Face Tracking System with Emotion Detection and Face Matching

Yaxi Lei
Brown University
yaxi_lei1@brown.edu

Qiran Gong
Brown University
qiran_gong@brown.edu

Da Huo
Brown University
da_huo@brown.edu

Abstract

Efficient face tracking system has became an essential application in modern society which has many use cases including security surveillance, identity checking, and many other purposes. In this project, we propose to implement a real-time face tracking system that includes emotion detection and face matching features. Both features can be used under many situations. Our system achieves efficient real-time multi-face detection with emotion labels.

1. Introduction

Face detection is widely used in many modern applications. For this project, we aim to explore some of the state-of-the-art face detection techniques while incorporating our own models to build a real-time face tracking system. The systems consists of 3 main parts – face detection, face matching, and emotion detection. For the face detection part, our original plan was to implement the Multi-task Cascaded Convolutional Networks [1]. However, while implementing MTCNN[1], our model suffers a poor performance and we realized that it is very hard for us to outperform the existing MTCNN[1] libraries. Therefore, we decides to use existing MTCNN libraries for the face detection part and build new features upon face detection. For the face matching part of this project, the goal is to detect whether a face captured by MTCNN[1] exists in our database. We built a deep convolutional neural network which takes two images of face as input and outputs whether these two images are the face of the same person. Section 3.1 describes this part in detail. For the emotion detection part of this project, the goal is to detect the emotion of the captured face. We built a Convolutional Neural Network that takes a captured face image as input and classifies it into one of 7 emotions Angry, Disgust, Fear, Happy, Sad, Surprise, Neutral. We used the FER-2013 dataset [2] to train our model and achieved a decent result. section 3.2 describes this part in detail.

2. Related Work

Similar to siamese networks[3], in face matching part, we take two face images as model's input. Based on siamese networks, our model can combine feature extraction with similarity measurements and learn parameters, which forms an end-to-end framework. For the feature extraction part, many different deep network architecture have been used to great success in computer vision community, like VGG16[4], ResNet-50[5], etc. Both VGG16 and ResNet-50 introduce a concept: block. They treat multiple convolution layers as a block and stack blocks as their neural networks. Additionally, ResNet-50 has a key concept: residual. Residual is actually the output from last block and ResNet combines outputs from different blocks to avoid gradient vanishing problem. However, networks mentioned above are complicated and all trained on large scale dataset, for example: ImageNet[6]. Given that our dataset scale is relative small, we thus take a relative simple network, which is similar to LeNet[7]. Different from LeNet, we additionally add one convolution layer and remove max-pooling between convolution layers except the last one. The reason why we change LeNet like that is we have used MTCNN[1] to remove background information and we hope to keep as much information as we can extracted from people's faces.

3. Method

3.1. Face Matching

The goal of facial Matching is to retrieve whether new faces can find matches in the database. Therefore, the problem can be summarized as calculating the similarity between pictures. Before model training, we first use MTCNN[1] to extract people's faces from pictures. This step of data preprocessing is to filter some useless background information. After face extraction, we stack all faces together and generate random pairs of index. Labels are binary, meaning similar or not for each pair of faces. To overcome data unbalancing problem, we take down sampling method in machine learning.

Then, our method uses pair-wise training, which can be divided into two parts: feature extraction and similarity

measurement. For the first part, we uses 3-layer CNNs and 2 Dense layers and add a max-pooling layer to filter some redundant information. Because we hope to keep original information from faces, we only add one pooling layer. To avoid overfitting problems, drop out layers are added between dense layers. For the second part, we simply concatenate features extracted from above-mentioned CNNs architecture, and pass it into dense layers to do the binary classification. The whole model structure is shown in Figure 1.

