last layers of the network becomes almost negligible by the time it reaches the first few layers. This means that the earlier layers don't learn at all. This is called the problem of vanishing gradient. However, there are many solutions were proposed to handle this problem such the rectified linear activation function (ReLU) and Batch Normalization [1,27].

b) Degradation Problem : As CNN get deeper, the training accuracy gets saturated which show that weights are not easy to optimize [1,27].

3.2 Modular CNN architectures [16]

Modular CNN architectures are the modern architectures that inspect advanced and novel designs for building convolutional layers in a way which allows for more effective learning and solving Vanishing Gradients and Degradation problems. Almost all of these architectures are based on micro-architecture [28], that refers to the set of modules or building blocks used to build the network. A building block can consist of standard Convolutional, pooling, etc. layers.

3.2.1 Recent CNN Architectures

ResNet [5]: stand for Residual Neural Network, Kaiming He et al [5] introduced a novel architecture with "skip connections" concept. The authors [5] suggest approach to solve the problem of degradation by implementing residual blocks in which intermediate block layers learn a residual function with respect to previous block input. Authors [5] propose that stacking layers should not reduce network performance, as we could stack identity mappings as shown in Figure 1 on the existing network, and the resulting architecture would do the same thing. This mean that the training error for deeper layers of the network model will not be a higher than its shallow layers of the network.

The Skip Connections and Residual connections or identity mapping [5,6] within layers add the outputs from previous layers to the outputs for next layers or can skip multiple layers. This mean the deeper layers have more ability to be trained and learned than shallow layers in the network.

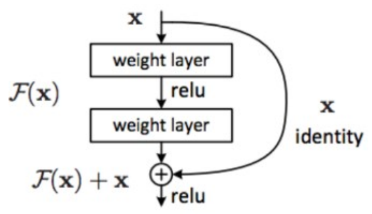

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 1: Residual learning: a building Block [6]

Residual blocks derivative its name from keyword residual, which is the difference between the predicted and target values. Authors [5,6] define the residual learning in the form H(x) = F(x) + x as in figure 1. This means even there is no residual, F(x)=0, model will save an identity mapping of the input, x. Theoretically this mean, the learned residual allows the network to do no worse.

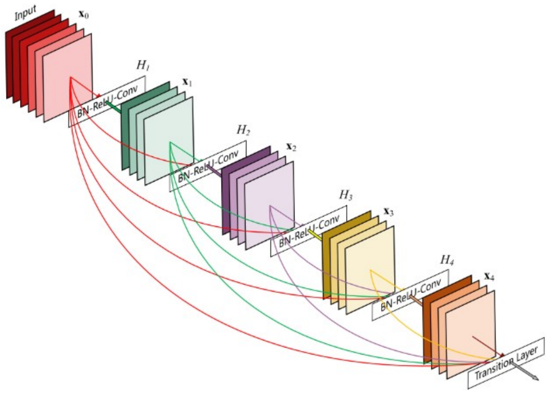


Figure 2: A 5-Layer Dense Block: A Building Block [7]

DenseNet [7]: Densely Connected Networks DenseNet is to some extent the logical behind Resnet of passing information from one layer to another, however in DenseNet each layer's feature map is concatenated to the input of every subsequent layer within a dense block This connection is done using concatenation not through summation as in ResNet.

This allows next layers to directly access the features from previous layers, also it allows to reuse features within the network.

The building block of DenseNet is the dense blocks. As shown in figure 2 each dense block has multiple convolution layers. A dense block is followed by a transition layer even its output can be proceeded to next dense block.

DenseNet have several advantages: it solves the vanishing-gradient problem, encourages propagation and reusing of features, and reduces the number of parameters as there is no need to learn redundant feature maps.

MobileNet [8,9]: it is lightweight in architecture. It uses depth wise separable convolutions which basically means it performs a single convolution on each color channel rather than combining all three and flattening it. It allows the input channels to be filtered. Depth wise separable convolution reduces the complexity and model size of the network, which is suitable for Mobile devices, or any devices with low computational power.

NASNetMobile [10]: is one of the variants of NASNet architecture for mobile platforms. It's based on Neural Architecture Search (NAS) method, NAS is Auto-search for the best convolution layers or cell applied to small datasets.

Then apply this cell to the larger dataset by stacking together more copies of this cell. These cells are called Normal and Reduction cells as shown in figure 3 and figure 4.

Normal Cell: Convolutional cells that return a feature map of the same dimension.

Reduction Cell: Convolutional cells that return a feature map with height and width is reduced by a factor of two.

A cell consists of only a few operations (several separable convolutions and pooling) and is repeated multiple times, according to the required capacity of the network. The mobile version (NASNet-Mobile) consists of 12 cells.

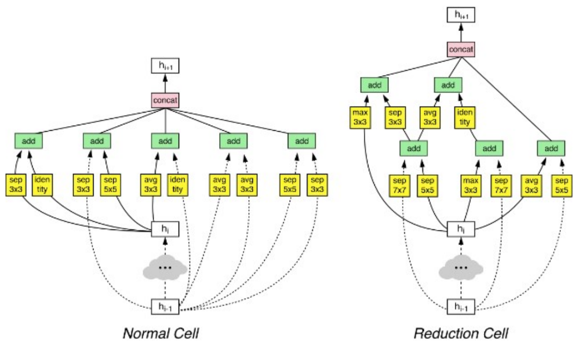


Figure 3: Normal And Reduction Cells: Building Blocks [10]

