Individual Final Report
Matt Wilchek

Overview

- In this project, we tried to apply deep learning concepts to train a model in classifying emotion from human behavior.
- I originally had the idea of exploring emotional recognition after learning about Microsoft's Cognitive Services API that is built in Azure to perform emotional recognition from pictures. I thought the concept would be interesting to see how possible it would be to apply concepts learned in class to this type of problem.
- After researching about possible datasets to begin working from, I was able to narrow my search down to a dataset by Massachusetts Institute of Technology and Carnegie Mellon University. I decided to go with the one from CMU due to how the original structure of the images and labels were organized.
- My partner and I then communicated an outline to plan next steps to begin. He ultimately provided the foundation for the image processing we used and the initial model architecture that I later refined. We worked together improving our model and summarizing our results with the presentation and paper.

Individual Work

- In the beginning of our project, we both began to process the dataset in a similar procedure that we used for the data for Exam 1; which was very different between each other. I noticed right away that some of the images didn't have labels tied to them, so I manually began looking at images and applying labels. I also noticed some of the labels for the images were not correct and adjusted them. This was a problem I didn't think was possible to automate due to the requirement of deciding what the best label would be for the picture.
- After re-classifying each image and bucketing each in their respective emotion labels, each having their own emotion directories, I developed a script that would process them and allow randomized augmentation to increase the dataset.
- At this point, my partner and I came together how we were going about preparing the data and I decided to just follow the process of data pre-processing that my partner completed. We both added the missing labels that existed in the dataset, but didn't change any of the labels I thought were misclassified. I also believed that the images in the beginning of the sequence should be left out due to the non-emotional states people had. The sequences began with people from no emotion to an emotional state. Including these images I believed could impact the overall learning for any model.
- My partner and I agreed about these images being included for the sequence that could impact the model, however we disagreed about the process of implementing this exclusion. From my experience of manually reviewing each image and sequence I believed that if we gave a percentage of images to cut from in the beginning we would lose or keep worthless images. The number of pictures varies for each sequence and some subjects began exhibiting emotional facial features earlier than others, therefore I believed we should divide and conquer a manual review of the images to decide which ones to keep or lose. In the end, I sided with my partner's plan to just cut the number based on a percentage and forgo the manual review process to save time.
- After my partner finished refining the data pre-processing with a Keras Image generator, configuration file, and developed a rough draft version for a Convolutional Neural Network model, I performed research about how it could be improved.
- From my googling, I stumbled upon an interesting article from Stanford University that described how important other activation functions can be for detecting smaller features within facial changes, such as 'elu' and 'tanh'.

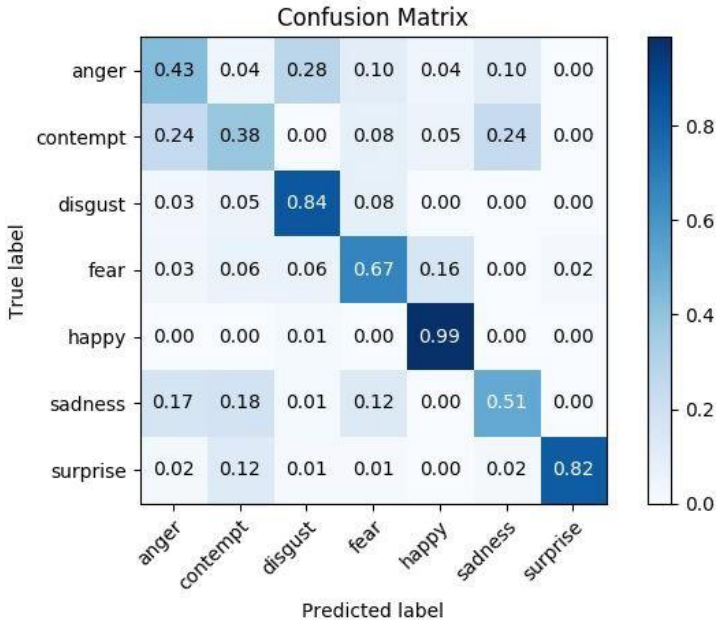
- After learning more about the importance of the different parameters needed for Convolutional Neural Networks in facial recognition, I decided to develop a separate tuning script that could run hundreds of possible models with different configurations to gain a better idea how the model could be improved using a special library I used in Exam 1 called ‘Talos’. I created a dictionary of parameters that included different learning rates, number of epochs, dropout rates, number of neurons, optimizers, and of course different activation functions. I initially placed my dictionary of possible configurations universally throughout the model my partner made.
- I soon realized this was a mistake since we had five layers of architecture; one input, one output, and three hidden. I then revised my tuning script so that the parameters could be different for each respective layer; this is what ultimately allowed our model to greatly improve our results. However, this hyper parameter tuning process had two huge downsides.
    - One, was that each tuning execution would ultimately execute from 2000 to 4000 different Convolutional Neural Networks with different parameters. This in turn, often made the execution of this script take from 10 to 12 hours to complete. Therefore, carefully timing the number of models to perform, number of epochs, and batch sizes were absolutely critical for the script to finish successfully and within time constraints to review.
    - Second, was that as I refined our final model that we present in our findings often times my partner would implement new data-preprocessing concepts that would be fed into each tuned model that I would report to him on. This resulted in me adjusting my tuning to start all over again on the different pre-processed dataset. The first tuning restart occurred when my partner decided to adjust the Keras Image generator to include small augmentations. The second time was after my partner decided to implement the MTCNN Facial Detection to the original images. In the end, due to the complexity increase of the original dataset each epoch for each model also became more complex often times increasing in execution time. Near the end of this project most models would complete between 15-30seconds per epoch.
- After my partner and I refined the model to the latest version, I then also collected emotional images similar to that of our original dataset from family and friends, applied our latest pre-processing MTCNN facial detection on them, then tested the images against our model using a separate predict script that I developed. The final accuracy was close to 33% against the actual classifications which we discussed in the presentation and report.

