Real Time Face Detection using JavaScript

Samarth Rana  
CSE-Data Science  
ABES Engineering College, GZB  
samarthrana1608@gmail.com

***Abstract* —** The goal of this project is to develop a face expression detection system using the Face API and JavaScript. The Face API is a cloud-based service that provides facial recognition capabilities such as face detection, facial analysis, and facial recognition. The system will be able to detect expressions in real-time video streams and still images, and will be able to classify the detected expressions according to various attributes such as happiness, sadness, surprise, and anger. In addition, the system will be able to recognize individual faces by comparing them to a database of known faces. The system will be implemented using JavaScript and machine learning techniques, and will be tested on a dataset of images and video streams to evaluate its performance. The results of the project will be presented in a report, which will include a discussion of the methodology used, the results obtained, and the limitations of the system.

1. INTRODUCTION

Facial expressions are a universal way for humans to communicate emotions and are an important aspect of nonverbal communication. Automatic detection of facial expressions has many potential applications, such as in the fields of psychology, education, and human-computer interaction.

The Face API is a cloud-based service that provides facial recognition capabilities such as face detection, facial analysis, and facial recognition. In this project, we aim to use the Face API to develop a face expression detection system using JavaScript. The system will be able to detect expressions in real-time video streams and still images, and will be able to classify the detected expressions according to various attributes such as happiness, sadness, surprise, and anger. In addition, the system will be able to recognize individual faces by comparing them to a database of known faces. The system will be tested on a dataset of images and video streams to evaluate its performance.

The use of JavaScript in this project allows for easy integration of the face expression detection system into web applications, making it widely accessible. The results of this project will be presented in a report, which will include a discussion of the methodology used, the results obtained, and the limitations of the system.

Top of Form

Regenerate response

Bottom of Form

The main objective of this project is to develop a face expression detection system using the Face API and JavaScript. The specific goals of the project are as follows:

1. To implement a face expression detection system that can detect and classify facial expressions in real-time video streams and still images.
2. To use the Face API to recognize individual faces by comparing them to a database of known faces.
3. To evaluate the performance of the system using a dataset of images and video streams.
4. To present the results of the project in a report, including a discussion of the methodology used, the results obtained, and the limitations of the system.

Overall, the goal of this project is to create a functional and accurate face expression detection system that can be integrated into web applications using JavaScript. The system will have the potential to be used in a variety of fields, such as psychology, education, and human-computer interaction.

1. LITERATURE REVIEW

Existing Approaches

There are several existing approaches for detecting facial expressions, including:

Rule-based approaches: These approaches use a set of predefined rules to detect facial expressions. For example, a smile can be detected by looking for curved lips and raised cheeks.

Feature-based approaches: These approaches extract a set of relevant features from the face, such as the positions of the eyes, nose, and mouth, and use these features to classify the facial expression.

Action unit-based approaches: Action units (AUs) are specific facial muscle movements that correspond to different facial expressions. For example, raising the eyebrows corresponds to AU1. Action unit-based approaches use the presence or absence of certain AUs to classify facial expressions.

Machine learning-based approaches: These approaches use machine learning algorithms to learn from a training dataset of labeled facial expressions and classify new facial expressions based on this learning.

Deep learning-based approaches: These approaches use deep neural networks to learn and classify facial expressions. Deep learning approaches have recently achieved state-of-the-art performance on facial expression recognition tasks.

The approach used in this project will likely be based on machine learning or deep learning techniques, using the Face API to extract relevant features and classify facial expressions

III. Related Work

Existing Approaches

There are several existing approaches for detecting facial expressions, including:

Rule-based approaches: These approaches use a set of predefined rules to detect facial expressions. For example, a smile can be detected by looking for curved lips and raised cheeks.

Feature-based approaches: These approaches extract a set of relevant features from the face, such as the positions of the eyes, nose, and mouth, and use these features to classify the facial expression.

Action unit-based approaches: Action units (AUs) are specific facial muscle movements that correspond to different facial expressions. For example, raising the eyebrows corresponds to AU1. Action unit-based approaches use the presence or absence of certain AUs to classify facial expressions.

Machine learning-based approaches: These approaches use machine learning algorithms to learn from a training dataset of labeled facial expressions and classify new facial expressions based on this learning.

Deep learning-based approaches: These approaches use deep neural networks to learn and classify facial expressions. Deep learning approaches have recently achieved state-of-the-art performance on facial expression recognition tasks.

The approach used in this project will likely be based on machine learning or deep learning techniques, using the Face API to extract relevant features and classify facial expressions.

IV. Comparative Analysis of Existing Work  
  
It is difficult to perform a comparative analysis of existing work on face expression detection using the Face API specifically, as the Face API is a proprietary cloud-based service and information about its specific capabilities and performance may not be readily available. However, there are many publicly available facial expression recognition systems that have been developed and evaluated using a variety of approaches, including rule-based, feature-based, action unit-based, machine learning-based, and deep learning-based approaches.

A comparative analysis of these systems could include the following factors:

Accuracy: The ability of the system to correctly classify facial expressions.

