

Emotya

Project Engineering

Year 4

Aleksander Kosiak

Bachelor of Engineering (Honours) in Software and Electronic Engineering

Galway-Mayo Institute of Technology

2021/2022

**Declaration**

This project is presented in partial fulfilment of the requirements for the degree of Bachelor of Engineering (Honours) in Software and Electronic Engineering at Galway-Mayo Institute of Technology.

This project is my own work, except where otherwise accredited. Where the work of others has been used or incorporated during this project, this is acknowledged and referenced.

Aleksander Kosiak

**Table of Contents**

[1 Summary 5](#_Toc112924447)

[2 Poster 6](#_Toc112924448)

[3 Introduction 7](#_Toc112924449)

[4 Summary of Libraries 8](#_Toc112924450)

[5 Research & Complications 9](#_Toc112924451)

[6 Project Architecture 10](#_Toc112924452)

[7 Libraries & Use Case 11](#_Toc112924453)

[8 Project Plan 13](#_Toc112924454)

[9 Ethics 14](#_Toc112924455)

[10 Conclusion 15](#_Toc112924456)

[11 References 16](#_Toc112924457)

# Summary

This project was created with previous experience in mind. Having worked personally in a company that specialized in face detection and its elements, I found that emotion detection is an important part in the future development of AI interaction with humans. For this reason, I wanted to explore where I could go with this section of machine learning.

Using the Python language, what I do in this project is that I use a webcam to display our face, then grab each frame shown and predict the emotion in it. A library (Keras) uses an image dataset containing all the emotions we want to extract and compares it with the frame we receive from the webcam and assigns the emotion using another library (Numpy). Then, we can see that live result on a localhost webpage and use it for further development in other projects as I see fit. Python is great for machine learning because of its simplicity and the vast access to frameworks and libraries that are supported.

The result of all this is achieved by using multiple libraries working together to create the application and then using a framework that works with my programming language, to upload the result onto the web. I used Keras, OpenCV, Flask, Numpy and Python to deliver the solution.

# Poster

Graphical user interface, website

Description automatically generated

# Introduction

The goal of this project is to introduce the wide audience into the possibilities of machine learning/deep learning and use it to improve the quality of life or even help people in branching development of this application. I used reference points from various websites to get started on getting a cohesive application running with various libraries that will be mentioned further on in the report.

# Summary of Libraries

To get a broad grasp on what is going on in this project, I need to introduce the important libraries I used to create the app, from the simplest ones down to the most complex. I ran into multiple problems trying to figure out the best way to have all the libraries work in unison.

I am mainly using 4 libraries to create the application. There is also a multi file system that does its own task of creating the result.

OpenCV: Used to grab live feed, capture the frame and send it down to a file that predicts the emotion.

Keras: A complex library that uses its own algorithm to predict the emotion from the frame sent down. For reference on how to use this library to create this process I used this site. [1]

Numpy: Reads the indices from the predicted frame and returns the appropriate emotion from a list of emotions along.

Flask: The framework that creates us the webpage and runs the app to display the feed and emotion.

# Research & Complications

In the early stages of my project, I wanted to use Django as my framework for my website. Unfortunately, at first it was incompatible with the application I wanted to build unless I used some complex plugins that I had no experience with or using Virtual Machines such as WSL2. I found this to work until the virtual machine did not recognise the webcam as a USB device and refused to work. I had to find another framework that would be more compatible with all my libraries and platform and came across Flask. I researched it was compatible with my libraries, so I gave that a chance. The Flask framework is much more lite and Python friendly, which was exactly what I needed for my use case in this project.

# Project Architecture

Diagram

Description automatically generated

Figure 6‑ Architecture Diagram

# Libraries & Use Case

1. Numpy

A library that has not much code written into the code but plays an important role in predicting the emotion. It grabs the maximum value indices as a form of an integer number. The number represents the emotion that is predicted and assigns it from the list of emotions I created. The emotion name is returned.



1. OpenCV

OpenCV main function is to register the live feed I am using in the form of a webcam. I grab the frames that are returned from the camera and convert them into the grayscale. This is done as the grayscale simplifies the algorithm and reduces the time to calculate. Colour brings in a more difficult and elongated process. [2]

Text

Description automatically generated

OpenCV also sets the Haar Cascade Classifiers that help detect the features on the face. They are in a form of an XML file that have the defined patterns. Using this I find the object in the form of a face using the detectMultiScale function. The face object is looped to find coordinates of the image. The image is then resized.

1. Keras

To understand the process of predicting emotion, I had to learn what a model and a model weight is. A model consists of an input layer, multiple hidden layers and an output layer. Each layer contains a node. The input layer contains a node that represents the data we feed it to train the model. Hidden layers are where the computations happen to train the model the way we want to.

Each hidden layer contains multiple nodes depending on how much parameters are set. These nodes do the computation and feed it onto the next nodes in the next layers, also depending on how much layers we add to make the training as accurate as possible. These hidden layers represent the filters we put over the images to do various computations by the nodes to increase accuracy on tests. These filters are the model weights, basically the computations. At the end there is only one node in the output layer that holds the results of all these computations.

This is our trained model. It is also saved to a JSON. This JSON includes the metadata such as location and data of the model. The h5 file for the model weights stores the weights themselves and the model configuration all in the one file. Now using the model with the weights loaded in, we can use them on the frames from the webcam to predict the emotions against the trained model.

Text

Description automatically generated

1. Flask

The Flask app is built with an index page and a video feed page to display one frame. The index page is where you see the real time processing of emotion detection from the webcam, with each frame being checked. The app is run in the main file which runs each necessary file separately.

# Project Plan

Table

Description automatically generated with low confidence

# Ethics

Emotion detection or face detection in general may contain some ethical considerations when rolling this tech out into the hands of the public.

The top ethical concerns related to this industry include [3]:

* Racial bias
* Discrimination
* Lack of privacy
* Undisclosed consent
* Mass surveillance
* Data breaches
* No real modern legislation protecting the public

# Conclusion

The outcome of this project came out well as it displays the scope and purpose set out at the start. It improves on earlier versions where the accuracy of the emotions was totally off, which is now rectified with a reputable, trustworthy and most importantly, accurate datasets that display emotions from the app. The website functions and displays the important information. The development of this application was tough but the future applications and development to this project have a bright future for the AI and Machine learning industry.

# References

|  |  |
| --- | --- |
| [1] | K. Sethi, "Emotion Detection Using OpenCV and Keras," Medium, 23 June 2020. [Online]. Available: https://medium.com/swlh/emotion-detection-using-opencv-and-keras-771260bbd7f7. |
| [2] | G. W. C. Christopher Kanan, "Color-to-Grayscale: Does the Method Matter in Image Recognition?," Journal Plos, 10 January 2012. [Online]. Available: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0029740. |
| [3] | K. R. Gangarapu, "Ethics of Facial Recognition: Key Issues and Solutions," Learn Hub, 25 January 2022. [Online]. Available: https://learn.g2.com/ethics-of-facial-recognition. |