SENIOR DESIGN PROJECT

Horus

Final Report

Ufuk Palpas, Furkan Demir, Can Kılıç, Asya Doğa Özer, Muzaffer Köksal

Supervisor: Uğur Güdükbay

Jury Members: Erhan Dolak and Tağmaç Topal

May 6, 2022

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2

Table of Contents

1. Introduction

The goal of Horus is to recognize human emotions. Horus will detect the emotions of a single person or a group of people. The detection can be done with a camera or multiple cameras rapidly taking images/video in real-time. Horus will first detect the faces and then analyze their emotions. Horus will be able to analyze the user's emotions from saved videos or images besides real-time detection.

In the single user mode, Horus examines a saved video or real-time footage over a time period, it will report the various emotions detected from a single person. So it will attempt to give the subject's mood from long-term analysis of his emotions with respect to time. For Horus to analyze a user's face, there shouldn't be any obstacles, so if the user wears a mask, hat, etc. Horus will warn the user to remove any such obstacles. Horus will also have a crowd control mode where it will analyze the emotions of a group of people such as stand-up, movie, art activity audiences, or students in a classroom through multiple cameras or a single camera to provide feedback to show organizers, who can look at audiences' moods and decide on which content to keep and which to discard. Horus will also detect emotions on video conference calls such as Zoom. For this option, Horus will contain a screen capturing mode. It will analyze the emotions of the participants on the conference call and provide feedback to the organizers. Horus will also have a deception detection mode to tell whether a sentence said by the user is a lie according to the mimics and the tone.

The results will be shown in real-time on the desktop app with a small window or the mobile app with audio or visuals in various types of graphs that the user selects. The charts will contain detailed information about the scan, such as average, time analysis, camera statistics, etc., to give an appropriate result to the user.

Horus will have many usages. For example, a street artist can place a camera nearby and get feedback from the Horus according to the audience's emotions by using a mobile application that shows the general emotion of the crowd. Another use case can be handling tension and preventing aggression. For example, Horus can be used in a prison's common areas to keep an eye on prisoners. If Horus detects multiple nervous, angry prisoners, it will alert the guards.

2. Requirements Details

2.1 Functional Requirements

2.1.1 System Functionalities

- The system gives an opportunity to the user to select in which mode the data will be analyzed.
- The system requests permission to access the local storage of the user to access videos or images or to save downloaded results from Horus.
- The system requests permission to access the camera of the user's computer.
- The system gives feedback to the user according to the analyzed data which contains the emotions of the surrounding people, both visual and with the aid of an audible warning.
- The system is able to get inputs from multiple cameras at the same time.

2.1.2 User Functionalities

- The user can choose between single user mode, deception detection mode, screen capture mode and crowd control mode. Also to see the old results the user can choose the list last analyses option.
- In the single user mode, the user can see both the real time and cumulative results on the same page while the data is being analyzed.
- In the crowd control mode, the user can get both the real time and cumulative feedback from Horus according to the crowd's emotions.
- In the crowd control mode, the user can connect more than one capturing device.
- In the deception detection mode, the user can get positive or negative results about whether it is a lie or not, after the scanning and composing a sentence tasks are finished.
- In the screen capture mode, the user can see the real time and cumulative results from Horus according to the people's emotions.
- In the single user and crowd control modes the user can upload an image or video from their devices instead of getting input by camera.
- The user can change the type and the colors of the graphs in both real time analysis and in the old analysis section.
- The user can download a result to his/her device in the desired format.

- The user can connect to Horus via smartphone and earphones to get feedback remotely in single user, crowd control or screen capture mode or to see the results of the older scans.

2.2 Non-Functional Requirements

2.2.1 Reliability

- Horus detects the faces as correctly as possible.
- Horus detects the emotions as correctly as possible.
- Horus determines deception as much as possible.

2.2.2 User-Friendliness

- The user interface of our app should be simple and easily understandable.
- It does not require any pre-education to use it.
- The user will change the colors and types of the resulting graphs as s/he desires.
- The user will see the real time and cumulative results at the time of scanning easily on the same screen or from his/her phone.
- The user will reach old results and view them easily.

2.2.3 Efficiency

- It detects the general mood of the user with a long enough video.
- In crowd control mode, Horus calculates the average emotion of the crowd.
- The results of the detection are given according to time.
- Regardless of the type of the video conference app, Horus starts scanning with screen capture mode.

2.2.4 Performance

- Horus uses the resources (such as CPU, RAM etc.) of the computer which it runs on to give accurate results in real time with minimal delay to give immediate feedback to the user.

