DeepView: A Wireless Dynamic Facial Recognition System with Data Logging

Blaze R.Perater, Benjamin R. Sanglitan, and Cristina P. Dadula

Abstract

Conventional facial recognition techniques are nonversatile to changes in pose expression because they utilize static algorithms. This paper proposed a dynamic wireless facial recognition system with data logging capabilities using CNN. Face recognition methodology was divided into two stages: face detection and face recognition. For face detection, a Histogram of Oriented Gradients (HOG)-based technique was used in conjunction with Face Alignment through Affine Transformation for input image pre-processing. The facial recognition stage utilized an OpenFace implementation for the neural network, modified clustering for grouping identities and Density-Based Spatial Clustering of Applications with Noise (DBSCAN) for removing outliers. The accuracy was calculated at 87.03% with an average processing time of 13.7 ms at 10 fps frame rate. Images are sorted in archives of the data logger by time, date, camera number and picture of encounter for each distinct identity.

In addition, face searching enables the user to upload and external photo and search the database for a matching identity. The system has been successfully implemented in a real world scenario.

***Keywords*–face recognition, deep learning, CNN, python**

EEE Department, Mindanao State University
- General Santos City, General Santos City, Philippines
blaze.perater@msugensan.edu.ph,
benjamin.sanglitan@msugensan.edu.ph,
cristina.dadula@msugensan.edu.ph

I. Introduction

For humans, face recognition is an unsophisticated task even in adverse situations like bad lighting or facial changes due to aging. This basic task for our brains has become a real challenge in advanced computer vision in recent years [1]. Face recognition is of practical importance because it is one of the easiest and most convenient biometrics that can be utilized in surveillance, identity authentication and access control. [2].

As a result, governments all over the world use FR systems to identify potential and current threats. It was used during Super Bowl XXXV in 2001 in which the faces of 100,000 people was digitally scanned, analyzed, and cross-referenced with a database of wanted and suspected criminals. Dubai installed facial recognition cameras in its international airport in 2008 to capture the facial images of passengers from the flights and complement the iris scan mechanism to nab illegal entrants and wanted criminals [3]. Today, they integrated the FR system to a tunnel aquarium to encourages travelers to look around, and increase the quality of their face scan [4].

FR systems was also effectively used in biometric authentication. In the 1970s, the UK government enquiry was chaired by Lord Devlin, summarizing several cases where sincere and credible witnesses proved badly wrong when picking out perpetrators from photographs or police line-ups. These findings sparked new psychological research into eyewitness testimony - in particular recall and recognition of faces [5]. To improve justice execution, the US Federal Bureau of Investigation created a database that contains over 30 million mugshots of criminals and ID card images from 16 states. The FBI used FR software to identify and gain leads to arrest a fugitive who was on the run for over 14 years. After getting tips that the fugitive was traveling internationally, an FBI agent contacted the US Diplomatic Security Service to gain access to a passport image database. He then uploaded a photo of the fugitive's wanted posters to cross-reference him with the database. The software linked the fugitive's

face to a similar headshot on a passport with a different name listed [6].

Security-conscious businesses have also utilized FR systems such as Retailers and businesses of all kinds can look for suspected shoplifters or track their workers' chronograms [7]. There are a lot of social media applications which increase a user experience with FR technique like the Mastercard Identity Check which provides payment confirmation, and online retailer Alibaba takes online payments with its Smile to Pay [7]. Currently, FR systems are becoming so reliable. A research shows that there is a 1-in 50,000 chance of a phone with touch ID being unlocked with the wrong fingerprint but with advanced 3d facial modeling, the probability drops to nearly 1-in-1,000,000 [7], [8].

Face recognition was first addressed in Computer Vision by Landmark-based method which are geometric representation of facial features [9]. However, it was limited by differences in pose and expression. EigenFaces attempted to defy this but noisy images resulted in inaccurate representations [10]. The Fisherfaces method surpassed Eigenfaces [11] and Local Binary Pattern ventured to address factors in the input images but LBP can't extract the important structures from important area of face image completely [12]. Methods based on convolutional neural networks (CNNs) are not affected by this factors. This paper proposes a CNN-based facial recognition system with data logging capabilities. It will utilize three (3) cameras positioned in fixed locations with enough lighting and will enable frontal capturing of images.

