Facial and Gesture Recognition

Elena Zisimatou
Dept. of Digital Systems
University of Piraeus
Piraeus, Greece
elena_zisimatou@hotmail.com

Maria Fritzela
Dept. of Digital Systems
University of Piraeus
Piraeus, Greece
me2041@unipi.gr

Konstantina Mantalia
Dept. of Digital Systems
University of Piraeus
Piraeus, Greece
konna.madalia@gmail.com

***Abstract*— In this project we implement a facial and gesture recognition program, leveraging the open-source framework MediaPipe. More specifically, we utilize MediaPipe Holistic pipeline which integrates facial, gesture and pose recognition. Using the computer camera, we capture our facial expressions, matching them with a different emoji. The goal is to label the data collected, train machine learning algorithms and use it to recognize real time expressions. Logistic Regression, Support Vector Machines and Random Forest utilized for the classification part of the problem. Moreover, due to the high dimensionality of the problem, Principal Component Analysis (PCA) was performed.**

Keywords*— MediaPipe, Facial, Gesture, Emoji, Holistic, Machine Learning, PCA, Logistic Regression, Random Forest, Support Vector Machines

I. INTRODUCTION

In recent years, real time facial and gesture recognition has gained significant attention. A wide variety of real time applications utilize computer vision to recognize faces, from law enforcement to video surveillance cameras and commercial applications [1]. In this project, we utilize the MediaPipe Holistic, an open-source framework and implement a real time computer web camera-based program which captures the face and the pose of the user, captures, and stores the landmarks, and finally makes predictions regarding the class that the specific frame belongs to. In our case, classes are 9 emojis which are described in TABLE I. We created data capturing each time our face and pose based on the emoji pose description. Each picture frame is stored as a sequence of real numbers in a data frame with its class as a label. In this way, the problem is induced into a classification task. We use different kind of classification machine learning algorithms to train and test the data. Apart from this, Principal Component Analysis was leveraged to improve the training time and reduce the complexity of our models and to reduce the dimensions and the complexity of the model. Reducing the dimensions from 2004 to 50, and train Logistic Regression seems to achieve accurate results. Finally, when the user runs the program, a window pop ups and the emoji that her/his face looks like is displayed along with the probability.

II. RELATED WORK

Related studies in the specified field include pose recognition with tools such as Microsoft Kinect Sensor which was innovative because it did not require expensive and complicated equipment like its predecessors. As proposed on paper [2] by using skeleton data provided by Kinect sensor the users perform a set of specific tasks from a vocabulary which includes different poses. The features are extracted from skeleton data of each pose and subsequently the performance results of four machine learning algorithms: Support Vector Machines (SVM), artificial neural networks (ANN), k-nearest neighbors (KNN) and Bayes classifier are being compared. In paper [3], is proposed another approach to jointly handle human pose estimation as well as action recognition giving particular importance in the use of predicted poses on action recognition and taking advantage of shared computations between the two tasks. This method can be trained with single frames and video clips at the same time. Showing us that by using a carefully designed architecture multi-tasking human poses and action recognition can be handled efficiently. Another approach [4], models skeleton sequences as bags of time-stamped descriptors and it utilizes a new framework for action classification based on the Citation-KNN method.

III. PROJECT DESCRIPTION

The goal of this project is to create a pipeline for the recognition of facial expressions and gestures in real time through the computer webcam, matching them to a selection of emojis. Emojis are encoded picture characters widely used online in social media and forums to express emotion. Nowadays, with much of human communication and connection happening online, they are making an impact on language and society itself [5]. As of September 2020, there are over 3 thousand emoji in the Unicode Standard, separated into different categories, like "People and Body", "Objects", and "Food and Drink" [6]. For this project, a selection of 9 commonly used emoji from the "People and Body" category are used for the different classes. These are presented in the table below (TABLE I).