2.2.5 Security

- The photos/videos that our app captured will not be shared via any third parties by Horus.

2.2.6 Extendibility

- Horus can be extended to involve different types of scans (modes) in the future.

3. Final Architecture and Design Details

class VideoThread(QThread): Facilitates the single screen emotion detection functionality of Horus.

class VideoMultiThread(QThread): Facilitates the multiple screen emotion detection functionality of Horus.

class ScreenCaptureThread(QThread): Facilitates the screen capture functionality of Horus.

class LieDetectionThread(QThread): Facilitates the lie detection functionality of Horus.

class View_Analysis(QThread): Facilitates the data viewing and graphing functionality of Horus.

class DeceptionDetectionVoice(QThread): Facilitates the voice part of the lie detection functionality of Horus.

class Ui_MainWindow(object): Giant part of the GUI and functionalities.

There are some other files that do not have class architecture such as test_faces.py which is used for testing the accuracy of the previous trained model, horus_emo_ferplus.py which is used for testing the accuracy of the trained model, horusEmoReg.py which is used to test without using GUI. Beside these, there are some train files such as Untitled0.ipynb and the train file of the lie detection lie_train.ipynb.

Because we develop a computer vision project, the project is not object oriented and we do not have a comprehensive file architecture. Furthermore, since we have used Python programming language, we did not need structured file layout, we simply put .py files in the src directory.

Finally, for our dependency constraints, Horus can run on Python3.7, it does not support 3.8 and 3.9 right now but we are searching for a way to implement those too [1].

4. Development/Implementation Details

4.1. Emotion Detection

We started to develop our program with the model that we have trained. We first did comprehensive research about face detection, face recognition and emotion recognition models that are available on the internet. At the same time, we searched for datasets for emotions that are big enough to result in a good accuracy. After various attempts, we have found our datasets and developed the model that fits very well. From that point, we tried different hyperparameters to achieve the best accuracy that we could get, which is 82%.

4.1.1. Datasets

We used 2 different datasets for training this model. First dataset is named as FERPlus[2] which is a rearranged version of the FER2013[3] dataset. In this dataset each image has been labeled by 7 different taggers. These are neutral, happy, sad, angry, disgust, fear, surprised. We prefer this dataset due to the better quality ground truth value than the FER2013 dataset.

In this dataset all the image sizes are 48x48 pixel and grayscale. There are 28,709 images for training and validation and there are 3,589 images for testing purposes in this dataset.

The second dataset that we used in train is ExpW[4] (Expression in-the-Wild) dataset. This dataset contains images that are labeled in 7 different categories as in the FERPlus dataset. In this dataset there are 91,793 images. We used a combination of these two datasets and divided them into 3 parts: train, validation and test. Each part contains images in equal percentage from both datasets and it is divided like 80%, 10%, 10%. Image sizes are 48x48x3 pixels and images are RGB..

4.1.2 Model Details

We used a CNN model in combination of two datasets, FerPlus and ExpW. Our training procedure took almost a day sometimes but in the end, we reached our desired goal with 300 epochs. Model details can be seen from Figure 1. Our other hyperparameters are:

- Batch-size: 64
- Learning rate: 0.0005

Furthermore, we take the images of the faces' as an input in the format 48x48x1 to our model. We first detect a face, give it as an input in the format above to the model, and recognize its emotion. Train details. can be seen from Figure 2 and Figure 3

```
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 conv2d (Conv2D)              (None, 48, 48, 64)        640

 batch_normalization (BatchN  (None, 48, 48, 64)       256
 ormalization)

 activation (Activation)      (None, 48, 48, 64)        0

 max_pooling2d (MaxPooling2D  (None, 24, 24, 64)       0
 )

 dropout (Dropout)            (None, 24, 24, 64)        0

 conv2d_1 (Conv2D)            (None, 24, 24, 128)       204928

 batch_normalization_1 (Batc  (None, 24, 24, 128)      512
 hNormalization)

 activation_1 (Activation)    (None, 24, 24, 128)       0

 max_pooling2d_1 (MaxPooling  (None, 12, 12, 128)      0
 2D)
...
Total params: 4,478,727
Trainable params: 4,474,759
Non-trainable params: 3,968
_________________________________________________________________
```

Figure 1: Model summary.

