bibliography item[]

# A Comparison of Facial Feature Extraction Methods based on Professional Domain Clustering

Bithiah Yuan
*University of Freiburg - Department of Compute Science*

October 22, 2019

**Abstract**

Motivated by researches in the social sciences and computational methods, the correlation between facial features and professional talents are examined using face clustering. Face clustering is a technique that groups similar faces together based on their feature representations. Therefore, obtaining a robust and discriminative face embedding for this task is essential. In this paper, four open-source state-of-the-art feature extraction methods and common clustering algorithms are compared by their accuracy to cluster faces based on the person's professional domain. The results show that there is a slight correlation between facial features and professional talents and that a combination of FaceNet and K-Means or EM clustering yield good clustering results.

## 1 Introduction

Since a significant source of information and attributes can be derived from the human face by non-verbal communication [16], facial features have been studied extensively in the social science domain to predict a person's success in reaching reputable leadership positions. In particular, studies have shown that certain facial features contribute to higher salaries and more prestigious employments [21]. In application, the relationship between facial characteristics and social attributes can provide a powerful objective indicator for organizations to idenity and select effective leaders within their domain [21].

Caused by experiments based on human judgement, researches in the social sciences are limited in scalability, consistency, and generalization. Therefore, a growing number of studies have been extended and refined to computer vision and machine learning research due to the capability of using massive datasets and the large-scale processing capacity [16]. Particularly, through a computational framework, [16] trained a model to predict the outcomes of political elections based on the candidates' face images. This suggests that similar methods can be used to predict behavior in a broad range of human social relations, such as job placement [16].

Motivated by computational methods, the following paper aims to examine the correlation between facial features and professional talents through face clustering. The face clustering problem consists of not only the choice of the clustering algorithm, but also the face representation and similarity metric of the face images [27]. Due

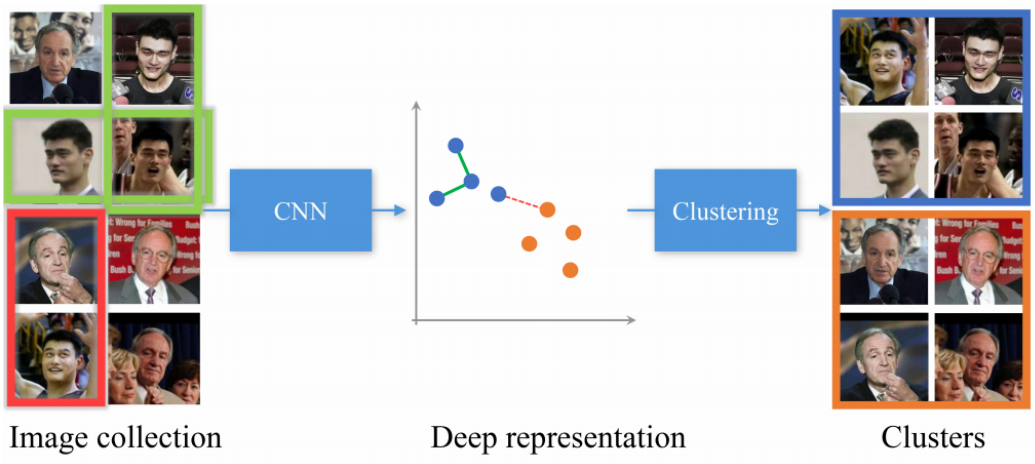


Figure 1: Face clustering workflow [27]

to the importance of face representations in face clustering, this paper compares four open-source state-of-the-art feature extraction methods based on deep convolutional neural networks and common algorithms for clustering the professional domains of faces.

# 2 Related Work

## 2.1 Face Clustering

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. The workflow shown in Figure 1. 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.

Face clustering systems are usually composed of the following four 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.

4. **Face Clustering:** Apply clustering algorithm.

