Real-time Smile Detection Using Histogram of Oriented Gradients with Support Vector Machine and Haar Cascade with Convolutional Neural Networks

Ahmed Afifi
McMaster University
1280 Main St W, Hamilton
afifia1@mcmaster.ca

Abstract

The aim of this paper is to outline and implement a method of detecting smiling faces provided a video feed in real-time. A Histogram of Gradients will be used to extract the features necessary to recognize the characteristics of smiling lips. Moreover, a Support Vector Machine will be implemented to create the best fit hyperplane that separates the two data classes (smiling, not-smiling) and form a model that can be used for classification. Similarly, a secondary approach will be used whereby the feature extraction will be done using the Haar cascade method and trained using Convolutional Neural Networks to try and achieve more accurate results with varying input conditions.

1. Introduction

Facial expressions indicate emotional sentiments and underpin sensations such as pleasure, happiness, sorrow, boredom, anger, contentment, and discontent. Perhaps the most common expression of human emotion, is the smile. There has been an emerging need for smile monitoring in real-world development and advancement to assist humans in a variety of potential applications, such as photo selection, user analysis, applications in the healthcare industry, artificial intelligence applications, and patient feeling conditions. Researchers and professionals involved in the development of this new technology have already expressed a strong interest in an efficient smile detection software that can provide accurate results.

1.1. Types of Smiles

There are two types of smiles, the first being the social smile, which is a voluntary facial expression that is created consciously, such as when meeting new people or taking a passport photo, and the other is a spontaneous grin, which is a natural response created unconsciously and is generated by emotional overflow.

1.2. Lip shape

The contour of each person's grin is distinct, but the smile has the same pattern, where the shape of the right and left edge lips being higher than the pattern of the centre lip. This simple fact can be used to differentiate the shape of smiling lips versus that of a neutral or frowning ones.

2. Related Works

Some extraction features for identifying smiles include haar-like feature extraction over grey colours and standardised locations such as hear, which is done by Malesevic and Jojic [1], and histogram equalisation, which is done by Syaputra [2] and focuses on the colour of each block. Some studies have also been able to accomplish the detection using Convolutional Neural Networks such as the one done by Zhang, Huang, Wu and Wang [3].Furthermore, a paper written by Ali and Dua [4] proposed targeting the real-time challenges of this application by amalgamation of geometric feature extraction (GFE) and regional local binary pattern (LBP) features extraction using autoencoders, which they believe to be dynamic and precise.

3. Main Approach

The main approach consists of 3 main stages: pre-processing, feature extraction and training (classifier). The dataset used to train and test the classifier in this approach is the GENKI-4K subset of the Machine Perception Laboratory (MPLab) GENKI database, which consists of 4000 face images labeled as either "smiling" or "non-smiling" by human coders. The pose of the faces is approximately frontal as determined by MPLab's automatic face detector [5].

3.1. Pre-processing

In order to provide the classifier with the best data for training, the GENKI-4K dataset was modified. The faces

in the pictures provided by the dataset were extracted using dlib's pre-built frontal face detector function which uses a 68-feature point algorithm to find face shapes in the pictures provided. The algorithm was not able to find the face in 122 pictures of the 4k set provided and 3878 pictures were saved separately to be used to train and test the SVM classifier.

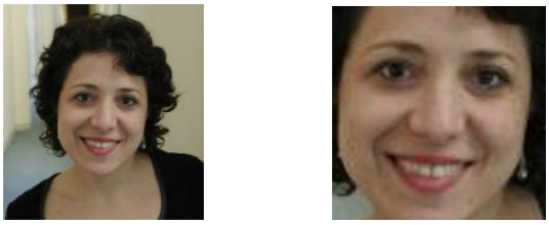

Figure 1. Before vs After pre-processing

3.2. Feature Extraction

The Histogram of Gradients (HOG) method was used to extract the features necessary to compare smiles provided from the real-time video feed and the data set of pictures. Since, the GENKI-4K data set contains pictures of roughly the same size (200x200 pixels), it was easy to approximate a position for the mouth from the extracted faces. This was done on every image using the function *extract_mouth()*, where the output is a gray scale cut-out of the mouth from the input image. This is also done on every frame from the input video stream in order to compare the mouth features with the model trained. The process of applying HOG follows 3 main stages outlined in figure 2.