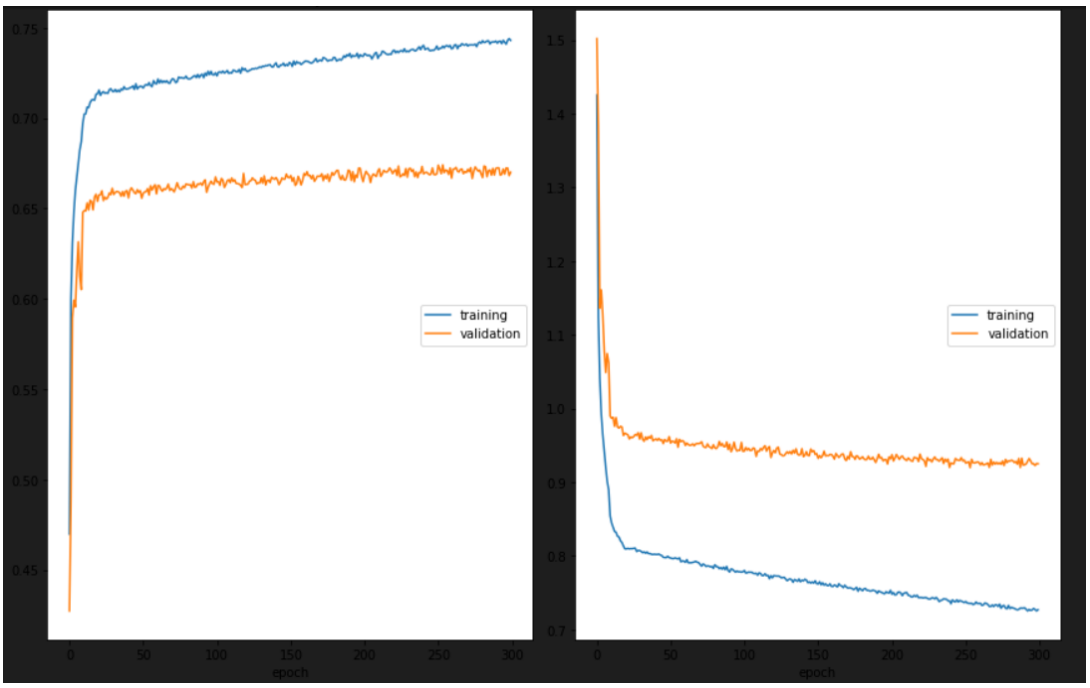

Figure 2: Accuracy/Loss graphs

```
accuracy
        training                (min:    0.470, max:    0.744, cur:    0.743)
        validation              (min:    0.427, max:    0.674, cur:    0.670)
Loss
        training                (min:    0.726, max:    1.425, cur:    0.727)
        validation              (min:    0.920, max:    1.502, cur:    0.925)

Epoch 300: saving model to model_weights.h5
1506/1506 [==============================] - 84s 55ms/step - loss: 0.7272 - accuracy: 0.7432 - val_loss: 0.9252 - val_accuracy: 0.6701 - lr:
1.0000e-05
```

Figure 3: Accuracy and Loss values

After we have finished with our model, we started to implement the interface of the program. We have used PyQt5 as it was the best choice for us to implement a user interface to our python program. We had various GUI interface design frameworks for Python such as PyQt, Kivy, Tkinter etc. To achieve a modern UI interface we decided that the best choice is the PyQt. After we developed the outline of the GUI, we started making links between paces and creating the main part of the program.

With PyQt5, we used QThreads for our functions and class references. And we used signals and slots to forward data between classes and functions. By doing this, we first made the calculations async and we divided all cameras in the system to threads so that. All cameras are able to work together without any waiting issues. Furthermore, this system makes GUI independent from the analysis.

For analysis we first need to detect the faces for this purpose we had many different options such as haarcascade or using a DNN model etc. We continued with the second option and used weights of a DNN model that is trained with CAFFE dataset for detecting the faces. We preferred DNN because it is more suitable for real time analysis. Then, we get the faces and give them to our model's weights, which are trained by us. After that we are able to get all the percentages about labels. We process the label percentage results and after that we convert them to realtime charts. Furthermore, at the same time, we save the processed video. And finally we save all the results, and calculate 4 different charts.

4.2. Deception Detection

We developed deception detection by training several models with an audio dataset and using the emotion detection module to extract micro emotions. Then we combined those to get a deception detector from both video and voice input.

4.2.1. Dataset

We used the Real-life deception detection dataset (2016) which is a multimodal dataset containing 121 videos with their transcripts [6]. The videos depict people talking and labeled as either deceptive or truthful. We extracted the voice of the videos as '.wav' files and used only the voice data to train the models. The dataset includes 61 lie data and 60 truthful data which is extremely balanced. We split the dataset into test and train sets with 3:7 ratio respectively.

