# Physical Layer Data Augmentation Techniques for Face Recognition and Face Emotion Classification Data

**Ashish Vinodkumar**
MIDS
Duke University
ashish.vinodkumar@duke.edu

**Pranav Manjunath**
MIDS
Duke University
pranav.manjunath@duke.edu

## Abstract (Ashish)

In this paper, we explore various physical layer augmentation techniques such as top-half masks, bottom-half masks, three-fourths mask, and an image augmentation technique, sharpness ratio, on the Face Recognition and Emotion Classification dataset using a VGG-16 model architecture. We learn that data augmentation in general boosts model performance and reduces overfitting. Specifically, masking one-fourth of the pixels in the physical layer, results in an accuracy **insert accuracy** similar to that of the baseline (full-image) model of **insert baseline accuracy**.

## 1 Background/Introduction (Ashish)

In the area of deep learning, data pre-processing is an important stage that ensures readiness of data for model training. Particularly in face detection and emotion classification applications, one of the important pre-processing techniques that is commonly applied is called data augmentation. Data augmentation is particularly helpful when the amount of images available is scarce and hard to be collected. In this particular project, we are interested in observing the effects of physical layer augmentation. Specifically, assess accuracy metrics of models trained on images that are masked at the “camera lens” level, either in the top-half, bottom-half, or three-fourths lens pixel level. Performing such physical layer data augmentation will directly limit the numbers of pixels incident on the camera lens. This paper looks to identify if such physical layer augmentation combined with general “post-image capture” data augmentation techniques such as sharpness ratio, can yield comparable or better results compared to the baseline non-augmentation mode.

## 2 Related Work (Ashish)

For this project, the team aims to follow an earlier work on assessing the effects of physical layer augmentation techniques, applied on a face detection and emotion classification dataset, by their testing accuracy and ROC curves.[1] Specifically, the team focuses on three physical layer augmentation techniques: top-half mask, bottom-half mask, one-third mask, along with one data augmentation technique sharpness ratio, to assess model performance metrics. The team shows that some augmentation techniques are able to extract face detection and emotion classification image statistics more effectively than others, which will lead to better predictive performance in the corresponding models.

## 3 Method (Ashish)

### 3.1 Data Pre-Processing

We obtained our dataset from Kaggle.[2] We had 2 separate datasets, one for Face Detection and the other for Face Emotion Classification. For the Face Detection dataset, we first parsed through the individual images in their respective folders, and created a root level folder with all images inside of it. For the Face Emotion Classification dataset, we parsed through the folders containing images for the 3 emotions of interest: happy, neutral, and sad. We then created a table schema consisting of the filepath of the image and its corresponding emotion label.

### 3.2 Model Architecture and Hyperparameters

For the Face Detection model, we performed transfer learning, by using an existing pre-trained model: CV2 Cascade Classifier. Using the CV2 Cascade classifier allowed us to detect the face coordinates allowing us to create a 64x64 image crop around the face.

For the Face Emotion Classification model, we followed the study by Porcu et al. (2020), and used VGG-16 as our model architecture. This VGG-16 model architecture allowed us to create the baseline model and all 3 physical layer augmentations along with 1 data augmentation experiment as outlined in the paper. For all the 4 models, we set the learning rate to 1e-3. We further experimented with a dropout parameter of 0.6, and a L2 regularization of 1e7.