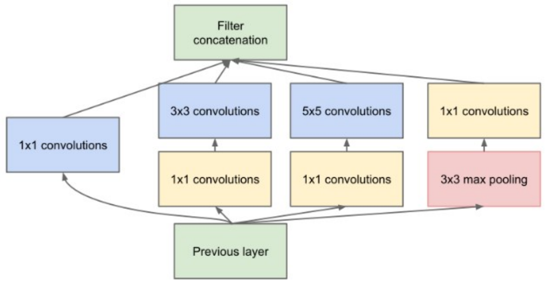


Figure 4: Inception Module: A Building Block [11]

Inception [11,12]: The architecture is based on a basic unit referred to as an "Inception cell" in which it applies sequence of convolutions at different scales as shown in figure 4 and then the results are aggregated. For reducing computation, 1x1 convolutions filters are used. For each cell, it learns a set of 1x1, 3x3, and 5x5 filters which can learn to extract features at different scales from the input. Max pooling with same padding is also used to save the dimensions as the output can be concatenated correctly. The output of these filters is then stacked along the channel dimension then it fed into the next layer.

Xception [13]: Xception is an extension of the Inception architecture which replaces the standard Inception modules with depth wise separable convolutions.

4. FER PROBLEM BASED CNN

FER based CNN techniques reduce the reliance on traditional techniques that based on handily extracting features by programmer and then training the classifier model, by enabling end to end learning [2] where the CNN models directly and automatically learn and extract features from input images and from the input images[2] as shown in figure 5.

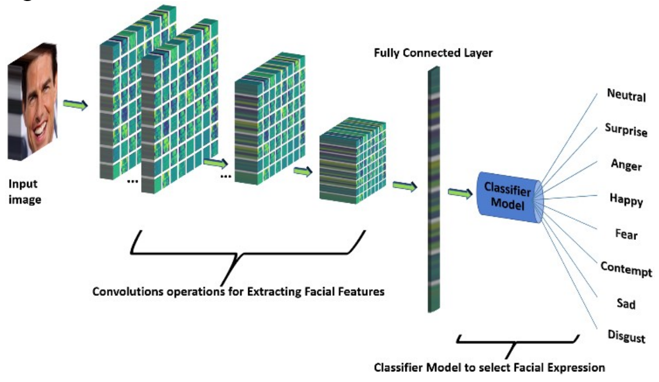

Input image
Fully Connected Layer
Convolutions operations for Extracting Facial Features
Classifier Model
Neutral
Surprise
Anger
Happy
Fear
Contempt
Sad
Disgust
Classifier Model to select Facial Expression

Figure 5: FER Based CNN – End To End Leraning

As shown in Fig 5 the CNN models apply series of convolutions and pooling operations to input image to extract features such as geometric features that describe the shape of the face and its components like nose, eyes, ears and its relations and positions and also the features can be appearance features that appear temporarily in the face during any kind of facial expression such as the intensity changes associated with different expressions and all features are mapped to one-dimension layer to be used as input to classification model that classify the expression based on the extracted features.

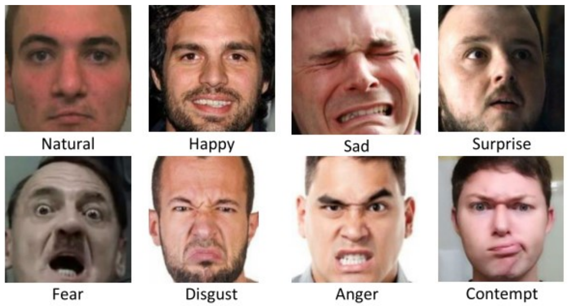


Figure 6: Samples Of Affected Data Where Images Are Queried From The Web.

5. EXPERIMENT SETUP

5.1 Dataset

AffectNet [14] contains about 1M facial images collected from the Internet by querying three major search engines using 1250 emotion related keywords in six different languages. About half of the retrieved images (~420K) are manually annotated.

In this study, we used only the images manually annotated and the image for eight emotions and excluding the images labeled with (None, Uncertain, No-Face) to have the final dataset with 291,651 images where 287,651 images are used for training and 4000 images for validation (500 images for each class). Number of manually annotated images in the training and validation set is shown in table 1 and the faces are allocated and cropped from images and resized to 224 × 224 in RGB color mode.

Table 1: Number of images for each expression.

Expression	Label	N#Images	Percentage
Neutral	0	75,374	25.84 %
Surprise	3	14,590	5 %
Anger	6	25,382	8.7 %
Happy	1	134,915	46.26 %
Fear	4	6,878	2.36 %
Contempt	7	4,250	1.45%
Sad	2	25,959	8.9 %
Disgust	5	4,303	1.47 %

5.2 Multistage Training

All experiments were done on colab platform, colab [29] is providing a free cloud service based on Jupiter Notebooks that supports free GPU. Not only is this a great tool for improving your coding skills, but it also allows absolutely anyone to develop deep learning applications using popular libraries such as PyTorch, TensorFlow, Keras, and OpenCV,

One of the big challenges that this study faced is the maximum lifetime of a running notebook on colab is 12 hours and to continue you have to restart the machine and resume your work

To overcome this challenge, we split the running epochs for each model into groups of epochs, each group range from 15 to 25 epochs and save the last model with all information including the state of the optimizer and loss parameters then reload the saved model and resume the training exactly where the model left off.

5.3 Models and configurations of hyperparameters

This study used the following pretrained models provided by Keras [4] framework as listed in table 2.

The pre-trained networks provided by Keras [6] framework is capable of recognizing 1,000 different object categories trained using ImageNet dataset [30] (1.2M images) to create a generalized model.

Table 2: Keras pretrained Models, number of parameters and size in MegaByte [12].

