CS 4200 Project Proposal: Intelligent Laptop

Ta-Wei Chien, June Lee, Charles Bickham

Abstract

The human face has very distinct characteristics that make each one of us unique. Because of this reason, facial recognition is now a widely accepted biometric to secure our personal devices such as cellphones and laptops. However, the perfect facial recognition program does not exist today; in fact, there are still many flaws with even the most popular software. Our team has surveyed the flaws that are found in most facial recognition software today and have crafted our own artificial intelligence solutions. We have come up with a program that is trained over time to recognize its master, using a webcam as a visual tool. We want to enable the user's laptop so that it may recognize its master better and better over time using our program.

Introduction

This project had initially piqued our group's interest as computer vision and facial recognition technology is all around us today, especially in our smartphones. As a field in artificial intelligence, it is easy to see that computer vision has heaps of potential to be a driving force for the future. We believe that this project will be a great learning experience for us all before we hopefully enter the computer science industry.

Although facial recognition and computer vision technology is something many of us have seen or even use on a daily basis, we believe our take on that technology is something that hasn't been done before. Most facial recognition programs today use a few initial photos or videos to use as data to recognize its owners. Instead of just a one-time video or picture, our program will continuously at times access the webcam to collect data on its owner and even catch certain common actions to have a more accurate and faster recognition in the future. We believe that this different form of data collecting will be great for those who wear accessories on their face or head often, like glasses, hats, and masks. It is a known fact that many people have some difficulty accessing their devices with accessories on, because other programs simply do not have enough data to recognize their owners with full confidence. This is a very significant issue in today's climate, as masks have become quite a necessity during COVID-19 and this point has been a big motivation for our project.

Related Works

There have been a lot of studies in the past about Facial Recognition that has inspired us to do this project. We have used the following articles to give us the inspiration for our project and to help guide us along the way.

The article "Simultaneous Facial Feature Tracking and Facial Expression Recognition" goes into detail on how facial recognition works and is approached. It explains that facial recognition is broken down into three characteristics. The first characteristic is at the bottom level in which the facial feature points around the facial component's eyebrows, mouth, etc. The middle level is the facial action units that represent a set of certain facial muscles such as the lid tightener, eyebrow raiser, etc. The top level characteristic of a facial recognition program represents six features that are known for the human emotion states. But in this article their approach was different than the typical mainstream approach. The article details on how they used a dynamic Bayesian network to recognize different facial evolvements using all three levels simultaneously. To clarify, Bayesian networks are a type of probabilistic graphical model that use Bayesian inferences for probability computations. Bayesian networks aim to model conditional dependence, and therefore causation, by representing conditional dependence by edges in a directed graph.

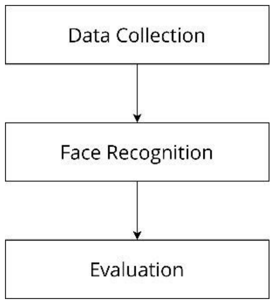
Through these relationships, one can efficiently conduct inference on the random variables in the graph through the use of factors. There were multiple experiments highlighted in the article in which they used to test their models as well.

In the article, "A General Approach on Facial Feature Extraction and Face Attributes," the author details how facial feature extraction is a method used to recognize the nose, mouth, and other components that make up the human face. According to the article, " Facial recognition system detects and returns the information regarding visual content with confidence scores in human face from digital image or frames from video source" (2). Many people come up with different algorithms to come up with the best way to recognize certain patterns within the face. Facial recognition takes into account visual biometrics that not only include your face but wearables too like glasses, headphones, and even masks. This is important today due to the increase of mask usage as a result of COVID-19. Facial recognition extracts more than 27 categories that include gender, mood, visual age, smile expression, face orientation, color, picture quality, etc. The information we get from it can be used in countless real-world applications. One of the more popular applications would be in the use of mobile camera technology, which includes facial recognition to unlock your device and applying face filters. In this paper, it gives a brief study on different attributes that were obtained from Microsoft FaceAPI, FacePlusPlus Cognitive Service Face++API and AmazonWebService Rekognition.