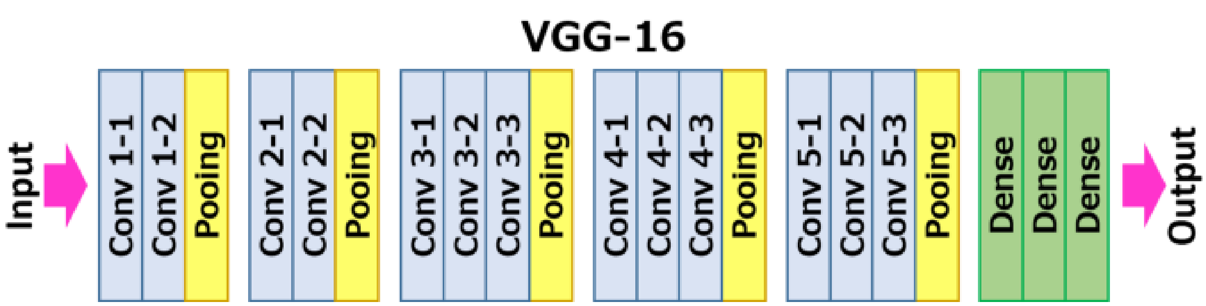


Figure 1: VGG-16 Model Architecture

### 3.3 Physical Layer Augmentation

To compare the effect of different physical and data augmentation techniques on the model's predictive performance, we used a baseline model that was trained on the entire 64x64 image, to then compare against the physical layer augmented models. We explored at least 3 distinct values for each physical layer augmentation and data augmentation (sharpness ratio), to ensure the best experimental result from each technique is used in the model comparison.

#### 3.3.1 Top-Half Mask

Top-Half Mask is a physical layer augmentation technique that involves reducing the number of pixels incident on the camera lens, i.e we block all pixels incident on the camera lens in the top-half of the image. Given that our original image is of size 64x64, removing the top-half pixels results in an image of 32x64. Below shows an image from the dataset, where the left represents the original non-masked 64x64 image, and the right represents an image with a top-half mask applied to it.

Figure 1: Original (64x64) Image

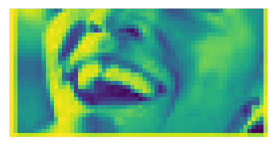

Figure 2: Top-Half Mask (32x64) Image

#### 4.3.2 Bottom-Half Mask

Bottom-Half Mask is a physical layer augmentation technique that involves reducing the number of pixels incident on the camera lens, i.e we block all pixels incident on the camera lens in the bottom-half of the image. Given that our original image is of size 64x64, removing the bottom-half pixels results in an image of 32x64. Below shows an image from the dataset, where the left represents the original non-masked 64x64 image, and the right represents an image with a bottom-half mask applied to it.

Figure 3: Original (64x64) Image

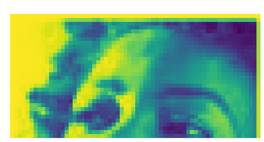

Figure 4: Bottom-Half Mask (32x64) Image

#### 4.3.3 One-Fourth Mask

One-Fourth Mask is a physical layer augmentation technique that involves reducing the number of pixels incident on the camera lens, i.e we block all pixels incident on the camera lens except for the middle three-fourth of the image. Given that our original image is of size 64x64, removing one-fourth pixels results in an image of 48x64. Below shows an image from the dataset, where the left represents the original non-masked 64x64 image, and the right represents an image with a one-fourth mask applied to it.

Figure 5: Original (64x64) Image

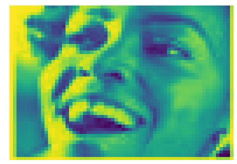

Figure 6: One-Fourth Mask (48x64) Image

### 3.4 Training and Evaluation

In terms of epoch number for model training, each model was trained on 25 epochs with each batch size equaling 64 images. During each epoch, the random shuffle for the training loader function was set to True to ensure the augmented and original images were blended in each batch of images. After each training epoch, each model was tested with the batch size of 64 images. Once all 25 epochs of testing is finished, the highest testing accuracy is saved for comparison purposes. For the hyperparameter used in each data augmentation technique, at least 3 different values were explored, and the one with highest testing accuracy was chosen as the final hyperparameter for that specific augmentation technique. Below are the final hyperparameters chosen for each augmentation technique:

Sharpness Ratio:

## 4 Experiment Results (Pranav)

## 5 Discussion/Conclusions (Ashish)

# 6 Reference


1. Pei, Zhao, et al. “(PDF) Face Recognition via Deep Learning Using Data Augmentation Based on Orthogonal Experiments.” *ResearchGate*, https://www.researchgate.net/publication/336061375_Face_Recognition_via_Deep_Learning_Using_Data_Augmentation_Based_on_Orthogonal_Experiments.
2. Porcu, Simone, et al. *Evaluation of Data Augmentation Techniques for Facial ...* https://www.researchgate.net/publication/345940392_Evaluation_of_Data_Augmentation_Techniques_for_Facial_Expression_Recognition_Systems.
3.


