# Visual Emotion Recognition Using Deep Neural Networks

Alexander I. Iliev[1, 2], Ameya Mote[1]

[1] SRH University Berlin, Charlottenburg, Germany
[2] Institute of Mathematics and Informatics, Bulgarian Academy of Sciences, Sofia, Bulgaria
ailiev@berkeley.edu,
AmeyaPrasanna.Mote@stud.srh-campus-berlin.de

**Abstract.** It has been proven historically how important feelings and expressions are. They form an important role in communications between individuals of different culture. In the present day, Globalization has led to exchange of vast number of ideas among people on Earth. This gives rise to a unique challenge of identifying what the person in front is speaking about and formulate opinions likewise. Failing to do that would often result in unfortunate consequences. This paper leads to make inroads in this field and provide a basis to other future researchers. We took images from a pre-existing video dataset and recognize the emotions behind it. Through a series of experiments, a final neural network model was created which gave an accuracy of 88%.

**Keywords:** Emotion Recognition; Image Analysis; Language Understanding; Deep Convolutional Neural Networks; Cross-Cultural Comparison.

## 1 Introduction

Emotion plays an important part in any culture. Research (Chai, Hafeez, Mohamad, & Malik, 2017) carried out further explains how Emotions have a massive influence on the cognitive functions of human. One example is how it affects the learning and memory of an individual. Similarly, previous research (Iliev, Mote, & Manoharan, 2021) by the same authors focused primarily on audio and speech analysis, talks how closely related emotions across different cultures are. This provides a basis for the current research and further expand on it according to the visual aspect. Paul Eckman in 1970, discovered 6 basic emotions, namely: happiness, anger, sadness, disgust, fear, surprise. In the coming years after that, neutral was also added to the list as 7th basic emotions (Ekman, 1992). Of course, there have been further research as to further divide the emotions into more types, but for this paper, only the 6 original types are taken into consideration.

## 1.1 Problem Description

Humans recognize the facial expressions from a pre-existing notion of what the facial aspect would be for a particular expression (Barret, Adolphs, & Pollak, 2019). For example, slightly curved lips might signify how a person has a negative state of emotion whereas raised eyebrows might mean the person is surprised. Often, such information is used to correctly identify the state of speech the person is speaking about. But the same could not be said about people from different cultural background. Recent research (Chai, Hafeez, Mohamad, & Malik, 2017) explains how universally recognized patterns for facial expressions might not be applicable to all cultural groups. To experiment on it, the U.S.A and HIMBA ethnic groups were considered. Over the course, it was observed that U.S.A group were much more likely to adhere to the universally recognized facial expressions patterns as compared to HIMBA. This led to a conclusion that facial expressions are based much more on the contextual concept. So, if there is an interaction between 2 culturally diverse individual's, they should not depend on the pre-existing data or else would lead to wrong classification of the emotion.

Before images, speech has long been thought of a source for classifying emotions. In research carried out in 2020 (Wani, Gunawan, & Qadri, 2020), using the SAVEE emotion dataset, researchers were able to determine for 4 basic emotion types, the CNN model was able to give an accuracy of 79.4%. While speech does help in asserting the various emotion types, the accuracy is far less than desired. Thus, in this research we wanted to focus on the facial features and determine whether they are an important factor while classifying emotions. For this, research termed as Facial emotion recognition using convolutional neural networks (FERC) (Mehendale, 2020), was carried out. In this paper, the authors have used 10000 images across 154 people. Using CNN, a 96% accuracy was generated. The same model was also used across other well-known image datasets such as Cohn–Kanade expression, Caltech faces, CMU and NIST datasets. While this research does signify how important facial features are, the images taken into consideration where in a controlled environment. In a real world, it is a seldom that a perfect lighting condition is achieved. As such, we focus more on creating real world images and testing our model on the same.

For our research, EINTERFACE'05 (Gendron, Roberson, van der Vyver, & Barrett, 2006) Dataset was selected. It is an audio-visual database that has 42 subjects across 14 European countries. After a series of experiments, the visual aspect was not deemed enough to classify emotions. Thus, they were converted into a series of images which led it to being the final dataset.

