**Detecting Human Emotion**

|  |
| --- |
| A Project Report  Presented to  The Faculty of the Computer Engineering Department |
| San Jose State University  In Partial Fulfillment  Of the Requirements for the Degree  Bachelor of Science in Software Engineering |

|  |
| --- |
| By |
| **Luis Arevalo**  **Avani Bhatnagar**  **Tyler Bruno**  **Francisco Mera**  **Cindy Yee** |
| 12/2018 |

|  |
| --- |
| Copyright © 2018 |
| **Luis Arevalo**  **Avani Bhatnagar**  **Tyler Bruno**  **Francisco Mera**  **Cindy Yee** |
| ALL RIGHTS RESERVED |

|  |
| --- |
| **APPROVED FOR THE COLLEGE OF ENGINEERING** |
|  |
| Professor Jerry Gao, Project Advisor |
|  |
| Professor Rod Fatoohi, Instructor |
|  |
| Dr. Xiao Su, Computer Engineering Department Chair |

**ABSTRACT**

**Detecting Human Emotion**

By Luis Arevalo, Avani Bhatnagar, Tyler Bruno, Francisco Mera, Cindy Yee

Deception detection is important for police investigations, court cases, and many other applications where deception detection is imperative. Polygraph tests are currently used in industry for deception detection purposes. Traditional deception detection techniques monitor the subject’s respiratory rate, pulse, blood pressure, and perspiration (Bellis, 2017). John Larson, a student who worked for the Berkeley police department, invented the first polygraph 97 years ago (Bellis, 2017). Meijer and Verschuere (2010), stated the following polygraph accuracy statistics: 74% to 89% for guilty examinees, with 1% to 13% false negatives, and 59% to 83% for innocent examinees, with a false positive ratio varying from 10% to 23%. With training, a person could pass a polygraph test. The need for an improved way to analyze a subject truthfulness is critical in the society we live in today.

**Acknowledgements**

We would like to express our sincere gratitude to our advisor Dr. Jerry Gao for his continuous support and guidance. He guided us throughout the development of the project.

We would also like to express our gratitude to sergeant deputy John Hallman at the Napa county sheriff’s department for providing us with invaluable advice for the requirements of the project.

**Table of Contents**

**Chapter 1 Introduction……………………………………………………………………..9**

1.1 Project Goals and Objectives

1.2 Problem and Motivation

1.3 Project Application and Impact

1.4 Project Results and Deliverables

1.5 Project Report Structure

**Chapter 2 Background and Related Work……………………………………………….11**

2.1 Background and Used Technologies

2.2 Literature Survey

2.3 State-of-the-art Summary

**Chapter 3 Project Requirements…………………………………………………………16**

3.1 Domain and Business Requirements

3.2 System (or Component) Functional Requirements

3.3 Non-functional Requirements

3.4 Context and Interface Requirements

3.5 Technology and Resource Requirements

**Chapter 4 System Design………………………………………………………………...20**

4.1 Architecture Design

4.2 Interface and Component Design

4.3 Structure and Logic Design

4.4 Design Constraints, Problems, Trade-offs, and Solutions

**Chapter 5 System Implementation……………………………………………………..25**

5.1 Implementation Overview

5.2 Implementation of Developed Solutions

5.3 Implementation Problems, Challenges, and Lessons Learned

**Chapter 6 Tools and Standards………………………………………………………...29**

6.1 Tools Used

6.2 Standards

**Chapter 7 Testing and Experiment…………………………………………………….30**

7.1 Testing and Experiment Scope

7.2 Testing and Experiment Approach

7.3 Testing and Experiment Results and Analysis

**Chapter 8 Conclusion and Future Work……………………………………………...33**

8.1 Conclusion

8.2 Future Work

**List of Figures**

Figure 1. Emotion detecting general architecture................................................................... 18

Figure 2. Deep Learning General Architecture …...…........................................................... 19 Figure 3. Component diagram of project ….………….……………………………………..20

Figure 4. Lie Detection Algorithm Logic and Structure...…………………………………...21

Figure 5. User Sequence Diagram…………………………………………………………...22

Figure 6. Hand to Face touch detection ………………………………………………...…...23

**List of Tables**

Table 1. Prior helpful classes..............................................................................................4,5

Table 2. Lie Detection Products Comparison List..............................................................6,7

* 1. Table 3. User stories with functional requirements……………………….………....…...9,10

Table 4. Hardware Requirements.......................................................................................11

Table 5. Software Requirements……………………………………………….………...11

# 

# 

# 

# 

# 

# 

# **Chapter 1. Introduction**

**1.1 Project Goals and Objectives**

The main goal of our project was for interviewers to detect when a subject is being deceitful in real-time and with recorded video and audio. One of our main objectives was to create deception detection software which is more portable, accurate, and easy to use than a conventional polygraph machine. Our software was designed for law enforcement officers to detect if a person being deceitful during or after an interrogation or interview. Our product uses both video and voice analysis to detect deception in a person’s behavior or speech. The current industry standard has a 60% accuracy rating for deception detection (Masip, 2017).

**1.2 Problem and Motivation**

Polygraph machines are expensive, cumbersome, and require professional training. In the law enforcement industry, departments often contract polygraph technicians due to these factors (Hallman, 2018). Since our software is not hardware dependent, the users can run our software on most modern computers. This eliminates the need for users to purchase specialised hardware for deception detection. Our software has also been designed to eliminate the need for intrusive sensors to be placed on the person being questioned. This allows for our deception detection software to be more mobile than Polygraph machines. With our software foundation law enforcement officers can take recorded audio from their body cameras and drop the audio file into our deception detection website.

**1.3 Project Application and Impact**

Our software is affordable to the common household, and more approachable than a polygraph. It is also simple to use compared to a polygraph. The user can simply use a computer with its built-in camera and microphone to run our detection software. This software is an affordable solution that can be used by law enforcement agencies around the world.

**Global and Industry Impacts:**

Since people lie 20% - 30% of the time, our application will have a universal impact on criminal justice (Boltz, Dyer, and Miller, 2010). Our application can be used for investigation and interviews to detect when a suspect is lying.

We developed our project to resemble a polygraph test which is widely used in different areas such as homicide investigations, court cases, etc. Our software is simple to use, therefore, eliminating the need for expensive training or contracted trained polygraph technicians. This will allow smaller agencies to adopt an easy to use deception detection that will help reduce crime all over the globe.

**Societal Impacts:**

The major impact our project would have of society is the lie detection field. As mentioned in our abstract, Meijer and Verschuere (2010), state these polygraph accuracy statistics: 74% to 89% for guilty examinees, with 1% to 13% false negatives, and 59% to 83% for innocent examinees, with a false positive ratio varying from 10% to 23%. Our software could potentially help prevent innocent suspects from being convicted. Our software will be more accessible to smaller law enforcement agencies and accessible throughout the world.

**Environmental Impacts:**

Our software has very little environmental impact in comparison to a polygraph machine. Furthermore, it is intended to be used with any modern day laptop outperforming the little energy used to produced in comparison to polygraph machine.

**1.4 Project Results and Deliverables**

We have done a considerable amount of research in the area of using machine learning and deep learning for deception detection. We had two teams working concurrently on the project, one was the audio team and the other was the visual team. Both teams developed different deception detection methods using the lie detecting research we initially gathered.