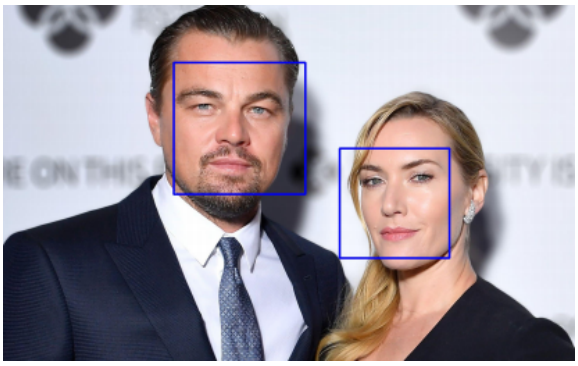

Figure 2: Face Detection [24]

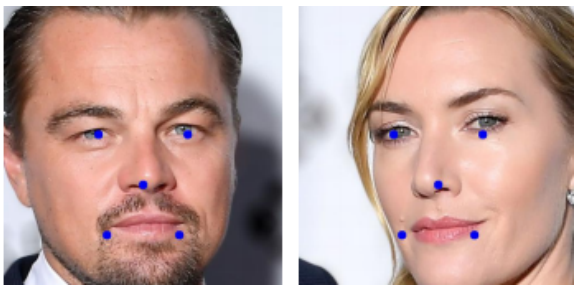

Figure 3: Face Alignment [24]

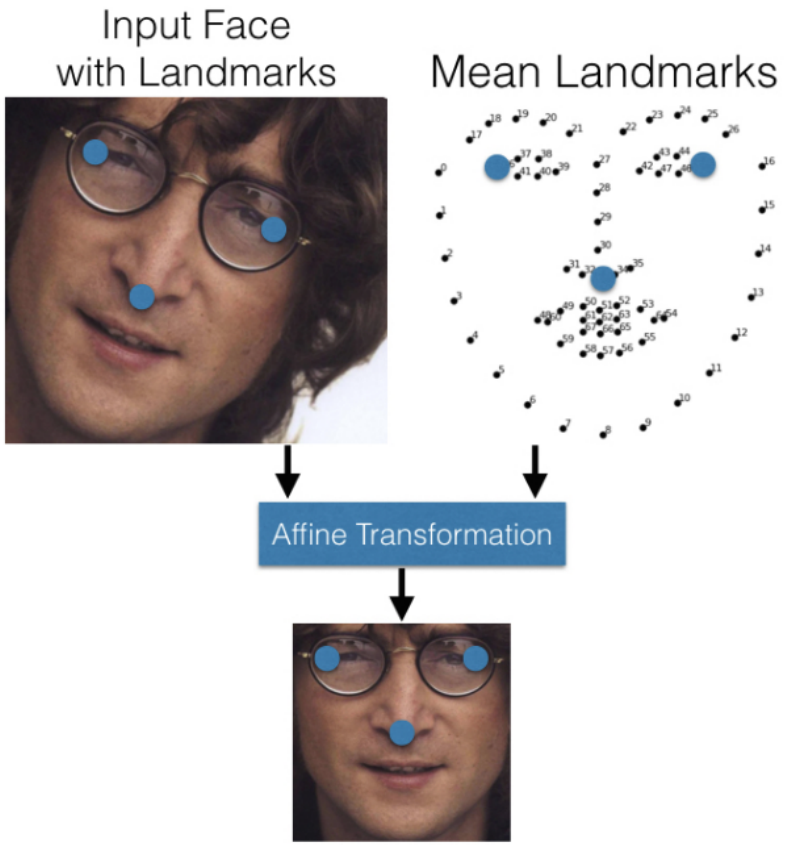


Figure 4: Applying affine transformation so that the face image is closer tothe set of mean landmarks. [5]

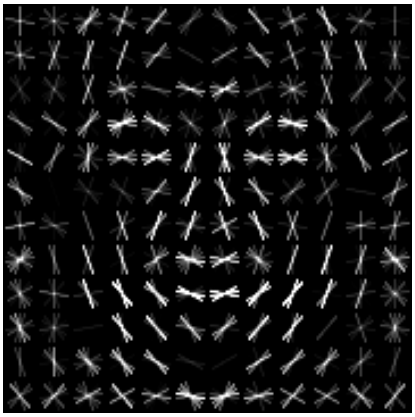

Figure 5: Trained HOG detector on multiple faces [24]

Figure 6: HOG representation of a face [12]

## 2.2 Face Detection

In this section a traditional face detector, HOG, and another based on deep neural networks, MTCNN is discussed.

### 2.2.1 Histograms of Oriented Gradients

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 identify appearances and shapes in an image [10].

HOG divides the image into small grids, where each grid accumulates a histogram of gradient directions or edge orientations over the pixels of the grid. The grids are then normalized for better invariance to illumination, shadowing, and other variations [10].