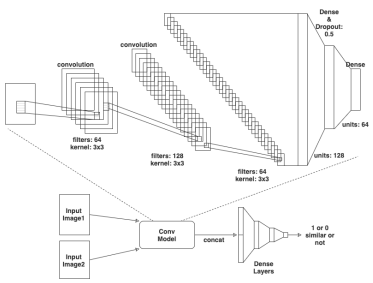

Figure 1. facial matching model architecture

3.2. Emotion Detection

For this part, we built a Convolutional Neural Network to classify the captured face image into one of 7 emotions Angry, Disgust, Fear, Happy, Sad, Surprise, Neutral and we trained our network using the Facial Expression Recognition 2013 (FER-2013) Dataset [2] which contains 35887 $48 * 48$ size images. Among those 37887 images, we used 28709 images as our training data, 3589 images as validation data, and 3589 images as testing data. The training, validation and testing data ratio is roughly about $8 : 1 : 1$.

As for our network architecture, initially we started with a fairly complex model structure which contains 15 layers. The architecture first have 12 convolutional layers and a max-pooling layer for every three convolutional layers. We used relu as our activation for all the convolutional layers. We also have 3 dense layers with relu activation after the convolutional layers. Figure 2 shows the detailed model architecture of our initial model. However, this model does not perform as good as we expected it to be. The model stuck on about 60% accuracy and most importantly, the model will predict "Angry" for the most of the time. The reason for this could be that the model is too complex given the limited data we have and it overfits too much on the training dataset. Thus, we switched to a simpler model structure.

Our new model structure consists a total of 7 layers which includes 4 convolutional layers and 3 dense layers. We used relu activation for all convolutional and dense layers and max-pooling in between of every convolutional layers. Figure 3 shows a detailed model structure of our new model.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 48, 48, 32)	320
conv2d_1 (Conv2D)	(None, 48, 48, 32)	9248
conv2d_2 (Conv2D)	(None, 48, 48, 32)	9248
max_pooling2d (MaxPooling2D)	(None, 24, 24, 32)	0
dropout (Dropout)	(None, 24, 24, 32)	0
conv2d_3 (Conv2D)	(None, 24, 24, 64)	18496
conv2d_4 (Conv2D)	(None, 24, 24, 64)	36928
conv2d_5 (Conv2D)	(None, 24, 24, 64)	36928
max_pooling2d_1 (MaxPooling2	(None, 12, 12, 64)	0
dropout_1 (Dropout)	(None, 12, 12, 64)	0
conv2d_6 (Conv2D)	(None, 12, 12, 128)	73856
conv2d_7 (Conv2D)	(None, 12, 12, 128)	147584
conv2d_8 (Conv2D)	(None, 12, 12, 128)	147584
max_pooling2d_2 (MaxPooling2	(None, 6, 6, 128)	0
dropout_2 (Dropout)	(None, 6, 6, 128)	0
conv2d_9 (Conv2D)	(None, 6, 6, 256)	295168
conv2d_10 (Conv2D)	(None, 6, 6, 256)	590080
conv2d_11 (Conv2D)	(None, 6, 6, 256)	590080
max_pooling2d_3 (MaxPooling2	(None, 3, 3, 256)	0
flatten (Flatten)	(None, 2304)	0
dense (Dense)	(None, 128)	295040
dropout_3 (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 64)	8256
dropout_4 (Dropout)	(None, 64)	0
dense_2 (Dense)	(None, 7)	455

Total params: 2,259,271
Trainable params: 2,259,271
Non-trainable params: 0

Figure 2. Initial Emotion Detection Model Architecture

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 48, 48, 32)	832
max_pooling2d (MaxPooling2D)	(None, 24, 24, 32)	0
dropout (Dropout)	(None, 24, 24, 32)	0
conv2d_1 (Conv2D)	(None, 24, 24, 64)	51264
max_pooling2d_1 (MaxPooling2	(None, 12, 12, 64)	0
dropout_1 (Dropout)	(None, 12, 12, 64)	0
conv2d_2 (Conv2D)	(None, 12, 12, 128)	204928
max_pooling2d_2 (MaxPooling2	(None, 6, 6, 128)	0
dropout_2 (Dropout)	(None, 6, 6, 128)	0
conv2d_3 (Conv2D)	(None, 6, 6, 256)	819456
max_pooling2d_3 (MaxPooling2	(None, 3, 3, 256)	0
dropout_3 (Dropout)	(None, 3, 3, 256)	0
flatten (Flatten)	(None, 2304)	0
dense (Dense)	(None, 1024)	2360320
dropout_4 (Dropout)	(None, 1024)	0
dense_1 (Dense)	(None, 128)	131200
dropout_5 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 7)	903