**1.5 Project Report Structure**

In this Report, we start by introducing the project which was done in this section. The second section is related to the background and literature of our project. It lists the background of our project including the technologies that we used and all the literature and state of art summary which is majority of the research done on this project. The third section lists the various project requirements such as domain and business requirements, system functional requirements, non-functional requirements, context and interface requirements, and technology and resource requirements. The fourth section briefly discusses the system design specifications. The design specifications include architecture, structure, constraints, logic design, interface and component design, as well as problems and solutions. The fifth section lists the system implementation details. These details include implementation overview, implementation of developed solutions, implementation problems, as well as challenges and lesson learned from overall implementation of the project. The sixth section lists the tools and standards. It specifies the selected hardware/software and where and how they are used. It also describes the standards used in our project related to requirements, interface, hardware and software, etc. The seventh section is related to the testing and experiment. It describes the scope, approach, result and analysis of the testing and experiment. The final section provides a conclusion of the project and discusses any future work related to this project.

**Chapter 2 Background and Related Work**

**2.1 Background and Used Technologies**

We first researched machine learning and deep learning projects to help give us the experience we need in order to reach our goal. This research helped us decide on the optimal algorithm to use for accurate results. We decided to use blinking and hand touching to detect lying at a subconscious level. For sound, we used emotion along with voice fluctuations and voice frequencies for deception detection.

On the audio team, we used PyAudioAnalysis with Paura2 to train our audio deception detection model. PyAudioAnalysis and Paura2 are Python libraries, created by Dr. Theodoros Giannakopoulos, director of Behavior Signals. PyAudioAnalysis is a machine learning tool built to analyze .wav audio files. We used PyAudioAnalysis to train and classify our model. Paura2 is used to record real-time audio.

On the visual team we detected real-time blinking using OpenCV, Python and dlib. We used the algorithm derived from the 2016 paper by Soukupova and Cech. We used the Rosebrock algorithm to measure the Eye Aspect ratio. The algorithm uses the Eye Aspect Ratio defined in Section 4.3 to detect if the eye is closed or open. This ratio remains constant when the eye is open, and suddenly drops to value close to 0 when the eye is closed (Rosebrock, 2017).

In addition to detecting real-time blinking visually, we detected the frequency of how often hands are touching the face. We used OpenCV, Python, and Tensorflow in addition to the dlib predictor. We started with the tensorflow model created by Dibia(2017), and then modified the script so that the face detected from the dlib predictor would be extracted. If the hand detected from the tensorflow model is near the face, derived by whether the bounding box surrounding the hands was found within any of the 7 rectangles of the face shown in figure 6, then the script would return a “hand touching face” signal.

We used a multithreaded process to combine hand and blink detection for deception analysis. The developed script gives an acceptable truthful threshold of blinks and hand gestures. The result will run a comparison and decide if the answer was a lie.

We used Python with Flask to build a website for our users. Front end software: HTML, CSS, JavaScript, and Bootstrap.

|  |  |  |
| --- | --- | --- |
| **Course Taken** | **Description** | **How it helped us** |
| CMPE 187 | Software Quality Engineering | Help in testing features |
| CS 146 | Data Structures and Algorithms | Help in implementing algorithms |
| CS 157A | Databases | Help give an idea of how to use databases |
| CS 151 | Object-Oriented Design | Helped us to use GitHub correctly. Also helped us generate a design for the project |
| CMPE 131/133 | Software Engineering | Helped us apply Software Engineering skills and helped us develop the web app for the project |
| CMPE 165 | SE Project Management | Helped us acquaint with the software development cycles and choose an effective Agile methodology. |

Table 1 - Prior helpful classes

**2.2 Literature Search**

In order for a machine to detect deceptive behavior, the people training the machine must first understand how others detect such behavior so, we conducted a thorough research before moving ahead with the implementation. According to Borza(2017), Deshmukh(2017), and Masip(2017), the most common ways to detect human lies include observing someone’s eyes for blinking, gazing, and reading behaviors, observing the face for microexpressions, observing their voice, and observing hand gestures that could represent nervous ticks like constantly touching their nose, hand movements that are not coordinated with the rest of the body, etc. Traditionally, polygraphs which have sensors connected to the person’s body are used to detect lies. There are two types of polygraph analysis that are widely used, the Concealed Information Test(CIT) and the Comparison Question Test(CQT), however both of these have countermeasures that the suspect can take to avoid exciting the machines. Polygraph machines record changes in emotions which could be due to several reasons which leads to some false positives. In addition, the 80% accuracy ratings allow for too much room for error, so they are not permissible to use in court, as found in Vriji(2016).

As an alternative to the polygraph machines, research has been developed for using machine learning into expression and facial emotion recognition by researchers such as Deshmukh(2017), Kundu(2017), Rutkin(2015), and Ugail(2011), voice analysis (Nasri,2016), and gesture detection (Neverova,2015), allowing for development into lie detection products (Nasri,2016) and (Rutkin,2015). Such research requires large amounts of data, whether they are images, audio or video input, both, or in any combination. The features would then be extracted, allowing for the algorithm to analyze the connections between the datasets and the model.

**2.3 State-of-the-art Summary**

At this time, there exists several products and services which are sold as-is with lie detection embedded without using the traditional polygraph. They might have different types of analysis such as voice stress analysis, video analysis, or by using an MRI. Table two depicts the differences in the different features that the different products and services currently available in the market provide.

Companies such as X13-VSA and eXsense have developed real-time voice analysis for lie detection based on the tone, pitch, and frequency received. WebScience has also developed a polygraph with software that can be used to further analyze the readings received from several sensors. From analyzing people’s eye for movement like pupil dilation and blinking time, Converus also sells software and hardware equipment for lie detection.

Currently, two of the most researched areas of emotion detection include the use of artificial neural networks and of the Support Vector Machine(SVM) classification algorithms. From Kundu(2017), features would be extracted from images or other input devices and the data would be fed into the neural networks in order to discover connections and decide on an answer.