The normalized local histograms of image gradient orientations in the grids are the features and are then trained (typically by a linear classifier) 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 be detected as the region of the face.

### 2.2.2 Multi-task Cascaded Convolutional Networks

Face detection and alignment can also be done using covolutional neural networks (CNNs). CNNs are a type of deep neural networks, as shown in Figure 7 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 $\theta_i$ 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].

Multi-task Cascaded Convolutional Networks (MTCNN) is a widely used method to predict face and landmark location. The framework has a cascaded (waterfall) structure with three stages of deep CNNs shown in Figure 8. [28]


Figure 7: Neural Network Traing Flow. [5]

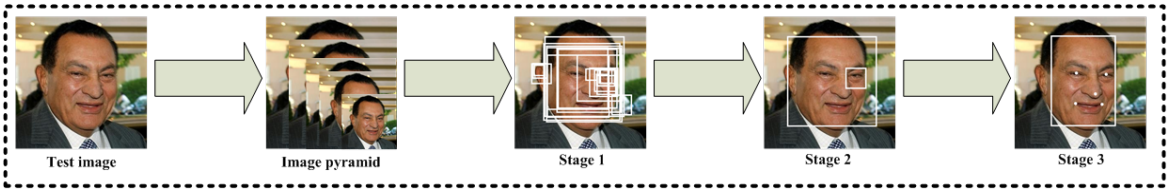


Figure 8: MTCNN: Cascaded structure with three stages of deep CNNs. [28]

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

1. **Proposal Network (P-Net):** Obtains candidates that will serve as potential positions of the bounding boxes [8].
2. **Refine Network (R-Net):** Uses the image and the results of the first prediction of the bounding boxes to reduce false positives and get the final box boundaries [8].
3. **Output Network (O-Net):** Outputs five facial landmark positions [28].

Both the prediction of the bounding box and facial landmark locations use regression to minimize the Euclidean loss between the candidate positions of the bounding boxes, landmark coordinates and the ground truth. The ground truth of the bounding box is the left, top, height, width of the box. The ground truth of the facial landmarks is 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 face clustering. However, challenges occur in real world images due to variations ranging from head poses and illumination conditions to aging and facial expressions. These images are referred as in-the-wild images [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 CNNs [5] since they are able to achieve very high accuracy by learning robust features due to the availability of large-scale in-the-wild face datasets on the web [24].

### 2.3.1 Convolutional Neural Networks

A common approach to training CNN models for face representation is using 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 faces and that the representation size per face is large and inefficient [26].

Another approach is to learn the features of the face directly by optimizing the distance between triplets of faces, in which the distances measure the similarity between faces [24] [26]. The approach uses the triplet loss function, which will be discussed in detail in the next section.

### 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 is to separate the distance between two aligned matching (positive) face images and an aligned non-matching (negative) face image by a distance margin. The result is a feature vector $f(x)$ 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 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. 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 $\alpha$ is a margin from the positive and negative pairs [24].

For $N$ possible triplets, the loss being minimized is:

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

.

Figure 10 further demonstrates using a neural network to compute the triplet loss and it's gradient by backpropagation through the network to the unique images in the learning step [5].

# 3 Face Feature Extraction Methods

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

## 3.1 FaceNet

FaceNet is a method that uses a deep CNN and the triplet loss function to directly optimize face embeddings. In [25]'s open-source implementation, the faces in the

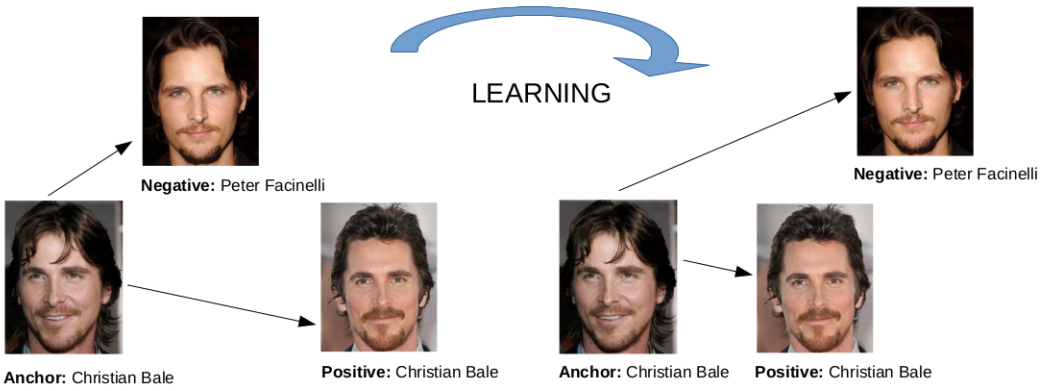


Figure 9: The loss of identical faces are minimized and the loss of distinct faces are maximized by the triplet loss function [1] [2].