## 1.2 Dataset Description

When working with images, there are certain things that should be considered. To mirror the real world, efforts were taken to modify the images. Inversion and Noise was added into the images so that the model could generalize. Another main factor that was added is the Monochrome. All images were changed into black and white images. Table 1 depicts the final dataset that was used for the Deep Convolutional Neural Network model.

**Table 1.** Dataset

| Set type | Happiness | Anger | Sadness | Disgust | Fear | Surprise |
|---|---|---|---|---|---|---|
| Train | 1922 | 1896 | 1922 | 1891 | 1922 | 1922 |
| Test | 241 | 237 | 241 | 237 | 241 | 241 |
| Validation | 240 | 237 | 240 | 236 | 240 | 240 |

Further, explanation of the various augmentation methods that are stated above is given.

Inversion: All the images were inverted to an angle of 15 degrees. This was done as all real-world images may not be straight and some would be inclined to certain angles.

Noise: Images may always not be clear and may have some granularity attached to it. A certain level of noise was added to the images to mirror that.

Monochrome: Black and White images are much easier to work with and reduce the computation time for the model.

Fig. 1, shows an image after using the above-mentioned augmentation techniques.

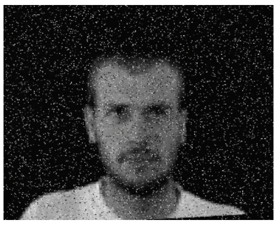


**Fig. 1.** After augmentation

### 1.3 System Hardware

To successfully train the model, there are certain requirements we should take in mind. This section explains the various system and its peripherals that have been used in the construction of the model.

IDE: To reduce on the computation time and fetch accurate results, we preferred to use Google Collab, instead of the local system. This gives the security of the data not being deleted accidently.

GPU: In the free hosted version of Google Collab, Google provides single 12GB Nvidia K80 GPU that can used by the users for up to 12 hours.

Google Drive: Due to the size of images being considerably big, we decided to upload the folder to the cloud and fetch the dataset from there.

### 1.4 Neural Network Architecture

After getting the required number of images, we further started talking about the kind of convolutional network model we wanted. The model needed to understand the granularities in the image and even with other factors, classify emotions accurately. Thus, Convolutional Neural Network was deemed perfect for this. CNN in essence, works the same as an artificial neural network. Each architecture in CNN comprises of an input layer, a hidden layer and an output layer. The notable difference between CNN and ANN is in the way it is deployed (O'Shea & Nash, 2015). CNNs are generally used for recognizing patterns across various images. This enables the user to use them into large image centric features. ANN fails in term of handling computational complexity whereas CNN offers ease which also helps in reducing the number of parameters required for the model. After several experiments with different number of layers and other hyperparameters, Fig.2, gives the final version of the model which also considered the computation time along with accuracy. Apart from this, efforts were also taken into checking how robust the model is for different sizes of images. So, 3 different sizes of images were also considered. In this architecture, we have used 6 hidden layers.

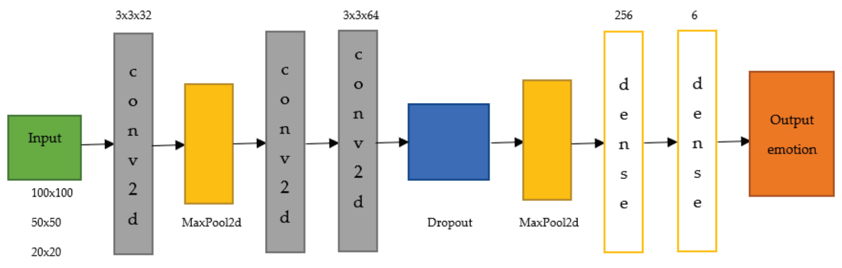
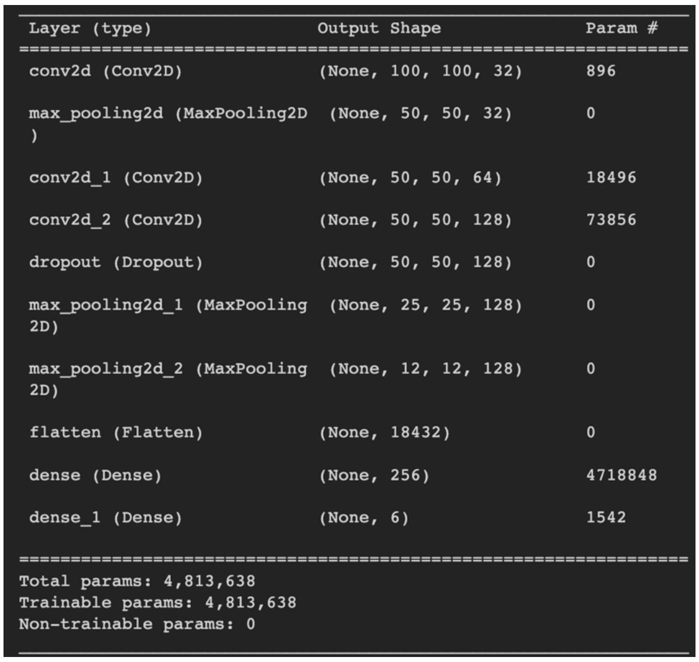