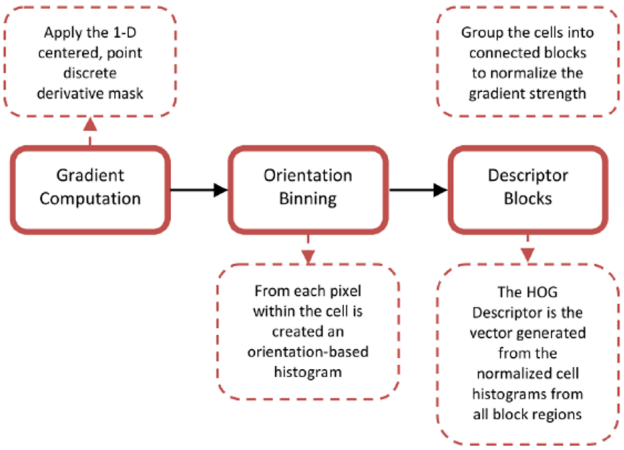

Figure 2. HOG Process

After calculating the vertical and horizontal gradients and finding the magnitude, a histogram is calculated based on 9 bins used to group ranges of orientation (0 - 180) and the result is used to determine the characteristics of the mouth shape in the mouth cut-out. The following parameters were used to in the pre-build OpenCV function *cv2.HOGDescriptor(winsize,blocksize,blockstride,cellsize,nbin)* to achieve the most accurate results:

```
# Hog Parameters for feature extraction
winsize=(64,64)
blocksize=(32,32)
blockstride=(16,16)
cellsize=(8,8)
nbin=9
```

Figure 3. HOG Parameters

3.3. Classifier

The *sklearn* package was used to import the necessary functions needed to train and test the Support Vector Machine (SVM) model used to predict smiles in the input video stream. The SVM looks for the best fitting hyperplane which can separate every point from the two data categories (smile, non-smile) to form a model that will be used to classify future inputs. Since SVM is a binary classification method, it works well for applications that only require a Yes vs No categorization of the inputs such as the well-known Dog/Cat classifier. Similarly, our application needs a 0/1 output from the model which is why SVM was chosen as the main approach for this project.

The remaining images from the pre-processing stage were separated into a 90/10 split where approximately 90% of the 3878 images (3500) were used to train the model and the remaining 10% (378) images were used to test the SVM model created. The images chosen for training and testing were randomized to using the python random package to eliminate any bias. After the training and testing processes were completed an accuracy of 87% was achieved using the formula in figure 4.

Moreover, the output the confusion matrix shown in figure 5, displays much higher values on the diagonal indicating that only 12 "non-smile" and 8 "smile" images were predicted incorrectly out of the 378 images tested.

3.4. Results and Remarks

Although the SVM model was able to achieve a result of 87% when tested with several input images from the GENKI-4K data-set of similar sizes and face orientations, the application of this model on a real-time input has

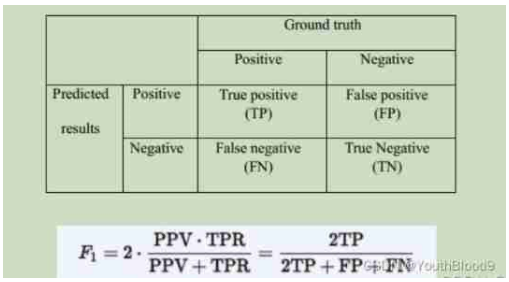

Figure 4. Accuracy Formula

```
Confusion Matrix =
 [[126   8]
 [ 12 184]]
```

Figure 5. Confusion Matrix

many more considerations to be taken into account such as lightning, and face angles. When tested on a real-time feed the accuracy was much less than 87% as it was heavily dependent on the filming environment and very sensitive to slight changes in brightness. The following figures are some of the results saved from live video feeds.

Figure 6. HOG + SVM Results

The accuracy and stability of such an application can be greatly improved by using a deep learning neural networks such as the Convolutional Neural Networks, which will be discussed in the next section.

4. Secondary Approach

For this approach the SMILEs dataset was used instead to provide the deep learning model with more samples and uniform sizes. The dataset contains 13,165 grayscale images of tightly cropped faces with size 64x64, labeled as "smiling" or "not smiling". In this approach the tensorflow, keras and sklearn packages are used to train the CNN model to determine whether the face detected in a live video feed is smiling in real-time.

4.1. Pre-processing and Feature Extraction

Although the close cropping of faces in the dataset of images makes it more convenient for training purposes, the images still contain background noise which needs to be removed in order to extract the desired ROI of the faces. This can be done use the Haar cascades method, which can be downloaded as an xml file cascade classifier and applied using OpenCV. This will be done by firstly loading in the images from disk and converting them to a fixed size of 28x28 pixels in case of any anomalies. We will be using the LeNet-5 architicutre to train our CNN model and the images will be converted into keras compatible arrays and added to a list for training by the LeNet serialized weights. The dataset will then be split into 80% for training and 20% for testing purposes.

4.2. Training and Testing

To train the LeNet model, we initialize the structure which takes the 28x28 input images using the LeNet.build() method. The LeNet is trained for a total of 20 epochs using the supplied class weights and the following results were achieved:

	precision	recall	f1-score	support
not_smiling	0.95	0.91	0.93	1895
smiling	0.79	0.88	0.83	738
accuracy			0.90	2633
macro avg	0.87	0.90	0.88	2633
weighted avg	0.91	0.90	0.90	2633

Figure 7. CNN Training Accuracy

Once the CNN is trained, the weights are serialized and saved. We can then create a learning curve to visualize the performance of the model. As shown in figure 7 above, the accuracy of the model is approximately 91% which is an upgrade to the HOG and SVM approach used initially.

As can be seen from figure 8, after the 5th epoch the total accuracy starts to stabilize to around the 91% mark. Further training past 10 epochs would result in a smaller validation loss but would not affect the weighted average average accuracy greatly.

