KAIST

# Facial Expression Recognition
# Milestone Report

**Abbas Mammadov** **20200844** | **Kaleb Mesfin Asfaw** **20200805** | **Zahra Bayramli** **20200812** | **Tivan Varghese George** **20200829**

School of Computing
Korea Advanced Institute of Science and Technology

## 1 Motivation

Humans have many avenues through which they transfer information; one such avenue being facial expressions. Facial expressions can reveal a lot of hidden information, including a person's emotional state, which can be used within contexts such as psychological assessment or legal interrogation. Considering the importance of human-computer interaction, a facial-expression emotion classifier becomes increasingly more desirable. Even though there already exist several high-accuracy models that distinguish simple expressions and emotions in simple and controlled settings, there is still room to improve in models that aim to identify complex emotions based on facial expressions, to a high-degree in a real world setting.

As such, in this project, we aim to create an efficient model that classifies emotion based on facial expressions in a real world setting. This means that our goal is to create a model that can accurately classify emotion independent of age, ethnicity, and appearance of a person. This model can potentially be of great service in the medical, psychiatric, legal, and leisure industries due to the planned generality of its classifications.

## 2 Methods

### 2.1 Dataset

We have used the FER2013 dataset which was collected automatically by the Google image search API. This large-scale data, which is available on Kaggle, has been studied well and played a crucial role in various research papers. The dataset consists of 35887 grayscale images of faces, which have been normalized to the 48x48 pixel scale. In FER2013, images of 7 facial expressions has been distributed as follows: Angry (4953), Disgust (547), Fear (5121), Happy (8989), Sad (6077), Surprise (4002), and Neutral (6198). We divided the dataset into train, validation, and test sets with ratio 80:10:10 (train- 28709, validation- 3589, test- 3589). During the training, we did data augmentation to increase the amount of data by adding slightly modified copies of existing images.

### 2.2 Models

**Resnet-50**

First, we decided to tackle this problem by using ResNet-50 as our baseline model. ResNet-50 is a convolutional neural network (CNN) with 50 layers depth. In order to classify images into 7 categories, we have added an output layer of 7 emotion classes. Using transfer learning techniques, we have also added two fully connected layers of sizes 1024 and 4096 to our model.

,

**VGG-19**
VGG-19 is another pre-trained model that we have explored and used in our problem. Being shallower than ResNet-50, it is a convolutional neural network, with 19 layers, pretrained on more than a million images from ImageNet database. It has generally been used in diverse and complex classification tasks. In our implementation, we also added dropout, and fully connected layers in addition to pretrained layers of the model.

## 3 Preliminary experiments

As we stated in our model section, we used ResNet-50 as our first pre-trained model (baseline). ResNet-50 in PyTorch has been pretrained on more than one million images of the 'ImageNet' dataset. At this moment, we are using only the FER2013 dataset. As a result of this, we will be exposed to some factors especially related to this dataset which prevent us from achieving high accuracy. All the results and their implementations can be found in our GitHub page.

To mention some, this FER2013 dataset consists of about 35,000 images, which is a bit smaller compared to the well known huge sized datasets that contributed to the boosting of accuracy of many models. It's also not well balanced. For instance, the number of the images labeled as "disgust" is almost incomparable with respect to the other emotions (more on this, later). Moreover, the fact that the dataset is entirely composed of grayscale images poses an obstacle to our model to perform well on the real world images.

As our baseline, we have added a last linear layer which classifies the images into the 7 categories of emotions. Using this architecture, we have got 0.6 accuracy on the validation dataset by training our model with batch size of 32, and epoch 50. Then, to maximize our accuracy, we utilized the concept of transfer learning here, and we have added 2 fully connected layers, each with sizes of 1024 and 4096 on our 'learned' layers. After training for 50 epochs, with batch size of 128, we got an accuracy of 0.73 on the training data and 0.64 on the validation data. However, we have realized that this method of transfer learning caused our model to overfit our data because we are adding even more layers over a pretrained model. This can be highly diagnosed from the following loss-curve that we have obtained from the ResNet-50 model, with 2 additional FC layers.

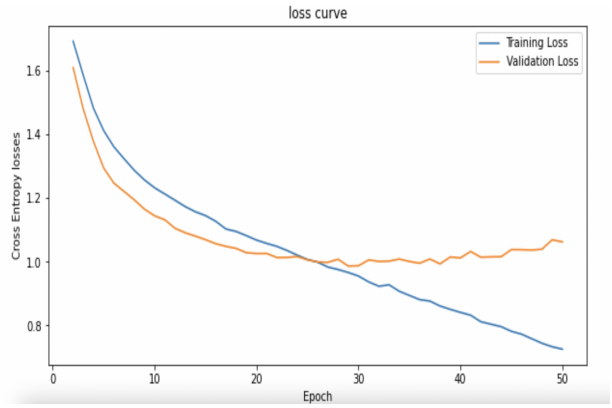


Figure 1: Loss curve of the ResNet-50 model (with 2 additional FC layers)