These articles had inspired us for the makings of this project and have gave us more information on similar past experiments that have been done using Computer Vision. The many articles were intended to be used as stepping stones to reach our ultimate goal.

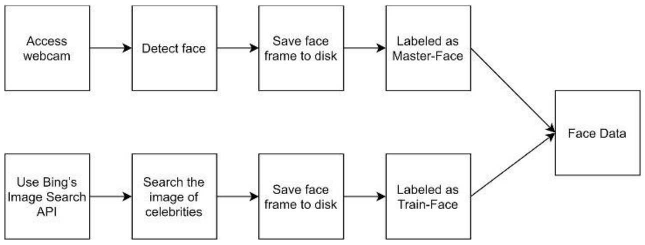
Our Method

We wanted to develop an artificial intelligence program that will protect the security of one's laptop using facial recognition technology. To achieve our goal, we decided to make the laptops enable verification as to whether the current user is the true owner through the camera. The work-flow of our mainly consisted of three integral parts: Data Collection, Face Recognition, and Evaluation.



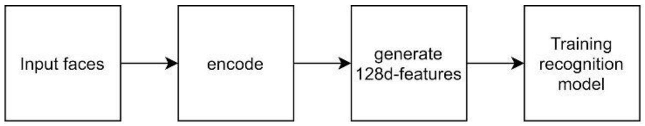
In the Data Collection part, there are two tasks we needed to do:

1. Collect the face data of the laptop owner
2. Collect the training face data



We wrote a Python script to do face enrollment via OpenCV webcam and Bing's Image Search API.
To gather the face image of the master, we needed to set the video camera up to detect the *(x, y)*-coordinates of the user's face in a video stream and save the frames containing their face to disk.
To collect the training face data, we mainly used Bing's Image Search API to download several face images of some singers on www.billboard.com and saved their faces into an individual directory.

In the Face Recognition part, we utilized a dlib facial recognition network. The output feature vector was 128-d (i.e., a list of 128 real-valued numbers) that is used to *quantify the face*. Training the network was done using ***triplets*** in this module:



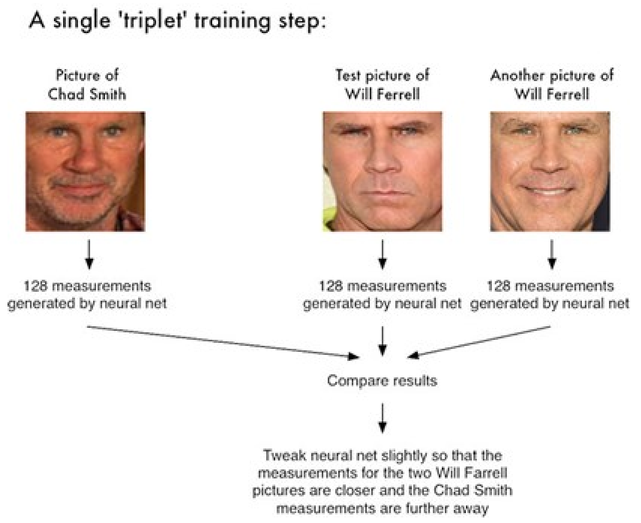
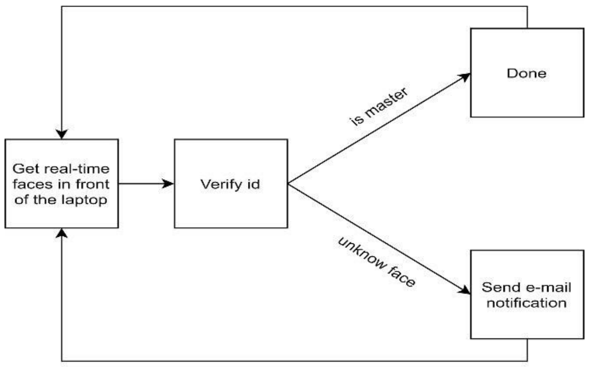

A single 'triplet' training step:
Picture of Chad Smith
Test picture of Will Ferrell
Another picture of Will Ferrell
128 measurements generated by neural net
128 measurements generated by neural net
128 measurements generated by neural net
Compare results
Tweak neural net slightly so that the measurements for the two Will Farrell pictures are closer and the Chad Smith measurements are further away