Total params: 3,568,903
Trainable params: 3,568,903
Non-trainable params: 0

Figure 3. New Emotion Detection Model Architecture

Initially we trained this model with a dense layer of size 128 and the model gives poor performance every time. After extensive testing and research, we found out that it could be the dense layer size is too small to catch all the features captured by convolutional layers. After we increase the dense layer size to 1024, the model starts to give decent results.

Although our new model gives a decent result, the model suffers from limited training data. To deal with this situation, we augmented our training data during the training process. We augumented the training data with the following parameters

```
datagen = ImageDataGenerator(
rescale=1./255,
width_shift_range=0.1,
height_shift_range=0.1,
zoom_range=0.1
)
```

4. Results

4.1. Face Matching

There are four concepts in statistics: true positive (TP), true negative (TN), false positive (FP) and false negative (FN). A true positive is an outcome where the model correctly predicts the positive class. Similarly, a true negative is an outcome where the model correctly predicts the negative class. A false positive is an outcome where the model incorrectly predicts the positive class. To better define f1 score, we first introduce another two terms: precision and recall. Precision measures model's performance on true samples, which is defined as $\frac{TP}{TP+FP}$. Recall measures model's capability of finding correct predictions on true samples, which can be defined as $\frac{TP}{TP+FN}$.

Considering that only using accuracy might cause bias on model's performance, thus for the face matching part, we choose accuracy and f1 score as evaluation metric. Accuracy is used to evaluate the closeness between prediction and labels, which is defined as $\frac{TP+TN}{TP+TN+FP+FN}$. F1 score is defined as $2 \cdot \frac{precision \cdot recall}{precision+recall}$ Our dataset is divided into training set, validation set and testing set by 8:1:1. After 100 training epochs, the accuracy and f1 score of our model can reach 0.9559 and 0.9549 respectively.

4.2. Emotion Detection

To evaluate our emotion detection model, we used the model accuracy which can be calculated as

$$Accuracy = \frac{TP}{TP + FN} \tag{1}$$

where TP is True Positive and FN is false negative.

For our initial model which has a more complex structure, the model's validation accuracy and training accuracy can be view in figure 4

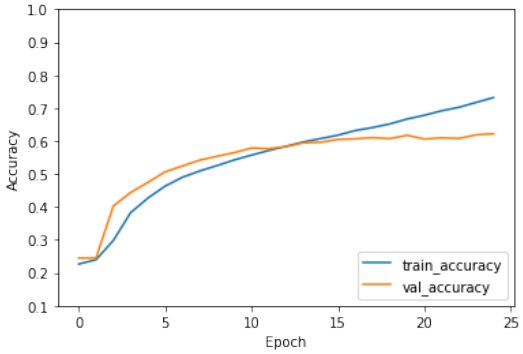

Figure 4. Initial Emotion Detection Model Accuracy

As we can see from the figure, the model starts to overfit around 60% accuracy. We tried to tune the model with varies hyperparameters and the best we can get is to start to overfit around 60% accuracy. For this model, the model's can reach more than 90% training accuracy with its complex architecture. However, it performs poorly on the testing dataset and validation dataset as it tends to always predict "Angry".