# 7 Appendix (Ashish)

## 7.1 Web Application

We utilized Flask, a micro web framework written in Python. Flask allowed us to effortlessly create a Front-End User Interface to showcase the procedures of the web-application. We used a combination of HTML, CSS, JavaScript, and Flask to orchestrate the web application. Specifically, we first created a home screen that would allow users to upload a face image of their choice. We then created a “Step 1” UI page that showcased how the CV2 Face Detection model was being utilized to identify the face in the image, and crop the image pertaining only to the face in a 64x64 frame. Finally, we created a “Step 2” UI page that showcased our custom baseline VGG-16 model and the respective masked (top-half, bottom-half, and one-fourth masked) models, and its classification performance against the baseline.

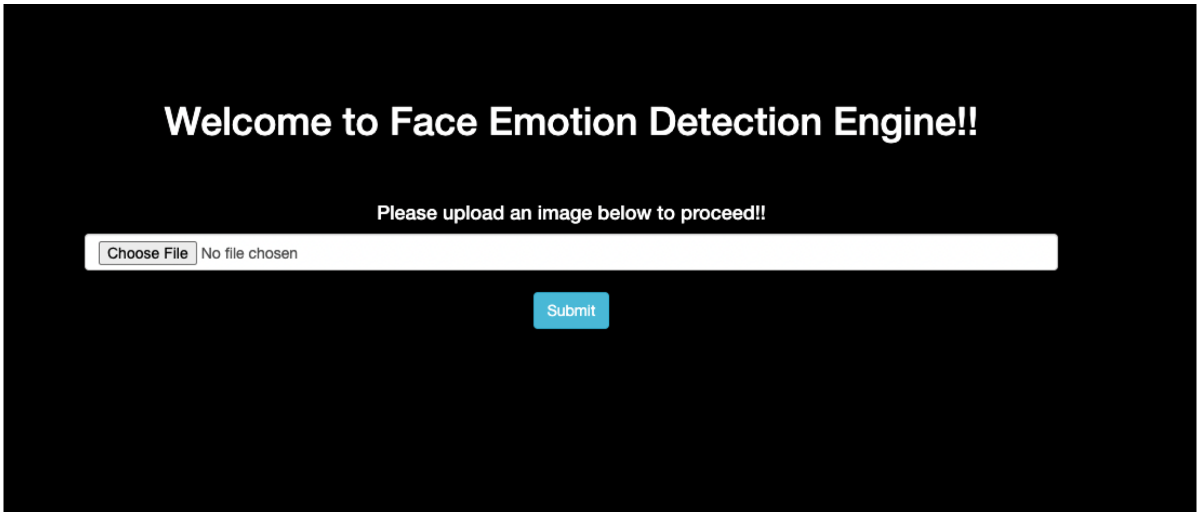


Figure 7: Home Page UI: Users can upload a face image of their choice.