**Fig. 2.** Convolutional Neural Network

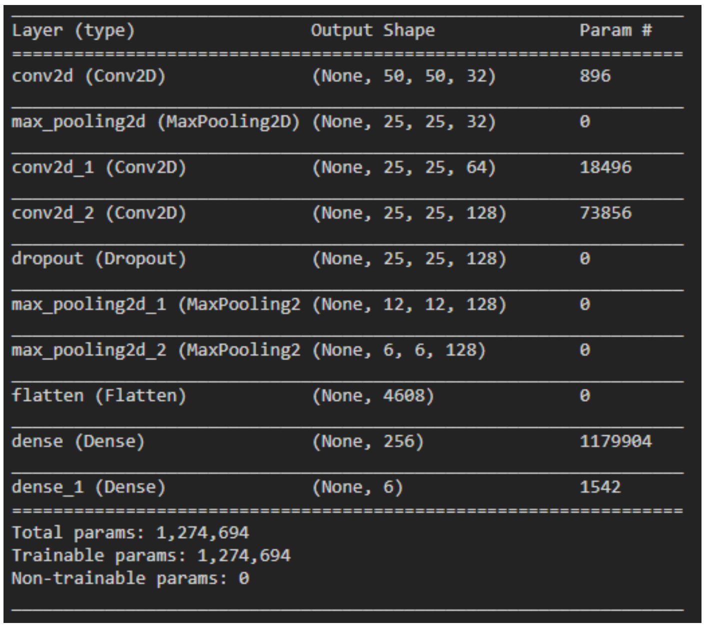
- Conv2d: The model uses 3 convolutional 2d layers with 3x3 filters. The initial Conv2d layer has 32 filters and each subsequent Conv2d layer increased by a factor of 2.
- MaxPool2d: It's a dimension reduction technique (Nagi, et al., 2011). We have used 2,2 stride matrix with a pooling size 2,2.
- Activation Function: We have used 'ReLU' and 'SoftMax' activation function. ReLU, called as the rectified linear unit has been immense in biological and mathematical faction. In simple words, it works by assigning threshold at 0, f(x) = max(0,x). It gives an output of 0 when x<0 and output of 1 when x>= 0 (Agarap, 2018). SoftMax is another activation function which is generally used at the output layer of any Deep learning model. It gives an output in terms of probabilities and commonly used for multi-class classification.
- Dropout: To remove the issue of overfitting, the model also employs dropout layer (Srivastava, Hinton, Krizhevsky, & Sutskever, 2014). A dropout layer with the drop rate of 0.1 was chosen. The rate basically helps us to drop a fraction of the inputs.
- Dense: It is a fully connected layer which also spews the output (Hue, 2020). It is the final stop before the output. Here we have 2 fully connected dense layers. The first FCN takes up 256-dimensional output units and the 2$^{nd}$ one helps us to get 6-dimensional output units which corresponds to the number of emotions we have used.

- In figures 3.1, 3.2, and 3.3, we test the flexibility of our model with 3 different image sizes. In the first image, we have a 100x100 image, we have 4,718,848 trainable parameters. Whereas in the second, 50x50 image size model, we have a 1,274,694 trainable parameter summary. The lowest image size is 20x20 pixels, which only has 131,328 parameters. Now that we have 3 different image sizes to compare our model for, we can test it on the dataset.