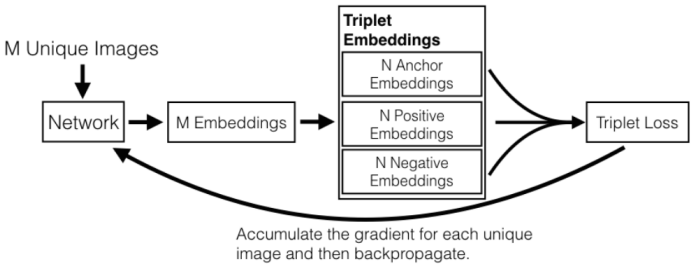


Figure 10: Learning the embeddings by optimizing the gradients of the triplet loss function [5].

images are first detected and aligned with MTCNN. The resulting aligned face images are 160 x 160 pixels and serve as the input for the FaceNet model.

The pre-trained model (20180402-114759) from [25] used in Section 5 was trained using the VGGFace2 dataset. The dataset contains 3.31 million images of 9,131 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 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, which is an open source library for developing machine learning software in C++ [19]. [13] used the HOG face detector from Dlib, in which the HOG representation was trained with a linear classifier (SVM) [18]. The face representation model again uses CNNs and the triplet loss function to learn the embeddings [20]. The resulting feature embedding is a 128-dimensional vector [20].

A dataset of about 3 million faces and 7,485 unique identities from a combination of the FaceScrub 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].

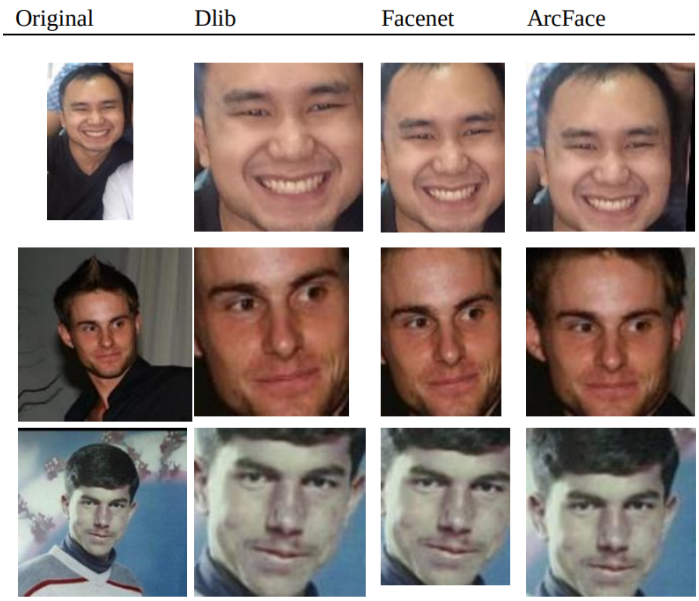


Figure 11: Comparsion of aligned faces between Dlib, FaceNet, and ArcFace [8]

### 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 the CASIA-WebFace and FaceScrub dataset. The face representation is obtained using a modification of FaceNet's architecture, in which the number of parameters are reduced. The pre-trained model has an accuracy of 92.92% on the LFW benchmark. 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 a face representation [11]. [15]'s implementation uses MTCNN for face detection to get aligned face images of 112 x 112 pixels. Then, ArcFace further adjusts a face image by rotating the image to a straight face as shown in Figure 11. Consequently, increasing the consistency of the facial feature positions and effectiveness when computing 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 high intensity of light reflection [8] as shown in Figure 12.

The pre-trained model (LResNet100E-IR) used in Section 5 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 model has an accuracy of 99.82% on the LFW benchmark [15].