II. Face Detection

For the face detection, given an input camera frame, an algorithm was constructed to show the coordinates of a region of interest (ROI) where the face of person is present. Face alignment was then applied to these ROI's for pre-processing.

A. Histogram of Oriented Gradients

Histogram of Oriented Gradients (HOG) is a feature descriptor applied in diverse field to characterize objects based on their shapes. It is a method that analyses every patch in an image by dividing it into smaller blocks. The intensity on how much the colors change is then observed in each block [13]. This is performed on different directions and compiled on a histogram. A linear classifier is then implemented to identify the object being detected, which in this study is the face.

The researchers employed the default Dlib face detector class which is a HOG-based face detector, and the frontal face Haar cascade found in the OpenCV library.

B. Face Alignment with Affine Transformation

The detected faces come in different shapes and sizes. Also, people can face the camera at different angles and orientations which induces variability that will lessen the consistency of the facial recognition stage. To minimize this, a face alignment technique called Affine Transformation has been utilized. This face alignment normalizes the input data before being fed to the facial recognition stage [14].

Affine transformation in geometry is simply a linear mapping that preserves points, straight lines and planes [15]. All relationships between the points, lines and planes remains after transformation. This technique is usually executed with geometric deformations that occur in non-ideal camera angles. For this study, Affine transformation was carried out to accomplish the following criteria:

1. The cropped region of interest must be centered on the face.
2. The eyes on the face must be rotated so that all the eyes from different images lies on the same horizontal line.
3. All the detected faces should be approximately identical in size after the transform.

The second criteria was done by determining two parameters-the angle and axis of rotation. First, the position of the eyes' center must be localized using the Dlib library. The location of the center is given as an ordered pair of x and y coordinates. To calculate the angle of rotation, the following formula was applied:

$$\Theta = \arctan \frac{d_y}{d_x} \tag{1}$$

$$d_y = y_{rightEyeCenter} - y_{leftEyeCenter} \tag{2}$$

$$d_x = x_{rightEyeCenter} - x_{leftEyeCenter} \tag{3}$$

The axis of rotation was placed on the midpoint between the two eyes. Midpoint is mathematically defined as:

$$M(x, y) = (\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2}) \tag{4}$$

III. Embeddings Calculation

A. Neural Network Approach

A neural network based on the Facenet architecture via the OpenFace implementation was deployed to perform the facial recognition. This model is composed of 166 layers with an input layer of 96x96x3 [16]. Unlike other neural networks which directly outputs the classification given an

input, the neural network that DeepView utilized outputs a 128-dimension embedding that describes the input face. This multidimensional embedding was the accompanying variable for realizing the identity of a face.

B. Face Representation

Faces were represented with an encoding which is a 128 dimension feature-vector. Faces have distinctive embeddings regardless of whether they are from the same individual or not. Factors like pose and emotion influences these embeddings. The neural network was trained to address these factors. It minimized the distance of faces from the same person, and maximized the distance of faces from different persons. The distance that is being described is the L2 Euclidean distance.

Humans see faces as shapes in three dimensional space. However, in neural network, it must describe the faces in a 128-dimensions space to achieve satisfactory results. This was the reason for embeddings existing in multiple dimensions. Even at higher dimensions, the mathematics concept performed in 3D space holds true. Hence, the distance between two 128-dimension encodings was calculated following the 'n'-dimensional Euclidean distance formula.

$$D_{ij} = \sqrt[2]{\sum_{v=1}^{128}(X_{vi} - X_{vj})^2} \tag{5}$$

Equation 5 transformed the complex idea of facial recognition into an elementary concept of finding the distance between two points.

The issue that arouse was to identify the appropriate threshold in order to accurately classify faces. This standard number was a hyperparameter that must be obtained before enforcing the whole DeepView system. The Facenet paper explained that a euclidean separation of 1.1 is the limiting value. All distances less than 1.1 implies that the embeddings are from the same identity while a greater number is from different individual [16]. The researchers found out the suitable threshold value for this application by comparing the accuracy of different threshold values.

IV. FACE RECOGNITION USING CLUSTERING

A. Face Clusters

A cluster is a collection of closely-related points. Each of the points in the cluster represents a person's face in a specific instance. A summary of a unique face identity was generated.