4.2.2. Models

We trained 5 different classifiers with our dataset. As data preprocessing we extracted MFCCs (Mel-frequency cepstral coefficients) for each data [7]. Then used those coefficients as features to train the models. Then, we shuffled the training data. We preferred to try different models to see the best working one regarding accuracy, precision, recall and F1 score. We trained the models many times after reshuffling data again and again to observe the average result. The below section demonstrates the average results for each of the classifiers.

4.2.2.1. Support Vector Machines

SVM model generally gave good results when compared to the other models. The accuracy was fine but not the best. The precision was low, however the recall was best amongs the

others. Since the dataset was balanced, we did not consider the F1 score. The scores can be seen in Figure 4.

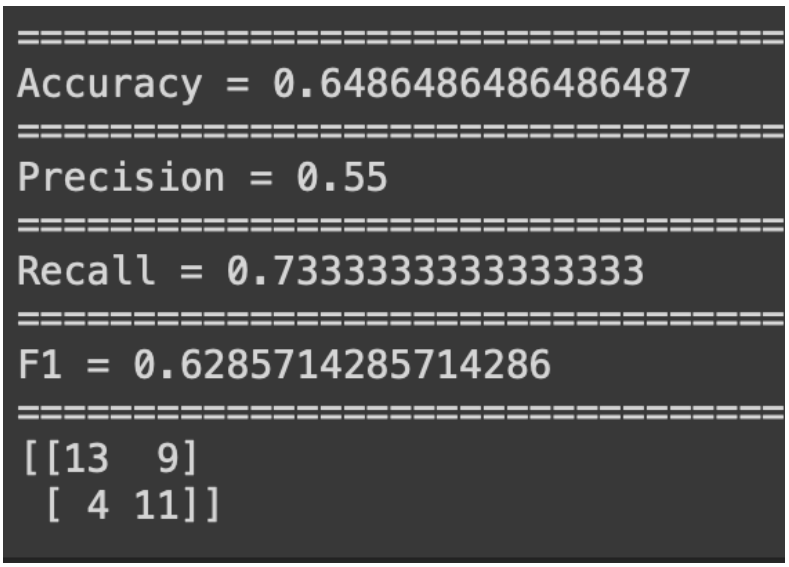

Figure 4: SVM results

4.2.2.2. Logistic Regression

The LR model was the best model with both high precision, accuracy and recall scores. Therefore, at the end we chose to use this model. The scores can be seen in Figure 5.

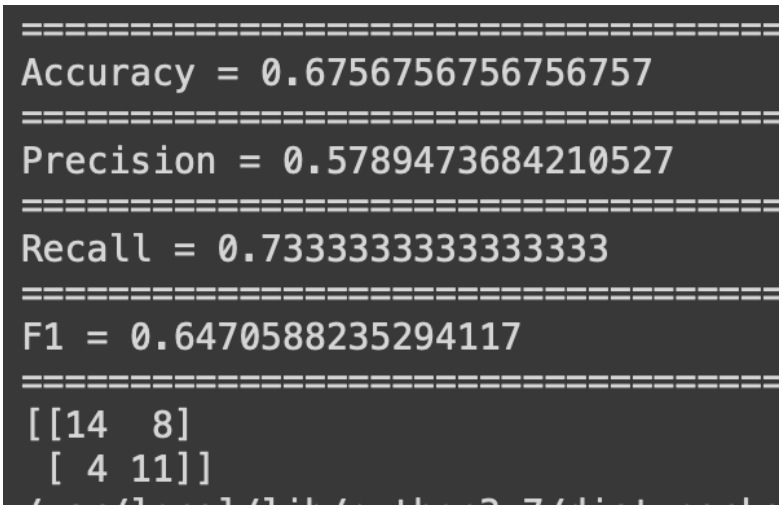

Figure 5: LR values

4.2.2.3. Decision Tree Classifier

The DTC model was one of the worst models that we tried. So we skipped this model but the results can be seen in Figure 6.

