<Facial Recognition>

For

Project Report

18-May-2022

Prepared by: Jordan Wallace, Sarah Gailey, Avery Helgesen, Blake Bridges, Daniel Pierce, Danny Obeid

Team Members: Avery Helgesen, Blake Bridges, Daniel Pierce, Danny Obeid, Jordan Wallace, Sarah Gailey

**Version History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Test | 5/2/22 | Initial Run | .01 |
| Front-end | 5/05/22 | Implemented front-end gui | .02 |
| Client Server Adjustments | 5/07/22 | Merging code, made adjustments to client server | .03 |
| Model Implemented | 5/08/22 | Added our model to the backend, replaced the placeholder | .04 |
| Camera Recognition | 5/14/22 | Added instantaneous camera upload feature | .05 |
| Perfection | 5/17/22 | Combined Front-end with Back-end to become fully functional | 1.00 |

**Table of Contents**

1. Deliverables

2. Executive Summary

3. Project Overview

3.1 Problem Statement

3.2 Scope

3.3 Project Team

4. Machine Learning Aspects

4.1 Dataset

4.2 Model Creation and Training

4.3 Inference

4.4 Evaluation of the Model

5. Software

5.1 System as a Software

5.2 System Architecture

5.3 User Interface Overview

6. Use Cases

7. References

8. Glossary

9. Appendices

# 

# 

# 

# 

# **1.** **Deliverables**

This document shall be submitted to Cougar Courses website Assignment8 by 11:59PM May 18th, 2022. Every student shall submit this document individually.

The following must be accessible by github account gheniabla and by gabla@csusm.edu.

All deliverables shall be available to access 11:59PM May 18th, 2022. Any content shall not be updated after the submission.

Code Repo (URL):

Server Code: <https://github.com/BlakeBridges/CS478FinalProject>

Model link: <https://www.mediafire.com/file/o2k5csw4ffvg0ms/blake.h5/file>

Presentation/Demo Video (if any) (URL):

# 

# **2.** **Executive Summary**

Give a quick executive summary of what you did and what are the general and most important capabilities of the system you investigated.

Please describe the most interesting/impressive aspect or experience you had in the project.

If you didn’t get the results originally planned, include it here.

For our semester group project, our team built a Deep Learning Facial Recognition tool that will allow user access if their face is recognized as a part of a specific dataset. This tool aims to mimic the facial recognition security tool most commonly used with smartphones. Our facial recognition program will use computer-generated filters via deep learning algorithms to transform the face images into numerical expressions that can be compared to determine many factors including loss, similarity, and accuracy. The user is able to take a photo of themselves in real time and upload it to our database which then compares the photo against the trained model and decides if it a match (1) or not (0).

The ability to train models in such a vast array of ways in order to get the desired results was an interesting aspect of the project. Watching how the quality and number of test images could affect our results emphasized the need for quality data sets to subsequently yield quality results.

Our model is not perfect yet, but we did find that accurate results were obtained at a greater rate if the uploaded picture of the user has similar lightning effects compared to the photos of the user in the model. If the lighting were to drastically change (i.e. much brighter or darker) our accuracy declined.

# 

# **3.** **Project Overview**

## **3.1** **Problem Statement**

The purpose of this project is to provide a safe, easy, and reliable way to access your private or personal information without the need for a long or forgettable password that could eventually be hacked.

## **3.2** **Project Team**

Name of the project: Facial Recognizer 3000 Turbo Deluxe Ultra

|  |  |  |  |
| --- | --- | --- | --- |
| Name of the Team member | Responsibility | Contribution % | Notes |
| Avery Helgesen | Deep Learning  Engineer | 16.67% | Implemented deep learning algorithms and trained models |
| Blake Bridges | Back-End Engineer | 16.67% | Built data structure to support for models and data |
| Daniel Pierce | Front-End Engineer | 16.67% | Creating front-end gui and client server |
| Danny Obeid | Back-End Engineer | 16.67% | Built data structure to support for models and data |
| Jordan Wallace | Associate Back-End Engineer, Documenteer | 16.67% | Assisted in back-end flask development, Documentation |
| Sarah Gailey | Associate Front-End Engineer, Documenteer | 16.67% | Front End Camera Recognition, Documentation |

**Please note:** The total contribution of all team members shall add up to 100%. For example, if you have 4 people in the group and each of them contributed equally, then each member’s contribution is 25%.

# 

# **4.** **Machine Learning Aspects**

Describe AI/ML aspects in general.

## **4.1** **Dataset**

• The test dataset we used is pictures of our group members’ faces; about 600 pictures of group members’ faces total: 300 ‘anchor’ pictures and 300 ‘positive’ pictures, more on the difference below.