TABLE I. EMOJIS USED AND THEIR POSES

Emoji	**Emoji Unicode Name**	**Pose Description**
🙂	Slightly Smiling Face	Subject is slightly smiling
😀	Grinning Face	Subject is smiling with teeth visible and/or mouth open
😐	Neutral Face	Subject is expressionless
☹️	White Frowning Face	Subject is frowning, possibly also slightly bowing head
😱	Face Screaming in Fear	Subject is simulating a feared expression, using their hands on the side of their face, with wide eyes and mouth open.
🤗	Hugging Face	Subject is holding their palms next to their face facing the camera, and/or smiling
🤫	Shushing Face	Subject is placing an index finger over pursed lips using either their right or left hand
😇	Smiling Face with Halo	Subject is smiling with their hands over their head forming a

Emoji	Emoji Unicode Name	Pose Description
		circle using their index fingers and thumbs
😈	Smiling Face with Horns	Subject is smiling with their hands on their head forming horns using their index fingers

These emoji were selected to incorporate a combination of facial, hand and pose characteristics. Because of limitations of the MediaPipe Landmarks, the tongue is not detected and neither is the opening or closing of the eyes. This makes the detection of emojis like the "winking face" impossible and thus are excluded from this study.

IV. TECHNOLOGIES USED

The technologies used for the implementation of the project are briefly presented in this section.

A. *MediaPipe*

MediaPipe is an open-source framework designed specifically for complex perception pipelines. In order to enable augmented reality, a typical application presents data such as video and audio at high frame rates to enhance the user experience. It is important for the user to be able to develop the same application in different platforms, but this task is essentially time consuming as it involves optimizing inference (being able to do inference as quickly as possible is very important for neural networks), and of course it is necessary for processing steps to run correctly and efficiently on our target device. By using the pipelines, as we mentioned above, MediaPipe addresses these challenges cause now the sensory data such as video and audio streams enter a graph of modular components (including inference models and media processing functions) and the output is object detection results and face landmark annotations.

The three main parts of MediaPipe are:

- A framework for inference (processing process in order to reach a conclusion depending on known facts) from sensory data
- A set of tools for performance evaluation
- A collection of reusable inference and processing components

Each component in the pipeline is a Calculator. In the graph the data flows via each calculator via data Streams whereas the basic data unit in MediaPipe is a Packet.

Packet: It is the basic data unit in MediaPipe. A packet is been treated as value class and can be copied cheaply.

Streams: Each node in the graph (a pipeline is defined as a directed graph of components where each component is a Calculator) is connected to another node via a Stream. A Stream carries a sequence of packets whose timestamps must be monotonically increasing. When using a Stream we should keep in mind that an output Stream can be connected to a variable number of input streams of the same type.

Calculators: Each node is implemented as a calculator. A calculator can receive zero or more input streams and output zero or more streams.

Graph: The basic context in MediaPipe is the Graph. All the processing takes place inside the graph. When a single graph run happens, the framework constructs calculator objects corresponding to a graph node and calls three specific actions. Open(), Process() and Close().

After a graph starts, the framework calls Open(). This specific function interprets the node configuration operations and prepares the calculator's per-graph-run state. This function may also write packets to calculator outputs. If an error occurs during Open call the graph can be terminated.

The framework calls the function Process() repeatedly whenever at least one input stream has a packet available. (by default all the inputs have the same timestamp) When we have parallel execution there is a probability that multiple function calls can be called at the same time. When an error happens the function Close() will be called and the graph will terminate.

Close() is the function that the framework will call after reaching the final level. This function is being called either way. Whether the function Open() was called and succeeded or in the case mentioned above where the graph terminated due to an error. After Close() returns, the calculator should be considered a dead node. During the final step the calculator object is destroyed as soon as the graph finishes running.

B. *Most common use case scenarios for MediaPipe*

MediaPipe is utilized in many cases, such as Selfie Segmentation, Face Mesh, Hand Tracking, Human Pose Detection and Tracking, Hair Segmentation, Object Detection and Tracking, Face Detection, Holistic Tracking, 3D Object Detection and many more applications.

C. *The Holistic Model*

MediaPipe Holistic aims to enable the user with a holistic perception of body actions, gesture and facial expressions. Holistic utilizes deep learning models to accurately detect key points, generating landmarks. In our case each landmark consists of coordinates x, y and z where z represents the landmark depth with the depth at center of the head being the origin. Visibility describes the likelihood of the landmark being visible (present and not occluded) in the image.