Model	Parameters	Size MB
ResNet50	25,636,712	98
ResNet101	44,707,176	171
ResNet152	60,419,944	232
DenseNet121	8,062,504	33
DenseNet169	14,307,880	57
DenseNet201	20,242,984	80
InceptionV3	23,851,784	92
ResNet50V2	25,613,800	98
ResNet101V2	44,675,560	171
ResNet152V2	60,380,648	232
MobileNet	4,253,864	16
MobileNetV2	3,538,984	14
NASNetMobile	5,326,716	23
Xception	22,910,480	88

For all experiments the following hyperparameters are used height and width for input images are 224 × 224, number of epochs is 100, batch size is 16, shuffle is true for training data, the categorical entropy used for calculating a loss while for optimizer parameter the Adam optimizer is used.

5.4 Metrics

The evaluation was performed by measuring the following metric:

Accuracy [31] is a ratio of correctly predicted observation to the total observations.

Loss is calculated by summation of all errors made by each example in training or validation sets and it indicate how the well or poorly a model is doing for these two sets of data after each or more iterations of optimization.

As this study is dealing with multi-class classification problem, we used categorical crossentropy as defined in Keras or what is called in machine learning as SoftMax cross entropy function to calculate the loss and optimize the model.

Precision [31] is the percentage of observations correctly predicted as positive to the total of observations predicted as positive.

Recall [31] is the percentage of observations correctly predicted as positive to the all observations in the actual class.

Precision-Recall is a useful measure of success of prediction when the classes are very imbalanced such as the classes in the dataset was used during this research. The precision and recall have the tradeoff relationship.

A model with high recall but low precision returns many results, but the most of these results of predicted classes are incorrect when compared to the training data classes. A model with high precision but low recall returns few results, but most of its predicted classes are correct when compared to the training data classes.

An ideal model is when it has high precision and high recall and return many results, with all results labeled correctly but that is only theoretically.

6. RESULTS

This section discusses in details the accuracy, precision and recall results for all experiments conducted during this study.

First before going into discuss the results, the following section shows that the results for first experiment to check the accuracy of MobileNetV2 model based on the Multistage training strategy against continuous training, The MobileNetV2 was chosen due to it is the smallest model among all available model in this study as shown in table 2 and it can be trained using 100 epochs on Colab in less than 12 hours, for the this experiment the model was trained using 100 epochs, for the Multistage training strategy the 100 epochs were divided into 3 groups (45, 30 and 25) epochs.

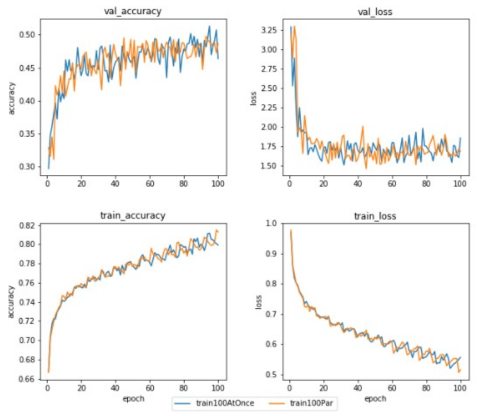


Figure 7: Mobilenetv2 Accuracy And Loss - Train100atonce (Continuous) VS Train100par (Multistage Training)

As shown in the figure 7, the accuracy and loss whether for continuous training or Multistage training are very similar whether with the accuracy and loss for training data or with validation data. Where the best accuracy for Multistage training .815 and .497 for training and validation data respectively, while for continuous training the best accuracy for Multistage training .812 and .513 for training and validation data respectively.

While the smallest losses value for Multistage training .505 and 1.46 for training and validation data respectively, while for continuous training the best accuracy for Multistage training .519 and 1.5 for training and validation data respectively.

Table 3: Models Validation Accuracy (Acc).

Model	Acc	Model	Acc
ResNet50	49.63 %	**ResNet50V2**	50.2 %
ResNet101	49.8 %	**ResNet101V2**	49.13 %
ResNet152	48.95 %	**ResNet152V2**	49.48 %
DenseNet121	51.53 %	**MobileNet**	50.8 %
DenseNet169	52.38 %	**MobileNetV2**	49.73 %
DenseNet201	52.3 %	**NASNetMobile**	51 %
InceptionV3	51.45 %	**Xception**	51.3 %

6.1 Accuracy and Loss

In this part, we discuss and compare between the experiment results for all models based on accuracy and loss, as shown in table 3, figure 8 and figure 9 that the results for Resnetv2 models achieved little bit better results than ResNet models.

Where the best accuracy results for all ResNet models using training, data are 95.4% with loss 1.3, 94.4 with loss 1.55 and 93.3% with loss 1.9 and the best accuracy results using validation data are 49.5% with loss 2.7, 50% with loss 1.5 and 49% with loss 1.9 for ResNet 50,101and 152 respectively.

While for Resnetv2 models the best accuracy results using training data are 95.4% with loss 1.4, 95.8% with loss 1.2 and 94% with loss 1.7 and the best accuracy results using validation data are 50% with loss 2.17, 49% with loss 1.85 and 49.5% with loss 2.26 for ResNetV2 50,101and 152 respectively.

