Facial expression analysis based on videos of presidential candidates

Andrew Whittum, Vivian Gunawan
Department of Computer Science
Boston University
Boston, MA 02215
awhittum@bu.edu, vgunawan@bu.edu

Abstract

The way we communicated is compromised of more than just speech, we convey our emotions through our facial expression and body language. Thus facial expression recognition have wide range of applications in human-machine interaction and other fields. In this paper, we categorize the facial expressions of 2019 and 2020 presidential candidate taken in various contexts on the campaign trail using deep learning techniques.

1 Introduction

There are 42 individual facial muscles in the face, contracting and relaxing different muscles allow us to create different expression. In the FER 2013 dataset, researchers categorize these faces into 7 different classes, happiness, neutral, sadness, anger, surprise, disgust, fear. Traditionally facial expression analysis models employs facial landmark detector and SVMs. However in the recent years, the usage of convolutional neural network have attained greater accuracy due to having a certain degree of translation, rotation and distortion invariance of an image.

Our facial expression analysis model is comprised of two parts: input pre-processing and a convolutional neural network. In the pre-processing step, we convert the videos into frames and extract the candidates faces. These facial frames are fed into the neural network to be categorized into 3 different classes, "Negative", "Neutral" and "Positive". In the sections below we will discuss further on the dataset given and our methodology.

2 Dataset

To train our facial expression analysis model, we were provided two sets of data. They consist of short video clips of presidential candidates from 2019 and 2020. The video clips are mostly sourced from speeches and news programs. Each video was titled with the name of the presidential candidate. We were also provided with labels of the facial expression of each candidate in question in a separate csv file. The 2019 dataset consist of 1320 videos with 11 different candidates, while the 2020 dataset consist of 1506 videos with only 6 of the candidates from 2019.

	Negative	Neutral	Positive	
2019	248	729	343	1320
2020	117	1033	356	1506
	365	1762	699	

Preprint. Under review.

2.1 Class imbalance

In the provided dataset we identified an issue of class imbalance; a disproportionate number of videos belonged to the "Neutral" class. In both the 2019 and 2020 datasets there were many fewer videos labeled "Positive" and the fewest labeled "Negative". To address this imbalance during training, we performed stratification when generating our folds during our 5-fold cross-validation. This splits the dataset randomly in a way that the same class distribution is maintained in each fold.

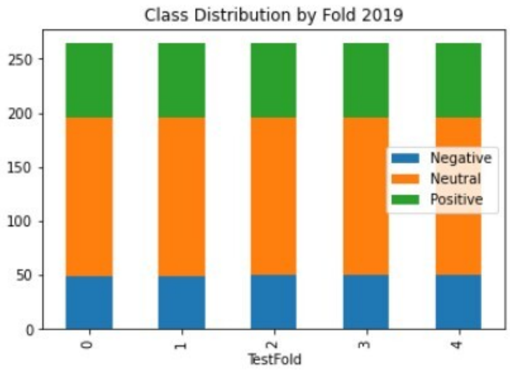


Figure 1: 5-fold stratified cross validation

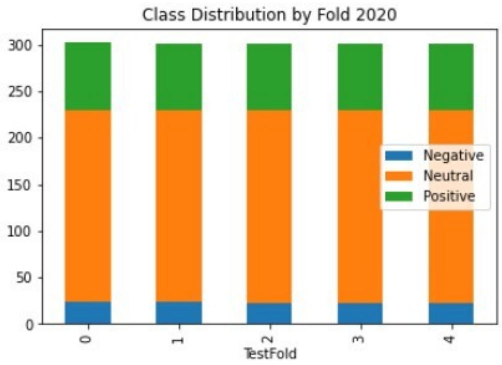


Figure 2: 5-fold stratified cross validation

3 Method

3.1 Pre-processing

Each of the videos is between approximately two and ten seconds long, with a frame rate of about 25 fps. The first step was extract each frame from the video, and then detect the face of the candidate in each frame. We used the openCV video capture method to extract frames from the videos and then applied the openCV SSD (single-shot detector) face detection method, which is based on the VGG architecture (Serengil 2020). We found that this detection method worked well even when the candidate was not directly facing the camera. A difficulty we encountered was when there were multiple people whose face's were clearly visible in the video at the same time. To deal with this we found reference images for each candidate where the candidates face was clearly visible, extracted the face from this image, and then we compared every face detected in the video against the relevant reference image. We tried three different methods of comparison, the dlib face verification method, the euclidean distance and the cosine distance, and we found that the cosine distance yielded the best results (Serengil 2020). There were still some faces that were incorrectly extracted by our method, but this was more effective than just using the first result returned by the face identifier. Finally, we resized the face into 299 by 299 pixels with RGB color channels and stores them as a .jpg file.