As we can see from the graph, the training error has declined considerably. It doesn't even seem to converge at this stage of iteration. It may have declined even more if we had trained our model for a couple of more epochs. This indicates that the model fitted our training data pretty well. However, the validation error has already started increasing. And from the graph, we can see that if we trained our model for more epochs, it would increase, because right now, it's in an increasing direction. (in other words, our model has high variance). We first assumed this problem might occur, and hence we have introduced a weight decay of 0.0001 as one of our methods of regularization. As it can be seen from our model architecture we have also used a dropout (with probability hyperparameter 0.5) thinking that it might reduce overfitting. We have tried to augment some artificial images using torchvision's inbuilt module 'transforms' as the FER2013 dataset has a relatively small amount of images for our model to learn how to classify the emotions. Therefore, for the data augmentation, we

have tried to flip some images horizontally (with probability of 0.5). We also have tried to generate some images by rotating the existing images on $\pm 10$ degrees and by $\pm 10\%$ horizontal and vertical shifting. However, our dataset was still not enough, and the fact that we are using a pretrained model, caused our model to quickly overfit the training data.

We also tried to analyze the prediction of our model towards each emotion.

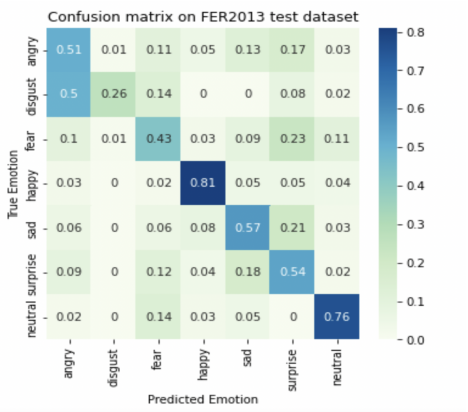


Figure 2: Confusion Matrix of the ResNet-50 model (with 2 additional FC layers)

As we can see from the above figure, we can see that most of the images have been correctly classified to their categories, but if we see the 'disgust' column, we can see that it's the category in which fewer actual 'disgust' images are correctly classified as 'disgust', and for our surprise, there is no any emotion (except very few 'angry' and 'fear' emotions) which has been misclassified as 'disgust'. This indicates that the model hasn't yet grasped the behavior of 'disgust' images. As we have discussed earlier, this is due to the presence of a small amount of images labeled as 'disgust'.

In addition to the 2 pretrained models we mentioned in the above, we also have used the 'VGG19' model, whose accuracy is 0.626 on the validation set. The loss curve of the model was a bit fluctuating especially on the validation set. We will work more on maximizing the accuracy of this model in our next report.

Throughout our experiment, we have used SGD - Nesterov (Accelerated) momentum /momentum factor = 0.9/, and a learning rate of 0.001. We used the typical loss function 'CrossEntropy' to calculate the loss of our model.

## 4 Next steps

Despite our attempts to reduce overfitting, it can be clearly seen from our graph that our best model (ResNet 50 with 2 FC layers) overfitted our training data. Hence, we are planning to maximize the weight decay hyperparameter so as to penalize some of the weights. We are also aiming to increase the probability hyperparameter of the dropout.

In this experiment, we have only used the FER2013 dataset, and its augmentation. Next, we will train our model using at least one more dataset. We are aiming to include most of this second dataset on our training part so that we can obtain more balanced images on some emotions. This improves our accuracy, especially on the 'disgust' label. Adding more data will also definitely reduce model overfitting. Because our model will now get enough resources to generalize emotions well. In addition to this, we will also try to design more ways of data augmentation. We also aim to perform a more detailed data analysis so as to obtain some more methods of normalization.

Regarding models, we aim to implement our own CNN model so as to compare the accuracy we can get with the pretrained ResNet, and VGG19 models. We will investigate how many iterations we can make to have an accuracy of at least as high as the pre-trained ones. We also plan to train some more complex CNN architectures like inceptions, SeNet Networks and potentially we will try to add a support vector machine model and compare its accuracy with the other deep learning framework. We will also try to perform a more detailed error analysis for each of our models.

## 5 Contributions

**Abbas Mammadov**

- Model building- VGG19
- Transfer learning models
- LaTex formatting for the reports

**Kaleb Mesfin Asfaw**

- Model building - Resnet50
- Error analysis

**Zahra Bayramli**

- Hyperparamter tuning
- Managed infrastructure

**Tivan Varghese George**

- Data pre-processing
- Data Augmentation

## References


[1]https://github.com/abbasmammadov/Facial-Expression-Recognition

[2]https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/data

[3]https://www.koreascience.or.kr/article/JAKO201809253681042.pdf?fbclid=IwAR3gnwoUoE$_arj0gKlfp8uM4Ih92eIwMcpChEw9CK$

[4]https://arxiv.org/pdf/1804.08348.pdf

[5]https://arxiv.org/pdf/1512.03385.pdf

[6]https://arxiv.org/pdf/1409.1556.pdf