As shown in figure 8 and figure 9 that the results for Mobile models achieved little bit better results than ResNet models where the best accuracy results for Mobile models for training data are 88% with loss .3 ,81% with loss .5 and 90% with loss .3 and the best accuracy results for validation data are 50.8% with loss 1.8 , 49.7% with loss 1.7 and 51% with loss 1.6 for MobileNet, V2 and NASNetMobile respectively.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

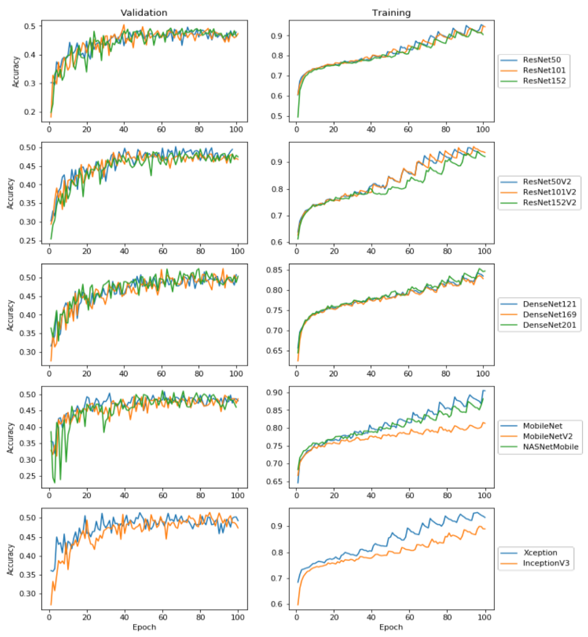


Figure 8: Models Accuracy For Both Training And Validation Dataset

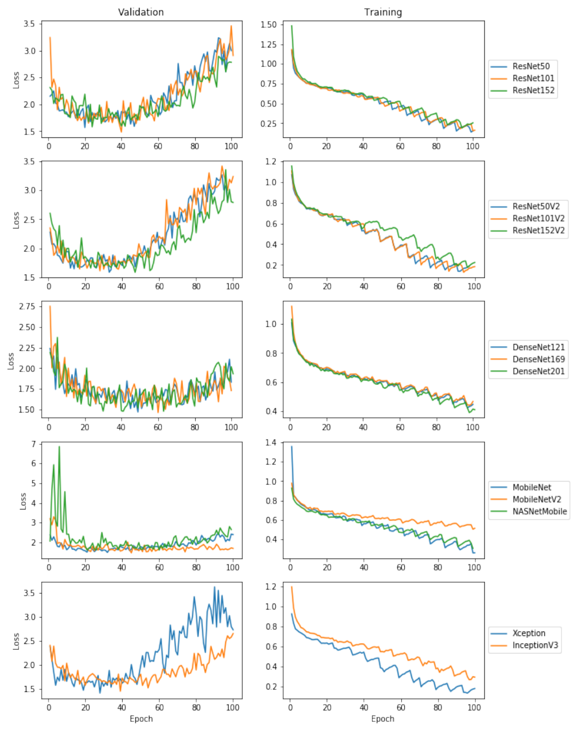


Figure 9: Models Loss For Both Training And Validation Dataset

Also, it can be noted that NASNetMobile is achieved little bit better result than MobileNet whether version 1 or version 2 as it has higher validation accuracy value and lowest loss value.

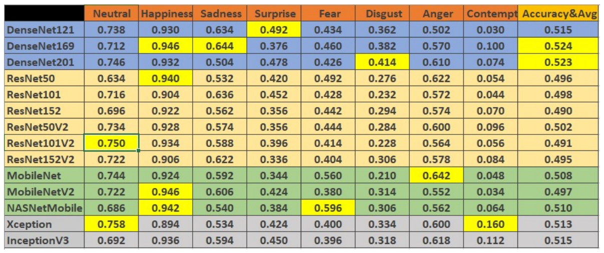
As shown in figure 8 and table 3 that the results for Xception and Inception models achieved little bit better results than ResNet models and MobileNet whether V1 or V2, where the best accuracy results for Xception and Inception models for training data are 95% with loss .13 and 89% with loss .27 and the best accuracy results for validation data are 51% with loss 1.6 and 51% with loss 1.8 for Xception and Inception respectively.

As a whole the DenseNet Models achieved better results than other models where the DenseNet169 and DenseNet201 achieved the best accuracy result.

6.2 Precision and Recall

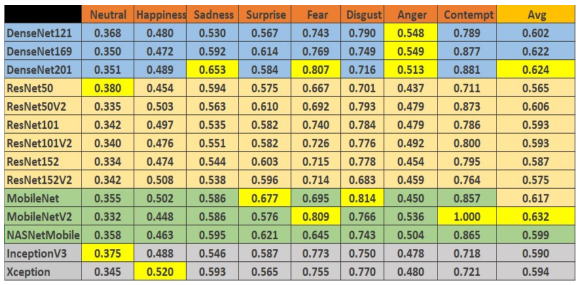
This section discusses and compares between the results for all models based on precession and recall.

It can be noted from figure 10 and figure 11 that, the DenseNet models achieved the highest recall in Happiness, Sadness, Fear and Anger expressions, while for precision achieved the highest values with Sadness, Fear and Anger expressions, while ResNet Models achieve the highest recall with Neutral and Happiness expressions while for precision achieves the highest values with Neutral expression.



	Neutral	Happiness	Sadness	Surprise	Fear	Disgust	Anger	Contempt	Accuracy&Avg
DenseNet121	0.738	0.930	0.634	0.492	0.434	0.362	0.502	0.030	0.515
DenseNet169	0.712	0.946	0.644	0.376	0.460	0.382	0.570	0.100	0.524
DenseNet201	0.746	0.932	0.504	0.478	0.426	0.414	0.610	0.074	0.523
ResNet50	0.634	0.940	0.532	0.420	0.492	0.276	0.622	0.054	0.496
ResNet101	0.716	0.904	0.636	0.452	0.428	0.232	0.572	0.044	0.498
ResNet152	0.696	0.922	0.562	0.356	0.442	0.294	0.574	0.070	0.490
ResNet50V2	0.734	0.928	0.574	0.356	0.444	0.284	0.600	0.096	0.502
ResNet101V2	0.750	0.934	0.588	0.396	0.414	0.228	0.564	0.056	0.491
ResNet152V2	0.722	0.906	0.622	0.336	0.404	0.306	0.578	0.084	0.495
MobileNet	0.744	0.924	0.592	0.344	0.560	0.210	0.642	0.048	0.508
MobileNetV2	0.722	0.946	0.606	0.424	0.380	0.314	0.552	0.034	0.497
NASNetMobile	0.686	0.942	0.540	0.384	0.596	0.306	0.562	0.064	0.510
Xception	0.758	0.894	0.534	0.424	0.400	0.334	0.600	0.160	0.513
InceptionV3	0.692	0.936	0.594	0.450	0.396	0.318	0.618	0.112	0.515