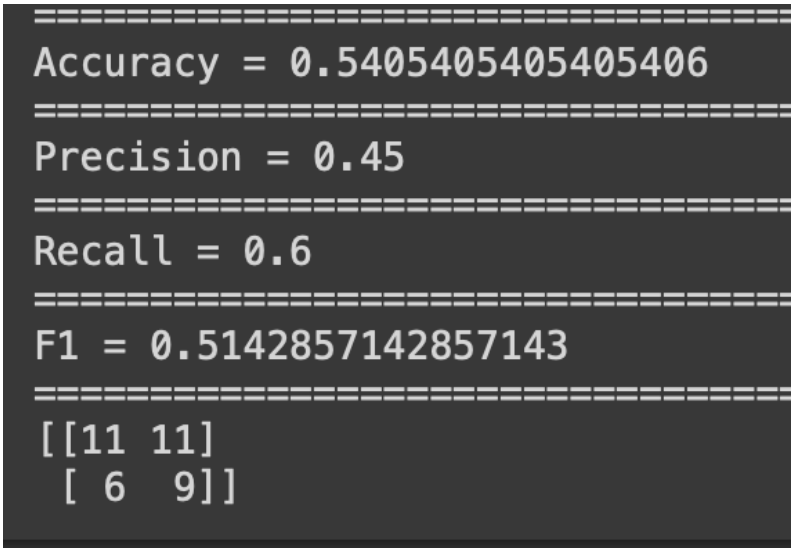

Figure 6: DTC results

4.2.2.4. Stochastic Gradient Descent Classifier

The SGD model had high accuracy but the recall was extremely low. It failed to classify data as lies. The scores can be seen in Figure 7.

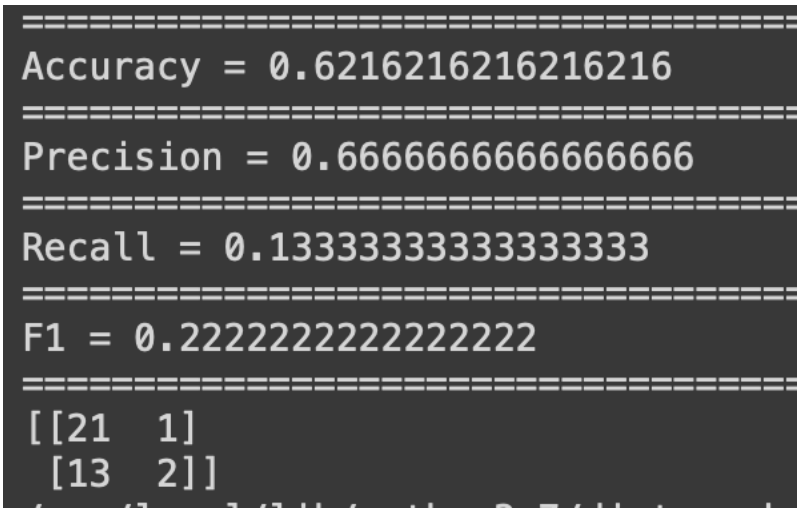

Figure 7: SGD results

4.2.2.5. Neural Network Model

We defined a sequential model as shown in Figure 8.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 20, 16)            65088

 batch_normalization (BatchN  (None, 20, 16)           64
 ormalization)

 lstm_1 (LSTM)               (None, 20, 8)             800

 flatten (Flatten)           (None, 160)               0

 dense (Dense)               (None, 2)                 322

=================================================================
Total params: 66,274
Trainable params: 66,242
Non-trainable params: 32
```

Figure 8: Model summary

The accuracy was the highest among the models. Precision, recall and F1 score was also fine but the model was not stable. Which means after each reshuffling, the scores changed drastically. Only a few times gave the result in Figure 9.