Each cluster will be represented by a centroid that is solved by:

$$P_{ij} = (x_{i1}, x_{i2}, ...x_{i128}) \tag{6}$$

$$\overline{P_i} = (\overline{x_{i1}}, \overline{x_{i2}}, ...\overline{x_{i128}}) \tag{7}$$

$$\overline{x_{ij}} = \frac{\sum_{i=1}^{n} X_{ij}}{n} \tag{8}$$

B. Identities

Existing clustering approaches are applied when a fixed amount of identities is already known from the start. Also, in other clustering algorithms, when introducing a new point to already established clusters, the whole algorithm should be performed again to all of the points which will add complexity to the system. To address this, a clustering algorithm based on k-nearest neighbor and k-means clustering was devised. Given a face encoding (point):

1. Compare the point with all the cluster centroids. Find the shortest point to centroid distance, take note of this cluster.
2. Compare the point with all the points of the closest cluster. Solve for the mean distance. If the average distance is less than the threshold, add this point to the cluster. If by adding this point, the cluster now has greater than 128 points, perform Density-Based Spatial Clustering of Applications with Noise (DBSCAN) to reduce the cluster to its core points only. Recompute the new centroid.
3. If the mean distance is greater than the threshold, this point will be the starting point of a new cluster. This point will also be the centroid of this newly-formed cluster.

Figure 1 summarizes the clustering method.

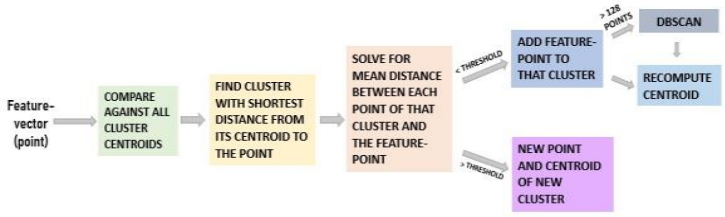

Fig. 1. Clustering Implementation Flowchart

C. Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

As points are added to the cluster, outliers must be removed using Density-Based Spatial Clustering of Applications with Noise(DBSCAN) to further establish the relationship of the points that symbolize the identityGiven a set of points in some space, DBSCAN groups together points that are closely packed. Marking as outliers points that lie alone in low-density regions [17].

The system executes a DBSCAN when points in a cluster exceeds 128 in number since the system uses 128-dimension embeddings. This algorithm will be used only when cleaning up existing clusters and not for establishing new clusters.

V. The Final System

Figure 2 shows the hardware setup of the system. All the functions and algorithms were packaged inside a dedicated computer.

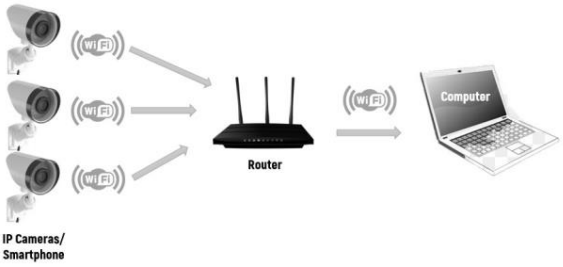

Fig. 2. Hardware Setup Diagram

A. Core Functions

The computer housed the following significant functions:

1. Connecting with the cameras and receiving image frame data from each of them.
2. Perform face detection and face alignment on the camera frames.
3. Calculate embeddings by neural network and recognize a face using face clustering.
4. Keep a database of the unique faces encountered as a unique identity. At the same time, keep a log of all the history of encounters with each identity.
5. Face Search functionality. The user can upload an external picture of a person and the system will perform a scan on its database if that particular person was encountered. Generate a logged data that is accessible in a spreadsheet file.

B. Multithreading

Executing all of the functions on a single thread was very inefficient and caused delays on the processing of the data. Therefore, various functions were performed on different threads. Multithreading was vital since the system also uses a queuing system which will be discussed on the next subsection.

The program has a separate thread for each of the following functions:

1. A thread for getting the frame data from the cameras.
2. Each camera has its allocated thread.
3. A thread for facial detection and alignment.
4. A thread for facial recognition and data logging.
5. A thread for maintaining a synchronized system clock.
6. A thread for graphical user interface (GUI) tools.
7. A thread for displaying video streams from the camera.

C. Queuing System

The face recognition aspect of the system is the most processing-intensive part. With the limited hardware the researchers have, it was necessary to devise a plan that provides an efficient way of handling processes without compromising the precision. Therefore, the DeepView system utilized a first in-first out (FIFO) queuing system during the facial recognition stage.