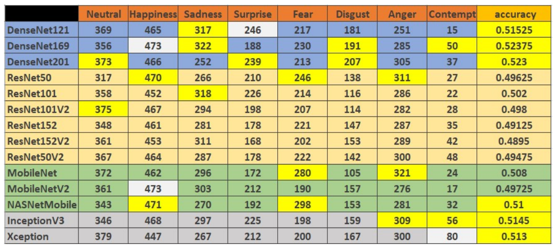
Figure 10: Recall For Each Expression VS Each Model -Yellow Highlight Show Highest Recall-



	Neutral	Happiness	Sadness	Surprise	Fear	Disgust	Anger	Contempt	Avg
DenseNet121	0.368	0.480	0.530	0.567	0.743	0.790	0.548	0.789	0.602
DenseNet169	0.350	0.472	0.592	0.614	0.769	0.749	0.549	0.877	0.622
DenseNet201	0.351	0.489	0.653	0.584	0.807	0.716	0.513	0.881	0.624
ResNet50	0.380	0.454	0.594	0.575	0.667	0.701	0.437	0.711	0.565
ResNet50V2	0.335	0.503	0.563	0.610	0.692	0.793	0.479	0.873	0.606
ResNet101	0.342	0.497	0.535	0.582	0.740	0.784	0.479	0.786	0.593
ResNet101V2	0.340	0.476	0.551	0.582	0.726	0.776	0.492	0.800	0.593
ResNet152	0.334	0.474	0.544	0.603	0.715	0.778	0.454	0.795	0.587
ResNet152V2	0.342	0.508	0.538	0.596	0.714	0.683	0.459	0.764	0.575
MobileNet	0.355	0.502	0.586	0.677	0.695	0.814	0.450	0.857	0.617
MobileNetV2	0.332	0.448	0.586	0.576	0.809	0.766	0.536	1.000	0.632
NASNetMobile	0.358	0.463	0.595	0.621	0.645	0.743	0.504	0.865	0.599
InceptionV3	0.375	0.488	0.546	0.587	0.773	0.750	0.478	0.718	0.590
Xception	0.345	0.520	0.593	0.565	0.755	0.770	0.480	0.721	0.594

Figure 11: Precision For Each Expression VS Each Model -Yellow Highlight Show Highest Precision-

While the Mobile models achieve the highest recall in Happiness, Fear and Anger expressions, but for precision achieved the highest value with Surprise, Fear and Disgust expressions, while InceptionV3 and Xception models achieved the highest recall with Neutral and Contempt expressions while for precision achieved the highest values with Neutral and Happiness.



	Neutral	Happiness	Sadness	Surprise	Fear	Disgust	Anger	Contempt	accuracy
DenseNet121	369	465	317	246	217	181	251	15	0.51525
DenseNet169	356	473	322	188	230	191	285	50	0.52375
DenseNet201	373	466	252	239	213	207	305	37	0.523
ResNet50	317	470	266	210	246	138	311	27	0.49625
ResNet101	358	452	318	226	214	116	286	22	0.502
ResNet101V2	375	467	294	198	207	114	282	28	0.498
ResNet152	348	461	281	178	221	147	287	35	0.49125
ResNet152V2	361	453	311	168	202	153	289	42	0.4895
ResNet50V2	367	464	287	178	222	142	300	48	0.49475
MobileNet	372	462	296	172	280	105	321	24	0.508
MobileNetV2	361	473	303	212	190	157	276	17	0.49725
NASNetMobile	343	471	270	192	298	153	281	32	0.51
InceptionV3	346	468	297	225	198	159	309	56	0.5145
Xception	379	447	267	212	200	167	300	80	0.513

Figure 12: Correctly Classified For Each Expression VS Each Model -Yellow Highlight Show Highest

Also, it can be noted from table 1 the distribution of classes in the dataset is imbalanced and some classes don't have sufficient data to fit and train the model, it can be noted that the fear, Disgust and Contempt expressions have very small training data 2% , 1.5% and 1.5% of the dataset respectively.

Although there are many techniques to overcome that problem [32] such as over-sampling data by copying the images from small classes many times until reach to number of samples in the majority class or instead of copying images, new images can be generated via data augmentation techniques, or down sampling the data, or using cost sensitive learning that was defined in keras as Class weights or using weighted loss functions but, in this research, we didn't apply any of the previous techniques, even we can study and observe how each CNN model will deal with the imbalanced dataset.

Also, it can be noted from figure 12 and table 1 although the distribution for expressions Fear and Anger in the training data is only 2% and 9%, respectively Mobil models could recognize correctly 298 images from 500 for Fear expressions and 321 images from 500 for Anger expressions and it achieved highest recognition rate than other models.

While DenseNet models archived the highest recognition rate for Disgust expression although it represented by only 2% of training data where DenseNet could recognize correctly 207 images of 500 images in the validation data.

Finally, we can put all together, the Densest achieved, on average the highest accuracy, recall and precision followed by Mobile models Inception and Xception followed by then in last place the ResNet models, although these sorting of models, but the results of accuracy are very close and the difference is ranging between 2% to 3% only, so to distinguish between them we add the other factors such as the size and number of parameters in each model.