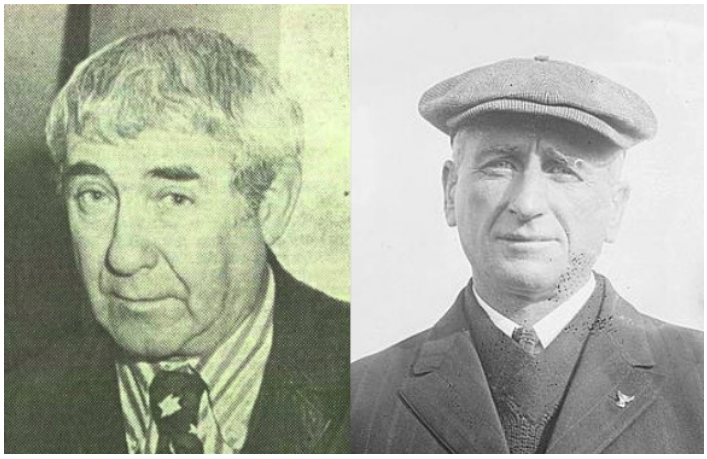

Figure 12: Images with high intensity of light reflection in which ArcFace is unable to compute the embeddings for [8]

# 4 Clustering Algorithms

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 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]. [23] also uses K-Means++ for initializing the centroids, in which it weights the data points from the closest centroid already chosen [6].

## 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. 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 [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 correspond 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 by maximizing the likelihood of the data given the assignments

### 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

This dataset consists of $2,180$ unique images of five categories of athletes obtained from a combination of the repositories of [3]. As shown in Figure 13, [3] scraped the images directly from the official sport competition websites, therefore, the images are 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 13: Example of face images from the dataset of Experiment 1: Five categories of athletes [3]

### 5.1.2 Experiment 2

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 Experiment 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 IDs, a query was formed to find the images of the people with the corresponding occupations. In order to reduce variation in the dataset, only images of males from North America and Europe were obtained. Since clustering algorithms like K-Means tend to generate similar sized clusters [27], the number of images for each occupation were limited to similar sizes. Due to the limited data source, the age of the face images would not be controlled. An example of a query for the occupation, manager, can be found in Appendix A. The images were then automatically downloaded using Python.

As shown in Figure 14, opposed to the dataset from Experiment 1, the images from Wikidata are in-the-wild 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 14: Example of face images from the dataset of Experiment 2: Five categories of professions [3]

### 5.1.3 Experiment 3

This dataset consists of 4,593 unique images of 11 different categories of professions. It is a combination of a subset of the dataset from Experiment 1 and 2 and images of five more professions downloaded from Wikidata. 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 were clustered by profession using different clustering algorithms. Since the number of different professions are known, the number of clusters corresponds to the number of professions in each experiment. The clustering algorithms were applied to the feature embeddings extracted from the methods


Figure 15: Example of a subset of face images from the dataset of Experiment 3. In combination with Figure 14, it makes up the 11 professions for Experiment 3

described in Section 3. Pre-trained models from each method was used in the experiment.

## 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 of 112 x 112 pixels.

   Table 1 shows which face detection method was used for each method and the number of features in the resulting embeddings.

| 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:** The embeddings were extracted using pre-trained models. Table 2 shows the details of the accuracy based on the LFW benchmark and training data size of the pre-trained models.

| Method | LFW Accuracy | Training Dataset Size |
|---|---|---|
| **FaceNet** | 0.9965 | 3.31 million images, 9,121 identitie |
| **Dlib** | 0.9938 | 3 million images, 7,485 identities |
| **OpenFace** | 0.9292 | 500,000 images |
| **ArcFace** | 0.9982 | 10 million images, 100,000 identities |
| **Human-Level** [5] | 0.9753 | |

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

The embeddings of FaceNet and ArcFace both have 512 features and the embeddings of Dlib and OpenFace have 128 features. The embeddings and corresponding labels (identity and profession) are also saved. 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 512 | 2,065 x 512 | 4,593 x 512 |

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

4. **Professional Domain Clustering of Faces:** The clustering algorithms from Section 4 were applied to the feature embeddings. The four images in Figure 16 belong to two professions. After detecting the faces and representing them as embeddings, Figure 17 shows the clustering results. The features were reduced to 2-D embeddings by PCA for visualization.

# 6 Results

## 6.1 Evaluation

Since the datasets provide 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].


Figure 16: The goal is to cluster the professions of the aligned images.

