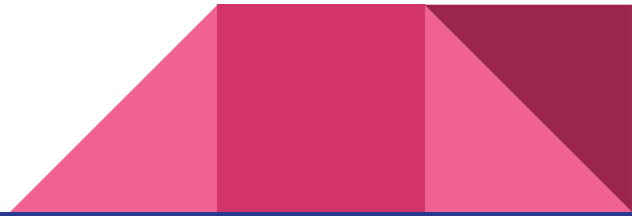
Facial Expression Recognition

Emmeline Tsen
Jacky Chow
Jerry Huang

Abstract

- Human face conveys a lot of information
- Facial detection has been a popular topic within AI
- Determine a person’s facial expression in an image
    - Help better understand what the person is feeling based on the image



Introduction

- Facial Expression Recognition (FER) has received a lot of attention in recent years within several fields
  - Medical
  - Security
  - Communications
  - Etc.
- Project consists of:
  - CNN Model & Model Tuning
  - Generative Adversarial Networks (GANs) to generate faces
  - CNN model to determine emotions from faces generated using GAN
  - TFX for Tensorflow Serving Model

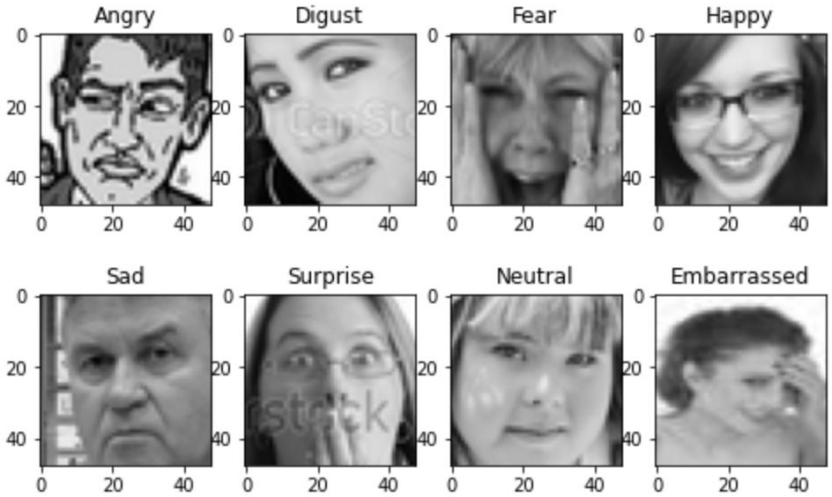
Related Work

- Mainstream methods
  - Traditional manual methods
    - Local Binary Patterns (LBP)
  - Shallow learning
    - SVM
    - Adaboost
  - Deep learning
    - CNN
    - RNN

Data

- Dataset
  - Fer2013 dataset
  - Amalgamation
    - Google image search: Embarrassed males and females
- Data Pre-processing
  - Image to Array
  - Converted images to black and white
  - Converted images to 48 * 48

```
show_img(data_valid)
```



Methods -- CNN

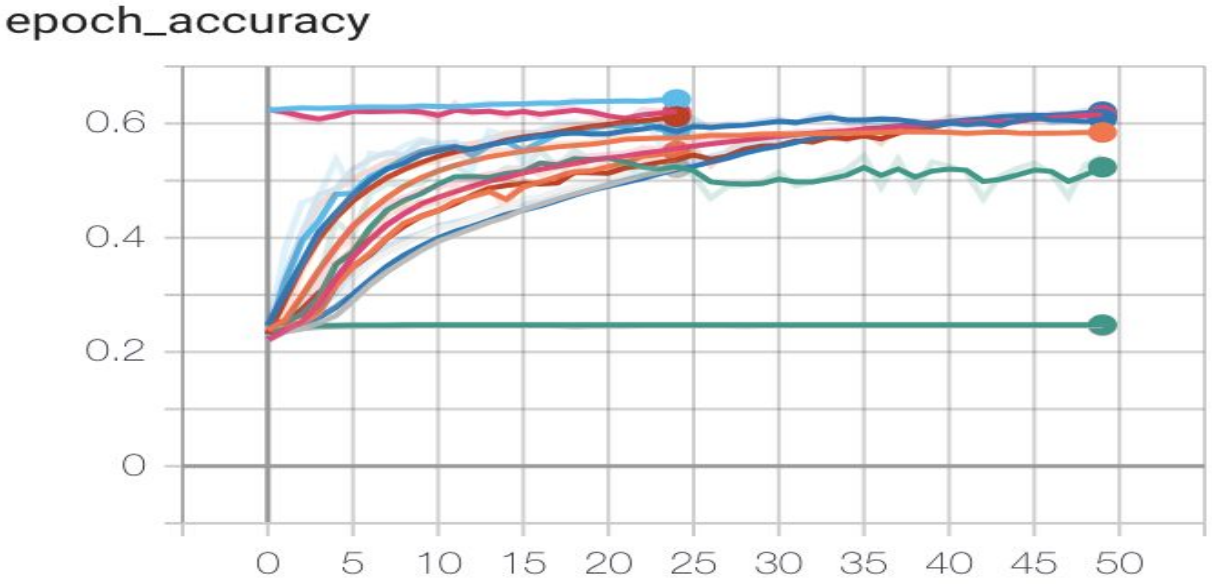
- Reshape the data into the fitting size (4 dimension array)
- Build the model
  - With data generator (rotation, shift, zoom)
  - Early stopping
- Hyperparameter Tuning
  - BatchNormalization
  - Dropout
  - Learning Rate
  - CNN Layers
  - Epochs
  - Optimizer