There are also products that are in development such as the Face Reading Technology by Ugail(2011) from the University of Bradford. They focus on micro expressions and thermal imaging on the face. They assume that the face has seven main facial expressions and that when there is deceptive behavior, there is increased brain activity and the face would heat up. Dartmouth is also in the process of developing a lie detection engine DART(Wu,2018) that takes in audio and visual input. They have been training their model using videos from court cases with known truths and lies.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Type | Voice Stress Analysis (VSA) | | | | | Polygraph | | | | Eye | MRI |
| Brand | X13-VSA | | | eXsense | | Web Science | | |  | Converus |  |
| Model | Home | Pro 3.0 | Cobra 3.6 | eX-sense Lie Detector | eX-Sense PRO | Polygraph Basic | Polygraph Expert | Polygraph Expert+ | The Home Lie Detector | EyeDetect | No Lie MRI |
| **OS** |  |  |  |  |  |  |  |  |  |  |  |
| Windows | Y | Y | Y | Y | Y | Y | Y | Y | Y | Y |  |
| Mac |  |  |  |  |  |  |  |  |  |  |  |
| iOS |  |  |  |  |  |  |  |  |  | Y |  |
| Android | Y | Y | Y |  |  |  |  |  |  | Y |  |
| Phone |  |  |  | Y | Y |  |  |  |  |  |  |
| **Time** |  |  |  |  |  |  |  |  |  |  |  |
| Real-Time | Y | Y | Y |  | Y |  |  |  | Y |  |  |
| Recorded | Y | Y | Y | Y | Y |  |  |  |  |  | Y |
| **Observing** |  |  |  |  |  |  |  |  |  |  |  |
| Voice frequency | Y | Y | Y | Y | Y |  |  |  |  |  |  |
| Sentence Structure |  |  |  | Y | Y |  |  |  |  |  |  |
| GSR Sensor |  |  |  |  |  | Y | Y | Y |  |  |  |
| BVP heart rate Sensor |  |  |  |  |  | Y | Y | Y |  |  |  |
| Abdominal Breathing Sensor |  |  |  |  |  |  | Y | Y |  |  |  |
| Temperature Sensor |  |  |  |  |  |  | Y | Y |  |  |  |
| Chest Breathing Sensor |  |  |  |  |  |  | Y | Y |  |  |  |
| Air pressure sensor |  |  |  |  |  | Optional | Optional | Optional |  |  |  |
| Eye Diameter |  |  |  |  |  |  |  |  |  | Y |  |
| Eye Movement |  |  |  |  |  |  |  |  |  | Y |  |
| Reading behavior |  |  |  |  |  |  |  |  |  | Y |  |
| blinks |  |  |  |  |  |  |  |  |  | Y |  |
| fixations |  |  |  |  |  |  |  |  |  |  |  |
| central nervous system |  |  |  |  |  |  |  |  |  |  | Y |
| pulse |  |  |  |  |  |  |  |  | Y |  |  |
| **Analysis Tools** |  |  |  |  |  |  |  |  |  |  |  |
| Classification |  | Y | Y |  |  |  |  |  |  |  |  |
| Frequency Graph |  |  | Y |  |  |  |  |  | Y |  |  |
| Intelligent Voice Stress Analysis | Y | Y | Y |  |  |  |  |  |  |  |  |
| Voice Polygraph |  |  |  |  |  | Y | Y | Y |  |  |  |
| **Question Type** |  |  |  |  |  | Y | Y | Y |  |  |  |
| Question Analysis |  |  |  | Y | Y |  |  |  |  | Y |  |
| T/F Questions |  |  |  | Y | Y |  |  |  |  |  |  |
| PRICE | $300 |  |  | $120 | $800 |  |  |  | $400 |  | $1000\* |

Table 2. Lie Detection Products Comparison List

From table two, it can be observed that there are a variety of products in the market that would detect lies, however there are no available products that would analyze visual data such as pupil distance and facial emotion, or audio data such as frequency and pitch at the same time. The cost of those technologies are also on the higher end, making them a less affordable solution.

**Chapter 3 Project Requirements**

**3.1 Domain and Business Requirements**

The domain requirements for our application could be used in justice institutions such as police offices, courts, or any other private detective services, and any other AI or ML related industries. If tested and proven to be successful, it can be used in place of polygraph machines as a cheaper, more accurate and more user-friendly alternative. It can also be used along with a polygraph machine to increase the overall accuracy in cases where there are enough resources.

Business requirements involve the features of the application and the project which will be made available to the users. Our project can detect human emotion from video and audio recordings. Our project can also detect human emotion from real-time video and audio. Our project should also be able to detect lies from reading people’s emotions and analyzing their tone, and any changes in their voice.

**3.2 System (or Component) Functional Requirements**

1. User shall be able to detect a person’s emotions using audio
2. User shall be able to detect whether a person is lying using combination of audio and video.
3. User shall be able to detect whether a person is lying in real-time.

**3.3 Non-functional Requirements**

1. Be able to see the likelihood of lying in percentage.
2. Be able to use it along with a polygraph to improve overall accuracy.
3. Accuracy at least equal to a polygraph machine
4. The GUI should be user-friendly and interactive
5. Results of the lie detection should take less than 2 minutes.
6. Deception Detection should be able to be used in real-time
7. Emotion Detection should be in real-time

## When creating the requirements for our project we consulted a sergeant deputy John Hallman at the Napa county sheriff department. He was a correctional officer for 2 years. He has been a sheriff deputy for 20+ years. He spent 10 years in SWAT, 12 years in gang investigation unit, 6 years as a computer forensic investigator, and 7 years as a detective. We created a comprehensive list of user stories with sergeant deputy Hallman’s help during the requirement elicitation process. Our first functional requirement was to detect a person’s emotion using facial and auditory cues. Detecting human emotion is the foundation for lie detecting. Detectives routinely try to elicit emotional responses to detect if a suspect is being deceptive or not. Emotion detection was our first priority because it was easier to implement than deception detection. Once we detected emotion, we focused on deception detection through facial and auditory cues. We found several datasets with multiple real life court testimonies, and other videos that have been categorized into lies and truths. We used this dataset to train our machine.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| User Stories With Functional Requirements | | | | | | |
| **ID** | **User story description (who, what, why)** | **Priority (1 to N)** | **Functional or Non-Functional (Feature)** | **Depen-dency** | **Story Points** | **Criticality** |
| 1 | Emotion Detection |  |  |  |  |  |
| 1.1 | As a user, I would like to be able to detect a person's emotions via video recording. | 1 | Functional | 0 | 8 | O |
| 1.2 | As a user, I would like to be able to detect a person's emotions via recorded audio. | 2 | Functional | 0 | 13 | O |
| 1.3 | As a user, I would like the emotions of the suspect displayed at specific points of the video recordings. | 9 | Non-Functional | 1.1, 1.2, 4.3 | 13 | D |
| 2 | Lie Detection |  |  |  |  |  |
| 2.1 | As a user, I would like to be able to detect if a person is lying via video recording. | 3 | Functional | 1.1 | 21 | MMF |
| 2.2 | As a user, I would like to be able to detect a person's lying via audio recording. | 4 | Functional | 1.2 | 21 | MMF |
| 2.3 | As a user, I would like to see the likelihood (%) of the suspect lying on the application. | 8 | Non-Functional | 1.3, 1.4 | 13 | MMF |
| 3 | Lie Detection From a device |  |  |  |  |  |
| 3.1 | As a user, I would like to be able to detect if someone is telling a lie from recordings taken from a body camera | 10 | Functional | 1.3,1.4 | 21 | O |
| 3.2 | As a user, I would like to be able to detect a person lying from cell phone audio. | 11 | Functional | 1.4 | 34 | O |
| 3.3 | As a user, I would like to be able to detect if a person is lying via cell phone video | 12 | Functional | 1.3 | 21 | O |
| 3.4 | As a user, I would like to be able to upload my video or audio recordings from a mobile device or computer so that I can detect if someone is lying from anywhere | 17 | Functional | 1.3, 1.4 | 21 | D |
| 3.5 | As a user, I would like to be able to detect lies in real-time by using a device like google glass | 13 | Functional | 1.3, 1.4 | 55 | O |
| 3.6 | As a user, I would like to see the likelihood (%) of the suspect lying on the application. | 14 | Non-Functional | 1.3, 1.4 | 13 | O |
| 4 | Misc |  |  |  |  |  |
| 4.1 | As a user, I would like to use this software in conjunction with a polygraph so that I am able to have 2 ways to confirm if the suspect is telling a truth or a lie. | 5 | Non-Functional | 1.3,1.4 | 21 | D |
| 4.2 | As a user, I would like to have the video lie detection at least as accurate as a polygraph machine | 6 | Non-Functional | 1.1,1.2,1.3,1.4 | 55 | O |
| 4.3 | As a user, I would like the software to have a nice GUI that is easy to learn so that my whole team doesn't need any specialized training | 7 | Non-Functional | 0 | 34 | O |
| 4.4 | As a user, I would like to be able to see if someone is lying in real-time so that I can ask better follow up questions. | 15 | Non-Functional | 1.3, 1.4 | 34 | O |
| 4.5 | As a user, I would like to see the results of lie and emotion detection from the video records to take no longer than 2 minutes. | 16 | Non-Functional | 1.3, 1.4 | 13 | MMF |

* 1. Table 3. User stories with functional requirements\*

**\* Note: Emotion Detection should be implemented first to help with lie detection**.

User**:** Anyone using our application to detect lies

Suspect: A person who is not necessarily guilty but who is being questioned.

