Abstract

Various machine learning algorithms have been implemented for the detection and classification of human faces for binary and multiclass classification using Python programming language with several image manipulation and machine learning library. Analysis is done at the end of each task to verify and compare the performance between different algorithms and methods. With this comparison, the best machine learning model is selected and trained to the corresponding dataset with accuracy achieved of more than 84% for all tasks. The code is accessible via author’s GitHub repositories.

Introduction

Face recognition is an extensive problem of identifying or verifying faces in images or videos. Human has evolved and been around with each other for as long as civilization exist. Hence, it is quite effortless for human to recognise and identify gender, age and face expression of a person. Nevertheless, this task is quite tough and complicated for a machine, given that some obstacle exists in the image/video such as having glasses, facial hairs and having low light intensity of an image or low-quality video. Regardless, it is not impossible for machine to identify it with high accuracy.

Face recognition has broad application in now digital world. It has been widely used in law enforcement and surveillance as well as entertainment industries. For example, facial recognition is heavily used in the immigration border all over the world for identification and verification of passport bearer.

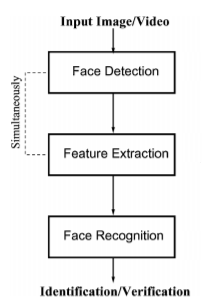
Throughout this module, students have learnt various machine learning model theories from supervised to unsupervised model, regressions and classifiers. The aim of this assignment is to train and test machine learning models and perform binary and multiclass classification for a given datasets.

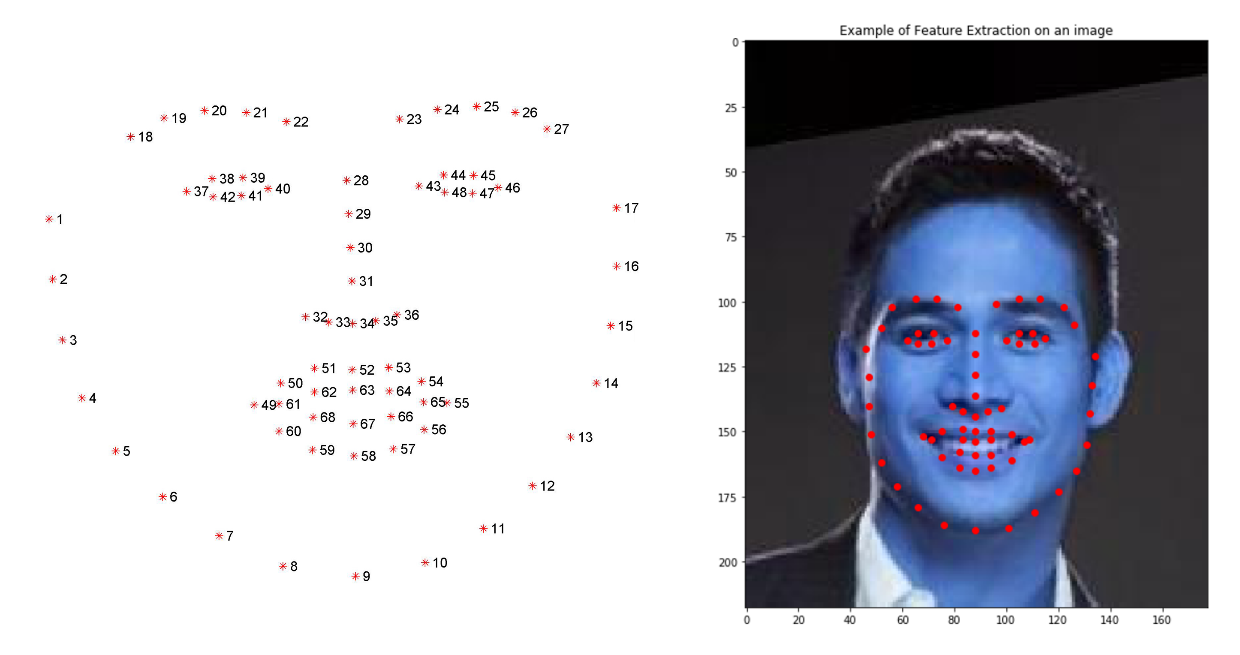
There are two datasets for this assignment; “CelebFaces Attributes Datasets” and “Cartoon Set”. For the first, the dataset consists of 5000 positive images of celebrities faces. The dataset is fetched from Yang et. al. paper [21] and is used for task A1, gender classification and task A2, smiling detector. Corresponding labels to the image have been provided in comma-separated values (csv) file.