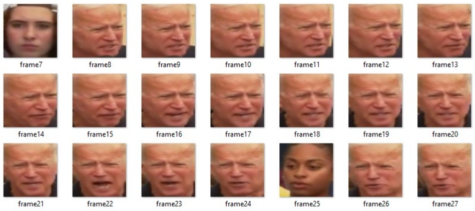


Figure 3: misidentified face

3.2 Model Selection

We began the model selection process by investigating various models for transfer learning. Transfer learning is a process by which a model is constructed and trained on a very large dataset and a large amount of compute power. Then the weights from the final model are saved and are provided open-source for anyone to use. We identified three models that had performed well in the ImageNet recognition challenge. These three networks that were InceptionV3, ResNet50, and VGG19. We ran these networks on our data with the pretrained "imagenet" weights. We changed the final output layer from 1000 nodes with a global average pooling layer and then an output layer with three output nodes. We ran our data through all three of these networks, but despite these being state of the art models, we received poor results despite extensive testing of different epochs and learning rates. All three networks classified nearly all of the images as "Neutral."

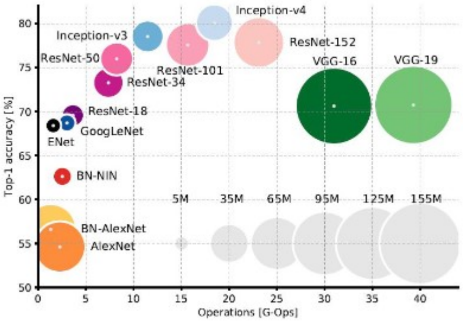
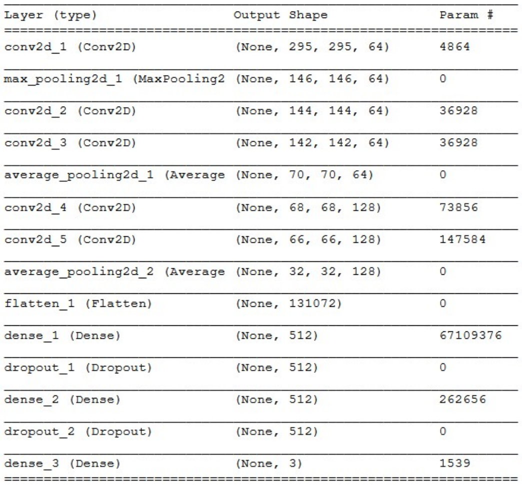


Figure 4: ImageNet Models (Milani, 2017)

Due to the low accuracy attained from the usage of transfer learning, we decided to use a much smaller network originally designed for classifying seven different emotions, altering only the final output layer to have three outputs (Serengil, 2018). This model used a softmax activation function for the output layer and a categorical crossentropy loss function. The input shape was the same as before (299, 299, 3). The full model structure is show below.