Figure-1: The concept of using triplets

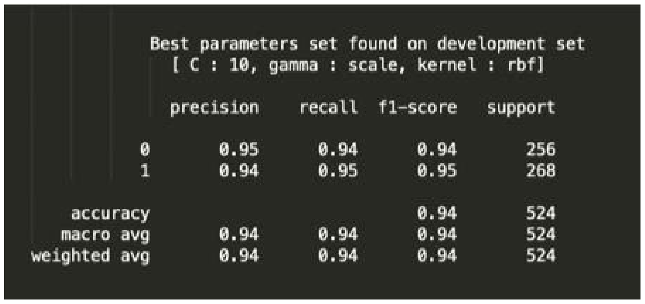
Through the recognition network, we encoded the data from the faces into a 128-d real-valued number, and used these arrays as the input of our facial recognition model which was based on SVM.

Besides the training model method, we also added another face recognition method in our program, the "face distance" method. We used the api of face_recognition, the python open source to compute the difference between the encoding of each face. This method made our program able to recognize the master in the situation that the face data of the master is not enough to train a machine learning model.

After implementing all of the face recognition methods, we approached the final part – Evaluation. First, we wrote out a function that allows the camera to get the face image of the current user, and verify the identity of the face via our previously encoded recognition model. Second, we cross-checked the result of the recognition, to find out if the laptop now remembers the owner's face. Finally, we invited several other users such as family and friends to use our laptop to check if we get an email notification that assures our model was working correctly with more inputted data.



Model Selection & Evaluation



```
Best parameters set found on development set
[ C : 10, gamma : scale, kernel : rbf]

              precision    recall  f1-score   support

           0       0.95      0.94      0.94       256
           1       0.94      0.95      0.95       268

    accuracy                           0.94       524
   macro avg       0.94      0.94      0.94       524
weighted avg       0.94      0.94      0.94       524
```

Our Distribution of Work

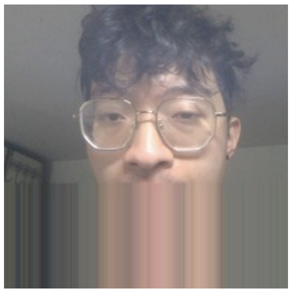
An initial step our team took to make sure that we are correctly collecting data, we used a Python script to do face enrollment via OpenCV webcam and Bing's Image Search API. Ta-Wei, who has a bit of experience with the API and facial recognition software, was mainly in charge of this part for about two to three weeks. He was responsible for getting the data collection ready with real in-person images and images of known faces from the web. Soon after, during the real facial recognition step Charles used the dlib facial recognition network to train the facial recognition model for two to three weeks. After that, Ta-Wei started the model selection for choosing the proper parameters of our recognition model and trained the demo model. And finally, June made sure that the software was able to evaluate the facial image of the current user using a function and our recognition model. During the time after most of the implementation, we made sure that the program outputs the correct answers and made any necessary tweaks. We also used the rest of the project's timeline to go through the final evaluation step and tried to re-tweak any final bugs.

Challenges We Faced

The challenges we faced were not few and far in between on such a large project like this. We were treading in completely unfamiliar territory and were forced to work on the project from the safety of our homes. Some of us were not familiar with Python and our time zones didn't match well either. At first, it was difficult for us to understand what we were reading in the research articles because we had little to no experience in the field of artificial intelligence. It took a lot of hours in class and our own research to start picking up most of what we were reading and seeing with other examples in computer vision.

Another significant challenge we faced was trying not to fall into the infamous hole of the programmer's "tutorial hell." In other words, it was easy to go down the rabbit hole of one tutorial after another, feeling like we were picking up translatable skills and making progress. However, sometimes we found it difficult and overwhelming when trying to apply what we were learning to more larger and complex systems. To successfully navigate our way through this "tutorial hell," we only took small things we needed from here and there and tried to immediately apply to our project. This constant method of learning and reapplying in the early stages of our programming kept our learning honest and true. As we got deeper into our project, we found the opposite problem to be true, where there wasn't a great plethora of tutorials that could help with very specific concepts or bugs. Because the field of computer vision is relatively new, gathering the best information was a challenge for our team.