Colab - https://github.com/emmelinetsen/faceAI/blob/master/Main_CNN_FER.ipynb

Experiments

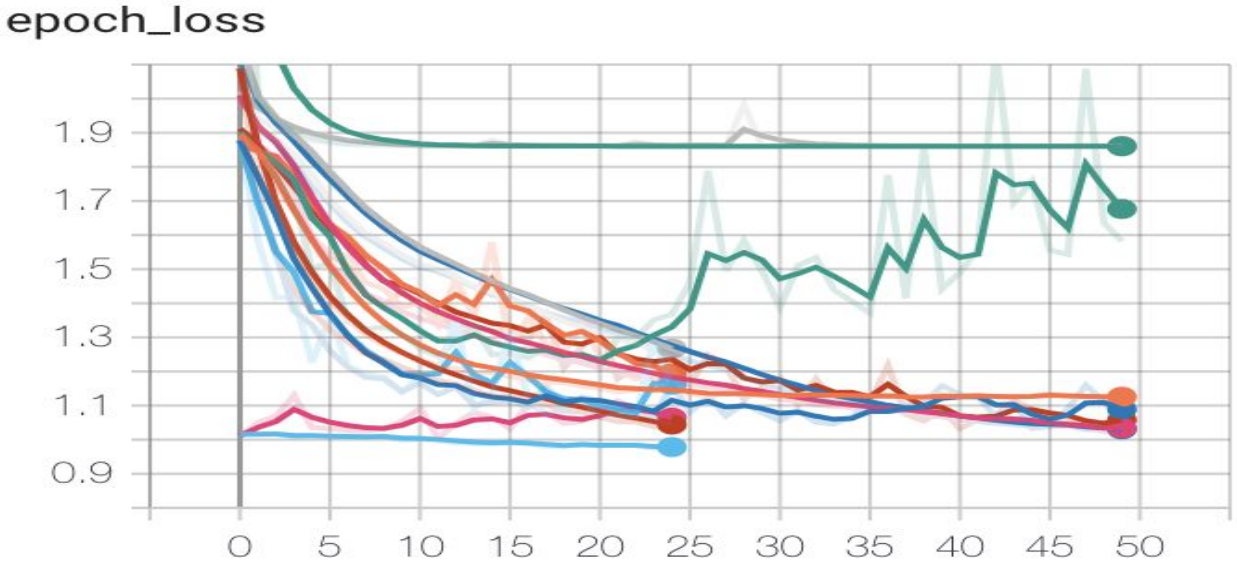
CNN Model	Drop out	Zoom	Epochs	Batch Normalize	Max Accuracy
1	0.25	yes	25	no	0.60
2	0.25	no	50	yes	0.61
3	0.25 (each layer)	no	50	yes	0.60
4	0.5	no	75	yes	0.64
5	0.75	no	50	yes	0.25

