CS640 Project: Face Emotion Recognition using Python

Zhenghang Yin, Lingyan Jiang, Lanqi Li
Boston University
Boston, Massachusetts 02215
{johnyin,lingyanj,llq}@bu.edu

Abstract

Recent years have witnessed a massive bloom in interest in facial recognition technologies. The goal of facial recognition systems is to find human faces in images or videos and then classify them by their identities or attributes. The motivation of the project is to optimize the facial emotion recognition in clips of presidental candidatial videos. We utilize several pre-trained models within the facial expression recognition system (FER) in the project to analyze each presidential candidate's emotion. We trained data via BU SCC[1] platform and collected labels generated by FER. Then we compared them with annotated labels and found ways to improve its predictable accuracy.

Introduction

The presidential and vice-presidential debates are very interesting. The presidential candidates quickly turned into emotional arguments that we could clearly see on their faces. However, even for humans, identifying facial expressions is difficult. Therefore, we were trying to find and tune a model to predict presidential emotions.

Facial expressions are one of the most powerful ways for depicting specific patterns in human behavior and describing the human emotional states. Facial expression recognition has been a dynamic topic in computer vision in recent years. It will be interesting for us to use FER to quantify each candidate's emotions. The machine learning model will be trained to classify the emotion shown on an image of a face. Thus, the computer program will tell us which kind of emotion on the face through images, like positive or negative.

However, it is difficult to identify facial expressions and the facial expressions recognition system often suffers from variations in expressions among individuals. The state-of-the-art methods in image-related tasks such as image classification and object detection are all based on Convolutional Neural Networks (CNNs)[2]. These tasks require CNN architectures with millions of parameters. The best way to analyze what emotions each candidate wore on their face is to look at the distributions of detected expressions throughout videos. To do that, we first need to construct a dataset of images from the debate videos, process them with the expression recognition model, and then aggregate and visualize the results.

In this project, we installed FER through pip and used the `MTCNN` network to do facial recognition. For each full debate video, we sampled roughly one from every twenty frames resulting in a few thousand images per person. We then cropped the video frame to contain only the face, which allowed the expression recognition software to perform better. The expression recognition model was then run on every image in the dataset and the top emotion detected in each image was tallied.

In this work, our main objective was to understand better and improve the performance of emotion recognition models in the process. We also took some approaches from recent publications, including transfer learning and ensembling, to enhance our model's accuracy. At the same time, we did not use any auxiliary data for my model other than the Dataset to train my models.

Related Works

Before starting the hands-on project, we first researched deep models on facial expression analysis. In the research process, we found four related works. After we learned about other people's methods of facial expression recognition, we began to choose the best performing model.

- Combination CNN with RNN[3].

  In the paper, Emotion Recognition on large video dataset was based on Convolutional Feature Extractor and Recurrent Neural Network, they explained that they considered the emotion recognition task as a classification as well as a regression task by processing encoded emotions in different video datasets using deep learning models (Rangulov and Fahim 2020). Their model combines the convolutional neural network (CNN) with the recurrent neural network (RNN) to predict dimensional emotions on video data. In their algorithm, CNN first extracted feature vectors from video frames. Second, RNN trained

[1] https://www.bu.edu/tech/support/research/computing-resources/scc

[2] https://en.wikipedia.org/wiki/Convolutional_neural_network

[3] https://en.wikipedia.org/wiki/Recurrent_neural_network

some feature vectors for exploiting the temporal dynamics of video. But analyzing how each neural network contributed to the system's overall performance, they discovered the problem of overfitting on an unbalanced dataset. The problem was solved by the downsampling technique to balance the dataset. And their proposed method was implemented using Tensorflow Keras.

- Facial Action Coding System(FACS).

  Facial Action Coding System(FACS) was a system to taxonomize human facial movements by their appearance on the face[4] and was developed by American psychologists Ekman and Friesen in 1978. They used Action Units(AU) to classify facial expressions which were the fundamental actions of individual muscles or group muscles. FACS was used for classifying human facial movements in the years 2001-2006.

