Facial Recognition on Google Photos

Umesh Kumar Gattem
Luddy School of Informatics, Computing and Engineering, Indiana University Bloomington

Motivation

- The goal is to create a complete pipeline for the Google Photos System that can perform person detection and recognition.
- This will involve utilizing the YOLO (You Only Look Once) object detection model, which can identify objects from the COCO dataset. However, in this case, the YOLO model will be repurposed to identify faces of individuals.
- The system will be able to analyze both individual and group photos, identify all faces in the image, create boundary boxes around each person, and accurately label each individual with their corresponding name.

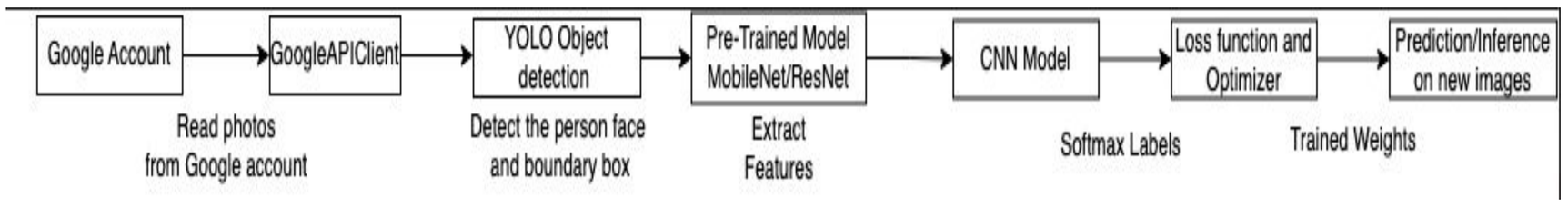
Challenges

- It is a challenging task to acquire a suitable dataset of our own images, especially when attempting to obtain individual images.
- In my experience, I had to manually sort through the photos on the Google Photos website by selecting images of specific individuals and filtering out any solo photos.
- Once this process was completed, I created an album for each person containing only their selected photos.
- Despite the difficulties in acquiring a suitable dataset, I was able to gather approximately 500 photos for a single individual.
- Training a model from scratch with a small dataset can be very challenging. Therefore, I opted to use pre-trained models such as Mobilenet for feature extraction instead.

Dataset

- I have selected 5 categories from the Google Photos Albums, comprising of approximately 400-500 images each, after manually sorting and filtering through individual photos.
- To read the albums from Google photos, I utilized GoogleAPIClient.
- However, to use Google APIs, we require a token file or a secret json file, which can be generated by creating credentials on the Google Cloud Console website.
- With this credentials json file, we can connect to the Google Cloud Console and access any of the Google APIs, including photos, drive, and Google mail.
- Once the credentials file is obtained, we can access any album from Google Photos, download and save the photos to local.

Model



Approach

- The process begins with creating individual photo albums through a manual and filtering process.
- These albums are then fed into the YOLO object detection model to detect the faces of individuals and generate boundary boxes around them.
- Next, the images are cropped using these boundary boxes and sent through pre-trained models such as Mobile Net, ResNet, and Inception for feature extraction.
- During feature extraction, the initial base model layer weights are kept untrained for a few epochs before fine-tuning the last few layers.
- Following that, custom layers like Global Pooling and Dense Layers are added to match the output labels dimensionality or number of classes.
- Softmax is then applied to the final layer, and the model's loss is calculated using a proper loss function such as Categorical Cross-Entropy and an optimizer such as Adam.
- Once the model is trained, it can be saved and utilized for prediction.