Meanwhile for the later dataset, it contains 10000 positive images of random cartoon faces with various combination of face shape, hairstyle with various colour, facial hair, and eye colour, fetched from google dataset [22]. Mentioned dataset is used for task B1, face shape recognition and task B2, eye colour recognition. Besides, corresponding labels to the cartoon dataset have also been provided in a csv file.

Literature Review

Facial recognition is pioneered by Woodrow Bledsoe with his paper from 1964 to 1966 [1,2], in which some people may label him as the father of facial recognition. However, it is not until recent decades facial recognition has been booming and is used almost anywhere possible.

 Zhao et. al. has produced extensive literature review on facial recognition in 2003 [3], particularly on still images as well as video-based face recognition. As mentioned in the paper, facial recognition system of a generic face consists of three main procedures; detection and rough normalisation of faces, feature extraction and normalisation of faces, and lastly identification and verification. However, it is quite common for the face detection step and feature extraction is done simultaneously given that both steps are highly correlated and dependent.

Feature extraction can be done in 3 different ways; 1) generic method based on edges, lines and curves, 2) feature-template-based method, 3) structural matching method. As for this, several algorithms have been developed in order to pinpoint the fiducial facial key points to extract facial information [5]. Three most famous algorithm and reasonably easy to be used are “Histogram of Oriented Gradient” (HOG) feature [4], Haar cascade classifier [6] and Viola-Jones algorithm [17]. These algorithms output facial key points in term of coordinates (x and y), localising the structure of the face relative to the image [18]. Image below shows the feature extraction mapped onto a single image.

As for gender classification using frontal face images, there are few published papers on how it is done. In Basha et. al. paper [7], author has used Continuous Wavelet Transforms algorithm for features selections for each male and female image. Then the features are trained using Support Vector Machine (SVM) with Linear kernel classifier to identify the gender of images. The paper claimed to have an accuracy of 98% using dataset called ORL database. In addition, gender classification can be done by using Local Binary Patterns (LBP) and SVM with radial basis function (RBF) kernel on unconstrained scenario as proposed by Caifeng Shan [8]. The LBP is used to describe faces and with the help of Adaboost, LBP features are carefully selected. Lastly, Li et. al. proposed almost similar method on gender classification [9] in 2012 except that more image processing is done prior to LBP features which includes edge contour detection using Sobel edge detection algorithm. Both methods claimed to achieve more than 90% accuracy.

Moving on to smiling detector, Hromada et. al. has developed algorithm for detecting smiling and compiled it to OpenCV open source library [10]. The feature extraction is done using ROCR library and it is classified using Haar Cascade classifier. The proposed method has achieved accuracy of 77% on using Japanese Female Facial Expression (JAFFE) database. Next, a paper by Whitehill et. al. has been published in 2009 [11]. Author has experimented using five different feature extractions which are Gabon Energy Filters (GEF), Box Filters (BF), Edge Orientation Histogram (EOH), BF + EOH, and LBP. As for the classifier algorithm, author uses GentleBoost which minimises X-square error between labels and model predictions [12] and linear SVM. From this experiment, author claimed by using GentleBoost works well with BF but terrible with GEF and vice versa with SVM. However, SVM works better with EOH.

Meanwhile, there are few typical pre-trained models using neural network such as DeepFace [13], FaceNet [14], VGGface [15], and SphereFace [16] that has been used for face recognition. DeepFace uses a nine-layer Convolution Neural Network (CNN) with few locally connected layers while for VGGface uses dataset fetched from the Internet and fine-tuned the networks via triplet loss function. As for FaceNet, by using huge private dataset, it trains the data on GoogleNet and adopted a triplet loss function like VGGface in order to achieve high performance. Lastly, SphereFace uses 64-layer ResNet architecture and use angular softmax (A-Softmax) loss to map face features with angular margin.

Description of Models

From literatures reviewed on previous section, some rough ideas have been developed on how to tackle the problem given in each task. With these ideas, author tries at least two different machine learning model for each task on Jupyter Notebook using ipynb file before proceeding with python file.