MMF: Minimum marketable Feature (High priority)

D: Desirable (Medium Priority)

**3.4 Context and Interface Requirements**

We used PyCharm IDE for development purposes and [github](https://github.com/DetectingHumanEmotion/detecting-human-emotion-webapp) to share our code. We used pipenv tool to create and manage a virtual environment for our project.

We used Flask as the web framework. We chose Flask since it is a Python based framework and hence we were able to integrate it with our code easily since major parts of our project are coded in Python. It also allows the development of simple pages which we can use for the frontend of our web application.

Our product includes a microphone and a camera as the visual and auditory interaction elements. These have been used to take the input which is then be processed using the different parts of our code, and all that is eventually tied together using a detection algorithm.

**3.5 Technology and Resource Requirements**

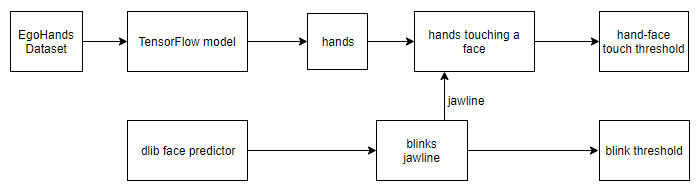


Figure 1. Software Architecture stack for visual deception detection

|  |  |
| --- | --- |
| **Device** | **Details** |
| PC / Mac | Works on both Operating Systems if installed and set up correctly |
| Microphone | Built-in mic of the computer or any external mic attached to the computer |
| Camera | Built-in webcam of the PC or any external camera attached to the computer |

# Table 4 - Hardware Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| **Technology** | **Version** | **Source** | **License** |
| **Python** | 3.6 |  | Open Source |
| **PyAudioAnalysis** | 0.2.5 | Theodoros Giannakopoulos | Open Source |
| **Paura2** | 0.1 | Theodoros Giannakopoulos | Open Source |
| **Scikit-learn** | 0.19.2 |  | BSD License |
| **Dlib** | 19.15 | Davis E King | Boost |
| **OpenCV** | 3.4 | Intel | Open Source BSD |

# Table 5 - Software Requirements

# 

# 

# 

# 

# 

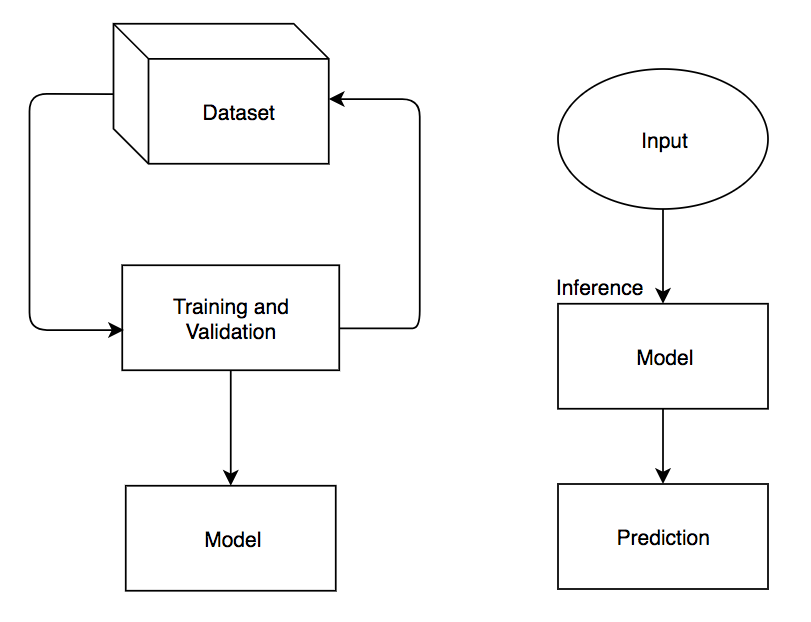
# 

# 

# 

# **Chapter 4 System Design**

**4.1 Architecture Design**



*Figure 2.* Deep learning general architecture

Figure 2 is a general and a very high-level architecture design for our trained prediction models. Hardware and software in addition to the models are all equally important for our emotion detecting machine. To accomplish our goal in building an emotion detecting machine, we will use a laptop webcam and microphone for visual and audio capture purposes. Figure 4 gives a more detailed description of our system.

**4.2 Interface and Component Design**

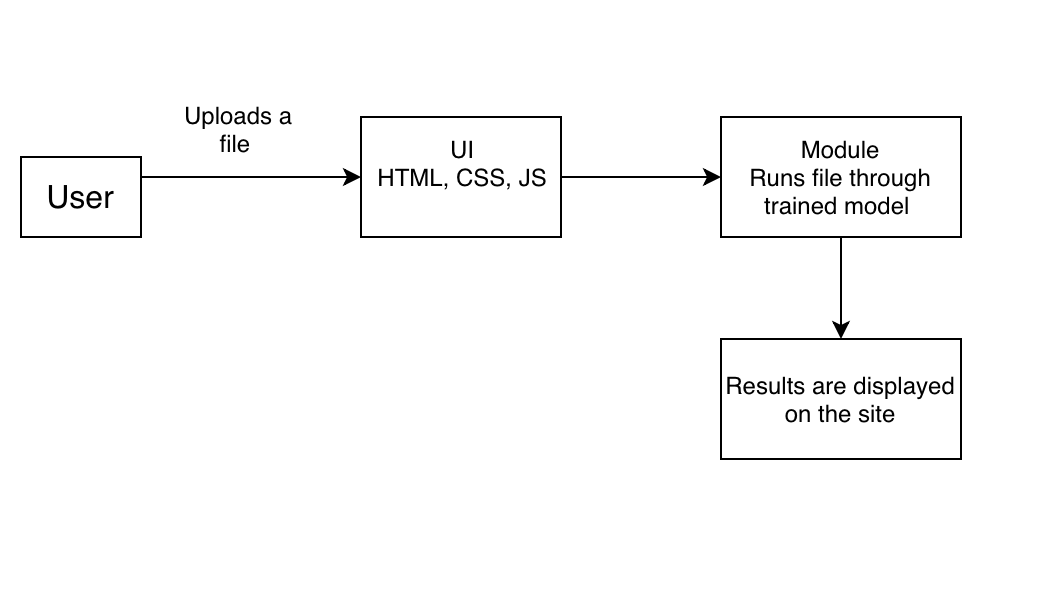
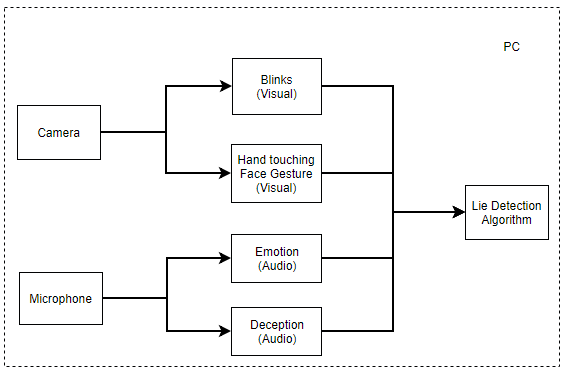
****