Accuracy



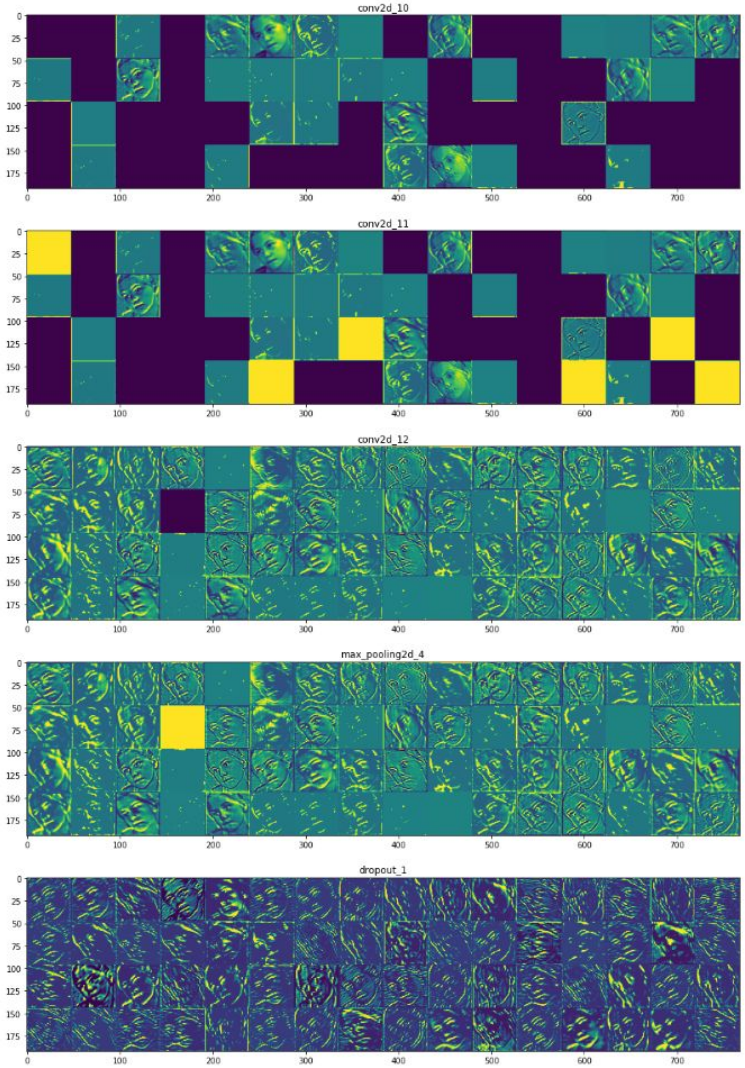
Tensorboard:
https://tensorboard.dev/experiment/GkvcQqC8SI2UVUlHvxPaDg/#scalars

Loss



Tensorboard:
https://tensorboard.dev/experiment/GkvcQqC8SI2UVUlHvxPaDg/#scalars

Visualizing Edges



ResNet

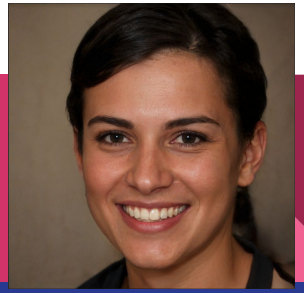
```
Train on 29141 samples, validate on 3643 samples
Epoch 1/20
29141/29141 [==============================] - 39s 1ms/step - loss: 0.4578 - accuracy: 0.8426 - val_loss: 1.9830 - val_accuracy: 0.5202
Epoch 2/20
29141/29141 [==============================] - 38s 1ms/step - loss: 0.1191 - accuracy: 0.9632 - val_loss: 2.1124 - val_accuracy: 0.5597
Epoch 3/20
29141/29141 [==============================] - 38s 1ms/step - loss: 0.0666 - accuracy: 0.9846 - val_loss: 2.2761 - val_accuracy: 0.5380
Epoch 4/20
29141/29141 [==============================] - 38s 1ms/step - loss: 0.0552 - accuracy: 0.9895 - val_loss: 2.3251 - val_accuracy: 0.5284
Epoch 5/20
29141/29141 [==============================] - 38s 1ms/step - loss: 0.0672 - accuracy: 0.9856 - val_loss: 2.6464 - val_accuracy: 0.4774
Epoch 6/20
29141/29141 [==============================] - 38s 1ms/step - loss: 0.1354 - accuracy: 0.9559 - val_loss: 2.6523 - val_accuracy: 0.4996
Epoch 7/20
29141/29141 [==============================] - 38s 1ms/step - loss: 0.1015 - accuracy: 0.9700 - val_loss: 2.4334 - val_accuracy: 0.5383
Epoch 8/20
```

OverFitting!!