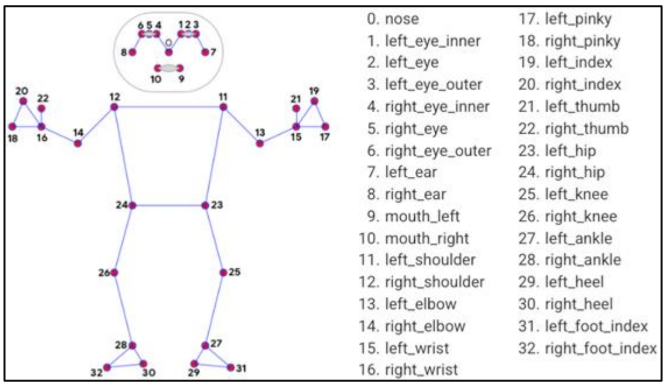

Figure 1: Pose Model Landmarks

D. *OpenCV*

OpenCV is an open-source library mainly used for computer vision, image processing, and machine learning. With the help of OpenCV we are able to process images and videos so that the implemented algorithm could be able to identify different objects such as dogs, cats, cars, traffic signals etc. even faces. The outputs are better when the data is real-time so OpenCV is ideal for processing videos and images.

V. Machine Learning Algorithms applied

For the purposes of our study, we tested the following supervised machine learning algorithms for classification: Logistic Regression (LR), Support Vector Machines (SVMs), and Random Forest (RF). In this section we are going to give a comprehensive presentation.

A. *Logistic Regression*

Logistic Regression is a probabilistic, machine learning model, whereby we solve classification problems, using regression [7]. The goal is to find the hyperplane that separates the examples of the classes with the optimal way. In terms of positive/negative (0/1) classification, the output of such algorithms is the probability of an observation to belong to the positive class.

Logistic Regression makes 4 the following assumption: The logarithm of the odds can be approached by a hyperplane

$b0 + b1x1 + ... + bnxn = b0 + \bar{b}X\ T = 0$, where:

- $x1, ..., xn$ are the input variables
- $b1, ..., bn$ the estimated coefficients
- $b0$ the estimated intercept

Finally,

$P(1) = S(b0 + b1x1 + ... + bnxn)$, where S is the sigmoid function. In most cases, if $P(1) \geq 0.5$ we classify the observation as 1 and if $P(1) < 0.5$ we classify it as 0. Nevertheless, we can set a threshold that the model will consider and make the prediction. If a coefficient of a feature is positive, it indicates that the increase of the independent variable contributes to the increase in the probability of the individual to belong to the positive class.).

B. *Support Vector Machines (SVMs)*

Support Vector Machines is a family of machine learning algorithms which are widely used for both classification (SVC) and regression tasks (SVR). SVM can effectively handle linear and non-linear problems [8]. It is a geometrical model which tries to approximate a hyperplane that separates the features into domains. Concerning the binary classification, the optimal hyperplane maximizes the margin between the two classes and the hyperplane. The points closest to the hyperplane are called support vectors. SVMs can approximate every possible decision boundary (non-linear), due to kernel tricks in which the features are transformed into a higher dimensional space, where their separation is feasible.

C. *Random Forest (RF)*

A Random Forest (RF) is an ensemble technique for building machine learning models, which takes advantage of the simplicity of Decision Trees to build a more flexible model that will be generalized more effectively to real data. With the term “ensemble” we mean all those methods that combine the predictions of other, simpler models.

The RF algorithm works as follows:

1. Create a subset of the data using the bootstrap method.
2. Create a Decision Tree using the data above and set as candidate variables a subset of the initial variables of the dataset. The RF evaluation is carried out by using the out-of-the bag data, the data that was never selected by bootstrap sampling.

VI. Principal component analysis

One of the most widely used feature extraction techniques is Principal Component Analysis (PCA). PCA is a multivariate technique that takes advantage of linear algebra methods to rotate the original coordinate system in a way such that the rotated features are statistically uncorrelated [9]. After the rotation, we can select only a subset of the new components according to how much of the initial variance they explain. The new components, the so-called principal components, are linear combination of the previous ones. The first principal component is required to have the largest possible variance, the second principal component the second largest possible variance with the constraint of being orthogonal to the first principal and so on [10]. PCA has been applied in many different fields like biology, natural language processing, signal processing, computer graphics etc.