```
==========================================
Accuracy = 0.6756756756756757
==========================================
Precision = 0.6
==========================================
Recall = 0.6
==========================================
F1 = 0.6
==========================================
[[16  6]
 [ 6  9]]
```

Figure 9: Average results

4.2.3. Micro Emotions

When the probability results of the models, which are for lie and for truth, are so similar that we define the difference is smaller than '0.2', then we used micro emotions to classify the data. If the count of micro emotions is more than 8, then we classified it as a lie.

In a video if an emotion is seen less than a predefined time interval, it is regarded as ‘Micro Emotion’ [8]. We defined the threshold as 10 frames. If an emotion is successively seen in less than 10 frames, then we marked it as a micro emotion. We counted the micro emotions afterwards for the input. Before counting the microframes, we applied a 5 point average filter to eliminate the fluctuations in the emotion detector module.

4.3. Mobile Application

We developed the mobile Horus application by using React Native and Javascript on Snack web application using Expo. Then we exported the app and added the bluetooth connection code to connect it to the main app.

5. Testing Details

5.1 Model Testing

Horus’s main functionality relies on the accuracy of its training model. For that reason we tried many different models and did extensive testing of those models to reach as accurate a model as we could. This part was also very time consuming as a single training session would take about a day to complete. In the current model, we have 82% accuracy with our testing set and given more time, that could also be improved. We also tested the accuracy of our model through images that are not inside the dataset.

5.2 Code Testing

Each subsystem was tested extensively before it was integrated with the other subsystems. This ensured we had less bugs as the development of the project went along. The approach was akin to agile development though not all techniques of agile were used.

6. Maintenance Plan and Details

Horus was designed in a way that it does not require any remote server and all the calculations, storing, system functionality in short work on the user’s local computer. Due to this design the maintenance of Horus is very easy. Additionally, Horus will be completely free and can be downloaded from GitHub. So, users can write their feedback or problems on GitHub and we can take them into consideration by doing a meeting on Discord as our primary communication source.

7. Other Project Elements

7.1.Consideration of Various Factors in Engineering Design

While doing a project, consideration of various factors is really important. These various factors are public health, public safety, public welfare, cultural factors, global factors, and social factors respectively. Horus, due to its features, does not have any impact on some factors and has an impact on others. The detailed representation can be seen in Table 1.

- **Public Health:** Public health has got a few effects on the design choices of the project. For instance, mask mandates due to Covid-19 would affect the performance of Horus as it requires open faces to work properly.
- **Public Safety:** Public safety has a prominent effect on the design choices of the project. Horus' design considers whether people's privacy is encroached upon. It takes measures to not single out an individual and only provide depersonalized data.
- **Public Welfare:** Public welfare has not got a significant effect on the design choices of the project. Horus' performance is affected by the quality of the hardware devices, therefore directly related to the budget of users.
- **Cultural Factors:** Cultural factors have a prominent effect on the design choices of the project. Horus is meant for most avenues where audiences are involved; however, cultural venues such as movie theaters and stand-up shows are some of the areas where Horus will be most useful. Horus will therefore be created with such venues in mind.
- **Environmental Factors:** Environmental factors have a considerable amount of effect on the design choices of the project. Bad weather conditions may affect the performance of Horus as substances such as dust may hinder the view of the cameras in open environments. In addition, such conditions may cause damage to cameras and the local hardware if precautions are not taken against it. Light conditions may also affect the performance of Horus as it will get harder to detect faces and their emotions if it is too dark or too bright.
- **Global Factors:** Global factors have some effect on the design choices of the project. Horus aims to reach a large number of users in various countries. For this purpose, Horus' interface will be in English.
- **Social Factors:** Social factors have a crucial effect on the design choices of the project. As digital environments become more prominent in daily social interactions, the need to interpret and project real life actions and emotions into digital media becomes more important. Horus allows the user to interpret such information and provides data that can be

used in digital social hubs. Therefore, it is highly relevant in this area. Horus will also consider issues of inclusivity in its modeling such as providing accurate results for different race groups.

- **Economic Factor:** Economic factors have a prominent effect on the design choices of the project. The quality and the performance of the CPUs and the GPUs will directly affect the developing and improvement processes of the Horus. The higher the performance, the less time the training process of the models will take, therefore there is a direct relation between the money spent on those devices and the developing speed.

Table 1: Factors that can affect analysis and design.

	Effect Level	Effect
Public Health	7	Masks have a negative effect on the accuracy of Horus.
Public Safety	8	Depersonalized data will be prioritized.
Public Welfare	2	Quality according to the users' budget.
Cultural Factors	7	Functionality and user experience will be geared towards cultural venues.
Global Factors	5	Horus should be English.
Social Factors	10	The need to interpret and project real life actions and emotions.
Environmental Factors	7	Light conditions and too many moving particles could affect face and emotion detection packages negatively.