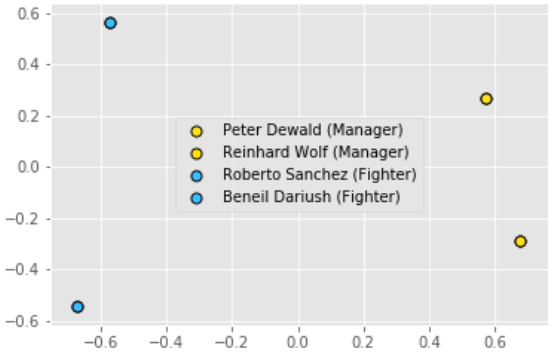


Figure 17: PCA plot after clustering using the embeddings of the faces from Figure 16.

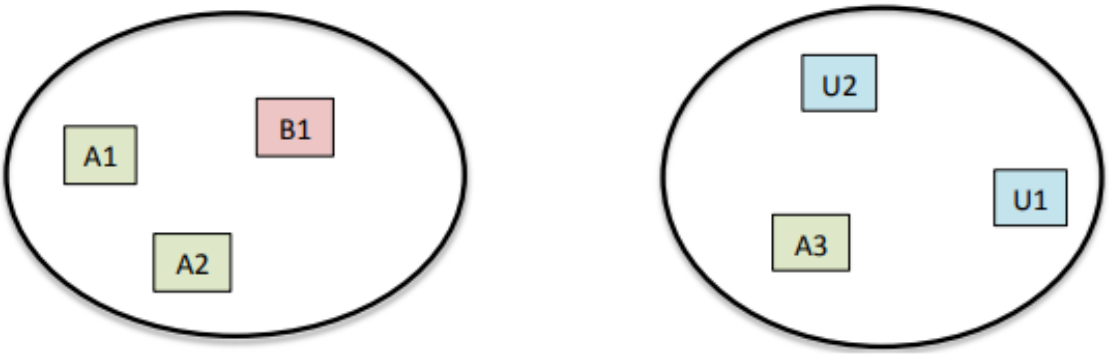


Figure 18: Example of a possible clustering output. Six data points are grouped into 2 clusters. A1, A2, and A3 have the same label, B1 has it's own label, and U1 and U2 have the same label [22].

### 6.1.1 Pairwise F-Measure

Suppose $L$ contains a set of actual clusters and $C$ contains contains a set of clusters from the output of the clustering algorithm. For example, shown in Figure 18,

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

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

Then the set of face pairs from the actual clusters, $L$, is

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

The set of face pairs from the clusters of the cluster algorithm, $C$, is

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

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

- **True Positives (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 18, $(A1, A2), (U1, U2)$ 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 18, $(A1, B1), (A2, B1), (A3, U1), (A3, U2)$ 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 18, $(A1, A3), (A2, A3)$ 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]

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

The F-Measure, Precision, Recall, and Runtime of the experiments are obtained by averaging the accuracy of the corresponding experiment 10 times.

## 6.2 Experiment 1

- **Comparison of Feature Extraction Methods and the F-Measure**

| 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: F-Measure obtained by each feature extraction and clustering method

- **FaceNet**

| Clustering Method | # of clusters | F-Measure | Precision | Recall | Runtime (seconds) |
|---|---|---|---|---|---|
| **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 (seconds) |
|---|---|---|---|---|---|
| **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 (seconds) |
|---|---|---|---|---|---|
| **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 (seconds) |
|---|---|---|---|---|---|
| **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 19: **True Positives** using FaceNet and K-Means. Clusters of Soccer Players and Fighting Champions.


Figure 20: **False Positives**


Figure 21: **False Negatives**

## 6.3 Experiment 2

- **Comparison of Feature Extraction Methods and the F-Measure**

| 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 F-Measure Comparison

- **FaceNet**

| Clustering Method | # of clusters | F-Measure | Precision | Recall | Runtime (seconds) |
|---|---|---|---|---|---|
| **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 (seconds) |
|---|---|---|---|---|---|
| **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(seconds) |
|---|---|---|---|---|---|
| **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 (seconds) |
|---|---|---|---|---|---|
| **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 22: Experiment 2: **True Positives** using FaceNet and K-Means. Cluster of Managers and Politicians

## 6.4 Experiment 3

- **Comparison of Feature Extraction Methods and the F-Measure**

| 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 F-Measure Comparison

- **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


Figure 23: Experiment 3: **True Positives** using FaceNet and K-Means. Cluster of Sport Coaches, Musicians, Lawyers, Entrepreneurs, and Actors


Figure 24: **False Positives**

Figure 25: **False Negatives**

## 6.5 Analysis

### 6.5.1 Experiment 1

As shown in Table 4, feature extraction using ArcFace provided the highest F-Measures for all the clustering algorithms except for Hierarchical Agglomerative Clustering where FaceNet provided a slightly higher F-Measure of 0.447 compared to 0.413 obtained from ArcFace. The highest F-Measure from Experiment 1 was 0.516 using a combination of ArcFace and K-Means clustering. The combination of ArcFace and EM-clustering also yielded the second highest F-Measure of 0.512.

