Correlation between Professional Domain and Facial Features based on Face Clustering

Bithiah Yuan

October 21, 2019

1 Introduction

A significant source of information and attributes can be derived from the human face by non-verbal communication [16]. As a result, facial features have been studied extensively in the social-science domain to predict success in reaching reputable leadership positions. In particular, studies have shown that certain facial features contribute to higher salaries and more prestigious employments for CEOs. In application, the relationship between facial characteristics and social attributes can provide an more powerful objective indicator for organizations to idenity and select effective leaders within their domain than broad facial cues such as attractiveness and competence. Results have shown that a human judge can identify business, military, and sports leaders from their faces with above-chance accuracy. However, these results are biased and do not imply the actual leadership qualities of a person [21].

Caused by behaviour experiments from human judgement, the the research of the social attributes and facial features in the social-sciences are limited in scalability, consistency, and generalization. For example, prior familiarity to the faces of the study and personal preferences can affect the results. Therefore, a growing number of social trait judgment studies have been extended and refined to computer vision and machine learning research due to the capability of using massive datasets and large-scale processing capacity [16].

Through a computational framework, [16] examined the relationship between facial traits and the social construction of leadership by a trained model that can predict the outcomes of political elections based on the perceived social attributes of a person's appearance. The results indicate that similar methods can be used to predict behavior in a broad range of human social relations, such as mate selection, job placement,and political and commercial negotiations [16].

Clustering analysis is an unsupervised learning technique that groups data points into clusters based on their similarities. It is useful in grouping a collection of unlabeled data with similar nature into clusters. [27] investigated clustering a large number of unlabeled face images into individual identities present in the data [27]. The workflow shown in Figure1. consists of obtaining face representations of a collection of unlabeled data by a deep neural network. The choice of clustering algorithm then groups the face images according to their identity.

Motivated by the researches in computational social trait judgment and [27], the following paper aims to examine the correlation between a person's profession based

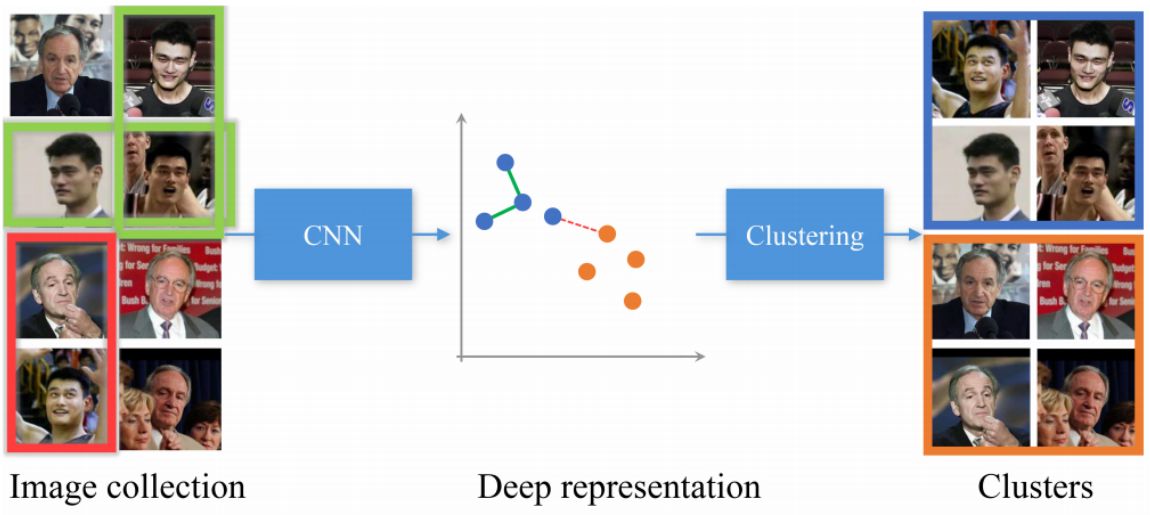

Figure 1: Face clustering workflow [27]

on their facial features through clustering face images. The clustering problem consists of the face representation and similarity metric of the face images and the choice of clustering algorithm [27]. Due to the importance of the underlying face representation in face clustering, this paper further compares different open-source state-of-the-art feature extraction methods based on deep learning.

2 Related Work

2.1 Face Recognition

Face recognition focuses on identifying or verifying the identity of subjects in images or videos [24].

Face recognition systems are usually composed of the following 4 steps [24]:

1. **Face Detection:**
   Detect the position of the faces in an image and returns the coordinates of a bounding box for each face as shown in Figure 2.

2. **Face Alignment:**
   Find a set of facial landmarks with the best affine transformation that fits a set of reference points located at fixed locations in the image. As shown in Figure 3, this step also includes resizing and cropping the image to the edges of the landmarks [5]. More specifically, as shown in Figure 4 given a set of mean landmark locations, the affine transformation makes the landmarks detected in the face image close to the mean [5].

3. **Face Representation:**
   Transform the pixel values of a face image into a low-dimensional discriminative feature vector, also known as an embedding.

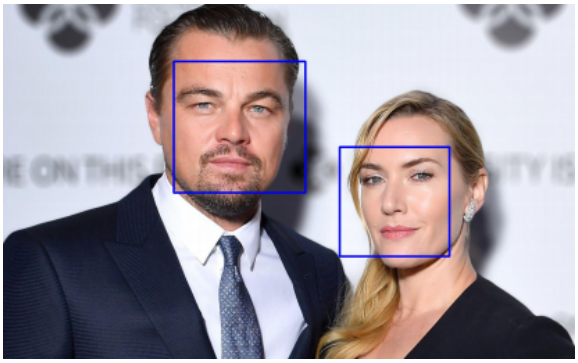

Figure 2: Face Detection [24]

Figure 3: Face Alignment [24]