PCA proved to be an extremely useful technique in our case. The initial data set consists of 2004 numerical features and 15979 rows. It means that our data are on the $\mathbb{R}^{2004}$ dimensional space. As a result, reduction of the dimensions is imperative cause the training phase of the models is time intensive. A transformation on the initial data set is conducted keeping only 50 components. On the figure below we plot the explained variance ration for each new component. It is worth saying that the new components explain the 0.999 of the initial variation of the data.

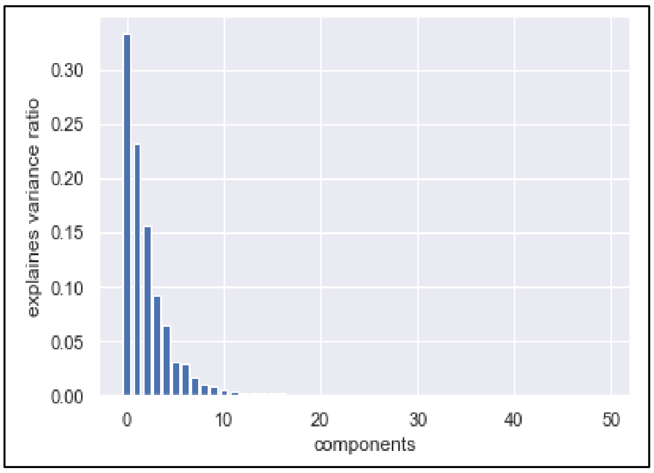

Figure 2: Principal Component Analysis Explained Variance Ratio

VII. Process

In this section, we describe the process followed during the creation of the facial and gesture recognition application.

A. *Strategy*

The strategy can be broken down into three discrete steps:

1. **Data Collection**: Using a webcam through OpenCV and MediaPipe’s Holistic Model, the pose, face, and

hand landmarks are collected for each one of the classes and combined into a single feature vector.

2. **Classification Model Training**: The data collection is split into a testing and training set. Then, a variety of classification models are trained on these landmarks and their performance on the testing set is evaluated.
3. **Testing and Application**: The trained model is used to make predictions in real time. MediaPipe Holistic detects the coordinates of the landmarks and the classifier maps it to one of the given classes in real time.

More details on the execution of these steps are described in the following sections.

1) Data Collection: Through rigorous experimentation, the data collection process proved not to be as straightforward as initially calculated. A couple of caveats were detected. Firstly, and possibly most obviously, the pose for each one of the classes had to be performed in different angles and variations, closer and further away from the camera, for the classification model to be able to generalize them accurately. The goal of this is to help the model understand and recognize the small variations of the pose that exist.
Secondly, it was soon understood that having only one volunteer perform the poses drove the classification models to work perfectly at recognizing their expressions in real time, however they fell very short in recognizing the emotions of different human subjects. Each person's facial and body expressions are truly unique, and training the model on only one subject constrains its perception of what each of the classes look like. For this reason, the data of multiple volunteers performing the poses were collected.

2) Classification Model Training: Multiple pipelines were created to train different classifiers on the data in an easy way. For all of the pipelines scaling was performed using sklearn's "Standard Scaler". The algorithms tested were Linear Regression, Support Vector Machines, and Random Forest. Each one of them was tested with the extra step of Principal Component Analysis (PCA) to reduce the dataset's dimensionality, and without it.

3) Testing and Application: Metrics for evaluating the performance of the model in real time were difficult to devise. Most of the assessment depended on noting the confidence returned by the model on each pose during live experiments, and recording and comparing demo videos. Two different approaches were implemented, one where the application presents the estimates that the model performs on each frame of the video independently, and a "sampling" approach, where the predictions of multiple frames are aggregated to present to the user the predicted class.

VIII. Results

It was quickly noticed that although the accuracy scores by the different algorithms on the test were useful in comparing their performance, they were not representative of their effectiveness in real time scenarios. It is the diversity of the training data that is critical to the final result. If the model has a very narrow perception of what each of the classes-poses look like, then it is set up for failure. Looking at the table below, one could theorize that all of the classifiers will perform outstandingly in recognizing the emoji poses in real time.