For each of the input camera, frames are placed inside a queue. The frames while in the queue are saved to the cache and accessed later during its turn on the queue. After face recognition, the particular face is deleted from the cache and the queue proceeds. Also, when this faces were detected, timestamps were also saved to preserve the accuracy during the data logging stage. Figure 3 shows the scheme used in queuing.

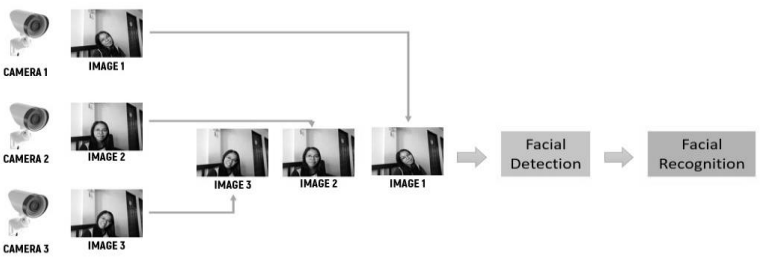

Fig.3. First in-First out Scheme

D. Data Logging

Data logging was accomplished for each of the unique identities encountered by the system. This data was saved to serialized object which can be then accessed and updated in the future when the system encounters the same identity.

Each identity has an archive of the following data:

1. Time
2. Date
3. Camera encountered

These data are being updated as the system encounters an identity over and over again. There is no upper limit to the number of entries that the system will store for each identity. Therefore, all the history with an identity will be saved in the database.

E. Exporting the Logged Data

The contents of the archive can be accessed as a spreadsheet file by running an external batch file inside the DeepView directory. The spreadsheet is formatted as follows:

- Each worksheet contains the history of a distinct individual.
- The first three columns are occupied by the following data: Date, time and camera.

F. Face Search

The system can receive an external photo to be cross referenced on DeepView's database. If the system finds a matching identity, the program displays a corresponding prompt whether a match is found or not.

G. Graphical User Interface (GUI)

The GUI has two screens: the primary window and the opening screen that must contain a trigger to launch the primary window as shown in Figure 4.

. Final Version of GUI Starting Screen

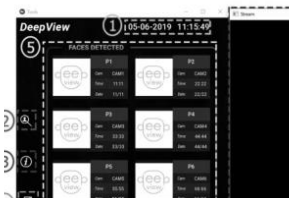

Fig. 4. Final Version of GUI Starting Screen

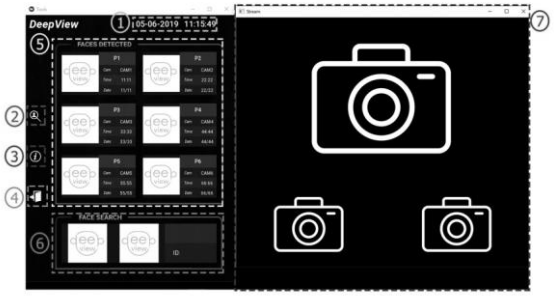

Fig. 5. Final Version of GUI Main Console

Figure 5 shows the final version of the GUI of the system. As in Figure 5, the GUI was successfully integrated to proposed components functionalities:

1. System time-displays the synced system date and time.
2. Face Search Button-from external image source, search the database for matched identity in the system.
3. Information Button-show the logo, authors and purpose of DeepView.
4. Logout Button-dropdowns *logout button* for exiting the main interface.
5. Faces Detected Panel-displays the history of recognized faces from the cameras. The faces is updated left to right and top to bottom.
6. Face Search Panel-displays the face search results. The left image shows the image uploaded externally while the right image shows the matched image in the database if there is any.
7. Camera Panel-shows the feed from the cameras. The primary camera is the feed with the largest video size located at the top while the secondary cameras are shown in the bottom.

VI. RESULTS

A set of 126 pictures containing 126 unique identities were gathered for testing. The images are gathered from the 3 cameras at different angles.

A. HOG

For the HOG-based face detection, the average execution time was 7.30547 ms. It has an accuracy of 79.37%, successfully detecting 100 faces out of the 126 images as shown in Figure 6.