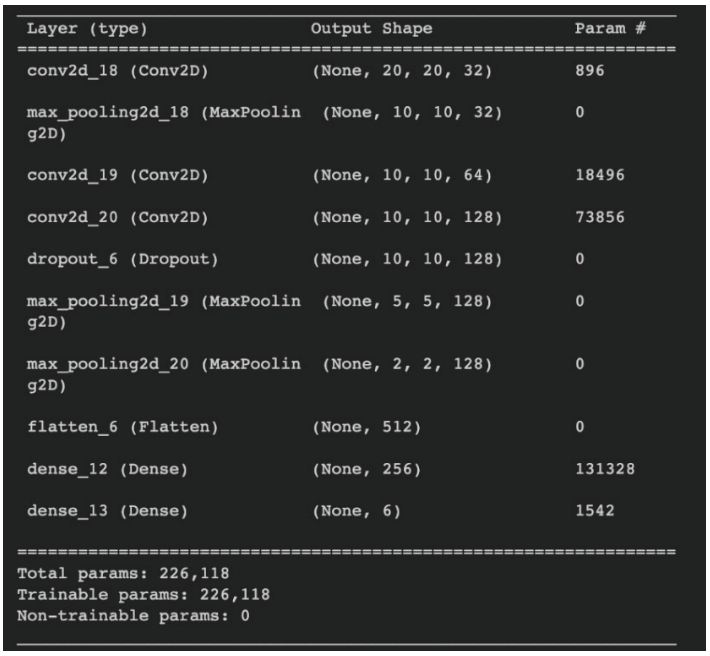
```
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 100, 100, 32)      896

 max_pooling2d (MaxPooling2D  (None, 50, 50, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 50, 50, 64)        18496

 conv2d_2 (Conv2D)           (None, 50, 50, 128)       73856

 dropout (Dropout)           (None, 50, 50, 128)       0

 max_pooling2d_1 (MaxPooling  (None, 25, 25, 128)      0
 2D)

 max_pooling2d_2 (MaxPooling  (None, 12, 12, 128)      0
 2D)

 flatten (Flatten)           (None, 18432)             0

 dense (Dense)               (None, 256)               4718848

 dense_1 (Dense)             (None, 6)                 1542

=================================================================
Total params: 4,813,638
Trainable params: 4,813,638
Non-trainable params: 0
```

**Fig. 3.1.** 100X100 CNN



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 50, 50, 32)        896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 25, 25, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 25, 25, 64)        18496
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 25, 25, 128)       73856
_________________________________________________________________
dropout (Dropout)            (None, 25, 25, 128)       0
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 12, 12, 128)       0
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 6, 6, 128)         0
_________________________________________________________________
flatten (Flatten)            (None, 4608)              0
_________________________________________________________________
dense (Dense)                (None, 256)               1179904
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 1542
=================================================================
Total params: 1,274,694
Trainable params: 1,274,694
Non-trainable params: 0
_________________________________________________________________
```

**Fig. 3.2.** 50X50 CNN



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_18 (Conv2D)          (None, 20, 20, 32)        896

 max_pooling2d_18 (MaxPoolin  (None, 10, 10, 32)       0
 g2D)

 conv2d_19 (Conv2D)          (None, 10, 10, 64)        18496

 conv2d_20 (Conv2D)          (None, 10, 10, 128)       73856

 dropout_6 (Dropout)         (None, 10, 10, 128)       0

 max_pooling2d_19 (MaxPoolin  (None, 5, 5, 128)        0
 g2D)

 max_pooling2d_20 (MaxPoolin  (None, 2, 2, 128)        0
 g2D)

 flatten_6 (Flatten)         (None, 512)               0

 dense_12 (Dense)            (None, 256)               131328

 dense_13 (Dense)            (None, 6)                 1542

=================================================================
Total params: 226,118
Trainable params: 226,118
Non-trainable params: 0
_________________________________________________________________
```

**Fig. 3.3.** 20X20 CNN

To experiment further, it was also decided to check the mean and standard deviation of original size of the images, i.e., 416x416 RGB. The code that was used to define it is given below in Fig.4.