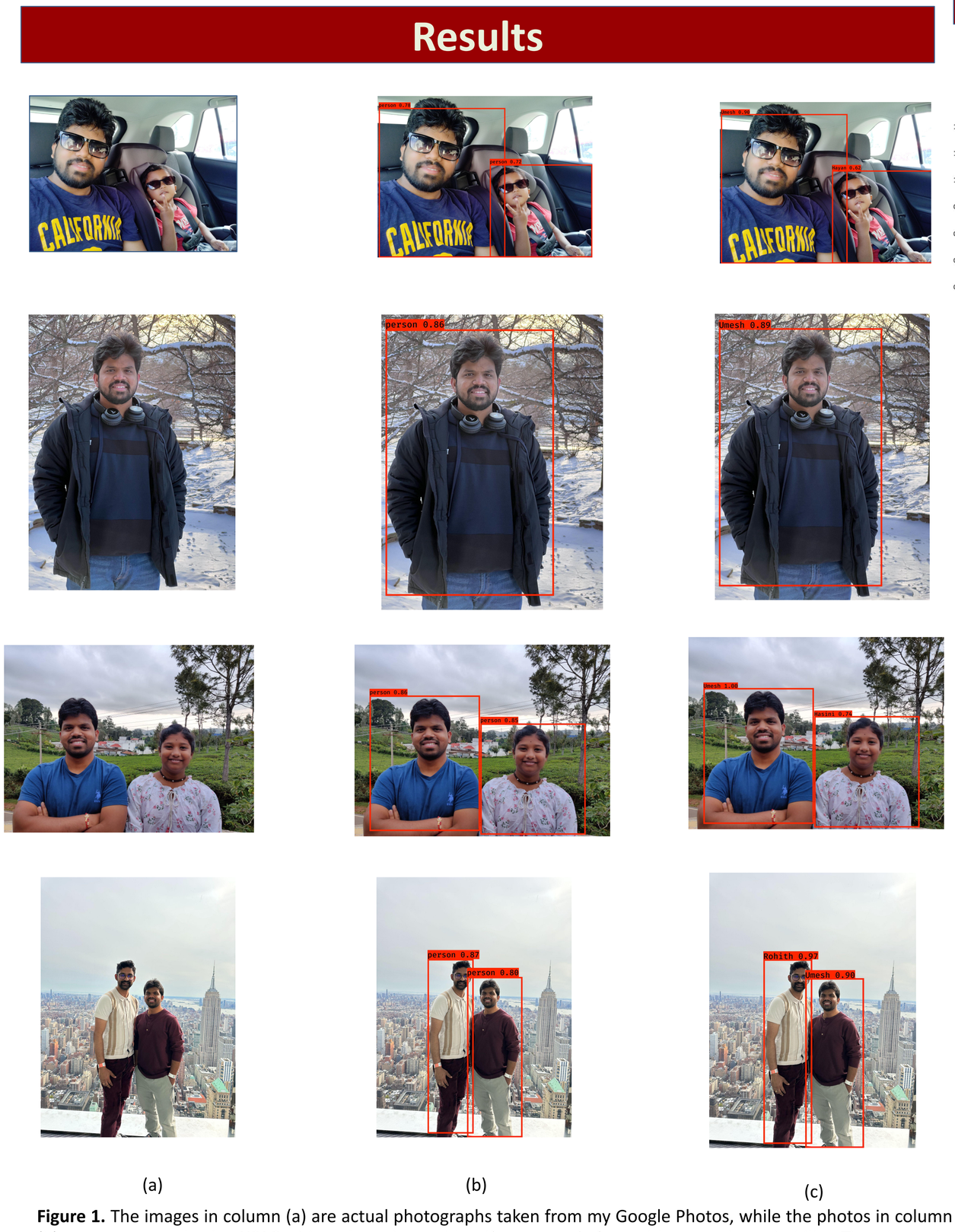
Experiments

- For this experiment, I utilized several base models, including MobileNet, ResNet, Inception V3, and Inception-ResNet.
- I conducted object detection using the YOLO model exclusively for all the base models.
- Additionally, I maintained uniformity across all the models by using the same hyperparameters, such as the number of fine-tuning layers, epochs, batch size, learning rate, loss function, and optimizer.
- Specifically, I used the first 120 layers as feature extraction layers and fine-tuned the remaining layers.
- I split the data into training, validation, and test sets, with a ratio of 60%, 20%, and 20%, respectively.

Object Detection	Base Model	Layers in Base model	Fine Tune layers	Epochs	Batch Size	Test Loss	Test Accuracy
YOLO	MobileNet	154	34	10	32	0.47	0.84
YOLO	ResNet	175	55	10	32	0.32	0.86
YOLO	Inception	311	191	10	32	0.38	0.89
YOLO	Inception-Resnet	780	660	10	32	0.26	0.92

Table 1. The Experiment table outlines various base models, their corresponding properties and hyperparameters, as well as the results of test loss and test accuracy.