Economic Factors	8	Required time for developers according to the project budget.

7.2.Ethics and Professional Responsibilities

Undoubtedly, as an application that uses visual data and being a computer vision project there are various professional and ethical issues. These issues are prominent for both the development and usage of the application.

We will consider several professional and ethical issues during and after the development process of Horus. One of them is data security. We have to get video input from the user to

do emotion recognition and that video input should not be shared with any kind of third party. Furthermore, we will ensure that the stored video and image data are not leaked. We will also ensure that user data will not be disclosed to anyone unless there is a security concern.

For the sake of professionality, the group meetings will be arranged each week. What has already been done and what will be done until the next meeting will be determined and implemented in the closest time span. The source code of the horus will be private and stored on Github servers. Contribution of the group members will be one of our main concerns.

7.3.Judgements and Impacts to Various Contexts

Global:

The inspiration for our project was in part influenced by the global pandemic that has affected the world for the past 2 years and increased the demand to monitor and process human interactions on camera such as the Zoom meetings that were popular during lockdowns. The data that we provide is specifically tailored towards such high population interactions where understanding the overall sentiment within the room or meeting is useful.

Economic:

Many business decisions require involved parties to gauge the opinions of the others to tailor their offerings or get a grasp of public opinion to determine what kind of product they can sell. This need was another decision factor for us as we designed Horus since we believe the ability to automate such a crucial part of business would be highly desirable and profitable.

As for the economic impact on the user, Horus does have some hardware requirements that the user must purchase. One of these requirements is an adequately powerful pc which vastly differs in price but in Turkey, can start from 10000tl and will likely increase given the volatile inflation and the loss of value of turkish lira against the dollar. Additionally, an array of cameras would need to be purchased, though the number of these cameras and their quality could differ from user to user. For most users a top of the line camera array would

not be needed and they would get adequate performance from what they already have or can purchase for a reasonable price.

Like it was mentioned in the global concerns, the pandemic has also had an effect on the economic concerns of Horus. The world is going through a supply chain crisis right now and computer chips are one of the most affected products. The shortage has impacted the price of these chips by a large margin and thus the economic prospects of Horus rests on a quick resolution of these global disruptions.

Environmental:

Horus relies on computers and cameras to work. The extraction process of the materials used in these products such as rare earth minerals are often environmentally destructive. However, we do not expect Horus to have a significant additional footprint on the environment, given the ubiquity of such products. Furthermore, Horus is meant more towards businesses and not towards the general public, so it will not dramatically increase the demand for computers and cameras.

Societal:

Technologies like facial recognition and emotion recognition are often viewed by the public as intrusive technologies that often violate privacy. While creating Horus, we made sure our impact on people's privacy was kept to a minimum. We do not collect personal information, nor do we record anything in any servers. Horus is also open source. By doing this, we aim to increase the transparency of our program and the trust in it. We also acknowledge other societal factors that are culture specific may be in play such as cultural sensibilities and tolerance of surveillance. We leave these to the user's discretion as we believe such issues can best be served locally. Applying a top down approach would severely limit the scope of Horus for a few edge cases.

7.4 Teamwork Details

7.4.1 Contributing and Functioning Effectively on the Team

We are all aware of the importance of teamwork. And this can be achieved by giving effort to the project equally. To keep track of this we used:

➢ Being active on the project's Discord channel while talking about the Horus.

➢ The succession of given tasks.

➢ To keep track of the stage of the given task through the weekly meetings.

To achieve equal workload tasks are planned according to full-stack development. To provide stability over the progress of tasks, we arrange evenly spaced meetings. In these meetings, we talked about what we did after the last meeting, and what we will do until the next meeting. We also discussed newly or previously thought ideas, and we determine the next tasks and assign them to the group members. We generally made the task sharing according to our previous experiments and volunteering. We also arranged spontaneous meetings if needed in cases such as quick questions of confusion about a particular task.

We maintained the communication through the Discord app via voice communication and screen share functionality. To function effectively most of the time instead of working alone we worked under a voice channel together both to motivate each other and to achieve synchronous communication. Synchronous communication allowed us to avoid waste of time and to minimize the mistakes due to communication errors. By doing this we were able to work a lot more effectively.

7.4.2 Helping Creating a Collaborative and Inclusive Environment

To create a collaborative and inclusive environment first, a big task is divided into smaller multiple tasks. So, every team member took a task according to a plan and gave effort to it. This is done to avoid giving some members a huge workload while giving others less. This way each member gave all their focus to a single or multiple assigned tasks. Since every team member achieved this, also Horus achieved good results and mostly completed in time without any problems. Also, merging these smaller tasks are done carefully and every team member tested whether the tasks they did fit the bigger task nicely and smoothly. Then, those smaller tasks forming a big task are also checked by the team leader of that work package to avoid problems in merging. This double checking minimized the mistakes that can occur.