Figure 3 - Component diagram of project

Figure 3 shows the component diagram of our project. Users interact with the UI of the system which is composed of HTML, CSS and JavaScript. Users will interact with this by uploading an audio file. The file is then run through our model and analyzed for deception. The results are then displayed on the site for analyses by the interviewer.

**4.3 Structure and Logic Design**

****Figure 4 - Lie Detection Algorithm Logic and Structure

As shown in figure four, the lie detection algorithm will consist of one camera input and a microphone from a laptop. We will then run a python script to process the video and audio footage to determine the number of times the person in question blinked, and how often they were touching their face. These results will further be added to the lie detection algorithm. A comprehensive result status, such as “Lie” or “Truth” message.

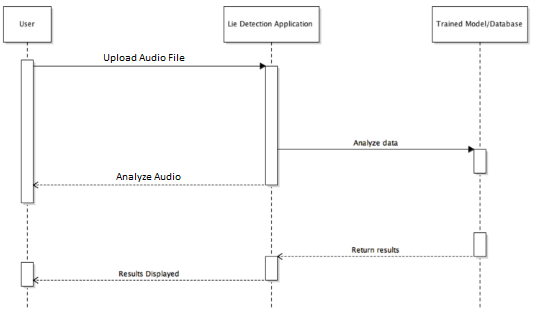


Figure 5. User Sequence Diagram

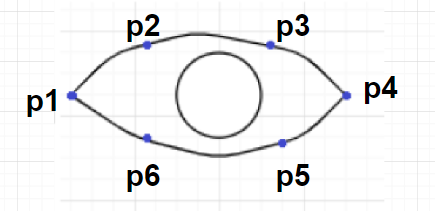


Figure 5. Eye associated landmarks

Figure 5 shows the facial landmarks that are associated with the eye which helped us detect whether the eye was closed or open.

An equation derived from Soukupova and Cech’s 2016 paper shows how these landmarks can be used to find a very interesting and helpful ratio called the eye aspect ratio (Rosebrock).

The eye aspect ratio (EAR) =

where p, p2, … p6 are the landmarks from Figure 5.

When the eye is open, this ratio stays constant, but whenever it shuts or blinks this ratio suddenly drops to 0. This is very helpful since now we can easily use this equation discovered in the 2016 paper to detect whether a person is blinking or not.

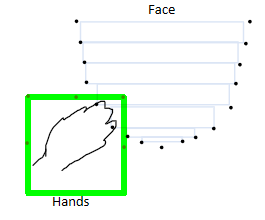


Figure 6. Hand to face touch detection

According to the study by Grunwald, Weiss, Mueller, and Rall, when someone touches their own face, it is an indicator that the person is most likely feeling stressed out. Touching one’s own face increases their emotional and cognitive load. The increased activity can be attributed to possible deceptive behavior. The frequency and duration of hands touching a face depicts whether a person is relaxed or agitated (Jensen & Kruse 2005).

**4.4 Design Constraints, Problems, Trade-offs, and Solutions**

**4.4.1 Design Constraints and Challenges**

Economic

Since we are college students we face some economic constraints. We don’t have the finances to afford state of the art hardware and devices. So, we are limited to cheaper alternatives like web cameras and computer microphones.

Resources

Since our project is a relatively new field. There aren’t many tools available for us to use. Some require a paid subscription or have very little to offer. When researching we found small pieces of the entire project so we are building most of it from scratch. Therefore, we have broken up the group into smaller groups and assigned individual tasks. Also the constraint of time, conflicting class assignments, and other projects.

Hardware/Software Constraints

We are limited to a small budget as we don’t have unlimited finances to buy the latest hardware and devices. We will also be faced with delays as on our computers are limited on how fast they display live video.

Reliability

Our group wants to aim for a high accuracy as we want our software to detect lying better than humans. However, we face limited datasets as there are very few available. Some are private and require payment in order to access them. Hence, it is difficult to find the appropriate datasets which would ensure that the system is accurate and reliable

**4.4.2 Design Solutions and Trade-offs**

For trade-offs we are limited to finances so were using web cameras, and computers microphones. We completed training our models on a PC with a GPU. We ran extensive testing to make our results as accurate as possible even with limited hardware.

Detection deception at 100% accuracy is very difficult. There are several ways to detect if someone is lying and these methods are not 100% guaranteed to catch a liar. So, we came up with some general techniques to cluster deceptive actions to improve our deception detection. For example if someone nods no but says yes then their answer isn’t honest. Using techniques like these we hope to improve our general accuracy to beat the average human at detecting lies.

For reliability it was difficult to find datasets that were accurate. Some had only a few clippings while others weren’t that great. A couple datasets were paid and we decided not to invest into the paid ones. Working around we use the datasets we have found and created some of our own to train our models.

**Chapter 5 System Implementation**

**5.1 Implementation Overview**

**Platforms:**

Our application runs on Windows 10, Mac OS, and Linux.

**Language used:**

Python is the main coding language used to detect emotion and deception from video and audio. We used Pipenv to package our dependencies when creating a virtual environment. Pipenv allowed us to clone the repo and quickly install all the required libraries in a more controlled environment. HTML, CSS, Javascript and Bootstrap with the Flask framework was used to create the web application.

**Implementation Dependencies:**

**Python Libraries**

Python3: Our project was developed using python3.6.6. Our project was not designed to work on python versions < 3.4.

Flask: Python web framework

PyAudioAnalysis: It is a Python library which is used for tasks such as audio feature, audio extraction, classification and segmentation. We used it for the audio part of emotion detection.

Paura2: It is a Python tool which records audio and analyze its content. It was built by the developers from PyAudioAnalysis.

Scikit-learn: Machine learning Python library. Version must be 0.19.2.

Pyaudio: provides port bindings for Port audio

Opencv-python: Open source python library for computer vision

Dlib and imutils: It is a toolkit which helps to make real-world machine learning applications. We used this to detect blinks.

**Environment Dependencies**

Portaudio: Open source cross-platform audio I/O. Paura2 library depends on this library.