As we were implementing our program, we found that there was a little problem when collecting in-person faces through the web-cam. Sometimes, the image would be blurred after we crop the face from the whole screen.



We were able to solve the issue by improving the frame rate of the webcam and adjusting the method we used to crop faces.

And there's another problem we met during the face encoding was the image pipeline and the preprocessing. We could not extract the encode of every face image successfully, especially for getting the face encoding for the face distance recognized method. We thought it might be three reasons: 1. We were using the pre-trained embed model; 2. It was hard to control the quality of the image recorded by the laptop camera. 3. We might have not found the best way to crop and rotate the face yet, so it was hard for the model to extract the encode of our input data.

Overfitting was also a big constraint so far. The light, the angle of the user's face or the user's facial expression might all be the reasons. If the user kept recording his/her faces in Room A, the model would highly be overfitting. When the user moved to Room B, the model would easily misclassify the identity of the faces. And there's another interesting thing we found, a model which was trained by around 500 user's faces which recorded in different places and time was much better than a model which was trained by 10000 user's faces but all recorded in the same place whole night. The most relative reason for this phenomenon might be the way we record the master's data. As we were using the laptop camera, it was likely that we were recording a video and kept fetching the extreme image as the training data.

Ultimately, we also faced a challenge because of the project deadline. There were still a few things that we wanted to implement into the project, such as a graphical user interface. We were also very keen on adjusting and retooling our model; however, limited time made us turn in what we have. Although this was a looming challenge for us while working on the project, our team believes that we gave it our best effort and have implemented a large majority of what we wanted.

What We Learned

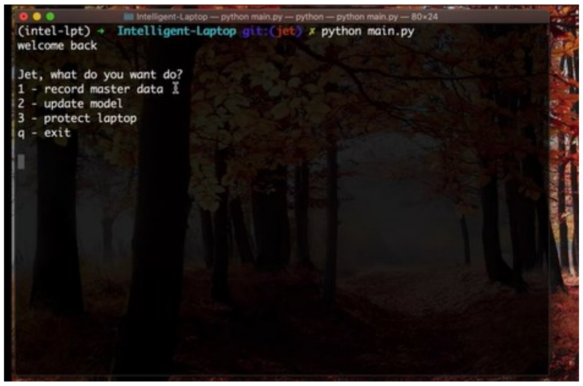
During the lifetime of the project, we learned a tremendous amount about the field of artificial intelligence and computer vision. We started as beginners about the world of artificial intelligence and knew even less about the field of computer vision within it. Some of us were not completely comfortable using Python, so we took our time to get familiar with the language so we could write out scripts for the program. As we did more research about the project by reading articles and watching videos, we learned how to use different APIs and libraries such as Bing's Image Search API. We learned the core concepts behind computer vision and the use of various techniques like triplets and facial recognition networks. We learned how to collect data that would help with our project and how to train our existing model. Our team also eventually acquired the skill of constantly tweaking our model for more accurate evaluations. Besides the obvious educational and technical lessons we learned, our team also learned how to function as a unit on large artificial intelligence projects such as this. We mastered the action of fairly distributing the load and how to remotely work with each other during these times.

What We Could Have Improved On

Overall, the team was more than satisfied with the final product. One of the things we could have improved on is time efficiency. For most of our members they were new to learning Python so a significant amount of time that was spent on the project was understanding python and how it works. If we would have dedicated more time to learning Python before we started coding our project it would have been smoother and we wouldn't have hit some of the road blocks that we did.