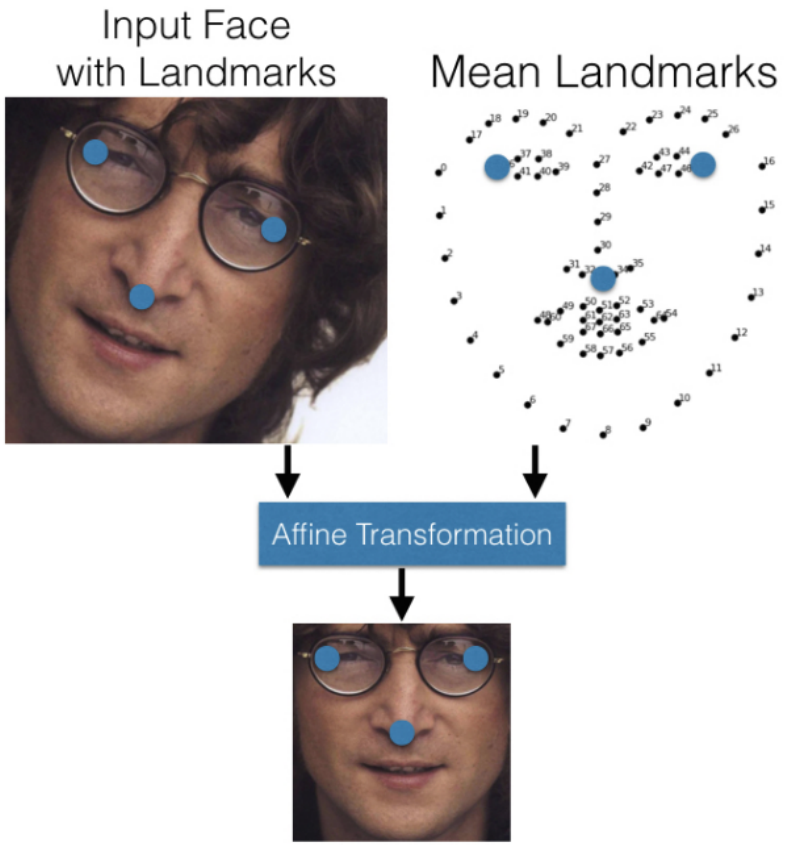

Figure 4: Applying affine transformation so that the face image is closer tothe set of mean landmarks. [5]

4. **Face Matching:**
   Compute similarity scores from feature vectors.

2.2 Face Detection

2.2.1 Histograms of Oriented Gradients

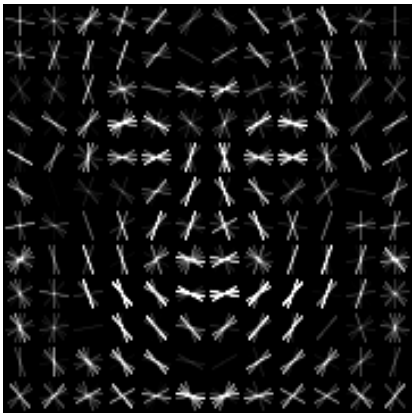

Figure 5: Trained HOG detector on multiple faces [24]

Figure 6: Hog representation of a face [12]

A traditional method for face detection is the Histograms of Oriented Gradients (HOG) descriptors, which utilizes the distribution of local intensity gradients or edge directions to characterize local object appearance and shape in an image. HOG divides the image into small grids, where each grid accumulates a histogram of gradient directions or edge orientations over the pixels of the cell. The combination of all the histogram in the cells form the face region. The cells are then normalized for better invariance to illumination, shadowing, and other variations. The normalized local histograms of image gradient orientations in a dense grid as features are then trained to classify the region of the face in an image [10]. When encountering a HOG representation of a new face image as shown in Figure 6, the part of the image that looks most similar to a trained HOG detector as shown in Figure 5 will form the region of the face.

2.2.2 Multi-task Cascaded Convolutional Networks

Another than using the tradition HOG representation and landmark detectors, face detection and alignment can also be done using CNNs. In particular, Multi-task Cascaded Convolutional Networks (MTCNN) is a widely used method to predict face and landmark location. The framework has a cascaded structure with three stages of deep CNNs shown in Figure 7. [28]

The image is first resized to different scales to build an image pyramid and is the input of the three stages:

1. **Proposal Network (P-Net):** Obtain candidates that will serve as potential positions of the bounding boxes [8].

2. **Refine Network (R-Net):** Using the image and the results of the first prediction of the bounding boxes, reduce false positives to get the final box boundaries [8].

3. **Output Network (O-Net):** Outputs five facial landmark positions [28].

Both the predicting the bounding box regression and facial landmark localization uses regression to minimize the Euclidean loss between the candidate positions of the

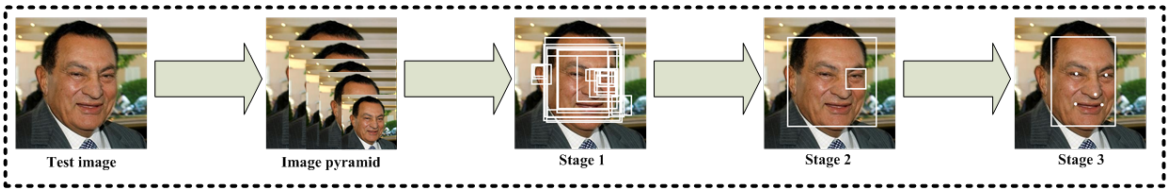

Figure 7: MTCNN: Cascaded structure with three stages of deep CNNs. [28]

bounding boxes, landmark coordinates and the ground truth. The ground truth of the bounding boxes are the left, top, height, width of the box. The ground truth of the facial landmarks are the coordinates of the left, right eye, nose, left and right mouth corner [28].

2.3 Face Representation

Face representation is conceivably the most important component in the system. However, challenges occur in real world (in-the-wild) images due to variations ranging from head poses and illumination conditions to aging and facial expressions [24].

Traditional techniques include using statistical methods such as Principal Component Analysis (PCA) to represent faces as a combination of eigenvectors [5]. The top-performing face representation techniques use deep learning methods based on convolutional neural networks (CNNs) [5] since they are able to achieve very high accuracy by learning robust features due to the availability of large-scale faces in-the-wild datasets on the web [24].

2.3.1 Convolutional Neural Networks

As shown in Figure 8 a neural network feeds the input into many layers of function compositions followed by a loss function which measures how well the neural network models the data. Each layer is parameterized by a vector or matrix θ_i and the aim is to optimize the loss function iteratively by finding the optimal gradients $\delta L \,/\, \delta\theta_i$ which are computed with the backpropagation [5].

Residual networks (ResNets) is a popular network architecture for face recognition. ResNets introduces a shortcut connection to learn a residual mapping which contributes to information flow across layers and allows the training of much deeper architectures [24].

A common approach to training CNN models for face recognition is use a classification approach, where each face image in the training set corresponds to a class. When recognizing a new face image, the classification layer is discarded and the features of the previous layer are used as face representations. The downsides of this approach is that it doesn't generalize well to new face faces and that the representation size per face is large and inefficient [26].

Another approach is to learn the features for face representation directly by optimizing the distance between pairs or triplets of faces, in which the distances measure the similarity between faces [24] [26].

