Lambton College

Big Data Analytics

BDM 3035 - Big Data Capstone Project

PhD. Meysam Effati

Mississauga, Ontario

July 30th, 2024

Deep Face Detection Model – Milestone Report # 5

- Brayan Leonardo Gil Guevara (C0902422)
- Eduardo Williams (C0896405)
- Marzieh Mohammadi Kokaneh (C089839)
- Rohit Kumar (C0895100)
- Saurabh Laltaprasad Gangwar (C0894380)

Contents

Figures

Tables

Introduction

The face recognition industry has been crucial for security purposes over the years, however, once COVID hit in late 2019, more requirements for this technology have been raised.

This document explores the use of deep learning for facial recognition. The Idea comes from creating a local dataset having different photos of people on a team and training the model to detect the identity of each person. We will discuss how to collect and prepare data, as well as develop models for this combined task. The target for this project is security companies for letting people to entrance the office or not.

In this delivery we are approaching in topics such as: model saving and deployment.

Progress Report

In previous reports we detailed how data (pictures) was collected and labeled. Also, how we performed the augmentation process for having a heterogeneous dataset. Additionally, we reported about the model definition and the model fitting customized class, as well as classification and regression loss calculations methods. Also, we detailed information about callbacks and how they influence the training process. Finally, we talked about some evaluation methods like Intersection Over Union, which gave us a clearer understanding of how accurate regression predictions (coordinates) are. In this delivery we are going to save and deploy our application.

Loading weights

Since we didn't use **restore_best_weights** parameter for **EarlyStopping** callback, we must restore them after training and before saving the model.

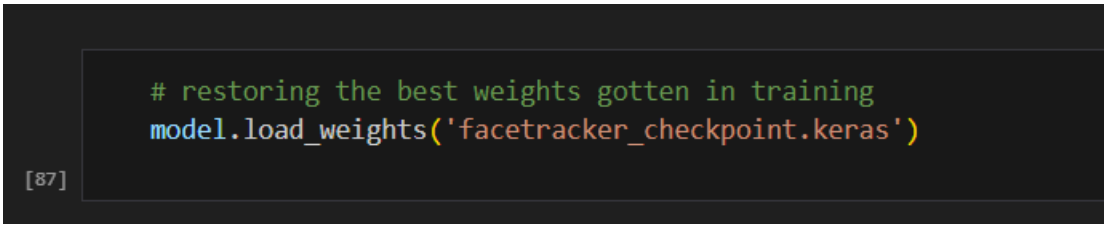

Figure 1 Loading weights

Saving the model

After loading weights we saved the model, as shown in Figure 2.

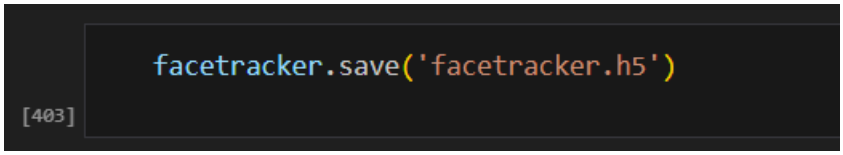

Figure 2 Saving the model

The model was saved as a h5 file. According to fileformat.com an H5 model is one of the Hierarchical Data Formats (HDF) used to store large amount of data. It is used to store large amount of data in the form of multidimensional arrays. The format is primarily used to store scientific data that is well-organized for quick retrieval and analysis. H5 was introduced as a more enhanced file format to H4. It was originally developed by the National Centre for Supercomputing Applications and is now supported by The HDF Group.

After saving the model, we tried it with some unseen pictures which were taken from other kind of mobile phone in order to test its robustness, getting good results.

Model deployment

Finally, we were able to deploy our application which has the following components:

Component	Description
facetracker.h5	Generated face detector model
app.py	Controller python / flask program
index.html	User interface

Table 1 Application components

First of all, we generated a local environment from Visual Studio Code where imported all required libraries. Then, we created the **app.py** program which acts as an orchestrator and has the following main sections:

- **Root app routing** (Figure 3) which is in charge of invoking and rendering the html page.