Figure 19 shows a subset of the clustering results of the true positives derived from using FaceNet and K-Means clustering. Figure 20 shows the false positives and Figure

21 shows the false negatives. Qualitatively speaking, the results make sense as the face pairs from Figure 20 resemble similar features and the face pairs from Figure 21 have large dissimilarities due to variation in ethnicity and appearances.

### 6.5.2 Experiment 2

Even though Experiment 1 and Experiment 2 had a similar number of face images (2,180 and 2,065) and the same number of professions and clusters, the results differ significantly when clustering the embeddings from ArcFace. More specifically, the combination of ArcFace and K-Means clustering has a F-Measure of 0.294 compared to 0.561 obtained from Experiment 1.

The main difference between the datasets from Experiment 1 and Experiment 2 is that Experiment 1 consists of high-quality head shots of athletes, in which the variation of position, lighting, facial expression,and age of the faces are low. Whereas Experiment 2 consists of in-the-wild images and varying professions. As mentioned in Section 3.4, ArcFace is not good at computing face representations of images with high intensity of light reflection. Therefore, this may be a contributing factor to the lower F-Measure.

Nevertheless, the results from FaceNet are consistent and the highest F-Measure, 0.443, from Experiment 2 was obtained by the combination of FaceNet and EM clustering.

### 6.5.3 Experiment 3

Table 14 shows that the F-Measure of Experiment 3 dropped by half compared to Experiment 1 and 2 with the highest F-Measure of 0.210 obtained from the combination of FaceNet and EM clustering. This may be due the increasing variations by doubling the data size, the number of professions and clusters.

Figure 23 shows the clusters of a subset of the true positives. It can be observed that the age variation is high, particularly, the faces of entrepreneurs and actors are younger than the sport coaches and lawyers.

Shown in Figure 24, a younger Sport Coach, d-2) Nicolas Manaudou was clustered with the fighting champion d-1) Leonardo Santos. Since the faces of the dataset of athletes consists of younger faces, this clustering decision is qualitatively reasonable. A similar reasoning can be applied to the false negatives shown in Figure 25.

### 6.5.4 Runtime

Table 19 shows the runtime of detecting and extracting the features from the raw input images. FaceNet was the most efficient for Experiment 1 and 2, while OpenFace was the most efficient for Experiment 3. On the other hand, Dlib had the longest runtime for all the experiments. Therefore, although Dlib had higher accuracies in the experiments, it is not efficient for larger datasets. Moreover, even though OpenFace had lower accuracies in the experiments, it is efficient for larger datasets.

# 7 Conclusion

Overall, the results show that when the raw input images are head shots with low variations, ArcFace is a good choice for the feature extraction method as it provided the

| 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

highest accuracy. However, when the raw images are in-the-wild with many variations, then FaceNet is a better choice in terms of consistency, accuracy, and efficiency.

Although both K-Means and EM Clustering provided the highest accuracy for the experiments, the runtime shows that K-Means clustering is a lot more efficient. As a result, it is recommended to use FaceNet as the feature extraction method and K-Means clustering if the dataset is large.

The highest F-Measure 0.516 from Experiment 1 and 0.443 from Experiment 2 indicate a positive correlation between facial features and the person's professional domain. The highest F-Measure of 0.210 from Experiment 3 indicates a slight correlation.

Unlike clustering faces to recognize a person's identity, it is not essential to use in-the-wild images to determine a person's profession from their face features. The results would be stronger if the dataset was not in-the-wild and could be controlled to have more high-quality head shots of faces with low variation. However, since the available face images are taken at public events, this would be difficult to achieve.

In summary, the results show qualitatively and quantitatively that there is a slight correlation between facial features and the person's profession and that FaceNet combined with K-Means or EM clustering yields good clustering results.

# Appendix A SPARQL Query

An example of a query that returns the names and image links to male managers with a citizenship from Europe or North America.

```
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.