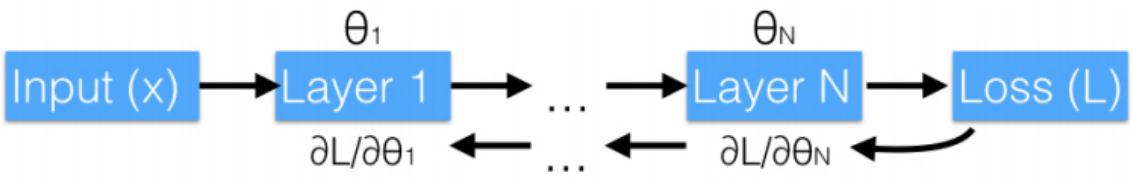

Figure 8: Neural Network Traing Flow. [5]

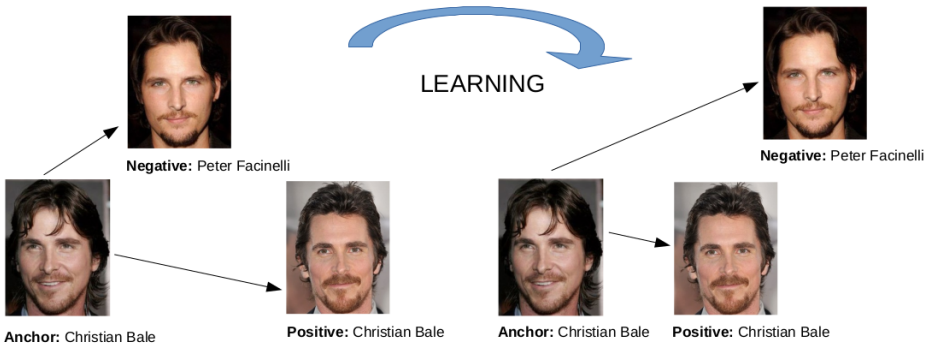

Figure 9: The loss of identical faces are minimized and the loss of distinct faces are maximized by the triplet loss function [1] [2].

2.3.2 Triplet Loss Function

When learning the face features directly, the choice of loss function has a great influence on the accuracy. One of the most used metric is the triplet loss function. The goal of the loss is to separate the distance between two aligned matching (positive) face images and a non-matching aligned (negative) face image by a distance margin. The result is a feature vector $f(x)$, known as embeddings, from a face image x to a compact Euclidean feature space in $\mathbb{R}^d$. The distance of the embeddings will be small if the faces are identical and large if the faces are distinct [26].

More specifically, as shown in the example in Figure 9. the distance between an anchor face image, x_i^a is minimized by the loss and will be closer to all other positive face images x_i^p than the negative face images x_i^n where the distance is maximized by the loss. For each triplet i, the following condition needs to be satisfied:

$$\|f(x_i^a) - f(x_i^p)\|_2^2 + \alpha < \|f(x_i^a) - f(x_i^n)\|_2^2$$

where α is a margin that from the positive and negative pairs [24].

For N possible triplets, the loss being minimized is:

$$L = \sum_i^N \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha\right]_+$$

[26].

As shown in Figure 10 using a neural network, the triplet loss is computed and it's gradient is backpropagated through the network to the unique images [5].

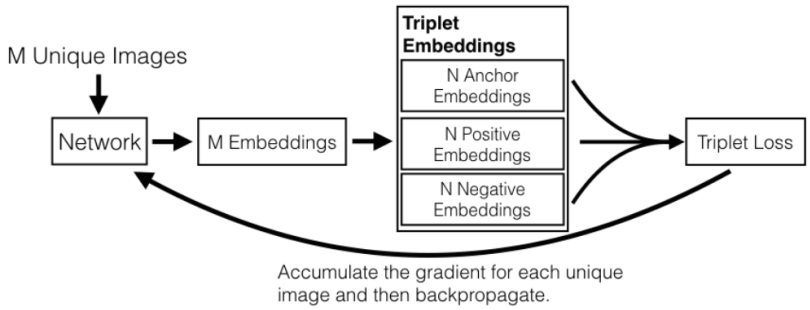

Figure 10: Learning the embeddings by optimizing the gradients of the triplet loss function [5].

3 Face Representation Methods

The following section examines open-source state-of-the-art face feature extraction methods.

3.1 FaceNet

FaceNet is a method that uses a deep CNN along with the GoogLeNet style Inception models and the triplet loss function to directly optimize the face embeddings. The faces of images are first detected and aligned with MTCNN [25], the resulting face images are 160 x 160 pixels and serve as the input for the FaceNet model. The structure of FaceNet consists of a batch input layer and a deep CNN followed by L_2 normalization, which results in the face embedding. This is followed by the triplet loss during training as shown in Figure 11 [26].

Between 100 to 200 million face images consisting of about 8 million different identities were used for training. The large dataset of labelled faces consist of various poses, illuminations, and other variations. [26].

The pre-trained model (20180402-114759) of open-source FaceNet implementation [25] used in the experiment from Section 5 was trained using the VGGFace2 dataset. The faces of the dataset was detected and aligned using MTCNN . The dataset contains 3.31 million images of 9131 identities, with an average of 362.6 images for each person. The images were downloaded from Google Image Search and have large variations in pose, age, illumination, ethnicity and profession. The model uses the Inception ResNet v1 architecture and has an accuracy of 99.65% on the Labeled Faces in the Wild (LFW) benchmark [25]. [25]'s implementation results in a 512-dimensional feature vector.

3.2 Dlib

[13] built a face recognition method using Dlib-ml, which is an open source library for developing machine learning software in C++ [19]. [13] used the HOG face detector from Dlib, which the HOG representation was trained with a linear classifier (SVM) [18]. The face representation model uses the ResNet architecture with 29 convolution layers and the triplet loss function to learn the embeddings [20]. The resulting feature embedding is a 128-dimensional vector [20].

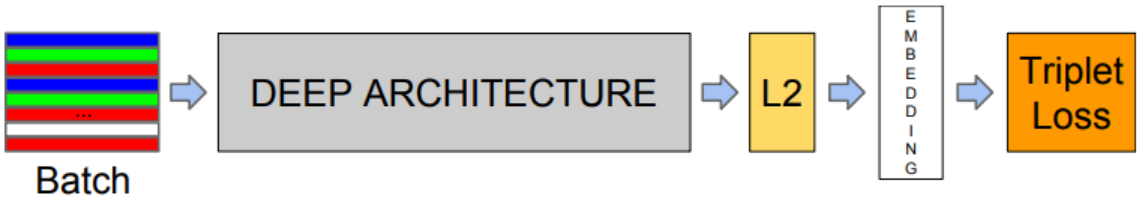

Figure 11: FaceNet model structure [5]

A dataset of about 3 million faces and 7485 unique identities from a combination of the face scrub and VGG dataset as well as a large number of other images scraped from the internet was used for training. The pre-trained model has an accuracy of 99.38% on the LFW benchmark [20].