It can be noted that mobile based models have the smallest size range from 16 to 23 MB and parameters range from 3.5 Million(M) to 5.3M with best accuracy 51% achieved by NASNetMobile, followed by DenseNet models with a size range from 33 to 80 MB and parameters range from 8M to 20M with highest accuracy 52% achieved by Densent169, Followed by Xception and inception models with average size 90 MB and parameters 23M with accuracy 51%

Also, Mobile Models achieved the highest rate for detecting expressions Fear and Anger that represented by only 2% and 9 % respectively of the training data while DenseNet models achieved the highest recognition rate for Disgust expression that represented only by 2% of the training data.

6.3 Face Pose and Top Layer Features Visualization

In this section we are benchmarking the model models based on the face pose and visualizing the features detected by first layer convulsions, so four different images for happy expression as shown in Figure 13 are used to test face pose and visualize the features in the first layer for each model, the four images have different views (front, right, left and occluded).

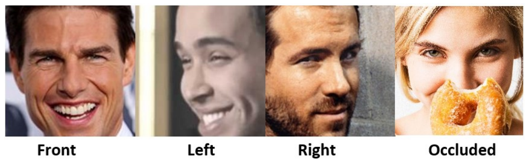


Figure 13: Four Face Images With Different Pose

Due to the number of images for each layers for the four images, in the research paper We select only two models for two different families to visualize the features in the first layer, in this paper we visualize only the following two modes DenseNet121 from DenseNet family and NASNetMobile from Mobil family as shown in Figure 10 and Figure 11.

As shown in figures 14,15,16,17 there are 64 features detected by the first layer for DenseNet121 while in NASNetMobile there are 32 features as shown in Figure s 18,19,20,21, and as shown in all figure s from figure 14 to figure 21 there are a variety of features types detected by the two models such as spatial features to detect eyes, teeth, lips and others face parts and also detect the appearance features such as intensity associated with different expressions such as wrinkles, bulges, forefront, regions surrounding the mouth and eyes.