**5.2 Implementation of Developed Solutions**

There are 2 main facets of our deception detection software. The first facet is using a person’s facial expressions to detect if a person is being deceptive or not. To accomplish this, we use machine learning to detect the face and eyes (Bartlett, Littlewort,2004). Through the magic of machine learning, we can count the number of times that a person blinks or touches his or her face (Rosebrock, 2017). From the research that we conducted, we can predict if a person is being deceptive or not by the number of times a person blinks or touches his or her face within a specified amount of time(Soukupova 2016, Grunwald 2014). A person that is telling a lie will blink and/or touch his face more frequently than if he was telling the truth (Rutkin 2015, EyeDetect, NoLieMRI). Blinking detection was accomplished using OpenCV, Python and dlib. Dlib is a very useful library which helped us use machine learning in this part of our system. The code basically uses dlib library to get the facial landmarks associated with the eye - p1 to p6 as seen in figure 5. These are used to calculate a EAR (Eye Aspect Ratio) which is close to a constant value when the eyes are open, but goes down to 0 when eyes shut and hence detect blinking.

For the audio deception detection portion, we use a more typical machine learning approach. A typical machine learning problem is to classifying pictures. For example, if we have a picture of an Iris flower and we want to know if it is an Iris Setosa or an Iris Virginica machine learning can easily solve this problem. If we were going to classify the picture using the typical programmatic way, we would have to evaluate the pixels and try to distinguish between petal length, width, height, color, etc. This method would need a comprehensive amount of “if” statements. This approach would be very difficult to program because of all the corner cases the programmer would have to account for. The machine learning approach would be much easier to implement. To solve this problem using machine learning, the programmer would have to have two datasets. The first dataset would be several pictures of the Iris Setosa and the second dataset would be several pictures of the Iris Virginica. A machine learning algorithm would be used to train a model of these two different datasets. This trained model will be able to see the differences between the two different datasets. Given a picture of an Iris the model would predict if the Iris is a Setosa or Virginica (Brownlee 2018). The accuracy of the model depends on the datasets used to train the model. The model will be more accurate if the datasets are large and have multiple images of the flowers (Nasri, Ouarda & Alimi 2016, Giannakopoulous 2015). Training a machine learning model to detect truth and deception in audio is very similar. We have found audio recordings of real court cases split into truths and lies. We use a python package called PyAudioAnalysis to train our machine with the truth and lie datasets. PyAudioAnalysis can see the patterns of vocal fluctuations and pitch of someone telling the truth vs telling a lie (Giannakopoulos 2015). Once we have the model trained, we can then use the model to classify the truthfulness of a person’s response. To increase the accuracy of the system, it is important to have a large dataset of different people telling truths and lies (Nasri, Ouarda & Alimi 2016). Paura2 was used to record audio in realtime. Paura2 splits the audio into separate recordings whenever there is a short pause in the realtime recording. Then, PyAudioAnalysis is used to classify whether the or not the recording is truthful or not for each of the recordings (Bruno, 2018).

**Integrated Solution for the entire project:**

Our project has audio and the visual parts which are together used to detect deception. The main purpose is to provide an overall analysis of the subject’s voice frequency changes, voice fluctuations, eye movements, and hand movements. The approach we used to achieve this purpose is as follows: We divided the project into two parts visual and audio. We first worked on detecting emotions using audio. Then we moved on to working on detecting deception.

For audio, we used PyAudioAnalysis to train data sets using the audio from real life court cases. This taught the machine the differences between the two data sets (Truthful datasets and Untruthful datasets) and led to the machine being able to differentiate between truths and lies. This was followed by using Paura2 to record audio in real-time and split the audio whenever there is a pause in the audio. These audio recordings were then used by PyAudioAnalysis to detect deception. Deception detection using audio was achieved for real-time as well as recorded audio.

For video, we used OpenCV to detect the faces in recordings as well as real-time. From our research, we found that blinking the eyes and touching the face too frequently was a sign of lying so we focused on these two parts. The two major components of visual deception detection were eye blink detection and hand tracking for the face. This was achieved using imutils, dlib, OpenCV, and tensorflow. The eye blink detection was achieved by calculating the Eye Aspect Ratio discussed in Section 4.3 using the eye associated landmarks as shown in Figure 5. When the eyes are open, this ratio is a constant value and whenever the ratio suddenly falls down close to 0, it is detected as a blink and the counter for the blink goes up. For hand tracking, we detect whenever the hand touches the face using imutils, tensorflow and OpenCV. Both the programs work for real-time as well as recorded video. After achieving these two visual tasks individually, we combined the two scripts together so that running the program would detect blinks as well as hand tracking in the same window.

After the visual and audio parts were individually working, we planned to integrate everything in one program that we could run it to start the detection. We combined the code to call the audio deception detection, eye blink detection and hand tracking detection together. We used threading and multi processing to get everything to work together. Finally, this deception detection program is run to trigger all the individual parts and the final truth/lie prediction and emotion analysis is printed in the terminal.

**5.3 Implementation Problems, Challenges, and Lesson Learned**

**Implementation Problems and Challenges:**

Our group faced implementation problems throughout the course of the project such as import errors, dependencies errors, training models, finding datasets, and component integration. The integration of all individual programs led to several issues. Since our product is unique, there were not examples to reference. We ran into errors while trying to make the application work on Linux as well as Windows. It was hard to import libraries due to too many dependencies and interdependencies. The different dependency version conflicts was also one of the issues we faced. Another huge challenge for us was the time constraint since we had a huge learning curve followed by a lot of implementation and testing. It was also difficult to find truth/lie datasets to train our model. With some of the python libraries we used, they also had some small errors and python version errors. For example, both PyAudioAnalysis and Paura2 were originally written in Python 2.7. Our team decided we will all use Python 3.6 for our project so that we don’t have any version issues between our code. We had to convert the code from Python 2.7 to Python 3.6. Another issue that arose with these libraries were the library dependencies. We spent a month resolving all the dependency import errors we were getting with PyAudioAnalysis and Paura2. These fixes were somewhat complex because we would have to revert some of the dependencies to an older version because the newer version would not work due to deprecated functions.

**Lessons Learned:**

We learned a lot throughout the development of our project. We learned the importance of using a virtual environment when developing our program. We had the issue where we get the program working on one platform but when trying to get it to run another platform required downloading the libraries again as well as troubleshooting version issues. Virtual environment also helped create a more consistent environment for all systems since pipenv can specify the required imported library dependencies. It helped all the team members to import the dependencies without facing any issues. We also learnt that for the successful completion of a project, it is essential to come up with an architectural design for the project. It is also very important for all group members to understand the design of the system and agree upon it before moving further so that there are no conflicts later in the project.

**Chapter 6 Tools and Standards**

**6.1. Tools Used**

Our Project uses several tools. The main language we used for most of the programming is Python. Python is a powerful language that can be used for many different purposes. There are a lot of open source machine learning libraries written for Python. We are also used Python Flask to develop our website. We used HTML, CSS, and Javascript for the front end development. For user authentication on the website we decided to use Okta since it is an easy to implement Single Sign-On service. We used Pycharm as the IDE since all our scripts are mostly written in Python.

There were two main Python libraries that we used for audio emotion and deception detection. PyAudioAnalysis was the library that we used to train our machine learning model. Paura2 was used for the pseudo-real-time classification between truth and lies as well as the several types of emotion. Both PyAudioAnalysis and Paura2 were written by Dr. Theodoros Giannakopoulos. Dr. Giannakopoulos is the director of machine learning at Behavioral Signals. Dr. Giannakopoulos has his PhD in Audio Analysis.