3.3 OpenFace

OpenFace uses Dlib for face detection and alignment. The input images after alignment are 96 x 96 pixels. OpenFace was trained with 500,000 images from a combination of CASIA-WebFace and FaceScrub. The face representation is obtained using a modification of FaceNet's architecture, which the number of parameters are reduced. The resulting feature embedding is a 128 dimensional vector [5].

3.4 ArcFace

[11] introduced a new loss function, additive angular margin (ArcFace), that uses a geometric interpretation for learning the discriminative features for face representation [11]. After applying MTCNN for face detection and alignment [15] get the 112 x 112 face input images, ArcFace further adjusts a face image by rotating an image to a straight face as shown in the comparsion in Figure 12. Consequently, the positions of eyebrows, eyes, nose, and mouth in different images are consistent and increases the effective when computing the similarities between the embeddings. The resulting embedding is a 512-dimensional vector [11].

A downside of ArcFace is that it cannot compute face representations of images with the high intensity of light reflection [8] as shown in Figure 13.

The model is trained on MS1MV2 which is a refinement of the MS-Celeb-1M dataset. The training dataset contains about 10 million images of 100,000 top celebrities selected from one million celebrities in terms of their web appearance frequency [11]. The pre-trained model (LResNet100E-IR) used in the experiment in the next section was trained on the MS1MV2 dataset using the ResNet100 architecture which achieved an accuracy of 0.9982 on the LFW benchmark [15].

4 Clustering Algorithms

Clustering can be considered the most important unsupervised learning problem and aims to find a structure in a collection of unlabeled data. Objects that are similar between each other are grouped in the same cluster, while dissimilar objects belong to other clusters [17].

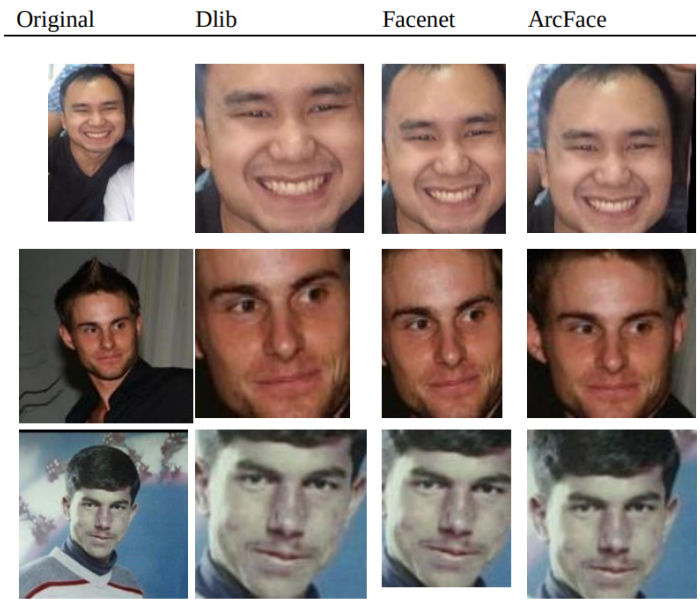

Figure 12: Comparsion of Face Detection between Dlib, FaceNet, and ArcFace [8]

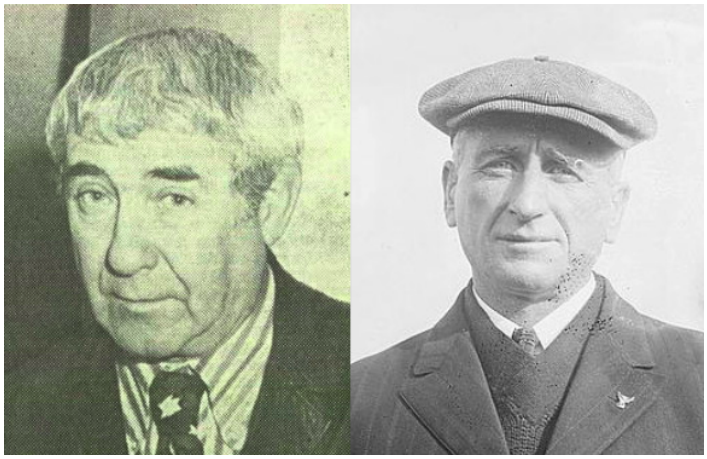

Figure 13: Images with high intensity of light reflection in which ArcFace is unable to compute the embeddings for [8]

After obtaining the face embeddings, each face represents a data point in the dataset. The similarity between each face can therefore be computed and grouped using different metrics clustering algorithms. The following section will present the algorithms use in Section 5.

4.1 K-Means

Lloyd's K-Means algorithm is a commonly used clustering method due to it's simplicity and efficiency. It aims to minimize the average squared distance between points in the same cluster [6]. The algorithm first initializes k random centroids, which are random datapoints chosen from the dataset. Then each point is assigned to the closest centroid. The centroids are then recomputed as the average of all datapoints assigned to it. The latter two steps are repeated until the centroids don't change significantly. The implementation in [23] computes the difference between the old and new centroids and terminates until the difference is less than a threshold of 0.0001 [6].

Even though the converagence of this algorithm is guranteed, one of the drawbacks is that it assumes the datapoints of the clusters have the same variation, therefore responding poorly to clusters with irregular shapes. Moreover, the algorithm is highly sensitve to the initialization of the centroids which can noteably affect the accuracy of the clusters [23]. K-means++ is a method developed by [6] to address this issue, in which it initializes the centroids at random from the data points, but weights the data points from the closest centroid already chosen [6]. Therefore, the initialized centroids will be (generally) far from each other [23]. Experiments from [6] show that k-means++ outperforms k-means in terms of both accuracy and speed.

TODO: plots of bad initialization

4.2 Spectral

Spectral Clustering is an efficient clustering algorithm where it uses a low-dimensional embedding of the distance matrix between the data points, followed by the K-Means algorithm in the low dimensional space [23].

4.3 Hierarchical Agglomerative

Hierarchical clustering is a method that merges or splits the clusters successively to form a nested tree of clusters, known as the dendrogram. In particular, Hierarchical Agglomerative Clustering is a bottom up approach where the root of the tree consists of the data points as their own clusters, then the clusters are successively merged together to form unique clusters at the leaves [23].

The merging technique is determined by the linkage criteria. Section 5 uses Ward's method for the linkage criteria, in which it minimizes the variance of the clusters by the sum of squared differences. Although Ward's method gives the clusters the most regular sizes, the distance used in the clustering is limited to Euclidean metrics [23].