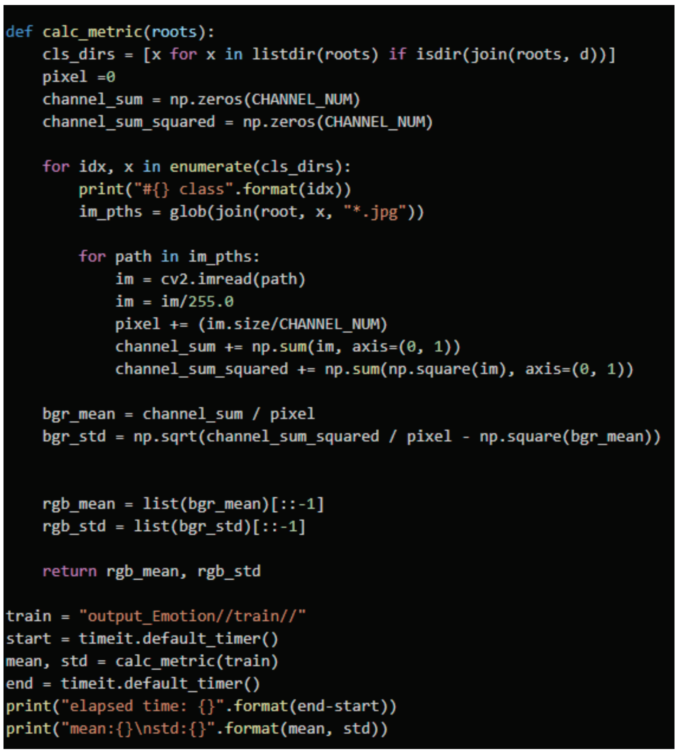

```python
def calc_metric(roots):
    cls_dirs = [x for x in listdir(roots) if isdir(join(roots, d))]
    pixel =0
    channel_sum = np.zeros(CHANNEL_NUM)
    channel_sum_squared = np.zeros(CHANNEL_NUM)

    for idx, x in enumerate(cls_dirs):
        print("#{} class".format(idx))
        im_pths = glob(join(root, x, "*.jpg"))

        for path in im_pths:
            im = cv2.imread(path)
            im = im/255.0
            pixel += (im.size/CHANNEL_NUM)
            channel_sum += np.sum(im, axis=(0, 1))
            channel_sum_squared += np.sum(np.square(im), axis=(0, 1))

    bgr_mean = channel_sum / pixel
    bgr_std = np.sqrt(channel_sum_squared / pixel - np.square(bgr_mean))


    rgb_mean = list(bgr_mean)[::-1]
    rgb_std = list(bgr_std)[::-1]

    return rgb_mean, rgb_std

train = "output_Emotion//train//"
start = timeit.default_timer()
mean, std = calc_metric(train)
end = timeit.default_timer()
print("elapsed time: {}".format(end-start))
print("mean:{}\nstd:{}".format(mean, std))
```

**Fig. 4.** Mean & Standard deviation code

The output

```
mean: [0.21988911368700648, 0.21988911368700648, 0.21988911368700648]
std: [0.2091009344354088, 0.2091009344354088, 0.2091009344354088]
```

## 2 Results

Table 2, as shown here, practically relays the information of how different systems perform for the image sizes we have. After referring to the table, we concluded that GPU is the best hardware to perform the computation of the model successfully.