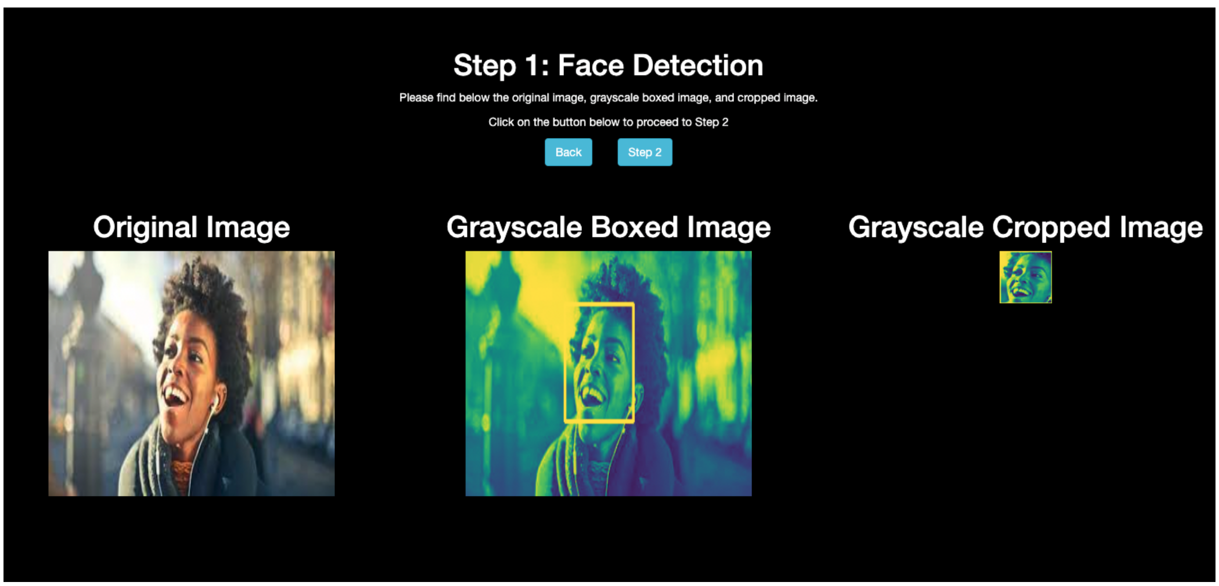


Figure 8: Step 1 Page UI: Users can see CV2 Cascade Classifier for Face Detection.

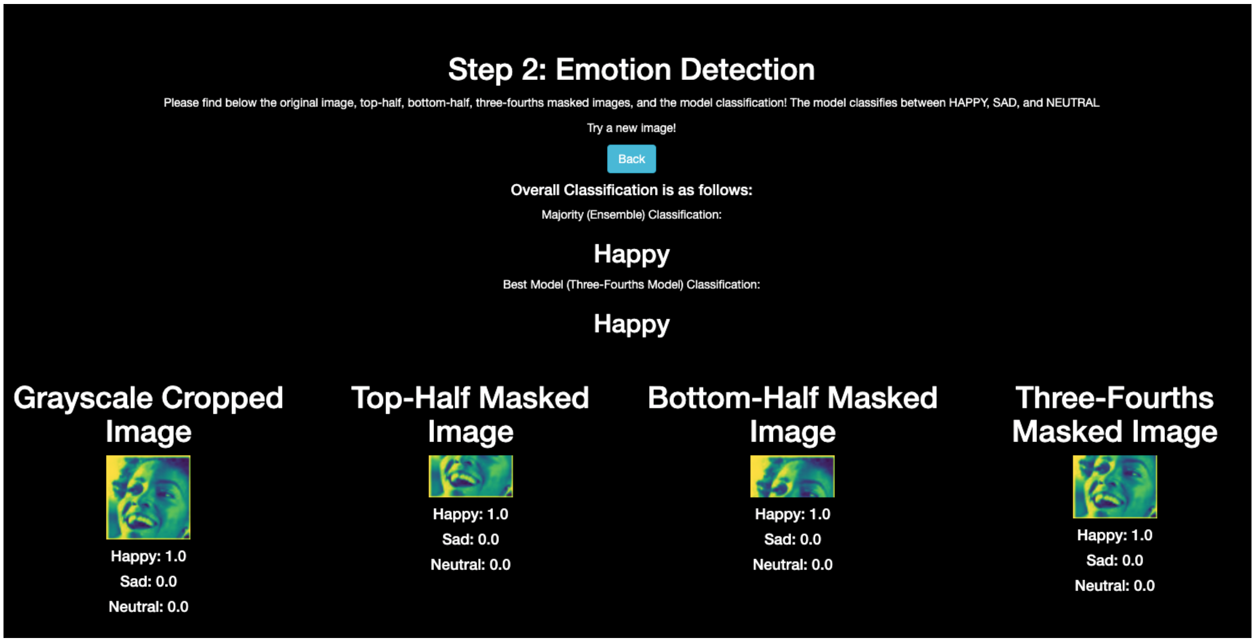


Figure 9: Step 2 Page UI: Users can see Masked models and their classification predictions.