4.4 Expectation–Maximization

The Expectation–Maximization (EM) algorithm is based on the Gaussian mixture model (GMM), which is a probabilistic method that assumes all the data points are

generated from a mixture of Gaussian distributions with unknown latent parameters (mean, covariance) for each distribution [23]. The Gaussian distributions corresponds to the clusters and since the information of which distribution each data point belongs to is unknown the goal of the EM algorithm is to estimate the latent parameters for each distribution. After randomly initializing the latent parameters, EM executes the following two steps until convergence [23].

1. **Expectation:** For each data point, compute the probability of it being generated by each latent parameter of the model. Each data point is then assigned to a cluster based on the probability.

2. **Maximize:** Re-estimate the latent parameters using the by maximizing the likelihood of the data given the assignments

The EM algorithm is similar to K-means, but instead the assignment of data points uses soft clustering. Instead of being either in a cluster or not (hard clustering), EM computes a probability of for assigning the data points [17].

4.5 Birch

Birch clustering is an efficient data reduction algorithm in which it reduces the input data to a set of subclusters. For a given dataset, the algorithm builds the Characteristic Feature Tree (CFT) where the data points are compressed to a set of Characteristic Feature nodes (CF Nodes). When a new data point is inserted into the CF Tree, it is merged with the nearest leaf of the subcluster of the root. These subclusters contain the necessary information for clustering [23]

5 Experiment

5.1 Dataset

5.1.1 Experiment 1: Head shots of 5 types of athletes

This dataset consists of $2,180$ unique images of five categories of athletes obtained from a combination of repositories of [3]. As shown in Figure 14, [3] scraped the images directly from the official sport competition websites, therefore, the images are all high quality straight head shots [3]. The following shows the distribution of images:

- **NBA Basketball Players:** 225 images
- **UFC Fighting Champions:** 560 images
- **FIFA Soccer Players:** 735 images
- **PGA TOUR Golf Players:** 558 images
- **ATP Tour Tennis Players:** 102 images

Figure 14: Example of face images from the dataset of Experiment 1: Five categories of athletes [3]

5.1.2 Experiment 2: In-The-Wild images of 5 varying professions

This dataset consists of 2,065 unique images of five different categories of professions. It is a combination of a subset of the dataset from 5.1.1 and a new dataset acquired using Wikidata's SPARQL query service.

Wikidata is an open-source knowledge base for structured data. After choosing the professions and finding their entity ID's a query was formed to find people and the corresponding image links with of the occupation. In order to reduce variation in the dataset, additional queries were composed to obtain only males from North America and Europe. Since clustering algorithms like K-Means tend to generate similar sized clusters [27], the number of images for each occupation are limited to similar sizes. An example of a query for the manager occupation can be found in A. Then, the images were automatically downloaded using Python. As shown in Figure 15, opposed to the dataset from 5.1.1, the images from Wikidata are in-the-wild images with varying head poses, illumination conditions, and other variations. The following shows the distribution of images:

- **Managers:** 371 images
- **Politicians:** 449 images
- **Military Officers:** 407 images
- **Architects:** 388 images
- **FIFA Soccer Players:** 450 images

Figure 15: Example of face images from the dataset of Experiment 2: Five categories of professions [3]

5.1.3 Experiment 3: In-The-Wild images of 11 varying professions

This dataset consists of 4,593 unique images of 11 different categories of professions. It is again a combination of a subset of the dataset from 5.1.1, 5.1.2, and images of five more professions downloaded from Wikidata using the same method as 5.1.2. The following shows the distribution of images:

- **Managers:** 371 images
- **Entrepreneur:** 485 images
- **Politicians:** 449 images
- **Lawyer:** 394 images
- **Military Officers:** 407 images
- **Sport Coaches:** 334 images
- **FIFA Soccer Players:** 450 images
- **UFC Fighting Champions:** 537 images
- **Architects:** 388 images
- **Actor:** 403 images
- **Musician:** 375 images

5.2 Set-Up

In the experiments faces of people are clustered by profession using different clustering algorithms. Since the number of different professions are known, the number of clusters selected corresponds to the number of professions in each experiment. The cluster algorithms were applied to the feature embeddings using the methods described in Section 3. Pre-trained models from each method was used in the experiment.

Figure 16: Example of a subset of face images from the dataset of Experiment 3. In combination with 15, the professions make up the 11 professions for Experiment 3

5.3 Framework

Computer Device Details:

- Processor: Intel Core i5-4570 CPU, 3.20GHz x 4
- 16 GB RAM

The following describes the steps to obtain the results:

1. **Load Images:** Each image is located inside a directory with the same filename
2. **Face Detection:** The FaceNet implementation uses MTCNN and the detected faces are of 160 x 160 pixels. Dlib uses the HOG face detector. OpenFace uses the HOG detector and the detected faces are of 96 x 96 pixels. ArcFace uses MTCNN and the detected faces are 112 x 112 pixels. Based on the open-source implementations, [25], [13], [4], [15], Table 1 shows which face detection method was used for each method and the number of features in the resulting embeddings. Table 2 shows the details of the accuracy based on the LWF benchmark, data size, and architecture of the pre-trained models.

Method	Face Detector	Number of Features
FaceNet	MTCNN (160 x 160 px)	512
Dlib	HOG	128
OpenFace	HOG (96 x 96 px)	128
ArcFace	MTCNN (112 x 112 px)	512

Table 1: The face detection method used and the number of features of the embeddings extracted from each method [25], [13], [4], [15].

3. **Extract Feature Embeddings:** Using the pre-trained models provided by the implementations, the embeddings can be extracted. The embeddings of FaceNet and ArcFace both have 512 features and the embeddings of Dlib and

Method	LWF Accuracy	Dataset Size	Architecture
FaceNet	0.9965	3.31 million images, 9,121 identities	Inception ResNet v1
Dlib	0.9938	3 million images, 7,485 identities	ResNet 29
OpenFace	0.9292	500,000 images	Inception ResNet v1 (modified)
ArcFace	0.9982	10 million images, 100,000 identities	ResNet 100
Human-Level [5]	0.9753		

Table 2: Accuracy based on the LWF benchmark, data size, and architecture of the pre-trained models [25], [13], [4], [15].

OpenFace have 128 features. The embeddings and corresponding labels (identity and profession) of FaceNet, Dlib, and ArcFace are saved as binary files that can be loaded directly using NumPy in Python and the Openface embeddings and labels are saved as csv files. Table 3 shows the dimensions of the feature embeddings extracted from each method.