How to Operate the Intelligent Laptop Program

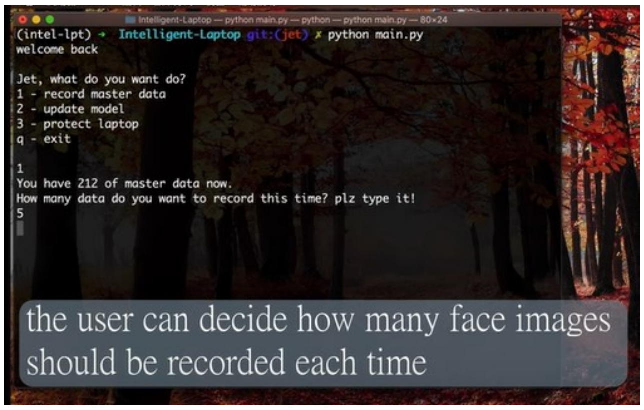
After downloading our source code from GitHub, the user should open a terminal and run our code from the main.py file. The user will then be greeted by our interface asking if they want to: 1) Record master data, 2) Update the model, 3) Protect the Computer, or 4) Exit the program.



```
Intelligent-Laptop — python main.py — python — python main.py — 80×24
(intel-lpt) → Intelligent-Laptop git:(jet) ✗ python main.py
welcome back

Jet, what do you want do?
1 - record master data
2 - update model
3 - protect laptop
q - exit
```

If the user picks the first option, which is recording master data, then they will be prompted to choose how many face images they would want to record each time.



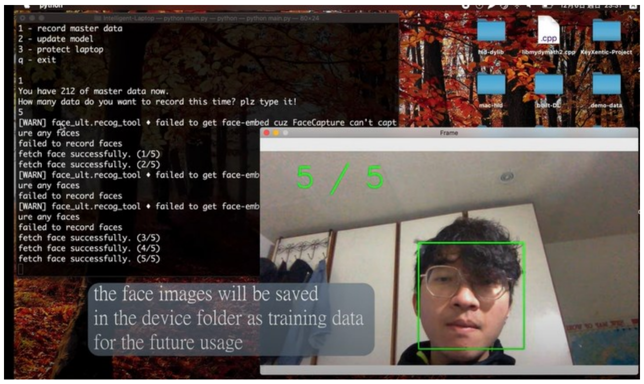
```
Intelligent-Laptop — python main.py — python — python main.py — 80×24
(intel-lpt) → Intelligent-Laptop git:(jet) ✗ python main.py
welcome back

Jet, what do you want do?
1 - record master data
2 - update model
3 - protect laptop
q - exit

1
You have 212 of master data now.
How many data do you want to record this time? plz type it!
5
```

the user can decide how many face images should be recorded each time

The face capturing tool then pulls up the user's webcam device and captures the user's face as many times as the user had inputted before. During this process, it is also important for the user to take clear and recognizable photos as they will be used for data to be stored in the device folder as training data to be used for the future.

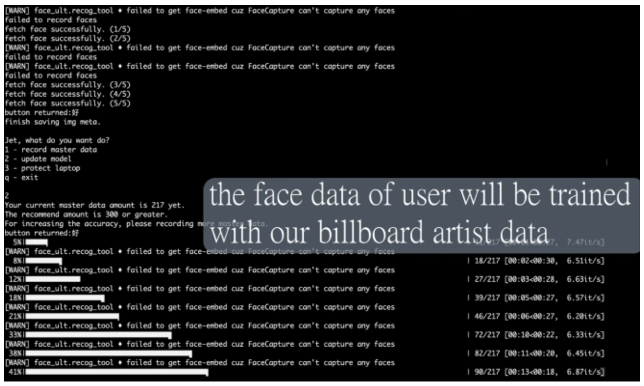


```
1 - record master data
2 - update model
3 - protect laptop
q - exit

1
You have 212 of master data now.
How many data do you want to record this time? plz type it!
5
[WARN] face_ult.recog_tool ♦ failed to get face-embed cuz FaceCapture can't capt
ure any faces
failed to record faces
fetch face successfully. (1/5)
fetch face successfully. (2/5)
[WARN] face_ult.recog_tool ♦ failed to get face-emb
ure any faces
failed to record faces
[WARN] face_ult.recog_tool ♦ failed to get face-emb
ure any faces
failed to record faces
fetch face successfully. (3/5)
fetch face successfully. (4/5)
fetch face successfully. (5/5)
```

the face images will be saved in the device folder as training data for the future usage

After the user confirms that they have collected a reasonable amount of master data, the interface will be pulled up again.