TABLE II. Test set Performance of different classifiers

Model	**Accuracy on test set**
LR	0.9987484355444305
SVM	0.9874843554443054
RF	0.9879015435961619
LR + PCA	0.9974968710888611
SVM + PCA	0.9870671672924489
RF + PCA	0.9939507717980809

However, these accuracy scores are not representative of their real time performance. In order to compare the classifiers accuracy in real time, a custom scoring system was developed. While having a test subject perform the poses, the result of the model along with its probability were noted. If the model recognized the pose performed during the experiments with confidence (probability > 0.8) then it was given a "Good" score. If the model recognized the pose without much confidence, a "Med" score was noted for that pose. Finally, if the model mostly failed to recognize a pose, it was given a "Poor" score.

In the table below, the performances of different classification models for each one of the classes in real time are presented: (Good=High Probability(>0.8) - 2 Points, Med=Low Probability - 1 Point, Poor=class not recognized - 0 Points). The S symbol represents the application of sampling.

TABLE III. Experimental scores of Classifiers

Emoji	**Classifiers**					
	LR	***LR (S)***	***SVM***	***SVM (S)***	***RF***	***RF (S)***
☹️	Good	Good	Good	Good	Good	Med
😐	Poor	Poor	Good	Good	Med	Good
🙂	Med	Good	Med	Good	Med	Med
😀	Good	Good	Med	Med	Med	Med
🤗	Good	Good	Good	Good	Med	Good
😱	Med	Good	Good	Good	Med	Good
😇	Poor	Poor	Poor	Poor	Good	Good
😈	Good	Good	Good	Good	Good	Good
🤫	Good	Good	Good	Good	Poor	Poor
Total Points	12	14	14	15	10	13

It is very interesting to see how different models performed better on different poses. For example, most models struggled with the recognition of the "Smiling Face with Halo" pose, but Random Forest produced very accurate predictions, but generally scored lower on the others. Also, the sampling of multiple frames to produce a result led to more accurate predictions.

IX. Conclusion- Future Work

We have presented the procedure of creating a facial and gesture recognition application in detail. Based on the accuracy scores we see that Logistic Regression achieves the best accuracy results, while during experiments SVM also performed very well.

Regarding future work, we could extend the application in order to incorporate a richer assortment of emoji.

References

[1] M. Meenakshi, “Real-Time Facial Recognition System—Design, Implementation and Validation”, 2012 Journal of Signal Processing Theory and Applications, pp. 1-18, doi: 10.7726/jspta.2013.1001

[2] Y. Choubik and A. Mahmoudi, "Machine Learning for Real Time Poses Classification Using Kinect Skeleton Data," 2016 13th International Conference on Computer Graphics, Imaging and Visualization (CGiV), 2016, pp. 307-311, doi: 10.1109/CGiV.2016.66.

[3] D. Luvizon, D. Picard and H. Tabia, "Multi-task Deep Learning for Real-Time 3D Human Pose Estimation and Action Recognition", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1-1, 2020, doi: 10.1109/TPAMI.2020.2976014.

[4] S. Ubalde, F. Gomez-Fernandez, N. Goussies and M. Mejail, "Skeleton-based action recognition using Citation-kNN on bags of time-stamped pose descriptors", *2016 IEEE International Conference on Image Processing (ICIP)*, 2016, doi: 10.1109/ICIP.2016.7532920.

[5] P. Seargeant, “The Emoji Revolution: How Technology is Shaping the Future of Communication”, Cambridge: Cambridge University Press, 2019, doi: 10.1017/9781108677387.

[6] "Emoji Counts, v13.1", *Unicode.org*, 2021. [Online]. Available: https://unicode.org/emoji/charts/emoji-counts.html. [Accessed: 05-Jul- 2021]

[7] [1]D. Montgomery, E. Peck and G. Vining, *Introduction to linear regression analysis (5th ed.)*. Hoboken, NJ: Wiley, 2012..

[8] Sofia A. Papapostolou. “Classification with Support Vector Machines”. Thessaloniki, 2017.

[9] X. Xu, T. Liang, J. Zhu, D. Zheng and T. Sun, "Review of classical dimensionality reduction and sample selection methods for large-scale data processing", Neurocomputing, vol. 328, pp. 5-15, 2019, doi: 10.1016/j.neucom.2018.02.100.

[10] S. Holand, “PRINCIPAL COMPONENTS ANALYSIS (PCA)”, Department of Geology, University of Georgia, Athens, GA 30602-2501, 2019. [Online]. Available: https://strata.uga.edu/software/pdf/pcaTutorial.pdf. [Accessed: 05-Jul- 2021]