  As we read in the paper Automated Facial Expression Recognition, Based on FACS Action Units written by Lien et al. and Kanade et al., they developed a computer vision system that automatically recognizes individual action units or action unit combinations in the upper face using Hidden Markov Models (HMMs) (Lien et al. 1998). Their approach to facial expression recognition was based on the Facial Action Coding System(FACS), which separates expressions into upper and lower face action. As to extract facial expression information, their team uses three approaches: (1) facial feature point tracking, (2) dense flow tracking with principal component analysis (PCA), and (3) high gradient component detection (i.e., furrow detection). For the dataset, this study includes more than 260 image sequences and 5000 images. Subjects ranged in age (18-35) and ethnicity (Caucasian, African-American, and Asian/Indian). The recognition results of the upper face expressions using feature point tracking, dense flow tracking, and high gradient component detection are good. They reach 85%, 93%, and 85% respectively.

- Facial expressions recognition Graph convolutional network (FER-GCN).

  The Graph Convolutional Network (GCN) (Liu, Zhang, and Zhou 2020) had great performance in learning correlative feature representation for specific tasks, which could share the messages in graphs and reconstruct the hidden states of each node to focus more on the significant information. It introduced a GCN based end-to-end framework for dynamic FER tasks. A GCN layer had been applied between CNN and RNN to learn more facial expression features to capture dynamic expression variation.

  The GCN layer updated the individual features of each frame and learns an adjacency matrix which represents the inter-dependency among frames (Liu et al. 2020). The LSTM was further applied to learn their long-term dependencies to model the variation. Then they adopt the learned adjacency matrix of the GCN layer to represent expression intensities in time series, which could decrease the influence of the weak expressional features from neutral frames and exploit more expressional contributing ones from peak frames for final classification. Compared to state-of-the-art approaches, their method is much more robust.

- The FaceChannel: A Fast & Furious Deep Neural Network for Facial Expression Recognition

  The FaceChannel (Barros, Churamani, and Sciutti 2020) was a light-weight convolution neural network, with around 2 million updatable parameters. They extended it by adapting the topology of the VGG16 model, which has 10 convolutional layers and 4 polling layers. They trained the model using the experimental protocol given by each dataset, like Affect Net (Mollahosseini, Hasani, and Mahoor 2019), OMG-Emotion (Barros et al. 2018), FER+[5], FABO. They got accuracy when evaluating the Face Channel with the different datasets. They also compared the number of different deep learning models' parameters, such as MobileNet (Howard et al. 2017), VGG13+, AlexNet[6], and VGGFace+[7]. The FaceChannel had by far the lowest number of parameters although presenting a better or same performance as the other models. The FaceChannel was able to learn general features, even with a high-lighted architecture. It was another testament to its strength at quick adaptation towards novel scenarios.

- Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Network.

  In this paper, they proposed an unified cascaded CNNs to improve the accuracy of multi-view face detection and design an online hard sample mining method. The proposed CNN had three stages. Firstly, they exploited a fully convolutional neural network P-Net to obtain the candidate windows and bounding box regression. It randomly cropped several patches from WIDER FACE to collect positives, negatives and part face and crop faces from CelebA as landmark faces. Then they used the estimated bounding box regression vectors in a similar manner as (Farfade, Saberian, and Li 2015) to calibrate the candidates and employed non-maximum suppression (NMS) to merge highly overlapped candidates. Secondly, all candidates were fed to another CNN R-Net, which rejected a lot of false candidates. It used the first stage of their framework to detect faces. Lastly, they used O-Net, which was similar to R-Net, to collect data but used the first two stages of their framework to detect faces.