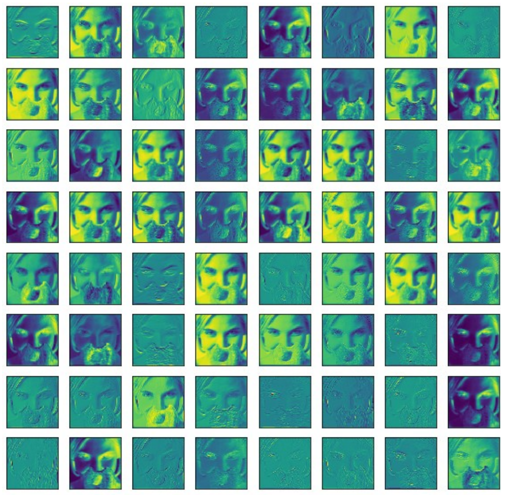


Figure 14: First Layer Features For Densenet121 – Occluded

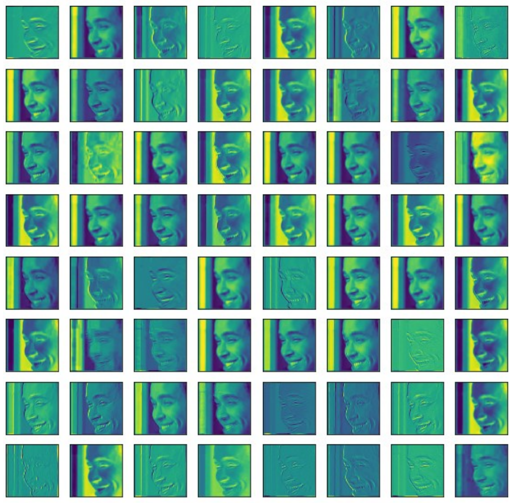


Figure 16: First Layer Features For Densenet121- Right

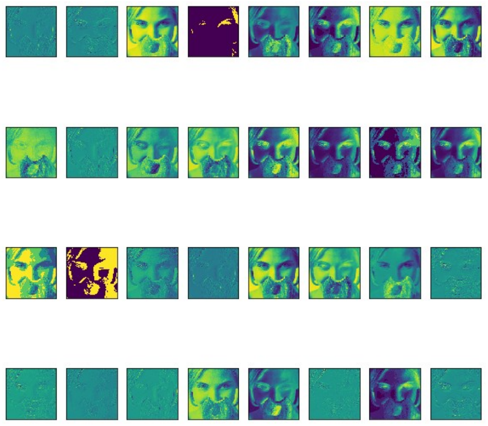


Figure 18: First Layer Features For Nasnetmobile – Occluded

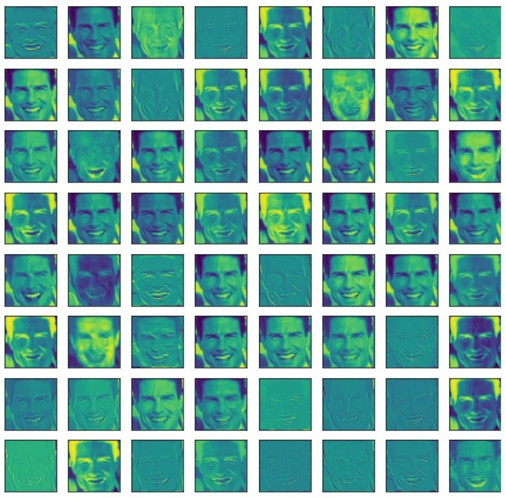


Figure 15: First Layer Features For Densenet121- Front

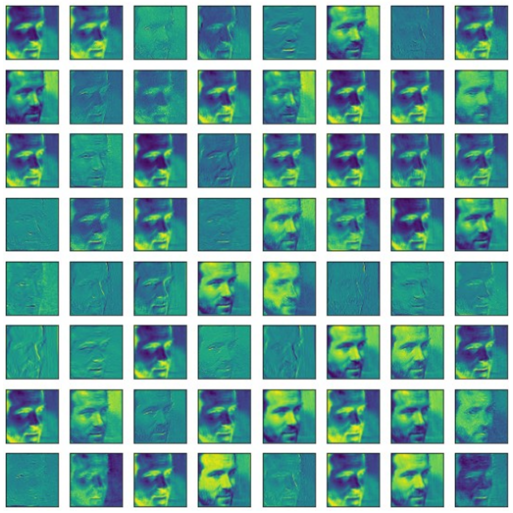


Figure 17: First Layer Features For Densenet121 - Left

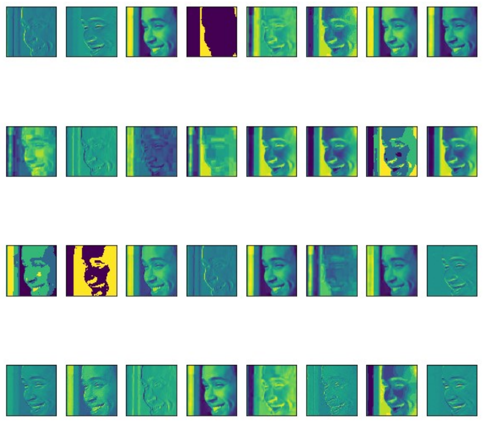


Figure 19: First Layer Features For Nasnetmobile - Right

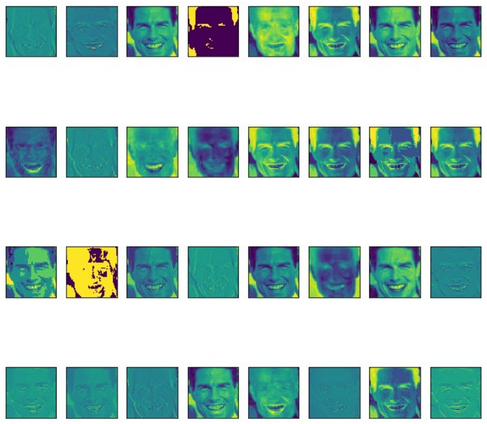


Figure 20: First Layer Features For Nasnetmobile – Front

Table 4: Classification Results For Each Model For The Four Happy Expression With Different Pose. Numbers From 0 To 7 Refer To Classified Expression Label.

Model	Front	Occ	Right	Left
DenseNet121	1	1	1	1
DenseNet169	1	1	7	1
DenseNet201	1	1	1	1
ResNet50	1	0	1	1
ResNet101	1	0	0	1
ResNet152	1	6	0	1
ResNet50V2	1	1	5	1
ResNet101V2	1	1	6	1
ResNet152V2	1	2	1	1
MobileNet	1	1	6	1
MobileNetV2	1	1	3	1
Xception	6	1	6	1
InceptionV3	1	5	6	1
NASNetMobile	1	1	5	1

When we tested the two models using these four images shown in Figure 9 the DenseNet169 classified front, left and occluded images correctly while it gave wrong result with right image, DenseNet169 classified the right image (Happy Expression) as Contempt expression where DenseNet121 and 201 are classified all four images correctly, although the DenseNet169 give a wrong classification, but the predication probability for the two classes are very similar where the recognition probabilities are 9.445 and 9.849 for a Happy and contempt expression respectively.

While NASNetMobile gave the same classification result, such as DenseNet169, it classified front, left and occluded images correctly while it gave wrong result with right image where it classified it as Fear expression, but unlike DenseNet169 the difference between the recognition probabilities of two classes where the predicted probabilities were 4.266 and 9.544 for a Happy and Fear expressions respectively.

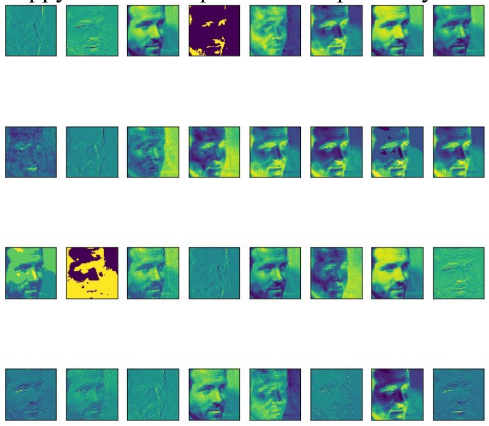


Figure 21: First Layer Features For Nasnetmobile - Left

Also, it can be noted that from table 4 that all models are correctly classified the left image also in front image expect Xception model classified it as Anger expression.

Also, it can be noted from table 4 that most of models give wrong classification for Right Happy expression image expect DenseNet121, DenseNet201, ResNet50 and ResNet152V2.

While for Occluded image it can be noted that from table 4 that All ResNet Model are gave wrong classification while for ResNetV2 models only ResNet152V2 give wrong classification, Also InceptionV3 model gave wrong classification where it classified the occluded Happy expression as Disgust.

7. CONCLUSION AND FUTURE WORK

The paper presented a benchmarking study for fourteen CNN models and provided an immediate and comprehensive comparison for helping in the choosing of the appropriate model according to constraints in resources for applications and practical deployments where we analyzed fourteen Convolutional neural network models based on different architectures such as DenseNet, ResNet, MobileNet and Inception and Xception provided by cross framework.