**Table 2.** Per epoch computation time

| **Image size** | **CPU** | **GPU** |
|---|---|---|
| 100x100 | 430s | 67s |

| | | |
|---|---|---|
| 50x50 | 137s | 50s |
| 20x20 | 66s | 36s |

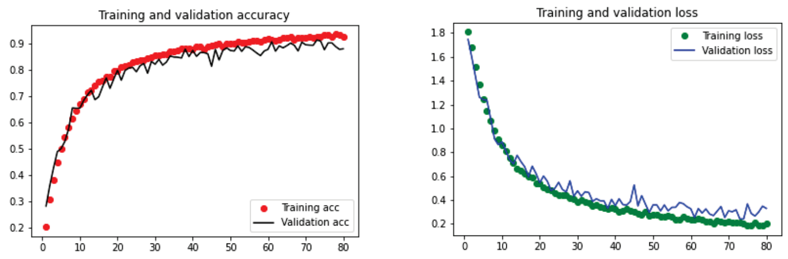
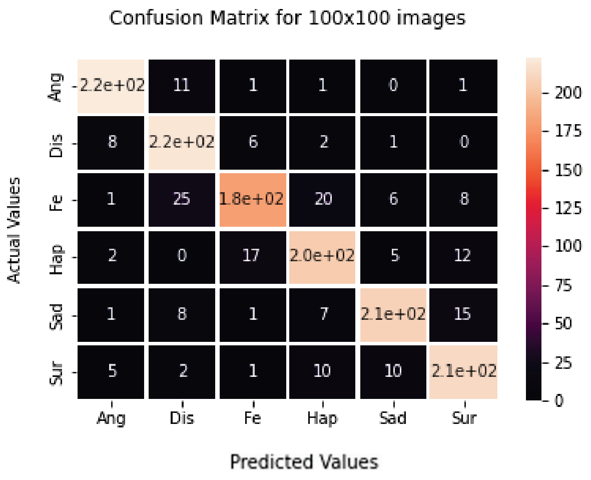


**Fig. 5.** Accuracy and loss graph (100x100)

After experimenting with all images, we concluded that the image size with highest computation time was 100x100 with approximately 74 minutes only. The above figure on the left shows us the training and the validation accuracy whereas the image on the right shows the training and validation loss. While the model comprises on the time, it does not the metrics. That is evident from the fact that a test accuracy of 90.75% was reached with train accuracy of 96%. Diving a bit deep, following Fig. 6, shows the confusion matrix and how different emotions performed against each other.



| | Ang | Dis | Fe | Hap | Sad | Sur |
|---|---|---|---|---|---|---|
| Ang | 2.2e+02 | 11 | 1 | 1 | 0 | 1 |
| Dis | 8 | 2.2e+02 | 6 | 2 | 1 | 0 |
| Fe | 1 | 25 | 1.8e+02 | 20 | 6 | 8 |
| Hap | 2 | 0 | 17 | 2.0e+02 | 5 | 12 |
| Sad | 1 | 8 | 1 | 7 | 2.1e+02 | 15 |
| Sur | 5 | 2 | 1 | 10 | 10 | 2.1e+02 |

**Fig. 6.** Confusion matrix(100x100)

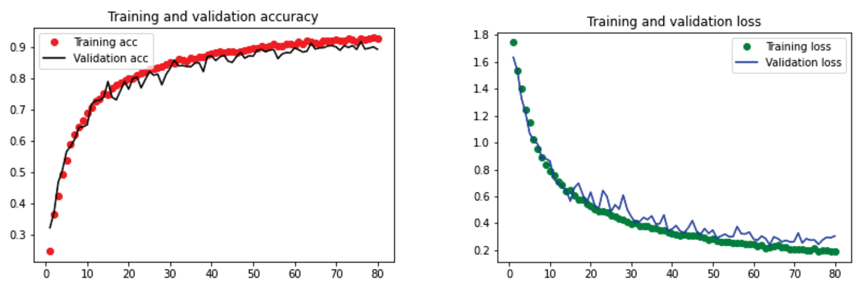


**Fig. 7.** Accuracy and loss graph (50x50)

Having talked about 2 extreme cases, after series of some more experiments, 50x50 image size was confirmed to the best among other sizes. The proof is given by the fig.7 7 given above, which the Test accuracy of 50x50 is 87% and the Train accuracy is 94%. An optimum computation time was also achieved with the model taking 55 minutes to compute.

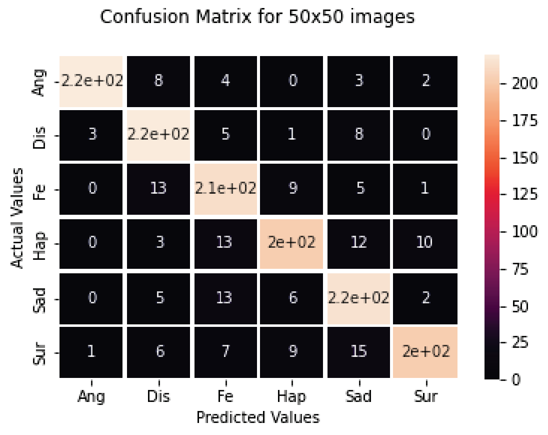


**Fig. 8.** Confusion matrix(50x50)

As evident from the confusion matrix given in fig.8, all the emotions fare better when compared against each other. But a detailed information can help us understand better.