Method	Experiment 1	Experiment 2	Experiment 3
FaceNet	2,180 x 512	2,065 x 512	4,593 x 512
Dlib	2,180 x 128	2,065 x 128	4,593 x 128
OpenFace	2,180 x 128	2,065 x 128	4,593 x 128
ArcFace	2,180 x 510	2,065 x 512	4,593 x 512

Table 3: Dimensions of the feature embeddings extracted from each method

4. **Profession Clustering:** The clustering algorithms from Section 4 were applied to the feature embeddings of the different methods. Since the number of different professions are known, the number of clusters selected corresponds to the number of professions in each experiment. The example in Figure 17 has two professions. After detecting the faces and representing them as embeddings, Figure 18 shows the clustering results. The embeddings are visualized using PCA.

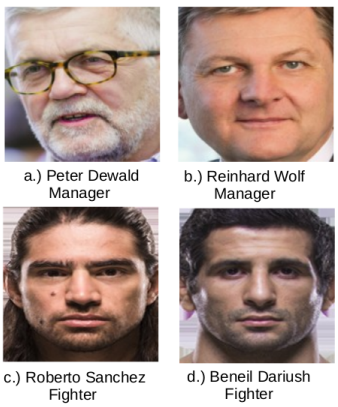

Figure 17: The goal is to cluster the professions of the aligned images.

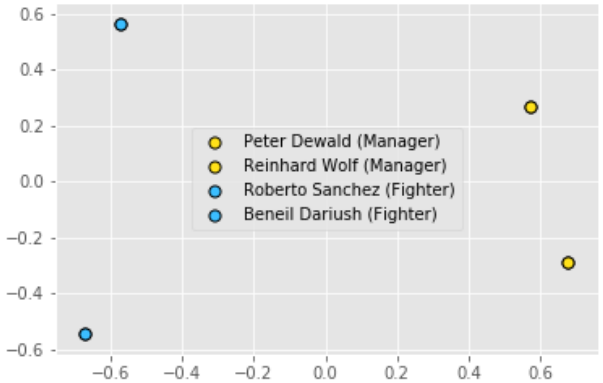

Figure 18: PCA plot after clustering using the embeddings of the faces from Figure 17

Figure 19: **True Positives** using FaceNet and K-Means. Clusters of Soccer Players and Fighting Champions.

6 Results

6.1 Evaluation

Since the dataset provides the labels of the correct profession for each face image, the accuracy of the clusters can be evaluated corresponding to the known profession labels. Due to efficiency, the clustering accuracy is evaluated with the Pairwise F-Measure [22].

6.1.1 Pairwise F-Measure

Suppose $L = \{l_1, ..., l_n\}$ contains the actual labels for each face and $C = \{c_1, ..., c_n\}$ are the labels from the output of the clustering algorithm. For example, shown in Figure 20, the actual labels are

$$L = \{\{A1, A2, A3\}, \{B1\}, \{U1, U2\}\}$$

and the labels from the output of the cluster algorithm are

$$C = \{\{A1, A2, B1\}, \{A3, U1, U2\}\}$$

Then the set of face pairs from the actual labels are

$$P = \{(A1, A2), (A1, A3), (A2, A3), (U1, U2)\}$$

The set of face pairs from the labels of the cluster algorithm are

$$Q = \{(A1, A2), (A1, B1), (A2, B1), (A3, U1), (A3, U2), (U1, U2)\}$$

For each face pair (i, j) of each cluster [7]:

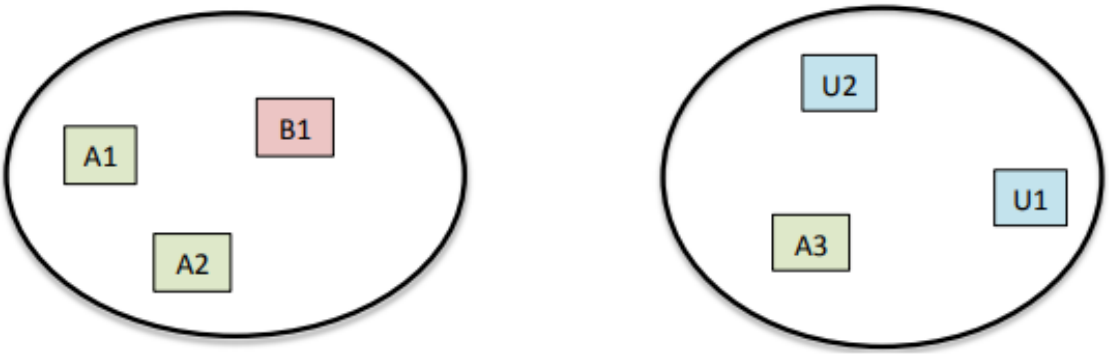

Figure 20: Example of a possible clustering output. Six data points are grouped into 2 clusters. A1, A2, and A3 have the same label, B1 has it's own label, and samples U1 and U2 have the same label [22].

- **True Positive (TP):** The number of face pairs (i, j) that are correctly clustered into the same cluster.

$$TP = |\{(i, j) \text{ where } c_i = c_j \text{ and } l_i = l_j\}|$$

In Figure 20, $(A1, A2), (U1, U2)$ are matching pairs and

$$TP = |P \cap Q| = |\{(A1, A2), (U1, U2)\}| = 2$$

- **False Positives (FP):** The number of face pairs (i, j) that are incorrectly clustered to the same cluster.

$$FP = |\{(i, j) \text{ where } c_i = c_j \text{ and } l_i \neq l_j\}|$$

In Figure 20, $(A1, B1), (A2, B1), (A3, U1), (A3, U2)$ are mismatching pairs and

$$FP = |Q - P| = |\{(A1, B1), (A2, B1), (A3, U1), (A3, U2)\}| = 4$$

- **False Negatives (FN)**: The number of face pairs that are clustered to a different cluster.

$$TN = |\{(i, j) \text{ where } c_i \neq c_j \text{ and } l_i \neq l_j\}|$$

In Figure 20, $(A1, A3), (A2, A3)$ are same-class pairs in different clusters and

$$FN = |P - Q| = |\{(A1, A3), (A2, A3)\}| = 2$$

- **Pairwise Precision:** Fraction of face pairs that are correctly clustered together over the total number of pairs that belong to the same class [27].

$$Pairwise\ Precision = \frac{TP}{TP + FP}$$

- **Pairwise Recall:** Fraction of face pairs that are correctly clustered together over the total number of pairs that are in the same cluster [27].

$$Pairwise\ Recall = \frac{TP}{TP + FN}$$

- **Pairwise F-Measure:** Harmonic mean of Precision and Recall [27]

$$Pairwise\ F\text{-}Measure = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

If the output of the algorithm clusters all data points as individual clusters, it will have a high precision, but low recall. Whereas if the output clusters all data points in the same cluster, it will have a high recall, but low precision [22].

6.2 Experiment 1

The results were obtained by averaging 10 F-measure scores.