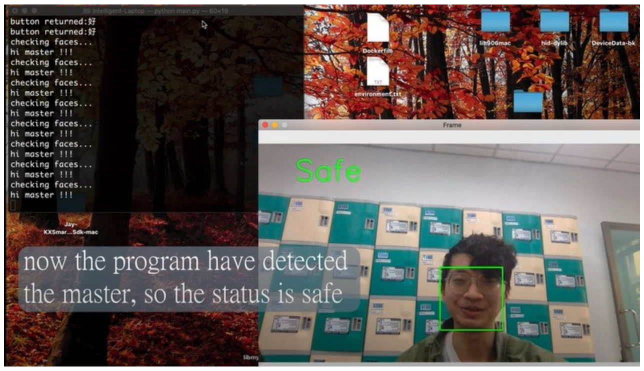
```
[WARN] face_ult.recog_tool ♦ failed to get face-embed cuz FaceCapture can't capture any faces
failed to record faces
fetch face successfully. (1/5)
fetch face successfully. (2/5)
[WARN] face_ult.recog_tool ♦ failed to get face-embed cuz FaceCapture can't capture any faces
failed to record faces
[WARN] face_ult.recog_tool ♦ failed to get face-embed cuz FaceCapture can't capture any faces
failed to record faces
fetch face successfully. (3/5)
fetch face successfully. (4/5)
fetch face successfully. (5/5)
button returned:好
finish saving img meta.

Jet, what do you want do?
1 - record master data
2 - update model
3 - protect laptop
q - exit

2
Your current master data amount is 217 yet.
The recommend amount is 300 or greater.
For increasing the accuracy, please recording more master data.
button returned:好
  5%|███                                                          | [illegible]/217 [[illegible]<00:27,  7.47it/s]
[WARN] face_ult.recog_tool ♦ failed to get face-embed cuz FaceCapture can't capture any faces
  8%|████                                                         | 18/217 [00:02<00:30,  6.51it/s]
[WARN] face_ult.recog_tool ♦ failed to get face-embed cuz FaceCapture can't capture any faces
 12%|██████                                                       | 27/217 [00:03<00:28,  6.63it/s]
[WARN] face_ult.recog_tool ♦ failed to get face-embed cuz FaceCapture can't capture any faces
 18%|█████████                                                    | 39/217 [00:05<00:27,  6.57it/s]
[WARN] face_ult.recog_tool ♦ failed to get face-embed cuz FaceCapture can't capture any faces
 21%|███████████                                                  | 46/217 [00:06<00:27,  6.20it/s]
[WARN] face_ult.recog_tool ♦ failed to get face-embed cuz FaceCapture can't capture any faces
 33%|█████████████████                                            | 72/217 [00:10<00:22,  6.33it/s]
[WARN] face_ult.recog_tool ♦ failed to get face-embed cuz FaceCapture can't capture any faces
 38%|████████████████████                                         | 82/217 [00:11<00:20,  6.45it/s]
[WARN] face_ult.recog_tool ♦ failed to get face-embed cuz FaceCapture can't capture any faces
 41%|█████████████████████                                        | 90/217 [00:13<00:18,  6.87it/s]
```

the face data of user will be trained with our billboard artist data

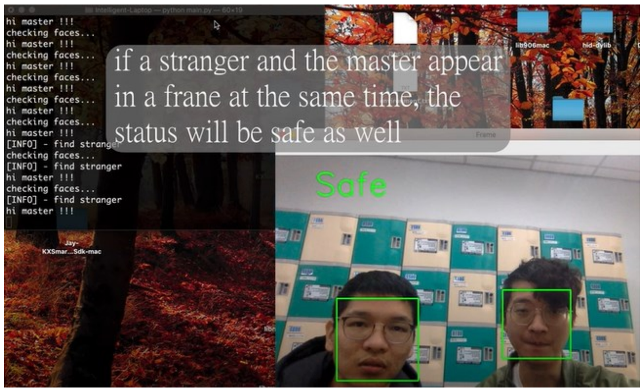
When the user wants to progress and update their model, they will press ‘2’ and run the model’s script. The face data of the user will then be trained with our database of Billboard artists. After the training of the model is finished, the new updated model and the program will show its average score of 10 times cross-validation.



```
button returned:好
button returned:好
checking faces...
hi master !!!
checking faces...
hi master !!!
checking faces...
hi master !!!
checking faces...
hi master !!!
checking faces...
hi master !!!
checking faces...
hi master !!!
checking faces...
hi master !!!
checking faces...
hi master !!!
```