For our second model which is the simpler one, the model's validation accuracy and training accuracy can be view in figure 5

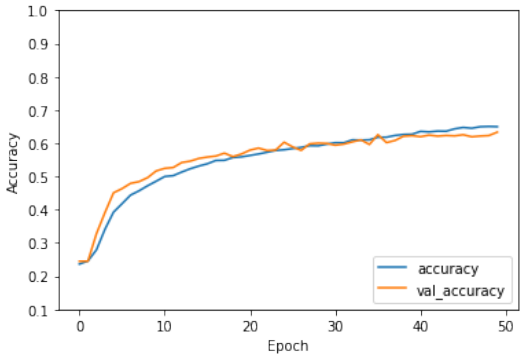

Figure 5. Second Emotion Detection Model Accuracy

As we can see from the figure, both the model's training accuracy and validation accuracy comes to a convergence at around 63% accuracy. The reason for this could be that given it's simple model structure, this is the best that this architecture can learn unlike our first model which can overfit to a high training accuracy. In addition, unlike the first model, this model gives a more reasonable prediction as it does not rely on predicting "Angry". It gives a more evenly distributed prediction and is particularly good at Happy, Surprise, Sad, and Angry emotions. Thus, we used this

model in our final implementation of the face tracking system.

4.3. Integrated System

As a final result, our system is capable of capturing multiple faces in a given video frame, give each face a emotion label, and match the face with a potential face database in real time. When running the program, an example of the video window is shown as figure 6

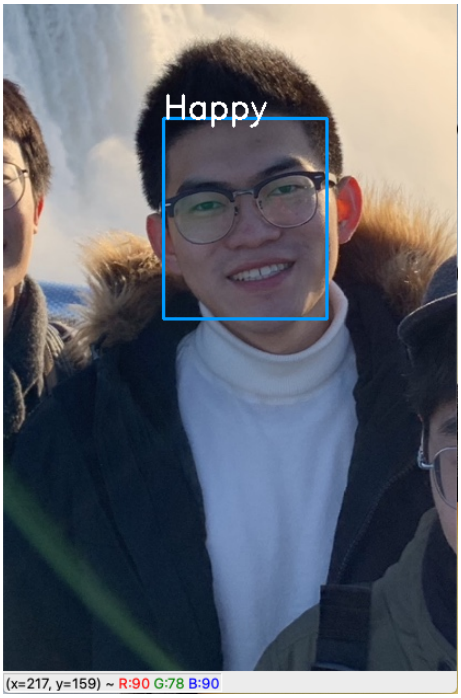

Figure 6. Program Example

5. Instructions on Running Our System

To run our system for a demo,

1. Clone the project repo

```
git clone https://github.com/
MangoManGeek/CV_final_project
```

2. install required packages from requirements.txt
3. run csci1430-final.py

```
python3 csci1430_final.py
```

6. Conclusion

In this project, our group was able to build a real-time face tracking system with emotion detection and face matching features using various Computer Vision and Deep learning techniques. We believe this system can be useful in many modern applications. However, processing image frames is a computation heavy task. We have to slow the frame rate in our current implementation in order to achieve real-time processing. Thus, our future work is to come up with a more efficient implementation or even model structure to provide a high frame per second rate.

References

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

[2] Ian Goodfellow, Dumitru Erhan, Pierre-Luc Carrier, Aaron Courville, Mehdi Mirza, Ben Hamner, Will Cukierski, Yichuan Tang, David Thaler, Dong-Hyun Lee, Yingbo Zhou, Chetan Ramaiah, Fangxiang Feng, Ruifan Li, Xiaojie Wang, Dimitris Athanasakis, John Shawe-Taylor, Maxim Milakov, John Park, Radu Ionescu, Marius Popescu, Cristian Grozea, James Bergstra, Jingjing Xie, Lukasz Romaszko, Bing Xu, Zhang Chuang, and Yoshua Bengio. Challenges in representation learning: A report on three machine learning contests, 2013.

[3] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546. IEEE, 2005.

[4] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[7] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Appendices

7. Team Member Contributions

Yaxi Lei: MTCNN model build and research; Overall system integration with all features; Presentation

Qiran Gong: Face Matching model build and research; Writing report

Da Huo: Emotion Detection model build and research; Writing report