Task A1

A picture containing screenshot, sky

Description automatically generatedFeature extraction landmarks produced by Sagonas et. al. [18,19,20] is used to fetch 68 fiducial key points of the image in terms of coordinates. The feature inputs are then trained using two different kernel of Support Vector Classification (SVC); Linear kernel and RBF kernel. The rationale for using Linear kernel is as proposed by Basha et. al. [7] meanwhile for the RBF kernel is inspired from Caifeng Shan’s paper [8]. With both classifiers are implemented, the performance of each model is evaluated in terms of accuracy score, precision score, recall score and f1 score.

A screenshot of a cell phone

Description automatically generatedThe result is tabulated below. As it can be seen, SVC with linear kernel has better performance than SVC with RBF kernel. Hence, author has proceeded task A1 with SVC with linear kernel.

Task A2

Smiling detector is done in two ways; 1) using HaarCascade classifier for face along with smile detector from OpenCV, 2) using feature extraction landmarks by Sagonas et. al. [18,19,20] and SVC. For the first method, the rationale of using this due to the algorithm is easy to use as this has been embedded inside OpenCV library which developed by Hromada et. al. [10]. As for the second method, author has been inspired by Whitehill et. al. paper [11] which apply Histogram of Oriented Gradient (HOG) feature extraction along with SVM classifier. From the second method, author experiments more on two different kernels of SVC; linear and RBF. Beside comparing between these two classifiers with 68 features, author also tests both classifiers with reduced features which contains only 37 coordinates. This 37-feature input only includes coordinate for jaw and mouth which is reasonable to detect smile. The performance for all five methods is tabulated below.

A screenshot of a cell phone

Description automatically generated

The result from Haar Cascade can be seen to have low accuracy and the worst recall score. This is due to the fact that Haar Cascade has terribly high false negative. A screenshot of a cell phone

Description automatically generatedFurthermore, SVC with linear kernel has slightly better performance compare to RBF kernel. Lastly, since performance between original features and reduced features has almost identical, author has decided to use reduced features as this uses lower computation power as well as shorter training time.

Task B1

A close up of a map

Description automatically generatedFor this task, two different methods are used. Firstly, data is feature extracted using feature extraction landmarks along with SVC Linear kernel as proposed by Basha et. al [7]. For the later method, author introduced Convolution Neural Network (CNN) Sequential model with 13 layers to train, test and validate the model. The rationale for this model is used due to feature extraction done in the first method has less accuracy in terms of the coordinates as well as some images cannot be detected. Figure below shows some examples on feature extraction done for cartoon set.

For the first image, feature extraction is well done. However, feature extraction cannot be done for the next two images even though the image is relatively fine. Meanwhile, the last two images have coordinates wrongly placed. Hence, author proceeds with CNN model to train the data.

Task B2

Eye colour recognition is done at first by using Haar Cascade Eyes classifier. For obvious reason, this is done due to the OpenCV library has built-in Haar Cascade Eyes classifier. Next, eye colour recognition is done by extracting the relevant Blue-Green-Red (BGR) colour produced by Histogram out of images cut using mask created by Hough Circles. The features produced is then trained using three different classifiers; Random Forest Classifier, K-Neighbour Classifier and SVC with Linear kernel. The rationale for this method is to extract relevant BRG for eye colour and trains using three different classifiers to compare between each classifier. The performance for all three classifiers is tabulated below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Classifier** | Random Forest | K-Neighbors | SVC (linear kernel) |
| **Accuracy Score** | 0.6478232618583496 | 0.6556205328135153 | 0.6335282651072125 |

As it can be seen, this method has relatively low accuracy. As for author knowledge, this is due to the image cropped using Hough Circles contains also skin tone either beneath or above the eye. Hence, the BGR colour extracted also contained the skin.

Lastly, author tries CNN Sequential model with 13 layers to classify the eye colour using images cropped by using coordinates produced by Haar Cascade Eyes classifier. The rationale for this is to save time and processing power as this shrink the image size to more than one tenth.

Implementation

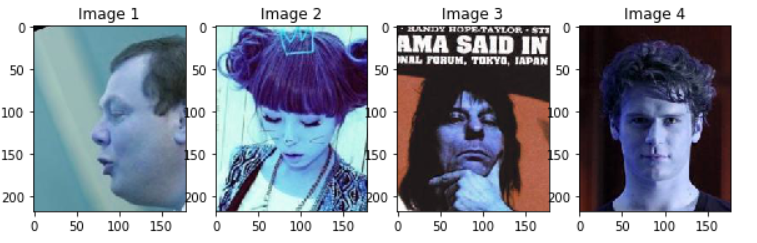
Some generic libraries are imported to be used for all the task. OS library is used to get and set directory to be accessed by python. NumPy library is used for mathematical manipulation of data while Pandas library is used to create and access dataframe. OpenCV library is used for image processing and manipulation with support from Keras library as well. Meanwhile, Scikit-Learn library is imported for some data processing and evaluation including splitting data and evaluate model performance. Lastly, matplotlib library is also imported and used for plotting graph. Apart from this generic library, some other library and function is used for specific task and will be discussed in subsection below.