- Real-time Convolutional Neural Networks for Emotion and Gender Classification They proposed a general CNN building framework which can design real time CNNs. The implements would provide face detection, gender classification and that achieved human-level performance when classifying emotions. A real- time visualization of the guided-gradient back-propagation proposed by Springenberg (Amodei et al. 2015) was implemented to validate the features learned by CNN.

[4] https://en.wikipedia.org/wiki/Facial_Action_Coding_System

[5] https://github.com/microsoft/FERPlus

[6] https://en.wikipedia.org/wiki/AlexNet

[7] https://github.com/rcmalli/keras-vggface

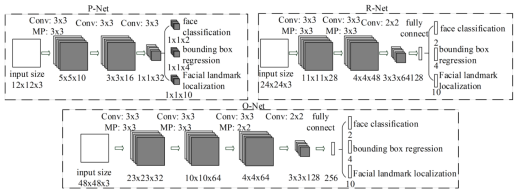

Figure 1: The architectures of P-Net, R-Net, and O-Net

They also proposed two models designed with the idea of creating the best accuracy over number of parameters ratio. Reducing the number of parameters could overcome two important problems. First, the use of small CNNs alleviated them from slow performances in hardware-constrained systems such as robot platforms. And second, the reduction of parameters provided a better gener- alization under Occam's razor framework. Their first model relied on the idea of eliminating completely the fully connected layers. The second architecture combined the deletion of the fully connected layer and the inclusion of the combined depth-wise separable convolutions and residual modules. Both architectures were trained with the ADAM optimizer (Kingma and Ba 2017).

Technology Methods

In the project, we used the FER package utility to perform the task of facial expression recognition. The package was available at PYPI[8], and could be installed via `pip install FER`. In the remaining of the section, we will discuss the structures of FER, the details of models used inside FER and how the FER package can be applied to the project.

Package structure

FER is a powerful and out-of-the-box tool which can be used to analyze and capture faces in images and videos, while predicting the gender and emotion of each face. Basically it provides two interfaces: **FER** (to process single-frame images) and **Video** (to process multiple-frame videos).

- **FER** is the core part of the tool. It uses TensorFlow[9] as the main deep learning framework for prediction and uses a two-stage model prediction procedure: in the first stage, the model will try to find all rectangles in the image where exists a face, and in the second stage, another model will try to recognize the emotion of the face in above rectangles.

  In two stages, FER allows users to choose different pretrained models. In the first stage, user can choose to use the default model (Arriaga, Valdenegro-Toro, and Plöger 2017) or MTCNN (Zhang et al. 2016), which will be introduced briefly in the next subsection. In the second stage, it will use the model in (Arriaga, Valdenegro-Toro, and Plöger 2017) to detect emotions.

[8] `https://github.com/justinshenk/fer`
[9] `https://pypi.org/project/tensorflow/`