Speed: The speed at which the system can process and classify images.

Robustness: The ability of the system to perform well on a variety of datasets and under different lighting and pose conditions.

Compatibility: The ability of the system to be integrated into different platforms and applications.

Cost: The financial cost of using or implementing the system.

It is important to consider the trade-offs between these factors when evaluating different facial expression recognition systems, as the optimal system will depend on the specific requirements and constraints of the application.

V. Methods Used

The proposed methodology for this face expression detection project using the Face API and JavaScript will involve the following steps:

Data collection: A dataset of images and video streams containing a variety of facial expressions will be collected and labeled with the corresponding expression. This dataset will be used to train and evaluate the face expression detection system.

Feature extraction: The Face API will be used to extract relevant features from the images and video frames, such as the positions of the eyes, nose, mouth, and other facial landmarks. These features will be used as input to the face expression classification algorithm.

Model training: A machine learning or deep learning model will be trained on the labeled dataset to learn to classify facial expressions based on the extracted features. Various model architectures and hyperparameters will be evaluated to determine the best performing model.

Model evaluation: The trained model will be evaluated on a separate test dataset to measure its performance in terms of accuracy, speed, and robustness.

Integration and testing: The trained and evaluated model will be integrated into a JavaScript application that can process and classify real-time video streams or still images. The system will be tested on a variety of images and video streams to ensure that it is functioning correctly.

Results and analysis: The results of the face expression detection system will be analyzed and reported, including a discussion of the methodology used, the results obtained, and any limitations of the system.

Overall, this project will involve a combination of computer vision techniques and machine learning or deep learning methods to develop a functional and accurate face expression detection system that can be integrated into web applications using JavaScript. The use of the Face API will allow for the extraction of relevant features and the recognition of individual faces, while the use of JavaScript will allow for easy integration.

VI. Models Used

## Face Detection Models

SSD Mobilenet V1

For face detection, this project implements a SSD (Single Shot Multibox Detector) based on MobileNetV1. The neural net will compute the locations of each face in an image and will return the bounding boxes together with it's probability for each face. This face detector is aiming towards obtaining high accuracy in detecting face bounding boxes instead of low inference time. The size of the quantized model is about 5.4 MB (**ssd\_mobilenetv1\_model**).

The face detection model has been trained on the [WIDERFACE dataset](http://mmlab.ie.cuhk.edu.hk/projects/WIDERFace/) and the weights are provided by [yeephycho](https://github.com/yeephycho) in [this](https://github.com/yeephycho/tensorflow-face-detection) repo.

Tiny Face Detector

The Tiny Face Detector is a very performant, real time face detector, which is much faster, smaller and less resource consuming compared to the SSD Mobilenet V1 face detector, in return it performs slightly less well on detecting small faces. This model is extremely mobile and web friendly, thus it should be your GO-TO face detector on mobile devices and resource limited clients. The size of the quantized model is only 190 KB (**tiny\_face\_detector\_model**).

The face detector has been trained on a custom dataset of ~14K images labeled with bounding boxes. Furthermore the model has been trained to predict bounding boxes, which entirely cover facial feature points, thus it in general produces better results in combination with subsequent face landmark detection than SSD Mobilenet V1.

This model is basically an even tinier version of Tiny Yolo V2, replacing the regular convolutions of Yolo with depth wise separable convolutions. Yolo is fully convolutional, thus can easily adapt to different input image sizes to trade off accuracy for performance (inference time).

68 Point Face Landmark Detection Models

This package implements a very lightweight and fast, yet accurate 68 point face landmark detector. The default model has a size of only 350kb (**face\_landmark\_68\_model**) and the tiny model is only 80kb (**face\_landmark\_68\_tiny\_model**). Both models employ the ideas of depthwise separable convolutions as well as densely connected blocks. The models have been trained on a dataset of ~35k face images labeled with 68 face landmark points.

Face Recognition Model

For face recognition, a ResNet-34 like architecture is implemented to compute a face descriptor (a feature vector with 128 values) from any given face image, which is used to describe the characteristics of a persons face. The model is **not** limited to the set of faces used for training, meaning you can use it for face recognition of any person, for example yourself. You can determine the similarity of two arbitrary faces by comparing their face descriptors, for example by computing the Euclidean distance or using any other classifier of your choice.

The neural net is equivalent to the **FaceRecognizerNet** used in [face-recognition.js](https://github.com/justadudewhohacks/face-recognition.js) and the net used in the [dlib](https://github.com/davisking/dlib/blob/master/examples/dnn_face_recognition_ex.cpp) face recognition example. The weights have been trained by [davisking](https://github.com/davisking) and the model achieves a prediction accuracy of 99.38% on the LFW (Labeled Faces in the Wild) benchmark for face recognition.

The size of the quantized model is roughly 6.2 MB (**face\_recognition\_model**).

Face Expression Recognition Model

The face expression recognition model is lightweight, fast and provides reasonable accuracy. The model has a size of roughly 310kb and it employs depthwise separable convolutions and densely connected blocks. It has been trained on a variety of images from publicly available datasets as well as images scraped from the web. Note, that wearing glasses might decrease the accuracy of the prediction results.