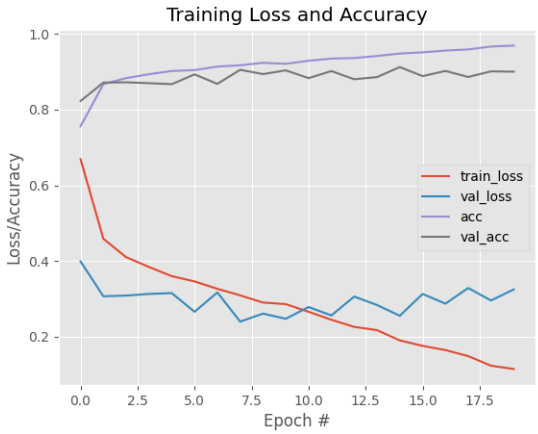

Figure 8. Learning Curve

4.3. Detection and Results

Now that the model has been trained and is ready to be used as a classifier for the video stream input, we can use the Haar Cascade method discussed in the per-processing section to extract the face and mouth shape in real-time and feed it to the model. This is done by looping over the frames from the captured web cam video stream and comparing each frame with the pre-trained LeNet model. The result is displayed as a string on the top of the detected face in real-time.

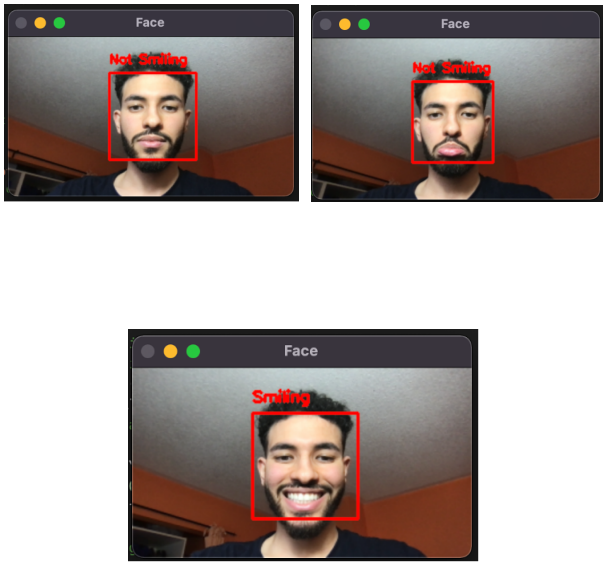

Figure 9. CNN Detection Results

5. Conclusion

In conclusion, a Histogram of Oriented Gradients (HOG) was used in this paper to extract the features necessary that differentiate a smiling mouth from one that is not. This method of feature extraction was combined with a Support Vector Machine (SVM) that was trained on a set of 4000 images of smiling and non-smiling faces and tested with a resulting accuracy of approximately 87%. This method worked well on still images but was not very effective in tracking smiles in real-time as it did not account for the difference in brightness levels and the constant change of face orientation and size in the video stream. The main approach could be further improved by normalizing the face orientation in the video feed and adjusting for maximum view and a more accurate indication of the mouth position. Furthermore, a separate method for mouth tracking could be useful in improving the accuracy of this approach for real-time tracking in particular as there is always a slight unintentional change of mouth positions within the provided video stream.

A second approach was also discussed in an attempt to improve the accuracy of the HOG and SVM method. The Haar Cascade algorithm was used instead of the Histogram of Oriented Gradients to extract the face features necessary to detect smiles in the input feed. Moreover, a Convolutional Neural Network was train using deep learning on a set of approximately 13,165 images that were pre-processed to show only a gray scale cutout of the smiling and non-smiling faces. This method was more successful in detecting smiles in real-time as the testing was able to return an accuracy of approximately 91%. Although this can be attributed to the larger data-set used in train the CNN model; this second approach has been proven to be more effective as it combines the advantages of several deep learning features which are better suited towards this type of application. This method could also be improved using a higher number of epochs but that runs the risk of over-fitting which is a balancing issue.

References

[1] Malesevic, Jovana Jojic, Jovana. (2012). One approach to smile detection. 2012 20th Telecommunications Forum, TELFOR 2012 - Proceedings. 1717-1720. 10.1109/TELFOR.2012.6419558.

[2] Syaputra, R., Syamsuar, D., amp; Negara, E. S. (2021, February 1). IOPscience. IOP Conference Series: Materials Science and Engineering. Retrieved April 9, 2022, from https://iopscience.iop.org/article/10.1088/1757-899X/1071/1/012027

[3] K. Zhang, Y. Huang, H. Wu and L. Wang, ”Facial smile detection based on deep learning features,” 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), 2015, pp. 534-538, doi: 10.1109/ACPR.2015.7486560.

[4] Ali, I., amp; Dua, M. (2020, April 16). Smile detection using data amalgamation. Procedia Computer Science. Retrieved April 9, 2022, from https://www.sciencedirect.com/science/article/pii/S1877050920308632

[5] The mplab genki database. INC. (n.d.). Retrieved April 9, 2022, from https://inc.ucsd.edu/mplab/398/