Task A1

Firstly, all the generic library is imported, and directory is set for the task. Next, feature extraction is done by calling a function named “extract\_features\_labels” that has been provided in the laboratory of this module. However, the function is edited and tailored for gender classification. Steps for this function is shown below.

1. Target image is loaded using “image” module fetched from Keras library, under “preprocessing” module.
2. Target image is then converted into array using method inside “image” module, resized into unsigned integer of 8 bits (uint8), and converted into grayscale by using OpenCV library.
3. By using dlib library, image is processed and coordinates of 68 fiducial coordinates of the face are returned.
4. Target image’s coordinates in terms of array is appended into a new array and the corresponding gender label to the image is appended to another array which is then converted from -1 and 1 to 0 and 1.
5. Lastly before returning both features and label of a target image, the data is converted into NumPy array.

Worth to note, out of 5000 positive images, 200 images cannot be feature extracted. Figure below shows some example of images that cannot be feature extracted.



As it can be seen, image 1 and 2 have different orientation and not front-faced. Meanwhile for image 3 the image is blocked by the hand while image 4 has shaded half of the face which fails the algorithm to detect the contour.

With 4800 images with 68 features in term of x and y coordinates, the data is reshaped into NumPy array of 136 (68 points with x and y coordinates). With this, the features and corresponding labels are split into training and testing with 0.2 test size using Scikit-Learn’s train\_test\_split method.

As the data will be trained using Support Vector Classification (SVC) with linear kernel, the hyper-parameter for the model is tuned using GridSearch from Scikit-Learn library. Although this is an exhaustive method, this method able to provide author some hindsight on values of C (cost of misclassification) and gamma (parameter of kernel). As for this, author has decided to use low C (0.01) and low gamma (0.001). Low C gives high bias but low variance due to small penalise on cost of misclassification but gives higher accuracy. However, author uses low gamma to compensate this to provide more variance with less bias.

The model is then trained with this hyperparameter along with linear kernel for SVC and validated using 5-fold cross validation from Scikit-Learn library to check if the model is overfitted. Next, learning curve is plotted to again verify this model.A screenshot of a cell phone

Description automatically generated

As it can be seen, with the training images given by the dataset, both training score and cross-validation score almost converge which indicates the number of datasets is reasonably good. Author feels the model has balanced bias-variance ratio as the gap between training score and cross-validation score is neither too far apart (indicating high variance) nor too near to each other (indicating high bias).

Task A2

Generic libraries are imported, and directories are set. Next, features extraction is done by using function called “extract\_features\_labels” as mentioned in previous task. This function is again edited and tailored to smiling detector. Steps for feature extraction are as follow:

1. Target image is loaded using Keras’s Image module.
2. Target image is converted into array using Keras’s Image method called img\_to\_array and target image is shrunken to uint8 size.
3. Target image is converted into grayscale using OpenCV library and image is processed and 68 fiducial key points of image is extracted using dlib library.
4. 31 coordinates are omitted, and only coordinates related to the jaw and mouth are extracted and appended to a new array.
5. The corresponding smiling label of the target image is appended to another array with the value of -1 and 1 is converted into 0 and 1.
6. Both 37-feature array and label array are converted into NumPy array.

As mentioned in previous task, feature extraction can only be done to 4800 images out of 5000 positive images.

The data is then reshaped and split into training and testing with 0.2 test size using Scikit-Learn’s train\_test\_split function. Hyperparameter is tuned to the following classification task using GridSearch from Scikit-Learn library and the C and gamma is found to be 0.1 and 0.001 respectively. This value has optimised between bias-variance tradeoff and tailored to high scoring in terms of accuracy.

Next, data is trained using SVC with Linear kernel with these hyperparameter. The data is validated using 5-fold cross validation using library from Scikit-Learn and learning curve is plotted to verify the model. Figure below shows the learning curve for smiling task.

A screenshot of a cell phone

Description automatically generated

This proves the bias-variance trade-off is approximately at optimum.