My Individual Scripts

The list below delineates the scripts I wrote or supported my partner in developing for the project.

- Preprocessing: This script was supposed to be used to create datasets and metadata for training our models.
    - data_split.py
- Modeling: These scripts represent the progression of our modeling efforts.
    - training_model_tuning.py
    - training_model_building_best_model.py
    - training_model_building_best_model_just_faces.py
- Evaluation: These scripts were used to evaluate the quality of our trained models.
    - model_predict.py

Results

- The example/baseline model had 21% accuracy that my partner first developed. The tuned model had 48% accuracy. The tuned model using face detection had 68% accuracy.
- The model had poor accuracy with the anger and contempt emotions. Also it had poor accuracy with the latest data collected from friends and family especially between surprise and fear. This wasn't surprising to me due to the previous noted issues I stated in regards to label miss-classification from CMU and including or dropping records that have impacted the training for these classifications. I also recall there not being many contempt records to begin with from the original dataset and it may have been better to just remove that emotion from the project entirely.
- The results identifying happiness and surprise were not as bad. Common traits between these two behaviors would include a smiling face versus an open mouth for each respective emotion.
- The matrix below shows our accuracy across classes of emotion for the best model we developed.
    - 

Conclusion

- Based on the initial dataset collected by Carnegie Mellon from 2000 to 2010, I think their initial project planning was a solid effort and very cutting edge to try and do in that timeframe. Due to their data collection, other organizations both academic and private are looking at this research as a baseline to develop more advance tools for emotional recognition.
- In their initial research paper, it was suggested that in order to have a strong model there dataset would need to be significantly bigger. After working with just this dataset from them to train a model, I would have to agree to that based on the initial evidence. I also am now speculating how accurate Microsoft's Cognitive Services API is with their Azure cloud most likely supporting an immensely large dataset to create a strong model.
- Upon manually reviewing almost every single image in CMU's dataset, I've realized that people have many different ways to express the same type of emotion we can think of to be common. For example, some subjects would be angry with a frown or grinned teeth, while others, mostly men from research, would show more neutral mouth shapes, but more intense stares and slightly puffed cheeks to show their frustration. To see such small differences in facial features made me realize that this task alone would be more difficult than I initially thought about.
- If more time was available to work on this project, especially not during a major holiday, I believe we could have made a better model with a better dataset backing it. Once we discovered the power of MTCNN facial detection, we realized that the training was definitely picking up more important facial features. The 'tanh' activation functions were very important to obtain the different curves around a mouth or forehead.
- More data could have easily been collected or scraped from google or we could have asked for more data from our friends, classmates, and family. We could have also tested some of the pre-trained models PyTorch offers such as ResNet.
- After witnessing the model tested on just a few images from people we knew it was fun to see some of the classifications actually working; which proves that the concept is still there and emotional recognition is entirely possible. I'm skeptical if the sequence of images was important, but I do think having images with more subtle facial features for certain emotions would be important.

Code from Other Sources

- In total, I custom developed three scripts for processing, modeling, and evaluation. I also supported the development for two modeling scripts my partner first started as well.
- Two of the scripts I made, the structure of the code was first templated out by either online documentation or code from our Exam 1 template. In order to create my large tuning script I read lots of documentation from Talos, Keras and/or StackOverflow which would be an estimated about ⅔. The other ⅓ of code was custom that I made to either run the appropriate functions in order or join/clean data appropriately.
- Other sites listed in the References provided inspiration. For example, I gained inspiration from a Stanford article about including 'tanh' as an activation function for tuning. This led us to refining our model which greatly improved its accuracy. I also did not use GitHub very much other than code I had in my own profile for development.

References

- Our Dataset from Carnegie Mellon
  - http://www.consortium.ri.cmu.edu/data/ck/CK+/
- Carnegie Mellon's first initial publication on modeling results using CK+
  - http://www.consortium.ri.cmu.edu/data/ck/CK+/CVPR2010_CK.pdf
- Medium Post from Nishank Sharma about using CNN in Keras for facial recognition
  - https://medium.com/themlblog/how-to-do-facial-emotion-recognition-using-a-cnn-b7bbae79cd8f
- Keras Modeling Documentation
  - https://keras.io/models/sequential/
- Keras Documentation on ImageDataGenerator
  - https://keras.io/preprocessing/image/
- Post from Adrian Rosebrock Describing ImageDataGenerator
  - https://www.pyimagesearch.com/2019/07/08/keras-imagedatagenerator-and-data-augmentation/
- Post about Facial Recognition
  - https://machinelearningmastery.com/how-to-perform-face-recognition-with-vggface2-convolutional-neural-network-in-keras/
- Official Talos Documentation
  - https://autonomio.github.io/docs_talos/#optimization-strategies
- Tutorial Using Talos and Keras
  - https://towardsdatascience.com/hyperparameter-optimization-with-keras-b82e6364ca53
- Stanford University article on important activation functions for facial recognition
  - http://cs231n.github.io/neural-networks-1/#commonly-used-activation-functions