• The training dataset is pictures of random celebrities’ faces about; 13,000

• The anchor and positive datasets used for our testing are inherently flawed to a degree, as they were all gathered in the same place, and at the same time. For a more proper model, images would need to be gathered over the course of several days, under different lighting conditions and in different places.

• The only problem with our current data is that, with all pictures trained in the same place, it can recognize an empty room as “you”, since all your images have that room, and so it recognizes your room as part of “you”.

## **4.2** **Model Creation and Training**

• We trained our model using pictures of random people’s faces from the internet. We used approximately 13,000 pictures to train our model.

• Our algorithm is a siamese detection model, meaning that two images are required to make each comparison.

• For training, each pair that contains an ‘anchor’ image and a ‘positive’ image are given a value of 1, meaning the people match, while for each pair containing an ‘anchor’ image and a ‘negative’ image, they are given a value of 0, meaning the people don’t match.

• The algorithm analyzes each picture, and then looks at the value associated with it to see its match number.

• Basically, the model is told which images match and which ones don’t for training, and it analyzes each image and it’s data to learn why pictures are considered a match.

## **4.3** **Inference**

• The inference process is rather simple: there is a folder of images called ‘validation\_images’, and this folder is full of random images pulled from our ‘positive’ folder, the same images used for training.

• One way or another, you feed an image of yourself from your webcam into the software, the raw initial code used cv2, but our flask deployment had to use a different method.

• Using the model, this image is compared against all the images in your ‘validation\_images’ folder, and instead of being told whether an image is a match or not, it of course has to guess based off of its training.

•It analyzes both images, and after deciding if the image is a match or not with the current ‘validation\_image’, it then moves and repeats the process for all validation images.

• The model designates a “1” if it believes the images to be a match, and a “0” if it believes the images to not be a match, and a running tally of all 1’s and 0’s is kept throughout execution.

• After all validation images have been compared, it then compares the ratio of 1’s to 0’s based on the ‘validation\_threshold’ value, so, for example, if the validation\_threshold is set to 0.6, then 60% of the decisions by the algorithm must be 1.

• If the threshold is met, then the pictures are said to be a match, and if the threshold is not met, then they are not a match.

• The input\_image is also destroyed each time the code runs.

## **4.4** **Evaluation of the Model**

• The strange thing with our model is that our accuracy is always 100% through training, as the algorithm is told the answer during training.

• The true metrics of the model are actually decided by the programmer, as you can set how strit you want the model to be based on the threshold values.

• Along with that, the model’s accuracy heavily depends on the data you use for it, so the more varied your images are, the more accurate the model can be in detecting your face, as it will be used to your face in multiple different facets, rather than just learning your face in the same position, and in the same room.,

# **5.** **Software**

## **5.1** **System as a Software**

• Input allows for user to upload a real-time image of themselves to be submitted to the server for facial recognition

• Processing this image involves training, testing, and comparing the datasets

• Output results in the system declaring if the image is a match or not from a specified location in the database

• The system does not store the real time photos in the database after completion

• This system compares images using deep learning algorithms and libraries to make a logical decision based off data and instructions

## **5.2** **System Architecture**

• Back-end is a restful Flask server that has an endpoint that is used for receiving a picture file and sending back success or failure.

• Front-end is built with combination of HTML, JQuery, and CSS

## **5.3** **User Interface Overview**

• User loads the initial web page where it asks for permission to use the webcam.

• Once granted permission, the user takes a photo of themselves using their webcam to login.

• The system then replies whether you are granted permission to login, or the facial recognition failed and you cannot access.

# 

# **6.** **Use Cases and tests**

• Unlocking access to personal data; Pictures, Files, Documents, Passcode Vault, etc.

• Facial recognition has becomes increasingly more popular in society

• Test Data was 13,000 celebrity faces

• Describe your use cases or tests you conducted.

# 

# **7.** **References**

• Provide a list of all documents and other sources of information referenced in your project. Include resource/document title, date, and author for each.

Useful Angle. “Capture Photo from a Camera Using Javascript.” *UsefulAngle*, https://usefulangle.com/post/352/javascript-capture-image-from-camera.

# 

# **8.** **Glossary**

• Define all terms and acronyms required to understand your project and this report.

Epoch - The number of epochs is a hyperparameter that defines the number times that the learning algorithm will work through the entire training dataset. One epoch means that each sample in the training dataset has had an opportunity to update the internal model parameters.

Loss - Loss is a number indicating how bad the model's prediction was on a single example.

Layers - Layers are the basic building blocks of neural networks in Keras. A layer consists of a tensor-in tensor-out computation function

Tensor - Tensors are the data structure used by machine learning systems, and getting to know them is an essential skill you should build early on. A tensor is a container for numerical data.

CNN - Within Deep Learning, a Convolutional Neural Network or CNN is a type of artificial neural network, which is widely used for image/object recognition and classification.

# **9.** **Appendices**

• Include any relevant appendices (if any).