AffectNet dataset is used to conduct experiments with 287,651 images for training and 4000 images for validation representing eight expressions Neutral, Happiness, Sadness, Surprise, Fear, Disgust, Anger and Contempt. The faces are allocated and cropped from images and resized to 224 × 224 in RGB color mode.

All experiments were done on Colab platform, To overcome the constraints of the running time of Colab, we split the 100 epochs for training into groups and save the last model with all information including the state of the optimizer and loss parameters then reload the saved model to resume the training exactly where the model left off.

The results show that the accuracy and loss values whether for continuous training or Multistage training are very similar whether with the training data or with validation data.

Also, the results show that the average of recognition accuracy for most models for a validation dataset is between 50% to 52% where Densest achieved, on average the highest accuracy, recall and precision followed by Mobile models followed by Inception and Xception then in last place the ResNet models.

Also results show that the lightweight versions of CNN models give similar results to complicated and large CNN models

In order to achieve more comprehensive results, in future we can modify the existing architectures by freezing or removing some layers due to these models were designed and used to recognize 1000 classes not only 8 classes or merging the features detected by the CNN models by different handcrafted features.

Finally, in future we can apply facial expression recognition problem on others deep learning architectures such as Deep Belief Network (DBN), Deep Autoencoder (DAE), Recurrent Neural Network (RNN) and Generative Aadversarial Network (GAN).

REFRENCES:

[1] Li, Shan, and Weihong Deng. "Deep Facial Expression Recognition: A Survey." arXiv preprint arXiv:1804.08348 (2018).

[2] Ko, Byoung Chul. "A Brief Review of Facial Emotion Recognition Based on Visual Information." sensors 18.2 (2018): 401

[3] Ng, Hong-Wei, et al. "Deep learning for emotion recognition on small datasets using transfer learning." Proceedings of the 2015 ACM on international conference on multimodal interaction. ACM, 2015.

[4] Keras Framework Applications, https://keras.io/applications/, 2020

[5] HE, Kaiming, et al. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. p. 770-778.

[6] He, K., Zhang, X., Ren, S., & Sun, J. (2016, October). Identity mappings in deep residual networks. In European conference on computer vision (pp. 630-645). Springer, Cham.

[7] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4700-4708).

[8] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861.

[9] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on

Computer Vision and Pattern Recognition (pp. 4510-4520).

[10] Zoph, B., Vasudevan, V., Shlens, J., & Le, Q. V. (2018). Learning transferable architectures for scalable image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 8697-8710).

[11] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2818-2826).

[12] SZEGEDY, Christian, et al. Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2015. p. 1-9.

[13] Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1251-1258).

[14] A. Mollahosseini; B. Hasani; M. H. Mahoor, "AffectNet: A Database for Facial Expression, Valence, and Arousal Computing in the Wild," in IEEE Transactions on Affective Computing, 2017.

[15] A Savoiu, J Wong, Recognizing Facial Expressions Using Deep Learning, 2016.

[16] Pramerdorfer, Christopher, and Martin Kampel. "Facial expression recognition using convolutional neural networks: state of the art." arXiv preprint arXiv:1612.02903 (2016).

[17] Hasani, Behzad, and Mohammad H. Mahoor. "Facial expression recognition using enhanced deep 3D convolutional neural networks." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2017.

[18] Hasani, Behzad, Pooran Singh Negi, and Mohammad H. Mahoor. "Bounded Residual Gradient Networks (BReG-Net) for Facial Affect Computing." arXiv preprint arXiv:1903.02110 (2019).

[19] Hasani, Behzad, and Mohammad H. Mahoor. "Spatio-temporal facial expression recognition using convolutional neural networks and conditional random fields." 2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017). IEEE, 2017.

[20] Tran, Elizabeth, et al. "Facial Expression Recognition Using a Large Out-of-Context Dataset." 2018 IEEE Winter Applications of Computer Vision Workshops (WACVW). IEEE, 2018.

[21] Peng, Min, et al. "From macro to micro expression recognition: deep learning on small datasets using transfer learning." 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018). IEEE, 2018.

[22] Li, Junnan, and Edmund Y. Lam. "Facial expression recognition using deep neural networks." 2015 IEEE International Conference on Imaging Systems and Techniques (IST). IEEE, 2015.

[23] Giannopoulos, Panagiotis, Isidoros Perikos, and Ioannis Hatzilygeroudis. "Deep learning approaches for facial emotion recognition: A case study on FER-2013." Advances in Hybridization of Intelligent Methods. Springer, Cham, 2018. 1-16.

[24] Xia, Xiao-Ling, Cui Xu, and Bing Nan. "Facial expression recognition based on tensorflow platform." ITM Web of Conferences. Vol. 12. EDP Sciences, 2017.

[25] Sang, Dinh Viet, and Pham Thai Ha. "Discriminative deep feature learning for facial emotion recognition." 2018 1st International Conference on Multimedia Analysis and Pattern Recognition (MAPR). IEEE, 2018.

[26] Zhang, Zhiqin. "Deep Face Emotion Recognition." Journal of Physics: Conference Series. Vol. 1087. No. 6. IOP Publishing, 2018.

[27] Li, F. F., Karpathy, A., & Johnson, J. (2017). Stanford CS class CS231n: Convolutional Neural Networks for Visual Recognition.

[28] Jordan, J. "Common Architectures in Convolutional Neural Networks." Jeremy Jordan (2018).

[29] Google Colab, https://colab.research.google.com ,2020

[30] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge.2014

[31] Joshi, Renuka. "Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures." blog.exsilio.com. https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/ (accessed May 7, 2020).

[32] Johnson, Justin M., and Taghi M. Khoshgoftaar. "Survey on deep learning with class imbalance." Journal of Big Data 6.1 (2019): 27.