Method	FaceNet	Dlib	OpenFace	ArcFace
K-Means	0.478	0.391	0.382	**0.516**
Spectral	0.445	0.361	0.345	**0.467**
HAC	**0.447**	0.362	0.391	0.413
EM	0.449	0.394	0.376	**0.512**
Birch	0.442	0.404	0.331	**0.453**

Table 4: Experiment 1: Comparison of Feature Extraction Methods and F-Measure of Clustering Algorithms

- **FaceNet**

Clustering Method	# of clusters	F-Measure	Precision	Recall	Runtime
K-Means	5	**0.478**	0.515	0.446	0.906
Spectral	5	0.445	0.507	0.397	0.662
HAC	5	0.447	0.463	0.432	0.669
EM	5	0.449	0.499	0.408	5.004
Birch	5	0.442	0.452	0.432	0.553

Table 5: Experiment 1 FaceNet

- **Dlib**

Clustering Method	# of clusters	F-Measure	Precision	Recall	Runtime
K-Means	5	0.391	0.415	0.370	0.277
Spectral	5	0.361	0.385	0.339	0.631
HAC	5	0.362	0.384	0.343	0.234
EM	5	0.394	0.419	0.372	0.594
Birch	5	**0.404**	0.325	0.533	0.086

Table 6: Experiment 1 Dlib

- **OpenFace**

Clustering Method	# of clusters	F-Measure	Precision	Recall	Runtime
K-Means	5	0.382	0.410	0.357	0.258
Spectral	5	0.345	0.359	0.332	0.665
HAC	5	**0.391**	0.369	0.417	0.226
EM	5	0.376	0.408	0.349	0.475
Birch	5	0.331	0.365	0.303	0.12

Table 7: Experiment 1 OpenFace

- **ArcFace**

Clustering Method	# of clusters	F-Measure	Precision	Recall	Runtime
K-Means	5	**0.516**	0.588	0.459	1.211
Spectral	5	0.467	0.533	0.415	0.804
HAC	5	0.413	0.459	0.376	0.756
EM	5	0.512	0.582	0.457	1.03
Birch	5	0.453	0.473	0.434	1.036

Table 8: Experiment 1 ArFace

Figure 21: **True Positives** using FaceNet and K-Means. Clusters of Soccer Players and Fighting Champions.

Figure 22: **False Positives**

Figure 23: **False Negatives**

6.3 Experiment 2

Method	FaceNet	Dlib	OpenFace	ArcFace
K-Means	**0.426**	0.405	0.306	0.294
Spectral	**0.392**	0.367	0.295	0.272
HAC	0.384	**0.392**	0.310	0.333
EM	**0.443**	0.411	0.310	0.277
Birch	**0.399**	0.383	0.294	0.304

Table 9: Experiment 2: Comparison of Feature Extraction Methods and F-Measure of Clustering Algorithms

- **FaceNet**

Clustering Method	# of clusters	F-Measure	Precision	Recall	Runtime
K-Means	5	0.426	0.415	0.439	0.880
Spectral	5	0.392	0.379	0.408	0.617
HAC	5	0.384	0.331	0.457	0.590
EM	5	**0.443**	0.427	0.461	5.337
Birch	5	0.399	0.368	0.435	0.499

Table 10: Experiment 2 FaceNet

- **Dlib**

Clustering Method	# of clusters	F-Measure	Precision	Recall	Runtime
K-Means	5	**0.405**	0.404	0.408	0.287
Spectral	5	0.367	0.365	0.370	0.640
HAC	5	0.392	0.348	0.450	0.201
EM	5	0.411	0.401	0.422	0.532
Birch	5	0.383	0.280	0.606	0.077

Table 11: Experiment 2 Dlib

- **OpenFace**

Clustering Method	# of clusters	F-Measure	Precision	Recall	Runtime
K-Means	5	0.306	0.306	0.307	0.344
Spectral	5	0.295	0.293	0.297	0.624
HAC	5	**0.310**	0.282	0.345	0.203
EM	5	**0.310**	0.310	0.311	0.398
Birch	5	0.294	0.276	0.314	0.137

Table 12: Experiment 2 OpenFace

- **ArcFace**

Clustering Method	# of clusters	F-Measure	Precision	Recall	Runtime
K-Means	5	0.294	0.283	0.306	1.332
Spectral	5	0.272	0.262	0.283	0.742
HAC	5	**0.333**	0.277	0.418	0.586
EM	5	0.277	0.268	0.288	0.803
Birch	5	0.304	0.255	0.376	0.837

Table 13: Experiment 2 ArFace

Figure 24: Experiment 2: Cluster of Managers and Politicians

Figure 25: Experiment 3: Cluster of Sport Coaches, Musicians, Lawyers, Entrepreneurs, and Actors

6.4 Experiment 3

Method	FaceNet	Dlib	OpenFace	ArcFace
K-Means	**0.203**	0.185	0.141	0.182
Spectral	**0.204**	0.166	0.143	0.159
HAC	0.177	**0.188**	0.131	0.174
EM	**0.210**	0.191	0.146	0.179
Birch	0.182	**0.193**	0.149	0.151

Table 14: Experiment 3: Comparison of Feature Extraction Methods and F-Measure of Clustering Algorithms

- **FaceNet**

Clustering Method	# of clusters	F-Measure	Precision	Recall	Runtime (seconds)
K-Means	11	0.203	0.202	0.204	3.698
Spectral	11	0.204	0.205	0.203	3.695
HAC	11	0.177	0.154	0.209	3.497
EM	11	**0.210**	0.206	0.215	25.618
Birch	11	0.182	0.172	0.193	2.271

Table 15: Experiment 3 FaceNet

- **Dlib**

Clustering Method	# of clusters	F-Measure	Precision	Recall	Runtime (seconds)
K-Means	11	0.185	0.181	0.190	1.260
Spectral	11	0.166	0.167	0.167	3.458
HAC	11	0.188	0.168	0.214	1.132
EM	11	0.191	0.186	0.196	3.296
Birch	11	**0.193**	0.140	0.310	0.207

Table 16: Experiment 3 Dlib

- **OpenFace**

Clustering Method	# of clusters	F-Measure	Precision	Recall	Runtime (seconds)
K-Means	11	0.141	0.139	0.145	1.162
Spectral	11	0.143	0.130	0.160	1.140
HAC	11	0.131	0.130	0.133	3.500
EM	11	0.146	0.143	0.150	3.233
Birch	11	**0.149**	0.137	0.164	0.492

Table 17: Experiment 3 OpenFace

- **ArcFace**

Clustering Method	# of clusters	F-Measure	Precision	Recall	Runtime (seconds)
K-Means	11	**0.182**	0.182	0.183	4.679
Spectral	11	0.159	0.159	0.159	4.112
HAC	11	0.174	0.156	0.197	3.600
EM	11	0.179	0.178	0.180	4.043
Birch	11	0.151	0.122	0.197	3.986

Table 18: Experiment 3 ArcFace

6.5 Analysis

Feature Extraction Method	Experiment 1	Experiment 2	Experiment 3
FaceNet	**00:15:06**	**00:25:27**	00:57:44
Dlib	00:28:19	00:50:44	01:52:57
OpenFace	00:16:01	00:27:59	**00:53:30**
ArcFace	00:16:04	00:42:44	01:34:38

Table 19: Runtime

Can't control age, big factor.