**Table 3.** 50x50 Image metrics

| **Emotions** | **Precision** | **Recall** |
|---|---|---|
| Anger | 0.96 | 0.98 |
| Disgust | 0.91 | 0.93 |
| Fear | 0.88 | 0.81 |
| Happiness | 0.91 | 0.80 |
| Sadness | 0.85 | 0.93 |
| Surprise | 0.88 | 0.93 |

Here in Table 3, the emotion model is most able to recognize clearly is Anger (0.96). Followed by Disgust and Happiness both having accuracy score of 0.91. Sadness is the least recognized emotion with score 0.85.

The 2 emotions that were easily recognized were Anger and Happiness with a score of 0.94 and 0.96 respectively. They were followed by Sadness (0.89), Disgust (0.88) and Surprise performed the last with an accuracy of 0.87. For research purposes, we have also decided to add the metric Recall to our paper to get better understanding.

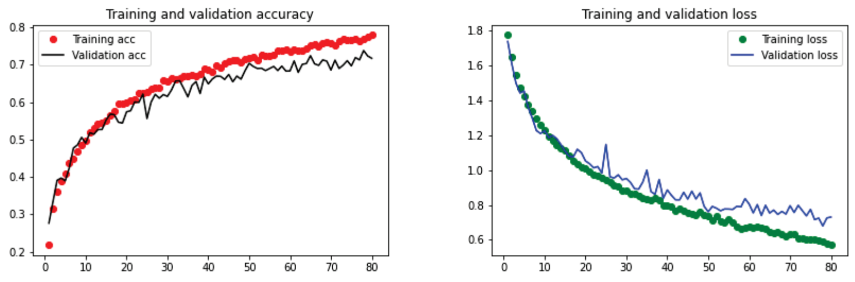
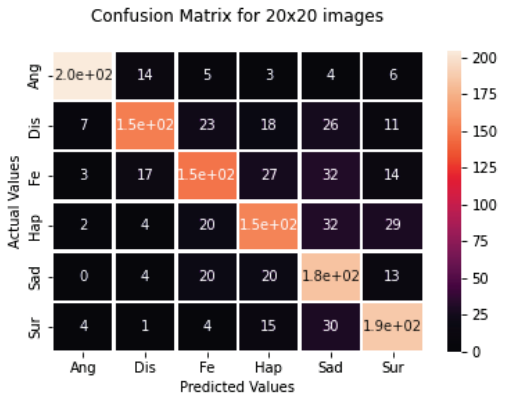


**Fig. 9.** Accuracy and loss graph (20x20)

In fig.9, since we talked about 100x100 image does comprise on time but not the accuracy, here we have 20x20 image size that comprises on exactly the opposite. The model had the fastest computation time of approximately 48 minutes. Although being fast, it was not predicting emotions accurately enough as evident from the Test accuracy of just 72.39% with Train accuracy of 78%. It is clear, this image size should not be deemed acceptable in works where accuracy is the major constraint. The confusion matrix (fig.10) gives us a clear idea of how the emotions performed.



| Actual \ Predicted | Ang | Dis | Fe | Hap | Sad | Sur |
|---|---|---|---|---|---|---|
| Ang | 2.0e+02 | 14 | 5 | 3 | 4 | 6 |
| Dis | 7 | 1.5e+02 | 23 | 18 | 26 | 11 |
| Fe | 3 | 17 | 1.5e+02 | 27 | 32 | 14 |
| Hap | 2 | 4 | 20 | 1.5e+02 | 32 | 29 |
| Sad | 0 | 4 | 20 | 20 | 1.8e+02 | 13 |
| Sur | 4 | 1 | 4 | 15 | 30 | 1.9e+02 |