now the program have detected the master, so the status is safe

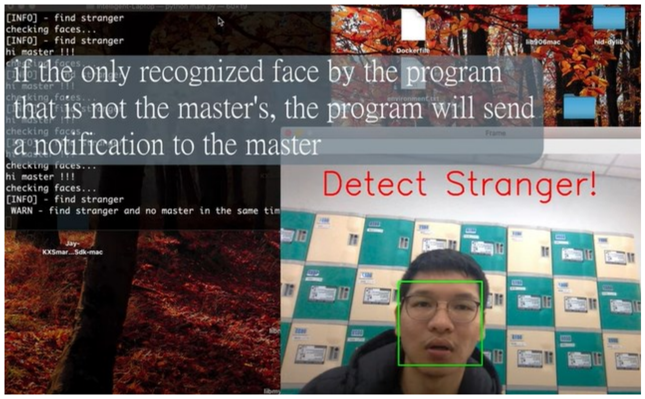
When the user presses “3” and wishes to protect the laptop, the program will go into our ‘protect mode.’ The Intelligent Laptop program will then launch our facial recognition program and use the user’s webcam to determine if the user is the saved master. If the user is deemed to be the master from the model then the computer will open and be greeted by the program. If not, the software will lock the user out and send an e-mail notification to the master, alerting that another unwanted user has tried to access the master’s laptop.



```
hi master !!!
checking faces...
hi master !!!
checking faces...
hi master !!!
checking faces...
hi master !!!
checking faces...
hi master !!!
checking faces...
hi master !!!
[INFO] - find stranger
checking faces...
[INFO] - find stranger
hi master !!!
checking faces...
[INFO] - find stranger
hi master !!!
```

if a stranger and the master appear in a frane at the same time, the status will be safe as well

On top of this, our program also has the ability to decipher who its master is even when mixed in with a stranger. If there is the master and a stranger in the same frame the laptop will deem the users safe and allow access.



```
[INFO] - find stranger
checking faces...
[INFO] - find stranger
hi master !!!
checking faces...
hi master !!!
checking faces...
[INFO] - find stranger
 WARN - find stranger and no master in the same tim
```

if the only recognized face by the program that is not the master's, the program will send a notification to the master

However, as soon as the master is out of the webcam's frame the program will immediately send an e-mail notification to the master alerting that an unknown user has attempted to access the laptop. Within this e-mail, the stranger’s face and attempted access time will be attached and sent to the master.

Watch our demo video here: https://www.youtube.com/watch?v=eVziLxWS-mTo&ab_channel=PuffVayne

Our project-repo: https://github.com/jet-chien/Intelligent-Laptop

Future Work

Some of the future work to improve this project is finding more of an efficient way for the image pipelining and processing. We would also like to look for different ways to improve the encoding of the faces. And lastly after expanding our dataset, we would like to try to build our model with Tensorflow or Pytorch using deep-learning.

References

1. Y. Li, S. Wang, Y. Zhao and Q. Ji, "Simultaneous Facial Feature Tracking and Facial Expression Recognition," in IEEE Transactions on Image Processing, vol. 22, no. 7, pp. 2559-2573, July 2013, doi: 10.1109/TIP.2013.2253477.

2. P. R. Police Patil, R. Pramod and S. Sandhya, "A General Approach on Facial Feature Extraction and Face Attributes," 2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS), Bengaluru, India, 2018, pp.151-155,

doi:10.1109/CSITSS.2018.8768504.

3. Yushu Feng, Huan Wang, Daniel T. Yi, and Roland Hu, “Triplet Distillation for Deep Face Recognition”

4. Figure – 1: https://www.pyimagesearch.com/wp-content/uploads/2018/06/face_recognition_opencv_triplet.jpg