Task B1

As previous tasks, generic library is imported, and directories are set using OS library. By opening the label’s csv file, filename of images and corresponding face shape is extracted and Pandas’ dataframe is created with “Filename” and “FaceShape” as columns. First five rows of dataframe is showed below. The counts for each label are also displayed below to verify data is balanced.

A picture containing object

Description automatically generatedA screenshot of a cell phone

Description automatically generated

Next, dataframe is split into training and testing using Scikit-Learn’s train\_test\_split function with default test size of 0.25. By using ImageDataGenerator method imported from Keras library under preprocessing.image, augmentation of image is done. The image is horizontal-flipped, vertical-flipped and rescaled to 8 bits. The training dataframe is also split into training and validation using this method by setting validation\_split to 0.25. Finally, with batch size of 32 and target size of 32 times 32 is set, and training and validation is produced using method “flow\_from\_dataframe” within Keras’ ImageDataGenerator.

Convolution Neural Network with Sequential model is then defined with 13 layers. The summary of the model is showed below. The model uses adaptive learning rate optimiser (Adam optimiser) and trained to achieve accuracy metric.

A screenshot of a cell phone

Description automatically generated

With the model is set, the data is trained using model.fit\_generator with epoch is set to 25. Training and validation loss are plotted against number of epoch and showed in figure below.

A screenshot of a map

Description automatically generated

As it can be seen, as epoch increases beyond 16, the losses change very slightly. Hence, it can be concluded that the training should stop at 16 to save processing power as well as time. Confusion matrix from validation dataset is plotted to verify the model’s training performance.

A screenshot of a cell phone

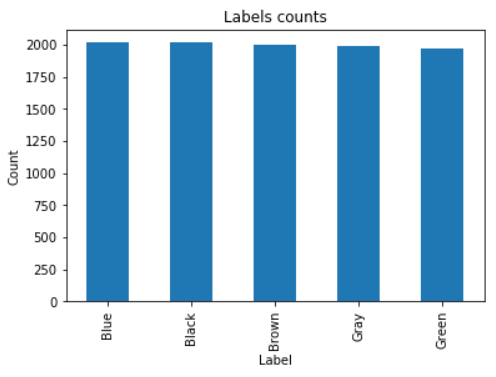
Description automatically generated

From this, it can be concluded the model has does not overfit the training data.

Task B2

Generic libraries are imported and directories for corresponding files are set as usual. A Pandas’ dataframe is created with columns of “FileName”, “EyeLabel” and “EyeColour”. The data for FileName and EyeLabel are then fetched from csv file and inserted into the dataframe. Meanwhile for EyeColour, using corresponding EyeLabel, the respective colour of “Brown”, “Blue”, “Green”, “Gray” and “Black” is appended to the dataframe. The first 10 data of the dataframe is showed as below. The data for each of labels is also plotted to verify the data is balanced.

A screenshot of a cell phone

Description automatically generated

Next, the dataframe is split into training and testing with 0.2 test size using train\_test\_split function from Scikit-Learn library with data is shuffled. As the task is concerned about the eye colour of target image, author has created a function to crop image from the dataset and saved the cropped image into another folder. The crop function uses rectangle produced by Haar Cascade’s detectMultiscale function from for eye cascade, named “haarcascade\_eye\_tree\_eyeglasses.xml”.

Next, data is augmented using ImageDataGenerator from Keras’s image library. All the images are rescaled to 8 bits image, with vertically and horizontally flipped. The validation split is set to 0.25. With ImageDataGenerator is set, training and validation dataset are extracted using method within ImageDataGenerator called “flow\_from\_dataframe” as demonstrated in previous task.

CNN model is then set with 13 layers and summary of the model is showed below. The model is also optimised using Adam optimiser.

A screenshot of a cell phone

Description automatically generated

The data is then trained with using “model.fit\_generator” while epoch initially set to 25. A graph of training and validation loss is plotted and showed below.

A close up of a map

Description automatically generated

Training and validation loss have small changes beyond 9 epochs. Hence, author decided to train the data until 9th epoch to save time and processing power. As for if model is overfitted or not, confusion matrix for validation dataset is created and showed.

A screenshot of a cell phone

Description automatically generated

As it can be seen clearly, the data is not overfitted. However, the accuracy is lower than expected. Regardless, author feels this is reasonable as some eye images are blocked by thick opaque glasses and there is absolutely no way to predict the eye colour.