- **POST prediction method** (Figure 4) that calls the model, which in turn makes the prediction.

```python
# defining root endpoint
@app.route('/')
def root():
    return index()  # redirecting to index page

# defining index page endpoint
@app.route('/index.html')
def index():
    return render_template('index.html')    # rendering from template
```

Figure 3 Root app routing

```python
# defining REST API endpoint for prediction
@app.route('/api/prediction', methods=['POST'])
def api_prediction_post():
    try:
        # data_json = request.json

        for file in request.files:
            print (file)

        if 'fileImg' in request.files:
            # retrieving sent image
            file = request.files['fileImg']

            # Reading the image via file.stream
            img = Image.open(file.stream)
```

Figure 4 Prediction method

For deployment and publication part we analyzed some platforms such as: **pythoneverywhere.com**, **ploomber.io**, **railway.app**, among others, which support Python and Flask applications. However, we got some problems or restrictions regarding to file types, file sizes, python versions, etc., so finally we selected Heroku. Heroku is a cloud platform that supports several programming languages. In addition, it has some useful e intuitive features.

Before to the deployment we were required to upload our source code to GitHub, as shown in the next Figure.

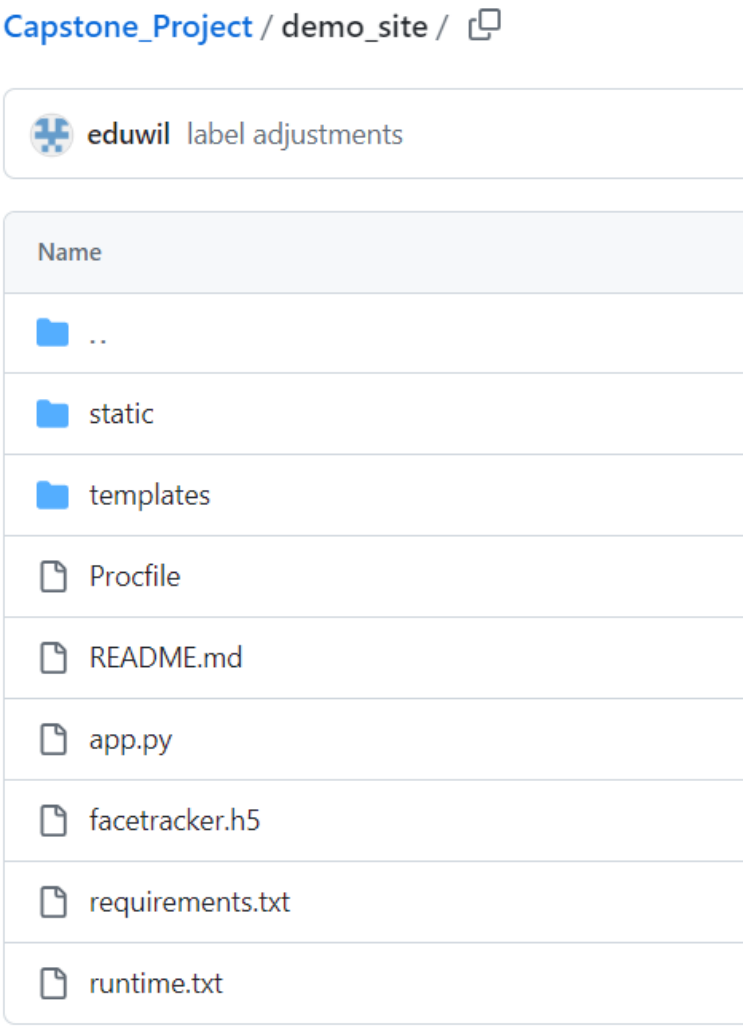

Figure 5 GitHub repository

Once this is done, we linked Heroku with GitHub, specified the repository of our application and finally clicked **Deploy Branch** button, which performed all necessary steps for publishing our application.