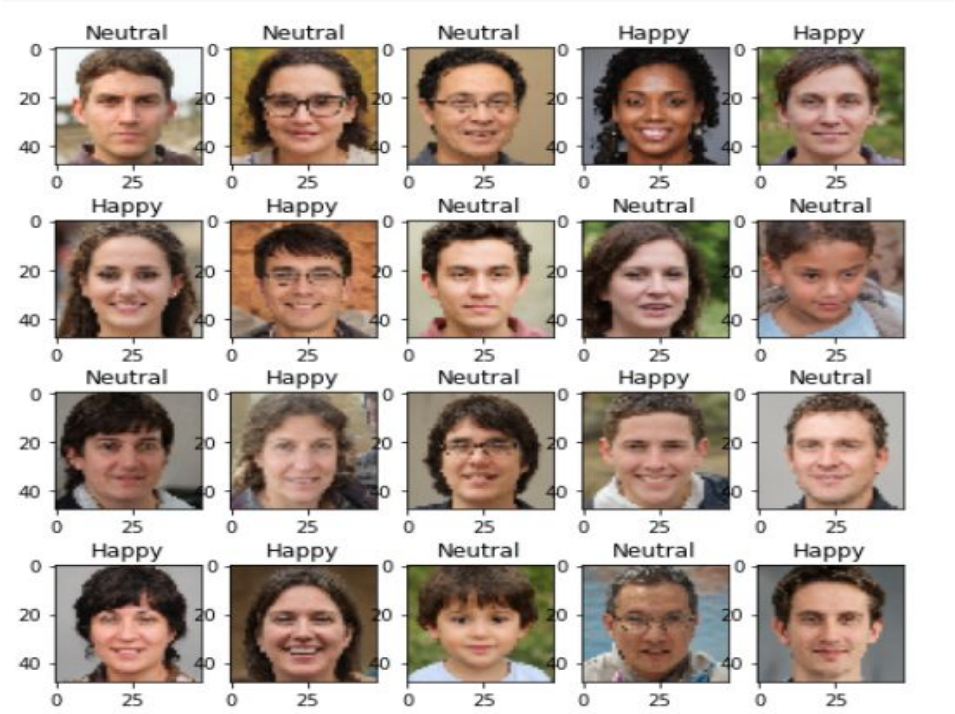
Generative Adversarial Networks (GANs)

- Generates new images that look very realistic
- Two parts to GANs:
  - Generative network - generates candidates
  - Discriminative network - evaluates to real/fake
- Using a generative and discriminative network to generate faces would take a long time to generate several faces - Colab
- Used pretrained model and Nvidia's SyleGAN2 to generate very realistic faces - Colab





CNN Model on Faces using GANs




```
[ ]  pred_df['label'].value_counts()

Happy      290
Neutral    209
Angry        1
Name: label, dtype: int64
```

Hosting on TFX Serving Model

```
[ ]  1 %%bash --bg
     2 nohup tensorflow_model_server \
     3   --rest_api_port=8502 \
     4   --model_name=cnn_model_2 \
     5   --model_base_path="${MODEL_DIR}" >server.log 2>&1
```

```
Starting job # 2 in a separate thread.
```

Make Predictions using TFX Serving Model

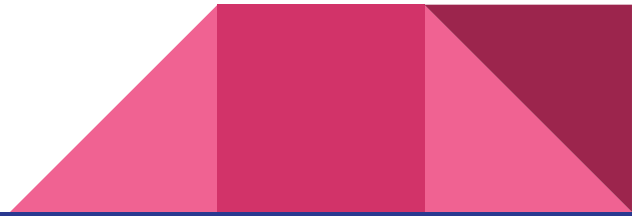
```
[ ]  1 import requests
     2 headers = {"content-type": "application/json"}
     3 json_response = requests.post('http://localhost:8502/v1/models/cnn_model_2:predict', data=data, headers=headers)
     4 predictions = json.loads(json_response.text)['predictions']
     5 output = predictions[0]
```

```
[ ]  1 from keras.preprocessing import image
     2
     3 classes = ['angry','disgust','fear','happy','sad','surprise','neutral','embarrassed']
     4
     5 img = image.load_img("/content/drive/My Drive/CMPE 258 - Deep Learning/258_final_project/data/stylegan2/image900.png",target_size=(48,48))
     6 img = np.asarray(img)
     7 plt.imshow(img)
     8 img = np.expand_dims(img, axis=0)
     9
    10 print(output)
    11 classes[np.argmax(output)]
```

```
[0.0458349213, 3.94001631e-09, 0.012209137, 0.00210829754, 0.109127142, 2.6029853e-05, 0.830694497, 1.98055497e-14]
'neutral'
```

Conclusion

- A lot to learn about a human’s face
- CNN was able to detect facial expression
  - Best accuracy: 64.53%
- Utilized GANs to help with testing
- TFX serving model to host our best model



References

[1]https://arxiv.org/pdf/1902.01019.pdf
[2]https://www.researchgate.net/publication/261498182_Face_expression_recognition_A_brief_overview_of_the_last_decade
[3]https://www-sciencedirect-com.libaccess.sjlibrary.org/science/article/abs/pii/S0030402616302807
[4]https://www-sciencedirect-com.libaccess.sjlibrary.org/science/article/pii/S0925231217313644

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

Thank you