Bar graph Comparsion of accuracy based on increasing the number of clusters for each algorithm

It can also be assumed that the quality of images in terms of pose, illumination, occlusion, etc. being considered is relatively low, since social media images, images taken at public events etc. are not generally captured in the most favorable conditions for face recognition. [27]

7 Conclusion

Appendix A SPARQL Query

An example of a query that returns the names and image links to people who are male with a citizenship from Europe or North America and the occupation of manager.

```
SELECT distinct ?name ?img
    WHERE{
          ?person wdt:P106 wd:Q2462658.
          ?person rdfs:label ?name.
          ?person wdt:P18 ?img.
          ?person wdt:P21 wd:Q6581097.

          ?person wdt:P27 ?country.
          {?country wdt:P30 wd:Q46}
          UNION {?country wdt:P30 wd:Q49}
```

```
  FILTER (LANG(?name) = 'en') .
  SERVICE wikibase:label { bd:serviceParam wikibase:language "en".}.
} LIMIT 600
```

- Property wdt:P106: Occupation
- Entity ID wd:Q2462658: Manager
- Property wdt:P18: Image
- Property wdt: P21: Gender
- Entity ID wd:Q6581097: Male
- Property wdt:P27: Country of Citizenship
- Property wdt:P30: Continent
- Entity ID wd:Q46: Europe
- Entity ID wd:Q49: North America

References

[1] https://i.pinimg.com/originals/76/c3/ea/76c3ea5bcd34a4d7435320c05651d494.jpg.

[2] https://www.slovenskenovice.si/images/slike/2018/04/28/239717.jpg.

[3] Soumitra Agarwal. https://github.com/SoumitraAgarwal?tab=repositories.

[4] Brandon Amos, Bartosz Ludwiczuk, and Mahadev Satyanarayanan. *OpenFace*. http://cmusatyalab.github.io/openface/. 2016.

[5] Brandon Amos, Bartosz Ludwiczuk, and Mahadev Satyanarayanan. *OpenFace: A general-purpose face recognition library with mobile applications*. Tech. rep. CMU-CS-16-118, CMU School of Computer Science, 2016.

[6] David Arthur and Sergei Vassilvitskii. "K-Means++: The Advantages of Careful Seeding". In: vol. 8. Jan. 2007, pp. 1027–1035. DOI: 10.1145/1283383.1283494.

[7] A.F. Bijl. "A comparison of clustering algorithms for face clustering". In: (July 2018).

[8] Adulwit Chinapas et al. "Personal Verification System Using ID Card and Face Photo". In: *International Journal of Machine Learning and Computing* 9 (Aug. 2019), pp. 407–412. DOI: 10.18178/ijmlc.2019.9.4.818.

[9] *Clustering*. https://scikit-learn.org/stable/modules/clustering.html. 2011.

[10] Navneet Dalal and Bill Triggs. "Histograms of Oriented Gradients for Human Detection". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2005)* 2 (June 2005).

[11] Jiankang Deng, Jia Guo, and Stefanos Zafeiriou. "ArcFace: Additive Angular Margin Loss for Deep Face Recognition". In: (Jan. 2018).

[12] *Face Recognition with Deep Learning*. https://www.hackevolve.com/face-recognition-deep-learning/. 2017.

[13] Adam Geitgey. *Face Recognition*. https://github.com/ageitgey/face_recognition. 2017.

[14] Adam Geitgey. *Machine Learning is Fun! Part 4: Modern Face Recognition with Deep Learning*. https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78. 2016.

[15] Jia Guo and Jiankang Deng. *InsightFace: 2D and 3D Face Analysis Project*. https://https://github.com/deepinsight/insightface. 2019.

[16] J. Joo, F. F. Steen, and S. Zhu. "Automated Facial Trait Judgment and Election Outcome Prediction: Social Dimensions of Face". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 3712–3720. DOI: 10.1109/ICCV.2015.423.

[17] Kang M. S. Jung Y. G. and J. Heo. "Clustering performance comparison using K-means and expectation maximization algorithms". In: *Biotechnology, biotechnological equipment* 28(sup1) (2014), S44–S48. DOI: 10.1080/13102818.2014.949045.

[18] Davis E. King. *Dlib 18.6 released: Make your own object detector!* http://blog.dlib.net/2014/02/dlib-186-released-make-your-own-object.html. 2014.

[19] Davis E. King. "Dlib-ml: A Machine Learning Toolkit". In: (July 2009).

[20] Davis E. King. *High Quality Face Recognition with Deep Metric Learning*. http://blog.dlib.net/2017/02/high-quality-face-recognition-with-deep.html. 2017.

[21] Christopher Olivola, Dawn Eubanks, and Jeffrey Lovelace. "The many (distinctive) faces of leadership: Inferring leadership domain from facial appearance". In: *The Leadership Quarterly* 25 (Oct. 2014). DOI: 10.1016/j.leaqua.2014.06.002.

[22] Charles Otto, Dayong Wang, and Anil Jain. "Clustering Millions of Faces by Identity". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PP (Apr. 2016). DOI: 10.1109/TPAMI.2017.2679100.

[23] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[24] Daniel Saez Trigueros, Li Meng, and Margaret Hartnett. "Face Recognition: From Traditional to Deep Learning Methods". In: (Oct. 2018).

[25] David Sandberg. *facenet*. https://github.com/davidsandberg/facenet. 2018.

[26] Florian Schroff, Dmitry Kalenichenko, and James Philbin. "FaceNet: A unified embedding for face recognition and clustering". In: June 2015, pp. 815–823. DOI: 10.1109/CVPR.2015.7298682.

[27] Yichun Shi, Charles Otto, and Anil Jain. "Face Clustering: Representation and Pairwise Constraints". In: *IEEE Transactions on Information Forensics and Security* PP (June 2017). DOI: 10.1109/TIFS.2018.2796999.

[28] Kaipeng Zhang et al. "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks". In: *IEEE Signal Processing Letters* 23 (Apr. 2016). DOI: 10.1109/LSP.2016.2603342.