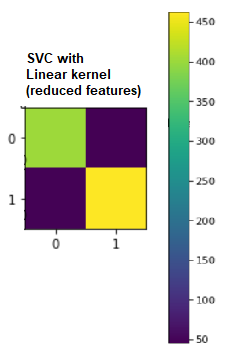
Experimental Results and Analysis

With each task’s classifier has been decided as well as the hyperparameter is tuned and optimised, the test set is used to predict the data with corresponding classifier for each task. Next, the accuracy score is measured using Scikit-Learn’s metric evaluation library. The training accuracy, validation accuracy and testing accuracy for each task is tabulated below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task** | **Model** | **Train Accuracy** | **Validation Accuracy** | **Test Accuracy** |
| **A1** | SVC (linear kernel) | 0.9336 | 0.9241 | 0.9146 |
| **A2** | SVC (linear kernel) | 0.8971 | 0.8863 | 0.9021 |
| **B1** | CNN (13 layers) | 0.9866 | 0.9826 | 0.9712 |
| **B2** | CNN (13 layers) | 0.8563 | 0.8664 | 0.8456 |

For each task, confusion matrix for test dataset is plotted and showed below. Figure 1,2,3 and 4 is the confusion matrix for gender classification, smiling detector, face shape detector and eye colour classification respectively.

A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated

This indicates the model’s true positive and true negative outperformed false negative and false negative.

Conclusion

There are four tasks in total, gender classification and smiling classification which both can be grouped into binary classification task, and face shape classification and eye colour classification, grouped into multiclass classification task. For all tasks, different methods for face detection and classification models are tested along with comparative analysis between them. Dataset for each binary and multiclass classification is celebrity faces and cartoon set respectively.

In general, binary classification is done by using support vector machine model while for multiclass, convolution neural network is introduced. Accuracy achieved by author for all the task has at least 84%. Implementation of neural networks for task B1 and B2 takes longer time compare to using support vector machine for task A1 and A2. In addition, using neural networks, there are some hidden layers that is not accessible.

The number of data provided is sufficient. However, machine learning’s rule of thumb usually implies higher number of data leads to better result.

1. Bledsoe, W. W. 1966a. Man-Machine Facial Recognition: Report on a Large-Scale Experiment, Technical Report PRI 22, Panoramic Research, Inc., Palo Alto, California.
2. Bledsoe, W. W. 1966b. Some Results on Multicategory Patten Recognition. Journal of the Association for Computing Machinery 13(2):304-316.
3. Face Recognition: A Literature Survey
4. Histograms of Oriented Gradients for Human Detection - Navneet Dalal and Bill Triggs
5. Facial Landmark Detection: A Literature Survey
6. Evaluation of Haar Cascade Classifiers for Face Detection
7. Face Gender Image Classification Using Various Wavelet Transform and Support Vector Machine with various Kernels.
8. Learning local binary patterns for gender classification on real-world face images
9. Gender classification by combining clothing, hair and facial component classifiers
10. Zygomatic Smile Detection: The Semi-Supervised Haar Training of a Fast and Frugal System: A Gift to OpenCV Community
11. Toward Practical Smile Detection
12. “Additive Logistic Regression: A Statistical View of Boosting
13. . Deepface: Closing the gap to human-level performance in face verification
14. Facenet: A unified embedding for face recognition and clustering
15. Deep face recognition. In BMVC
16. Sphereface: Deep hypersphere embedding for face recognition
17. Rapid Object Detection using a Boosted Cascade of Simple Features
18. 300 Faces In-The-Wild Challenge: database and results
19. A Semi-automatic Methodology for Facial Landmark Annotation
20. 300 Faces in-the-Wild Challenge: The First Facial Landmark Localization Challenge
21. From Facial Parts Responses to Face Detection: A Deep Learning Approach
22. Cartoon Set: An Image dataset of Random Cartoons 2019

@techreport{Bledsoe\_1966a, author = {Woodrow Bledsoe}, title = {Man-Machine Facial Recognition: Report on a Large-Scale Experiment}, series = {Technical Report PRI}, volume = {22}, publisher = {Panoramic Research Inc.}, address = {Palo Alto, California}, year = {1966}}

@article{Bledsoe\_1966b, author = {Woodrow Bledsoe}, title = {Some Results on Multicategory Patten Recognition}, journal = {Journal of the Association for Computing Machinery}, volume = {13}, year = {1966}, number = {2}, pages = {304--316}}