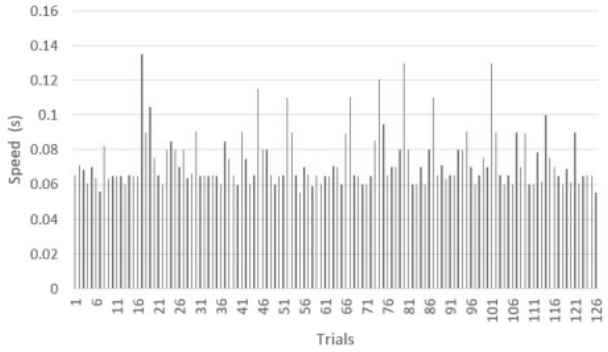

Fig. 6. HOG Accuracy Results for 126 images

B. Face Alignment using Affine Transformation

Using the same dataset of 126 images, benchmarking tests were conducted on the face alignment algorithm. Figure 7 shows the affine transformation rate.

The average execution time applying transformation to each face was 1.94026 ms.

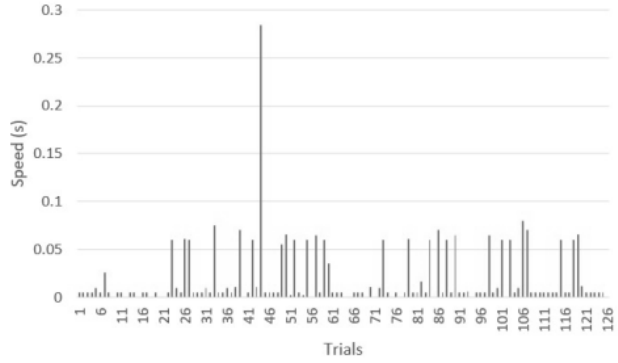

Fig. 7. Affine Transformation Rate for each trials

Figures 8 shows the image captured using the three (3) cameras. The face are not aligned. However, using Affine alignment, the image are properly aligned as shown in Figure 9.

Fig. 8. Images before Face Alignment as captured by the three (3) cameras

Fig. 9. Images after Face Alignment

C. Encoding Calculation Speed

When implemented, the neural network demonstrated a remarkable speed regardless of the slow construction time. The average time it took the network to assign an embedding to face was found to be 3.92 ms. Figure 10 summarizes the rate of the neural network performance on different images.

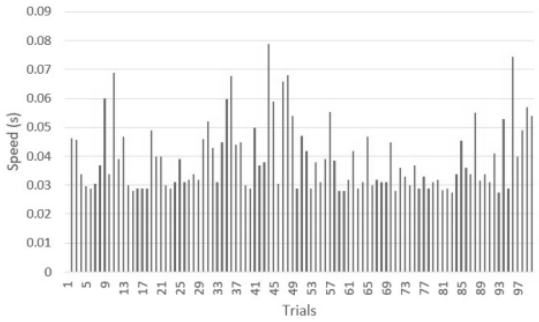

Fig. 10. Embeddings Computation Rate

D. Face Recognition Accuracy

The dataset was labeled and the relationship between the pictures was already predefined and the accuracy was examined. The test involved solving the Euclidean distance between two points and this was compared with a varying threshold value. The results were used to determine the threshold value that the system utilize on its varied set of functionalities such as point comparisons and clustering.

The previous sections demonstrated opposite patterns when comparing identities with a different or same individual. The challenge for the system was to find an ideal threshold value that would precisely balance the decision making in identity similarity. The suggested threshold value was 0.6 [18] as shown in Figure 11.

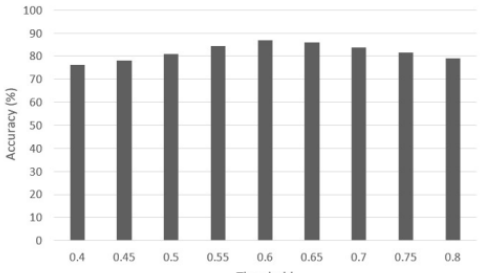

Fig. 11. Average Face Recognition Accuracy

Based on this findings, the researchers decided to settle with the recommended value of 0.6 with an average accuracy of 87.03%. This number will be used in distance and centroid comparisons and also in clustering computations.

However, the OpenFace implementation promised accuracy of 92.10% [18]. The underperformance can be attributed to the quality of the camera that was used for data gathering and variance in lighting was a great factor.