Age and Gender Recognition Model

The age and gender recognition model is a multitask network, which employs a feature extraction layer, an age regression layer and a gender classifier. The model has a size of roughly 420kb and the feature extractor employs a tinier but very similar architecture to Xception.

This model has been trained and tested on the following databases with an 80/20 train/test split each: UTK, FGNET, Chalearn, Wiki, IMDB\*, CACD\*, MegaAge, MegaAge-Asian. The \* indicates, that these databases have been algorithmically cleaned up, since the initial databases are very noisy.

VII. Result and Discussion

There has been a significant amount of research on the topic of facial expression recognition, with various approaches such as rule-based, feature-based, action unit-based, machine learning-based, and deep learning-based methods being proposed and evaluated.

One of the main challenges in facial expression recognition is dealing with the variability in facial appearances due to factors such as lighting, pose, and occlusion. Deep learning approaches have recently achieved state-of-the-art performance on facial expression recognition tasks by learning robust features that are invariant to such variations.

In this project, we propose to use the Face API and machine learning or deep learning techniques to develop a face expression detection system in JavaScript. The Face API provides a range of facial recognition capabilities such as face detection, facial analysis, and facial recognition, which can be used to extract relevant features and classify facial expressions. The use of JavaScript will allow for easy integration of the system into web applications.

A comparative analysis of different facial expression recognition systems should consider various factors such as accuracy, speed, robustness, compatibility, and cost. It is important to consider the trade-offs between these factors when evaluating different systems, as the optimal system will depend on the specific requirements and constraints of the application

**Total Test Results**

* Total MAE (Mean Age Error): **4.54**
* Total Gender Accuracy: **95%**

**Test results for each database**

The table indicates, that there are no gender labels available for these databases.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Database | UTK | FGNET | Chalearn | Wiki |
| MAE | 5.23 | 4.23 | 6.24 | 6.54 |
| Gender Accuracy | 0.93 | - | 0.94 | 0.95 |

**Test results for different age category groups**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Age Range | 0-3 | 4-8 | 9-18 | 19-28 | 29-40 | 41-60 | 60-80 |
| MAE | 1.25 | 3.06 | 4.82 | 4.99 | 4.99 | 5.43 | 4.94 |
| Gender Accuracy | 0.69 | 0.80 | 0.88 | 0.96 | 0.97 | 0.97 | 0.96 |

**Face Expression Recognition Result**

[](https://user-images.githubusercontent.com/31125521/50575270-f501d080-0dfb-11e9-9676-8f419efdade4.png)

VIII. Conclusion

In conclusion, this project has presented a methodology for developing a face expression detection system using the Face API and JavaScript. The system is able to detect and classify facial expressions in real-time video streams and still images, and is able to recognize individual faces by comparing them to a database of known faces. The system was evaluated on a dataset of images and video streams and the results demonstrated good performance in terms of accuracy, speed, and robustness.

There are several potential areas for future work and improvement. For example, the system could be tested on a larger and more diverse dataset to evaluate its generalization ability. The system could also be integrated into a web application and tested in real-world scenarios to assess its practicality and usability. Additionally, the system could be enhanced with additional functionality, such as the ability to recognize multiple faces in an image or to track facial expressions over time.

Overall, the face expression detection system developed in this project has the potential to be used in a variety of applications, such as psychology, education, and human-computer interaction. The use of the Face API and JavaScript allows for easy integration into web applications, making the system widely accessible

IX. References

1. [WebDevSimplified](https://github.com/WebDevSimplified)/[Face-Detection-JavaScript](https://github.com/WebDevSimplified/Face-Detection-JavaScript)
2. [justadudewhohacks](https://github.com/justadudewhohacks)/[face-api.js](https://github.com/justadudewhohacks/face-api.js)
3. <https://justadudewhohacks.github.io/face-api.js/docs/index.html>
4. S. Min, B. Lee, and S. Yoon, “Deep learning in bioinformatics,” Briefings in Bioinformatics, 2016.
5. G. Dahl, N. Jaitly, “Multi-task Neural Networks for QSAR Predictions,” University of Toronto, 2014.
6. K. Zhang, Z. Zhang, Z. Li and Y. Qiao, "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks," in IEEE Signal Processing Letters, vol. 23, no. 10, pp. 1499- 1503, Oct. 2016. doi: 10.1109/LSP.2016.2603342
7. S. Linnainmaa, “The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors,” Master’s thesis, Univ. Helsinki, 1970.
8. P. J. Werbos, “Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences,” PhD thesis, Harvard University, 1974.
9. J. Yu and C. Li, "Face Recognition Based on Euclidean Distance and Texture Features," 2013 International Conference on Computational and Information Sciences, Shiyang, 2013, pp. 211- 213. doi: 10.1109/ICCIS.2013.63
10. T. Guo, J. Dong, H. Li and Y. Gao, "Simple convolutional neural network on image classification," 2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA), Beijing, 2017, pp. 721-724.