Results

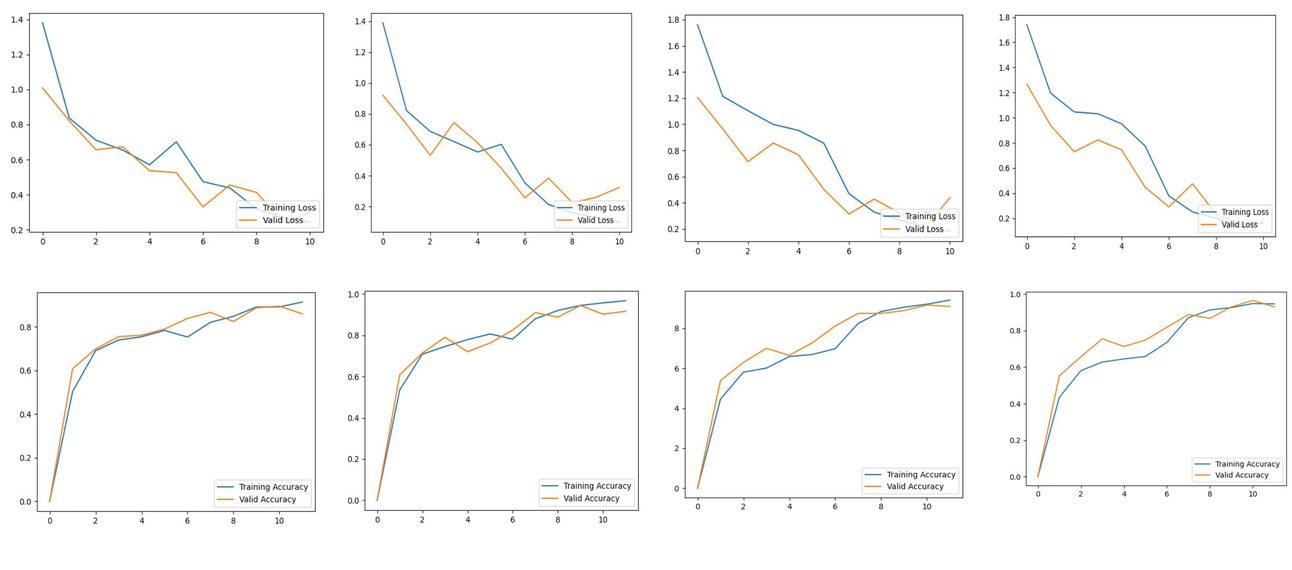


(a) (b) (c)

Figure 1. The images in column (a) are actual photographs taken from my Google Photos, while the photos in column (b) were generated using YOLO object detection to identify a person's face with a certain probability. The photos in column (c) are the resulting images after using one of the base models to predict and label the person's face with their actual name.

Graphs

The sections shows the training loss, validation loss, training accuracy and validation accuracy of all the different models. As mentioned in Table 1. all the models were trained for 10 epochs with 32 batch size with 60% of training data and 20% validation data.



(a) MobileNet (b) ResNet (c) Inception (d) Inception-ResNet

Figure 2. The above graph shows the Training Loss, Validation Loss, Training Accuracy and Validation Accuracy value for different base models.

Conclusion

- In conclusion, this project involved several stages, including creating individual photo albums, object detection using the YOLO model, feature extraction, and model training.
- The project also required accessing and downloading photos from Google Photos using GoogleAPIClient and the Google Cloud Console.
- The project demonstrated that with appropriate techniques and tools, it is possible to develop a facial recognition system that can automatically recognize and classify individuals in photos.
- The system has various potential applications, including personal photo management, security, and law enforcement. However, the project also highlights the importance of ethical considerations and potential privacy concerns that need to be addressed when developing and deploying facial recognition systems.

Future Work

- To improve the results across multiple classes, I aim to conduct additional data preprocessing steps to generate a larger number of individual images.
- My goal is to utilize this model on video input, wherein it will identify all individuals and label them accordingly while the video is playing.
- Additionally, I intend to generate image styles for these identified individuals using STYLEGAN.

References:

- ***https://github.com/umesh-gattem/CV_Project***
- **Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens: *Rethinking the Inception Architecture for Computer Vision***
- **Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun: *Deep Residual Learning for Image Recognition***