E. Camera Frame Rate

The frame rate of the camera must coincide with the average process time that the system will take on a single face. The average queue time for the system is equal to 13.7 ms. Using the formula of the frame rate, each camera can be set to approximately 24 frames per second. But to increase the stability of the system, the researchers decided to settle with a 10 fps frame rate.

VII. Conclusion

The researchers successfully developed a fully-functioning facial recognition system with data logging capabilities using well-established algorithms. The system employs a deep learning based facial recognition model. Instead of learning the faces, the model was trained in such a way that it minimizes distances for similar faces and maximizes distance for different faces. This model converts a face-picture to a feature vector that can be represented as a point in a multidimensional space. Before conversion, a HOG-based face detection and Affine transformation were utilized to preprocess a photo. Standard Euclidean geometry was used to determine relationship between points. Because of this paradigm an identity of a person can be represented as a cluster of closely related points. These clusters enables DeepView to re-identify the people it encounters. The proposed method yielded an 87.03% accuracy. This is significantly lower than its expected accuracy of 92.10%. DeepView has performed based on the resulted accuracy and can be deployed in real world scenario.

Acknowledgments

The authors would like to acknowledge the financial support of the Philippine Council for Industry, Energy and Emerging Technology Research and Development (PCIEERD) of the Department of Science and Technology (DOST) and Mindanao State University – General Santos City (MSU-GSC) for the establishment of our research laboratory.

References

[1] K. Sato, S. Shah, and J. K. Aggarwal, “Partial face recognition using radial basis function networks,” 1998.

[2] K. Assaleh, T. Shanableh, and K. Abuqaaud, “Face recognition using different surveillance cameras,” *IEEE*, 2013.

[3] V. Sathish. (2008) Dubai airport to get face recognition units. [Online]. Available: https://www.emirates247.com/eb247/news/national/dubaiairport-to-get-face-recognition-units-2008-10-28-1.57945

[4] T. Ong. (2017) Dubai airport is going to use facescanning virtual aquariums as security checkpoints. [Online]. Available: https://www.theverge.com/2017/10/10/16451842/dubaiairport-face-recognition-virtual-aquarium

[5] (2015) Face recognition. [Online]. Available: https://esrc.ukri.org/aboutus/50-years-of-esrc/50-achievements/face-recognition/

[6] P. Bump. (2018) Facial recognition in law enforcement – 6 current applications. [Online]. Available: https://emerj.com/ai-sectoroverviews/facial-recognition-in-law-enforcement/

[7] J. Carpenter. (2017) 5 benefits of biometric face recognition technology. [Online]. Available: https://blog.crossmatch.com/authentication/benefitsbiometric-face-recognition-technology/

[8] Apple. (2018) About face id advanced technology. [Online]. Available: https://support.apple.com/en-us/HT208108

[9] T. Kanade, “Picture processing system by computer complex and recognition of human faces,” 1974.

[10] D. Pissarenko, “Eigenface-based facial recognition,” *December 1st*, 2002.

[11] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, “Eigenfaces vs. fisherfaces: Recognition using class specific linear projection,” Yale University New Haven United States, Tech. Rep., 1997.

[12] J. Meng, Y. Gao, X. Wang, T. Lin, and J. Zhang, “Face recognition based on local binary patterns with threshold,” in *Granular Computing (GrC), 2010 IEEE International Conference on*. IEEE, 2010, pp. 352–356.

[13] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” vol. 1, 07 2005, pp. 886–893.

[14] X. Chai, S. Shan, and W. Gao, "Pose normalization for robust face recognition based on statistical affine transformation," in *Fourth International Conference on Information, Communications and Signal Processing, 2003 and the Fourth Pacific Rim Conference on Multimedia. Proceedings of the 2003 Joint*, vol. 3. IEEE, 2003, pp. 1413–1417.

[15] M. Berger, M. Cole, and S. Levy, *Geometry I*, ser. Universitext. Springer Berlin Heidelberg, 2009. [Online]. Available: https://books.google.com.ph/books?id=5W6cnfQegYcC

[16] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815– 823.

[17] H.-P. Kriegel and M. Pfeifle, "Density-based clustering of uncertain data," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 2005, pp. 672–677.

[18] B. Amos, B. Ludwiczuk, and M. Satyanarayanan, "Openface: A generalpurpose face recognition library with mobile applications," CMU-CS-16118, CMU School of Computer Science, Tech. Rep., 2016.