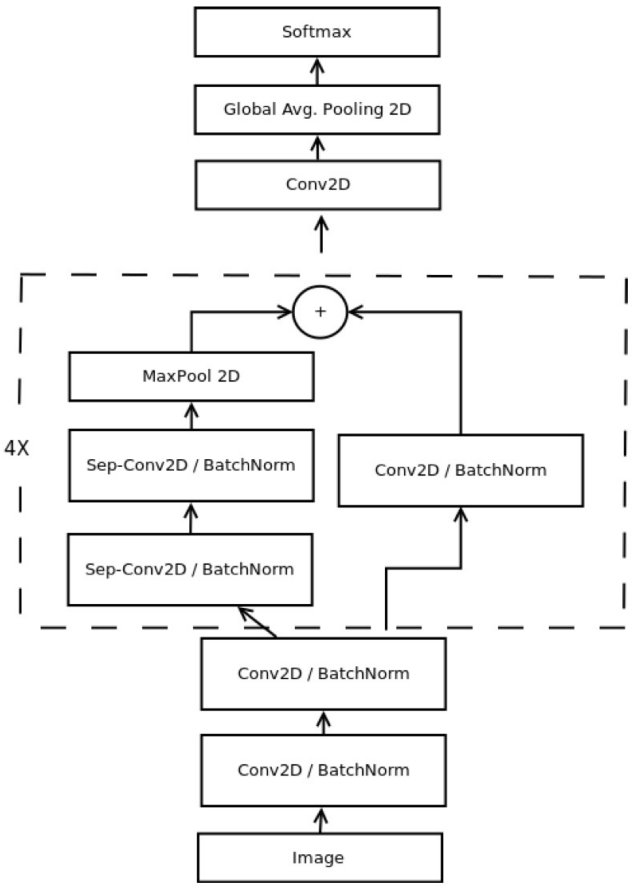

Figure 2: The model for real-time classification

When using FER, it's needed to call `FER.detect_emotions` function. The function will first preprocess the image, and then call `FER.find_faces` function to return a list of rectangle data representing the faces detected in the image. Next, it will crop the corresponding faces into small picture clips and passing them to the second model to predict the emotions.

In FER, there are 7 builtin emotion labels: (**0**: "angry", **1**: "disgust", **2**: "fear", **3**: "happy", **4**: "sad", **5**: "surprise", **6**: "neutral"). They're overlapping with the labels defined in the task, so they can be used directly.

- **Video** is an encapsulation of the interface FER. In general, it first divides the video into multiple frames at a certain frequency, and then calls `FER.detect_emotions` for each frame. Then for each frame, video will draw rectangles on the video, and recompresses frames into a new video.

Application

In this project, We exploited the FER package to do face emotion recognition in the presidential campaign videos. First the videos were passing into the **Video** class to ana-

lyze emotions one by one. Then, in each video, We selected the label with the highest probability for each frame as the predicted results. Each video was divided into equal parts according to a fixed interval (every 1000 frames, with time interval equals to $\frac{1000}{\text{fps}}$). Finally, We counted and chose the label with the maximum sum for each video.

The pseudocode of the project is similar to the following:

Algorithm 1 Main framework in our project.

```
detector ←new FER object
for each video, label_true ∈ Videos do
    data ← analyze(video, detector)
    data_labels ← max_index(data group_by frame)
    label_pred ← max_count(data_labels))
    Compare(label_pred, label_true)
end for
```

Here are some of the emotional analysis we made using this model.

Figure 3: Biden's emotional analysis

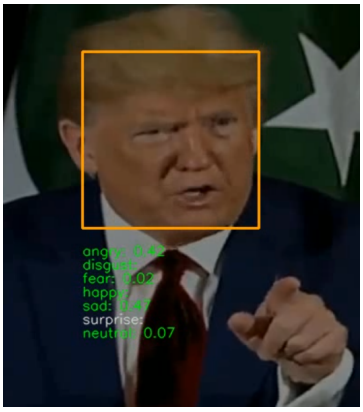

Figure 4: Trump's emotional analysis

Experiments and Results

In this section, we will introduce how we run the code, how to process the data results, and the final prediction results.

Experiments Steps

Considering the programs part that relies heavily on GPU can having problems of long-time running, we divided the task into two stages:

1. The first step was to take screenshots of each video file at a fixed frame rate, and use a pre-trained model to detect face rectangles and recognize emotions. This step required the use of GPU and was quite time-consuming. During the step, we three team members used the deep learning platform on BU SCC, trained 2019 (1,320 videos) and 2020 (1,506 videos) datasets for up to 24 hours.

   Two face detection algorithms are used in FER: the method in (Arriaga, Valdenegro-Toro, and Plöger 2017) (marked **MTCNN=False**) or (Zhang et al. 2016) (marked **MTCNN=True**). We ran all data sets separately in two methods, by specifying python command `detector = FER(MTCNN=True or False)`.