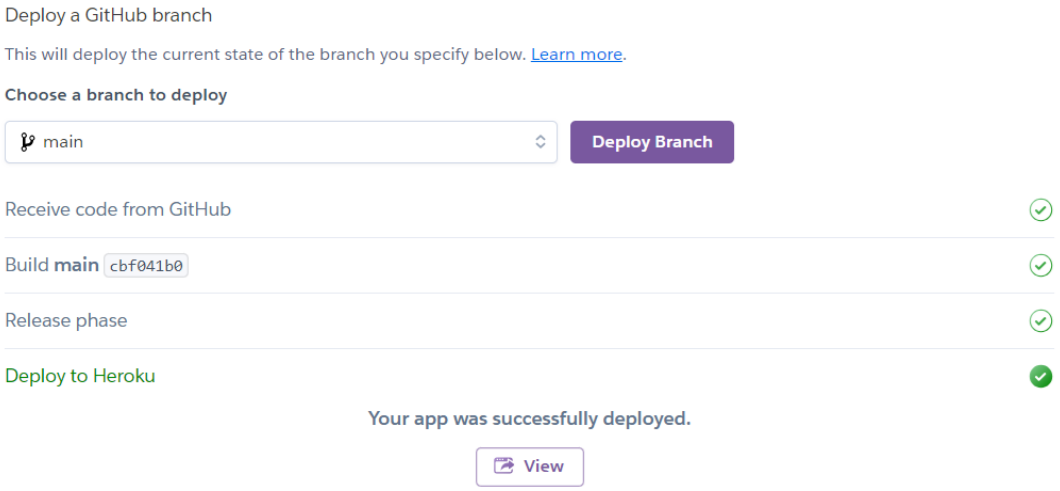

Figure 6 Deployment from Heroku

In the end, we had our application on the Internet, which can be accessed from https://grouph-facedetection-9f87e69b9513.herokuapp.com/

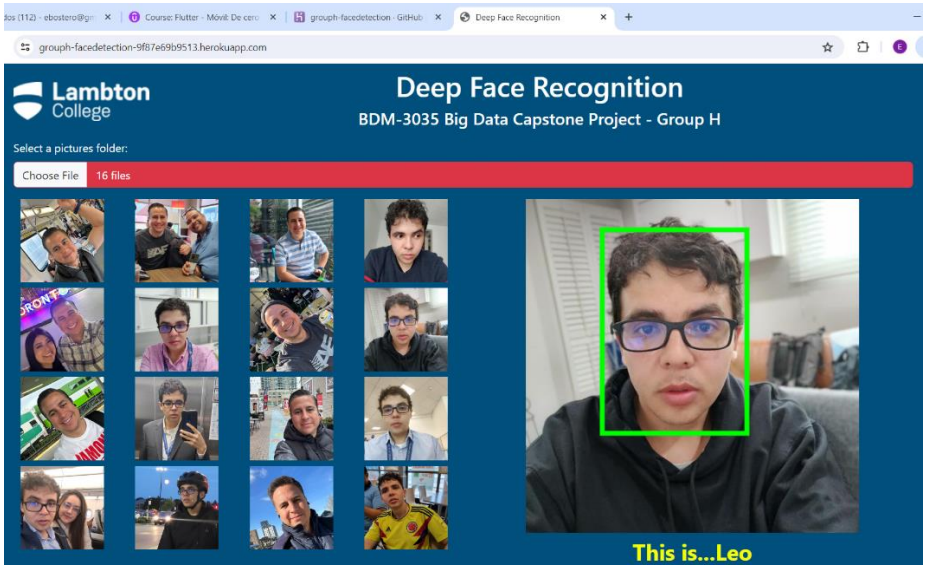

Figure 7 Web user interface

Next Steps

Once the model was saved and deployed, next steps are related to prepare for final deliveries and presentation.

Faced Challenges

We faced several challenges, especially on deployment part, since Heroku require some settings which we did not know.

Lessons Learned

We have learnt about deployment mechanisms, from generating a local environment up to loading source code in GitHub for its late deployment in a cloud environment like Heroku.

Conclusion

As we move forward, we acquire new knowledge and understanding about machine learning and deep learning applications, and how they can be published and invoked through API services and cloud technologies.

References

FileFormat. (n.d.). FileFormat. Retrieved from FileFormat: https://docs.fileformat.com/misc/h5/#:~:text=H5%20File%20Viewer-,What%20is%20an%20H5%20file%3F,for%20quick%20retrieval%20and%20analysis.