We also used OpenCV and dlib for eye blink detection and facial landmark identification. OpenCV is a library used for real-time computer vision. Dlib is a general cross-platform software library. It was used to calculate the Eye Aspect ratio. We used these tools since they would help us accomplish our project goals. We used the shape predictor by dlib which is a pre-trained facial landmark detector. We also used imutils which facilitates basic image processing functions. The reason we picked imutils is because it is easier to use with OpenCV and Python.

**6.2. Standards**

For programming in groups it is important to create standards so that everyone is using a similar environment and makes it easier for collaboration. We agreed to use Python 3.6 as our main programing language. We also agreed to use Pipenv which is a virtual environment for Python projects. Another standard we decided to use was our communication system. We decided to use Whatsapp and Whatsapp Web for our main communication. We chose Whatsapp because it is lightweight, easy to use, accessible on all platforms, and has file management. Another standard we used was Test Driven Development, we tested our code continuously to make sure everything worked before making big changes so that we could detect bugs and resolve them easily. For the documentation, all team members were supposed to document all the progress and keep track of all the references that were used in the process. For the source code, we encouraged the team to include comments to make it understandable for other team members to understand the logic and the flow of the program better.

**Chapter 7 Testing and Experiment**

**7.1 Testing and Experiment Scope**

There are 3 main components to our project, visual deception detection, audio emotion detection, and audio deception detection.

The objectives of testing were the following:

1. Ensure that the system could achieve emotion detection using audio.
2. Detect deception using audio.
3. Detect deception using video.

**7.2 Testing and Experiment Approach**

For the Audio Deception Detection we performed testing on our sample data using each algorithm (KNN, SVM, Random Forest, Gradient Boosting, and Extra Trees). Our testing data had audio data from real court cases. Since we knew the results of our testing data, we could verify when our model correctly classified the data. Also, since deception detection relies on a person’s deceptive tells which are caused by stress, for example being afraid of the consequences from revealing the truth, our testing produced real life results. Another factor we wanted to test was with our training data. Our training data was separated between 50+ truth audio files and 50+ lie audio files from real court cases. Some of these files had the cross examiner asking the person being questioned. For our initial test we trained our model with the cross examiner in the training data. Our second test we removed the crossexamener so that only the person being questioned would be heard. We hoped that by removing the examiner the model would be able to better recognize audio patterns of someone telling a lie.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Audio Deception Detection** | **Algorithm Used** | | | | |
| Model Used | SVM | KNN | Random Forest | Gradient Boosting | Extra Trees |
| Audio with defendant and crossexamner | 50% | 50% | 50% | 67% | 33% |
| Only Defendant Speaking Audio | 16% | 33% | 33% | 67% | 33% |

For the audio emotion detection we did not have satisfactory results for the accuracy. Our top accuracy was 43% with Extra Trees. Since our main goal was to deception detection, we had to postpone increasing the accuracy of the emotion detection with audio.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Audio Emotion Detection** | **Algorithm Used** | | | |
| Model Used | SVM | KNN | Random Forest | Extra Trees |
| Emotion Detection | 38% | 15% | 23% | 43% |

From the initial results we found that Gradient Boosting performed the best in both models. It was able to correctly classify the testing data 50% of the time. Also, the Audio that did not have the crossexamner filtered out performed the best overall.

For each of the gestures that may represent deceptive behavior, we tested them individually on our computers using our native webcam.

**7.3 Testing and Experiment Results and Analysis**

Test Results 1: after running the scripts and computing the CPU usage

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Detection method** | **CPU Usage** | **RAM** | **Processor** | **Libraries** | **OS** |
| Audio | 9% | 16GB | i7 | Pyaudioanalysis | Windows 10 |
| Blinking | 16% | 16GB | i7 | Dlib, OpenCV | Windows 10 |
| Hands | 50% | 8GB | i7 | TensorFlow, OpenCV | Windows 10 |

Final individual program results:

|  |  |  |
| --- | --- | --- |
| **Detection method** | **Results** | **Improvements** |
| Audio | Accuracy is 67%. Accuracy needs improvement. | Add more training datasets to the model to increase accuracy. |
| Blink | Average of 1/15 false blinks, Low frame rate | Improving the frame rate of the captured video would help the accuracy of the counts of each blink. This could be further improved by training the tensorflow model. |
| Hand | Hand position and angle determines detection of hands, difficult to detect. High CPU load | Improving the frame rate of the captured video would help the accuracy of the counts of each blink. This could be further improved by training the tensorflow model. |

Final Testing Results:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Detection method** | **CPU** | **RAM** | **FPS** | **OS** |
| Multi-threaded audio and video detection | 70% | 1,384 MB | 16 | Windows 10 |
| Multi-process audio and video detection | 70% | 1,361 MB | 16 | Windows 10 |

**Chapter 8 Conclusion and Future Work**

**8.1 Conclusion**

Deception detection is both an objective and subjective subject with many controversial arguments, but it is one of the most interesting topics constantly being researched. There are many methods of trying to detect lies, but we have to keep in mind that humans all have different physiological and psychological bodies. An indicator of deceptive behavior for one person might be normal behavior for another person. Deception detection is a complex process and polygraphs tend to lead to a lot of false positives (Lilienfeld, 2009). To overcome this, our project creates a system in which we uses vocal fluctuations, blinking and hand gestures to detect emotions and deception.

We have also built a UI which provides an interactive way for the user to upload recorded audio to detect the emotions and deception within the recording. Deception detection results are shown after the user uploads the audio file.

Even though we have accomplished a lot of our project, we still have things to work on. We were able to detect deception for each complete response by the user, but in future we plan to add more components for the visual deception detection. The section below discusses our future work.

**8.2 Future Work**

In this project, we only used a handful of methods to detect deceptive behavior, but if time permits we would like to include many other non intrusive methods, such as, using temperature sensors through IR cameras. Another non intrusive method we researched was heartbeat. Individuals who weren’t telling the truth had a higher heartbeat than those who did. Integrating a heart beat monitor into our system would greatly increase our accuracy. On the front end side, we want to incorporate a questionnaire that will record the user's response to each question in the questionnaire. The website will display a single question from the questionnaire. The person's speech and facial expressions will be recorded between each question. Once the user is done responding to the question the user will press the “next” button to continue to the next question. Once all the questions have been completed, the users deception and emotion response will be displayed for each question on the results page. We also want to add the ability for an interviewer to be able to add questions to the questionnaire from an interviewer webpage. The interviewer will have the ability to ask new questions based on the the interviewee’s response. We also want to allow multiple users to access to the results page and see the results in real-time for each question.

In addition, our service currently works on Windows 10, Mac OS, and Linux, but in the future, we hope to offer the service in other platforms such as android or IOS.

The webapp currently only works with recorded audio, but it could be further improved by making real-time audio and video processing.