   The following was a partial sample of the raw data generated by FER. The content within each braces indicated a frame. Then within a frame, there was multiple fields, `box` means the rectangle data about the faces, and the other fields `angry`, `happy`, etc. were scores of different emotions. For each faces, because the facial recognition models used in FER did softmax computation in the last step, the total score of all emotions is 1.0.


```
[
    {
        "box0": [430, 110, 84, 84],
        "angry0":0.43, "disgust0":0.0,
        "fear0":0.07, "happy0":0.02,
        "sad0":0.31, "surprise0":0.01,
        "neutral0": 0.17
    },
    {
        "box0": [429, 109, 85, 85],
        "angry0":0.53, "disgust0":0.0,
        "fear0":0.07, "happy0":0.01,
        "sad0":0.28, "surprise0":0.01,
        "neutral0": 0.1
    },
    ...
]
```


   Because both models were pretrained to find multiple faces in one frame. In this project, we simply discarded the extra faces except the default box_0 face.

2. The second step was to analyze intermediate raw data files, produce prediction results, and then compared them with golden labels.

   Because the difference between our project and the pre-trained model: there were 7 labels in the model, but only 3 labels in our project. So we did the following mapping:

labels in pretrained model		our project
happy	→	Positive
surprise	→	Positive
neutral	→	Neutral
fear	→	Neutral
sad	→	Neutral
disgust	→	Negative
angry	→	Negative

In order to balance the Positive, Neutral, and Negative 3 class labels, we sorted the results of FER and then converted them into new class labels.

Results

The results of the project are as the following:

Datasets	Counts	Use MTCNN?	accuracy
2019	1320	√	**60.05%**
			53.94%
2020	1506	√	55.35%
			57.75%

The result is in the following format:

filename	predict	golden	123		
			pos	neg	neut
biden_58_0	Neut	Neg	77	67	187
biden_124_0	Neut	Neut	13	5	42
biden_138_0	Neut	Neut	0	0	4
biden_139_2	Neut	Neut	0	3	120
biden_143_0	Neut	Neg	23	42	139
biden_143_2	Neut	Neut	14	88	179
biden_274_0	Pos	Neg	31	0	25
biden_308_0	Pos	Neg	55	1	20
biden_411_0	Neut	Neg	0	0	112
...					

Discussion

First, the results of this task have been described above. We can see that the results for two different data sets: the 2019 and 2020 presidential candidate data sets are not very satisfactory. In this paragraph, we focus on discussing its causes and our thinking.

Algorithm Bias

The FER (MTCNN=False or True) algorithm we found has a certain emotion recognition bias when its pretrain and labels are selected. In the Fer algorithm, it divides people's facial emotions into seven categories: angry, disgust, fear, happy, sad, surprise, and neutral.

Besides, we matched the given labels in the later stage. we mapped into three categories of emotions: Positive, Neutral, and Negative. As you can see in Fer's algorithm, there are more than half of the negative emotions in the labels. If it is a simple classification or detection of different emotions without any label constraints, there is surely no problem. But when only three types of emotions are needed, it is obvious that negative emotions account for the vast majority. This has brought the deviation and bias of the algorithm, which is one of the reasons why our team believes that the accuracy rate is kind of low.

Model Difference

In the process of our training model, our group unexpectedly found that for the 2019 data set, we use the `FER(MTCNN=True)` algorithm to achieve higher accuracy. For the 2020 data set, we use the `FER(MTCNN=False)` algorithm for better results.

When we observed the original raw data, we found that many faces of the 2019 mp4 data were very clear in the video, and many of them were directly shot videos instead of secondary videos selected from the news. But in 2020 mp4 data, many secondary videos selected from the news have relatively small faces. And some of them were videos of profile faces. It can be inferred that when the face recognition degree is higher and the video face is clearer, the effect of using `FER(MTCNN=True)` will be better. When the face situation in the video is in the opposite situation, the effect of using `FER(MTCNN=False)` will be better.