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 295, 295, 64)      4864
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 146, 146, 64)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 144, 144, 64)      36928
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 142, 142, 64)      36928
_________________________________________________________________
average_pooling2d_1 (Average (None, 70, 70, 64)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 68, 68, 128)       73856
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 66, 66, 128)       147584
_________________________________________________________________
average_pooling2d_2 (Average (None, 32, 32, 128)       0
_________________________________________________________________
flatten_1 (Flatten)          (None, 131072)            0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               67109376
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 512)               262656
_________________________________________________________________
dropout_2 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_3 (Dense)              (None, 3)                 1539
=================================================================
```

Figure 5: Model Structure

The input we passed into our model consisted of a random sample of 20 image frames from each video in the dataset. Each frame was was given the same label (positive, neutral, or negative) as the video from which it was extracted. The results show the classification of each individual frame from the test set.

We tried a variety of hyperparameters, we tested for learning rates of .001 and .0001 and different epoch sizes of 10, 50, and 100 epochs. Our final results, shown below, used a learning rate of .0001 and 100 epochs.

All of our testing was done on Boston University's Shared Computing Cluster using keras in Jupyter Notebooks. The time taken to train the networks for the final results was about four and a half hours.

4 Results

Below are the results of our facial expression recognition model. The confusion matrix is generated based on the average of the 5 folds, with each entry being the number of facial frames.

2019		True		
		Negative	Neutral	Positive
Predictions	Negative	378.8	401.6	134.8
	Neutral	478.8	2095.2	516
	Positive	136	405.2	689.2

2019	Accuracy	Precision	Recall	F1
Negative	0.780	0.414	0.381	0.397
Neutral	0.656	0.678	0.722	0.699
Positive	0.772	0.560	0.514	0.536
Weighted Average	0.709	0.598	0.604	0.600

2020		True		
		Negative	Neutral	Positive
Predictions	Negative	88.6	116.4	20
	Neutral	321.4	3393.4	785.8
	Positive	57.8	446.8	596.4

2020	Accuracy	Precision	Recall	F1
Negative	0.911	0.394	0.189	0.256
Neutral	0.713	0.754	0.858	0.803
Positive	0.775	0.542	0.425	0.477
Weighted Average	0.744	0.674	0.700	0.681

Overall, we achieved higher accuracy with the 2020 dataset compared to the 2019 dataset. The model achieved extremely high accuracy in identifying the "Negative" class in the 2020 dataset. Accuracy was above .7 for all categories in both years except for the Neutral category for 2019. Precision and recall were highest for the neutral category, followed by positive, and negative was the lowest for both years. Precision and recall were particularly low for the negative category in 2020, shown by an F1 score of 0.256.

5 Discussion

Overall, we can say that our most common mistake was incorrectly identifying positive or negative expressions as neutral. There were comparatively fewer errors where a positive face was classified as negative or vice versa.

We have identified several reasons why these misclassification errors may have occurred. The first is that the neutral category made up a disproportionate amount of the data, so this may have made the model too likely to categorize previously unseen data as neutral. The second is that even if a video was labeled as positive or negative, the candidate could have been making a different facial expression for some or most of the video. We weren't able to review all of the videos, but in a few videos we reviewed, we did find this to be the case. Finally, there is the problem of face misidentification where the face that was extracted from a frame was not that of the relevant candidate.

In future work, these problems could be remedied by using a more balanced dataset, or perhaps by oversampling or undersampling of certain categories. The video clips that are used could also be shorter and be clipped to just show the candidates face when he or she is making a certain facial expression, and remove the parts when the facial expression changes. Finally, a more accurate facial recognition algorithm may be used to avoid extracting any faces that aren't that of the candidate.

The other big problem that we faced was being unable to achieve good results by performing feature extraction using state of the art models. Despite extensive testing and research, we still aren't sure why we weren't able to achieve reasonable results with models such as InceptionV3. There is extensive literature surrounding using transfer learning for other image classification tasks, so it was disappointing that we were not able to achieve better results. Given more time we could maybe employ different form of transfer learning such as fine tuning, where we retrain more layers instead of just the final output layer.

References

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna. (2015). Rethinking the Inception Architecture for Computer Vision.

Christopher Pramerdorfer, Martin Kampel. (2016). Facial Expression Recognition using Convolutional Neural Networks: State of the Art.

I. J. Goodfellow, D. Erhan, P. L. Carrier, A. Courville, M. Mirza, B. Hamner, W. Cukierski, Y. Tang, D. Thaler, D.-H. Lee, Y. Zhou, C. Ramaiah, F. Feng, R. Li, X. Wang, D. Athanasakis, J. Shawe-Taylor, M. Milakov, J. Park, R. Ionescu, M. Popescu, C. Grozea, J. Bergstra, J. Xie, L. Romaszko, B. Xu, Z. Chuang, and Y. Bengio. Challenges in representation learning: A report on three machine learning contests. Neural Networks, 64:59–63, 2015. Special Issue on "Deep Learning of Representations"

Milani, Pedro, The Power of Inception: Tackling the Tiny ImageNet Challenge. 2017, Stanford University.

Serengil, Sefik, (2018, January 1). Facial Expression Recognition with Keras. Retrieved December 04, 2020 from https://sefiks.com/2018/01/01/facial-expression-recognition-with-keras/.

Serengil, Sefik, (2020, August 25). Deep Face Detection with OpenCV in Python. Retrieved December 04, 2020 from https://sefiks.com/2020/08/25/deep-face-detection-with-opencv-in-python/.

Serengil, Sefik, (2018, August 6). Deep Face Recognition with Keras. Retrieved December 04, 2020 from https://sefiks.com/2018/08/06/deep-face-recognition-with-keras/.

Pan, S.J. and Yang, Q., 2010. A survey on transfer learning. IEEE Transactions on knowledge and data engineering, 22(10), pp.1345–1359

Zhizhong Li, Derek Hoiem. (2017). Learning without Forgetting.