Another aspect we intend to improve on is eliminating the potential for implicit bias for our deception detection.Machine learning and Artificial Intelligence (AI) has received criticism for potential being biased towards specific groups of people. Machine learning models are considered black boxes. This means that it is difficult to understand the logic behind the classification process since it does not use standard programming techniques. This ethical dilemma is generally caused by the datasets used to train our audio deception detection model(Venkateswaran,2017). Our product is prone to implicit bias as well. For example, if our truth dataset has audio recordings that are from people under the age of 60 and the datasets for lie audio are of people over the age of 60, the model could tend to falsely classify senior citizens audio as being deceptive (Vrij, Fisher, 2016). This machine learning model is prone to have an implicit bias against other groups of people if we are not careful about how we choose our datasets (Wu, 2018, Venkateswaran, 2017). We want to have an accurate system that will correctly classify truth or lies regardless of a person’s age, race, gender, and accent. To avoid a situation like this, we have built our datasets to have almost the equivalent number of recordings of people with similar demographics in each dataset. This will help purge our model of implicit bias. We look to expand our datasets from people with different accents, races, ages, and genders (Yapo, Weiss, 2018). This also will increase our model’s accuracy as well as allow our deception detection software to be used in countries that speak English outside of the United States. We also plan on expanding our model to other languages. It would be unethical to have the model classify a person’s response just because of his or her demographic. Our goal is to correctly verify if someone is innocent and to vindicate someone who is guilty regardless of their demographic background. Therefore, using and testing our software outside of the United States is important to make our software robust and accurate (Bruno, 2018).

**References**

Bartlett, M. S., Littlewort, G., Lainscsek, C., Fasel, I. & Movellan, J. (2004). "Machine learning methods for fully automatic recognition of facial expressions and facial actions," 2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583), pp. 592-597 vol.1. doi: 10.1109/ICSMC.2004.1398364

Bellis, M. (2017). “Police Technology and Forensic Science”. Retrieved from https://www.thoughtco.com/police-technology-and-forensic-science-1991804

Boltz, M. G., Dyer, R. L., & Miller, A. R. (2010). Jo Are You Lying to Me? Temporal Cues for Deception. Journal of Language and Social Psychology, 29(4), 458-466. doi:10.1177/0261927x10385976

Borza, D., Danescu, R., Itu, R., & Darabant, A. S. (2017, December 14). High-Speed Video System for Micro-Expression Detection and Recognition. Retrieved February 20, 2018, from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5751645/

Brownlee, J. (2018, September 19). Your First Machine Learning Project in Python Step-By-Step. Retrieved from https://machinelearningmastery.com/machine-learning-in-python-step-by-step/

Bruno, T. C. (2018, November 28). Area V LO 3 Case Study [Scholarly project].

Deshmukh, R. S., Jagtap, V., & Paygude, S. (2017). "Facial emotion recognition system through machine learning approach," International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, 272-277. doi: 10.1109/ICCONS.2017.8250725

Dibia, V. Real-time Hand-Detection using Neural Networks (SSD) on Tensorflow, (2017), GitHub repository, https://github.com/victordibia/handtracking

EyeDetect: The Next Generation Lie Detector | Polygraph Alternative. (n.d.). Retrieved from https://converus.com/eyedetect/

Giannakopoulos T (2015) pyAudioAnalysis: An Open-Source Python Library for Audio Signal Analysis. PLoS ONE 10(12): e0144610. https://doi.org/10.1371/journal.pone.0144610

Grunwald, M., Weiss, T., Mueller, S., & Rall, L. (2014). EEG changes caused by spontaneous facial self-touch may represent emotion regulating processes and working memory maintenance. Retrieved from https://doi.org/10.1016/j.brainres.2014.02.002

Hallman, J. (2018, February 24). Law Enforcement Deception Detection Strategies [Telephone interview].

Jensen M. L. & Kruse J. (2005). Blob Analysis of the Head and Hands: A Method for Deception Detection

Kundu T. & Saravanan, C. (2017). "Advancements and recent trends in emotion recognition using facial image analysis and machine learning models," International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT), Mysuru, India, 2017, 1-6.

Lie Detector Software Ex-sense Retrieved from http://exsense.trusters.com/

Lie Detector Polygraph Retrieved from http://www.lie-detector.info/shop.php

Lilienfeld, S. (2009). The Polygraph Test Strikes - and Strikes Out - Again. The Skeptical Psychologist. Retrieved from https://www.psychologytoday.com/us/blog/the-skeptical-psychologist/200907/the-polygraph-test-strikes-and-strikes-out-again

Masip, J. (2017). Deception detection: State of the art and future prospects. Psicothema, 29(2), 149-159. doi:10.7334/psicothema2017.34

Meijer, E., Verschuere, B. (2010, July). The Polygraph and the Detection of Deception. Journal of Forensic Psychology Practice. Retrieved February 06, 2018, from http://psycnet.apa.org/record/2010-15958-005

Nasri, H., Ouarda, W., & Alimi, A. M. (2016). "ReLiDSS: Novel lie detection system from speech signal," 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA), Agadir, pp. 1-8. doi: 10.1109/AICCSA.2016.7945789

Neverova N., Wolf C., Taylor G.W., Nebout F. (2015) Multi-scale Deep Learning for Gesture Detection and Localization. In: Agapito L., Bronstein M., Rother C. (eds) Computer Vision - ECCV 2014 Workshops. ECCV 2014. Lecture Notes in Computer Science, vol 8925. Springer, Cham

New Truth Verification Technology. (n.d.). Retrieved from http://www.noliemri.com/

Rosebrock, A. (2017). “Eye Blink Detection with OpenCV, Python, and Dlib.” PyImageSearch, 24 Apr. 2017, from www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/.

Rutkin, A. (2015). Lie-detecting algorithm can spot a guilty face. New Scientist. 228. 22. 10.1016/S0262-4079(15)31553-0.

Soukupova, T. and Cech, J. (2016). “Real-Time Eye Blink Detection using Facial Landmarks.”, February 3-5, 2016

The Home Lie Detector Test. (n.d.). Retrieved from https://www.hammacher.com/Product/

To, K. (2002). Lie Detection: The Science and Development of the Polygraph. Illumin. Vol.V Issue.I Retrieved February 06, 2018, from: http://illumin.usc.edu/43/lie-detection-the-science-and-development-of-the-polygraph/

Ugail, H., M.H.Yap, B.Rajoub, “Face Reading Technology for Lie Detection” https://www.cl.cam.ac.uk/research/security/seminars/archive/slides/2011-10-25.pdf

Vrij, A., & Fisher, R. P. (2016). Which Lie Detection Tools are Ready for Use in the Criminal Justice System? Journal of Applied Research in Memory and Cognition, 5(3), 302-307. doi:10.1016/j.jarmac.2016.06.014

Wu, Z., Singh, B., Davis, L., V. S. Subrahmanian, V.S. (2018). “Deception Detection in Videos” https://arxiv.org/pdf/1712.04415.pdf

Venkateswaran, A. (2017, June 22). Ethics in Machine Learning – Towards Data Science. Retrieved from https://towardsdatascience.com/ethics-in-machine-learning-9fa5b1aadc12

Voice Stress Analysis | Lie Detector Software Apps | Advanced Voice Stress Analysis Technology X13vsa Retrieved from https://www.lie-detection.com/

Yapo, A., & Weiss, J. (2018, January 03). Ethical Implications Of Bias In Machine Learning[Scholarly project]. In Scholarspace.manoa.hawaii.edu. Retrieved November 16, 2018, from http://hdl.handle.net/10125/50557

EyeDetect: The Next Generation Lie Detector | Polygraph Alternative. (n.d.). Retrieved from https://converus.com/eyedetect/

New Truth Verification Technology. (n.d.). Retrieved from http://www.noliemri.com/