Conclusion and Future Work

In the paper, our motivation for this project is to optimize the facial emotion recognition in clips of presidential candidate videos. During the steps, our team used the deep learning platform on BU SCC, trained 2019 (1,320 videos) and 2020 (1,506 videos) datasets for up to 24 hours. We utilized several pre-trained models within the facial expression recognition system (FER): MTCNN = True or MTCNN = False in the project to analyze each presidential candidate's emotion.

After running the FER algorithm, we collected the labels generated by FER which is already built into the algorithm. Then, we mapped the 7 emotions building labels generated by the FER algorithm to three requested golden labels as negative, neutral, and positive.

As to the result, we reached 60.04% and 55.35% respectively when using MTCNN=True in the FER algorithm for the 2019 and 2020 dataset. While using MTCNN=False, we reached 53.74% and 57.75% respectively. We referred that the low accuracy may be related to the FER label bias and deviation and low video face definition, size, and shooting angle.

In the future, we will exploit the inherent correlation between face emotion recognization and the FER algorithm or even other algorithms, to further improve the performance.

References

[Amodei et al.2015] Amodei, D.; Anubhai, R.; Battenberg, E.; Case, C.; Casper, J.; Catanzaro, B.; Chen, J.; Chrzanowski, M.; Coates, A.; Diamos, G.; Elsen, E.; Engel, J.; Fan, L.; Fougner, C.; Han, T.; Hannun, A.; Jun, B.; LeGresley, P.; Lin, L.; Narang, S.; Ng, A.; Ozair, S.; Prenger, R.; Raiman, J.; Satheesh, S.; Seetapun, D.; Sengupta, S.; Wang, Y.; Wang, Z.; Wang, C.; Xiao, B.; Yogatama, D.; Zhan, J.; and Zhu, Z. 2015. Deep speech 2: End-to-end speech recognition in english and mandarin.

[Arriaga, Valdenegro-Toro, and Plöger2017] Arriaga, O.; Valdenegro-Toro, M.; and Plöger, P. 2017. Real-time convolutional neural networks for emotion and gender classification. *arXiv preprint arXiv:1710.07557*.

[Barros et al.2018] Barros, P.; Churamani, N.; Lakomkin, E.; Siqueira, H.; Sutherland, A.; and Wermter, S. 2018. The omg-emotion behavior dataset.

[Barros, Churamani, and Sciutti2020] Barros, P.; Churamani, N.; and Sciutti, A. 2020. The facechannel: A fast furious deep neural network for facial expression recognition.

[Farfade, Saberian, and Li2015] Farfade, S. S.; Saberian, M. J.; and Li, L. 2015. Multi-view face detection using deep convolutional neural networks. *CoRR* abs/1502.02766.

[Howard et al.2017] Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications.

[Kingma and Ba2017] Kingma, D. P., and Ba, J. 2017. Adam: A method for stochastic optimization.

[Lien et al.1998] Lien, J. J.; Kanade, T.; Cohn, J. F.; and Ching-Chung Li. 1998. Automated facial expression recognition based on facs action units. In *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, 390–395.

[Liu et al.2020] Liu, X.; Jin, L.; Han, X.; and You, J. 2020. Mutual information regularized identity-aware facial expressionrecognition in compressed video.

[Liu, Zhang, and Zhou2020] Liu, D.; Zhang, H.; and Zhou, P. 2020. Video-based facial expression recognition using graph convolutional networks.

[Mollahosseini, Hasani, and Mahoor2019] Mollahosseini, A.; Hasani, B.; and Mahoor, M. H. 2019. Affectnet: A database for facial expression, valence, and arousal computing in the wild. *IEEE Transactions on Affective Computing* 10(1):18–31.

[Rangulov and Fahim2020] Rangulov, D., and Fahim, M. 2020. Emotion recognition on large video dataset based on convolutional feature extractor and recurrent neural network.

[Zhang et al.2016] Zhang, K.; Zhang, Z.; Li, Z.; and Qiao, Y. 2016. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters* 23(10):1499–1503.