All of the group members had one or two main tasks and many small tasks. As aforementioned, we shared responsibility roles of the main tasks according to the requirements of the task and previous experiences of the assigned one. On the other hand,

we solved and planned to solve the design-related things, and crucial project decisions together involving opinions of each member.

7.4.3 Taking Lead Role and Sharing Leadership on the Team

Being under the management of a leader is a crucial factor in a team work to avoid misunderstandings and confusion. Moreover it helps the team to be in a more connected and organized state. To avoid overloading all the management jobs to one single person, we preferred to divide the work into teams and assign one leader to each team, thus the workload is divided into members equally.

To share the leadership, we have seven work packages: Face Detection, Emotion Detection, Deception Detection, User Interface, Reports, Mobile Application, and Testing, respectively. We assigned those work packages to different team members. The team members with work packages with lower workloads are given a second work package with again a lower work package resulting. This way some team members got one big work package while some got two work packages smaller than the others. The leaders and members of these work packages can be seen in Table 2.

Table 2: Work Packages and responsibilities of the team members

WP#	WP title	Leader	Members
WP1	Face Detection	Furkan Demir	Asya Doğa Özer, Furkan Demir
WP2	Emotion Detection	Ufuk Palpas	Ufuk Palpas, Muzaffer Köksal
WP3	Deception Detection	Asya Doğa Özer	Can Kılıç, Muzaffer Köksal
WP4	User Interface	Can Kılıç	Asya Doğa Özer, Furkan Demir, Ufuk Palpas
WP5	Reports	Can Kılıç	Can Kılıç, Asya Doğa Özer, Furkan Demir, Ufuk Palpas, Muzaffer Köksal
WP6	Mobile Application	Asya Doğa Özer	Can Kılıç, Asya Doğa Özer
WP7	Testing	Muzaffer Köksal	Furkan Demir, Ufuk Palpas

7.4.4 Meeting objectives

The promised objectives of Horus have mostly been accomplished. In its current situation, Horus is a working PC app that provides its user an user-friendly interface with all the functionality as promised on previous reports. However, we are all aware that there is still some room for new functions we need to implement but those functions will be basically the ideas that come to users' mind which gain shape as the Horus is used.

7.5 New Knowledge Acquired and Applied

Horus implemented and planned in a way that every team member will use his/her strongest skill. The members in order to gain new knowledge mostly used online sources like YouTube and stackoverflow.

We learned:

- ➢ Usage and implementation of PyQt5 that we applied for our interface and connection with the python files.
- ➢ React Native mobile app development that we applied for our mini console on mobile with bluetooth connection.
- ➢ Single and Multi-threaded camera control that we applied for our crowd control and screen capture modes.
- ➢ Training of CNN that we applied for our emotion detection model

8. Conclusion and Future Work

Horus started as an ambitious project that aimed to achieve multiple complex tasks in tandem to provide its users useful emotion data of groups and individuals. We worked hard on its development and are proud of what we have created, however there is always more that we can do to achieve a superior product. We believe with further development Horus could become even more robust, reliable and accessible.

For better, more reliable results, a better model could be trained to improve on the accuracy rate of the algorithm. Our time constraint limited the number of models we could try, especially since the time it takes to train a model was so long. This would in turn result in more accurate crowd emotion data and increase the value proposition of our product.

For a more robust Horus, we could also diversify the data we provide to the user and even create a more interactive experience that allows users to pick and choose the parameters of the data they receive. This would increase the robustness of Horus, since it would be viable for more use cases. This could also be achieved through different versions of Horus that we cater towards a specific market.

Horus' performance could be improved via a combination of multithreading and code refactoring so that it increases the frequency of computations which would both decrease its economical impact on its users(as they would require less powerful hardware) as well as increase its accessibility(as more people would be able to afford hardware that is needed for Horus). With enough resources, Horus' accessibility could be improved even further, by using servers that do the computations for our users instead of them needing computers.

Finally, due to time constraints, we were not able to create a UI that we felt was indicative of the quality of our project. Most of our time went to the backend part of the project as that was the most complex part. In the future, we would like to create better UI and UX.

9. References

[1] *[solved] no module named Numpy in python*. Python Pool. (2021, July 10). Retrieved May 6, 2022, from https://www.pythonpool.com/no-module-named-numpy-solved/

[2] Microsoft, "Microsoft/FERPlus: This is the FER+ new label annotations for the emotion FER dataset.," *GitHub*. [Online]. Available: https://github.com/microsoft/FERPlus. [Accessed: 06-May-2022].

[3] M. Sambare, "Fer-2013," *Kaggle*, 19-Jul-2020. [Online]. Available: https://www.kaggle.com/datasets/msambare/fer2013. [Accessed: 06-May-2022].

[4] Z. Zhang, P. Luo, C.-C. Loy, and X. Tang, "Learning social relation traits from face images," *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015.

[5] *React-native-ble-manager*. npm. (n.d.). Retrieved May 6, 2022, from https://www.npmjs.com/package/react-native-ble-manager

[6] *Downloads / demos: Language information and technologies*. Downloads / Demos | Language Information and Technologies. (n.d.). Retrieved May 6, 2022, from https://lit.eecs.umich.edu/downloads.html#undefined

[7] K. Rao, "Appendix A MFCC Features," doi: 10.1007/978-3-319-49220-9.

[8] "Deception Detection | How to Tell If Someone is Lying," *Paul Ekman Group*. https://www.paulekman.com/deception/deception-detection/