**Fig. 10.** Confusion matrix(20x20)

**Table 4.** 20x20 Image Metrics

| Emotions | Precision | Recall |
|---|---|---|
| Anger | 0.98 | 0.83 |
| Disgust | 0.71 | 0.77 |
| Fear | 0.74 | 0.54 |
| Happiness | 0.68 | 0.70 |
| Sadness | 0.61 | 0.79 |
| Surprise | 0.72 | 0.73 |

As seen in Table 4, the model was able to recognize Anger emotion the most with an accuracy of 0.98. The other emotions did not fair quite well at all with the 2nd best accuracy was achieved by Fear with 0.74, then Surprise (0.72) and Disgust (0.71). Sadness performed the worst of the emotions with the model achieving a score of just 0.61.

## 3 Conclusions

Ever since a growth in globalization, a hot exchange of ideas and information has taken place across various cultures. But it is well documented how difficult it becomes to exchange information across people not speaking the same language. This paper tries to find a common ground for people across various cultures speaking the same language. In a real-world scenario, the images/videos might not be perfect to base our model on and as such we need to prepare the CNN model in a way that it can recognize emotion even across an image with a lot of noise. It becomes clear that image size does play an important role in recognizing of any emotion across different cultures. In this experiment, 100x100 image size gives the best result in terms of accuracy whereas in terms of computation time, 20x20 image size would be the best. Thus, to keep a fine balance between both, 50x50 image size is chosen as the best one and going forward, all experiment could be done based on that image size. Across emotions, anger is the most recognized from all the cultures. The opposite could be said about surprise. The earlier research with speech combined with Realtime image extracts should provide a basis for even more accurate results. It remains to be seen if addition of audio would greatly increase the accuracy of the model or not. Further work also needs to be carried to check how implementing noise and introducing angles affected the accuracy.

## Acknowledgements.

The research is supported by the Project BG05M2OP001-1.001-0003 "Center of Excellence in Informatics and Information and Communication Technologies" funded by the OP "Science and education for smart growth 2014-2020", EU through the European structural and investment funds.

## References

Agarap, A. (2018). Deep Lerning using Rectified Linear Units (ReLU). *CoRR*.

Barret, L., Adolphs, R., & Pollak, S. (2019). Emotional Expressions Reconsidered: Challenges to inferring Emotion from human facial movement. *Psychological Science in the Public Interest*, 1-68.

Chai, M. T., Hafeez, U. A., Mohamad, N. M., & Malik, A. S. (2017). *The influence of Emotion on Learning and Memory.* Frontiers in Psychology.

Ekman, P. (1992). An argument for basic emotion. *Cognition d Emotion_6*, 169-200.

Gendron, M., Roberson, D., van der Vyver, J., & Barrett, L. (2006). The enter-face '05 audio-visual emotion database. *International Conference on Data Engineering Workshops.* Washington.

Hue, A. (2020, Jan 29). *Analyticcs-Vidya.* Retrieved from Medium: https://medium.com/analytics-vidhya/dense-or-convolutional-part-1-c75c59c5b4ad

Iliev, A., Mote, A., & Manoharan, A. (2021). Multi-Lingual Classification using Convolutional Neural Network. *Large-Sclae Scientific Computations.* Sofia,Bulgaria.

Mehendale, N. (2020). Facial emotion recognition using convolutional neural networks(FERC). *SN.Appl.Sci.2*.

Nagi, J., Ducatelle, F., Di Caro, G., Cireşan, D., Meier, U., Giusti, A., . . . Gambardella, L. (2011). Max-Pooling convolutional neural networks for vision-based hand gesture recognition. *2011 IEEE International Conference on Signal and Image Processing Applications*, (pp. 342-347).

O'Shea, K., & Nash, R. (2015). An Introduction to Convolutional Neural Network. *Neural and Evolutionary Computing*, 10.

Srivastava, N., Hinton, G., Krizhevsky, A., & Sutskever, I. (2014). Dropout: A simple way to prevent neural networks from. *Machine Learning Rese*.

Wani, T., Gunawan, T., & Qadri, S. (2020). Speech emotion recognition using convolution neural networks and Deep Stride convolutional neural networks. *ICWT'2020*, (pp. 1-6).

Received: June 12, 2022
Reviewed: June 24, 2022
Finally Accepted: July 31, 2022