



University of Calgary

PRISM: University of Calgary's Digital Repository

Graduate Studies

The Vault: Electronic Theses and Dissertations

2019-09-12

Intelligent Data Analysis for Early Warning: From Multiple Sources to Multiple Perspectives

Afra, Salim

Afra, S. (2019). Intelligent Data Analysis for Early Warning: From Multiple Sources to Multiple Perspectives (Unpublished doctoral thesis). University of Calgary, Calgary, AB.

<http://hdl.handle.net/1880/110990>

doctoral thesis

University of Calgary graduate students retain copyright ownership and moral rights for their thesis. You may use this material in any way that is permitted by the Copyright Act or through licensing that has been assigned to the document. For uses that are not allowable under copyright legislation or licensing, you are required to seek permission.

Downloaded from PRISM: <https://prism.ucalgary.ca>

UNIVERSITY OF CALGARY

Intelligent Data Analysis for Early Warning: From Multiple Sources to Multiple
Perspectives

by

Salim Afra

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF DOCTOR OF PHILOSOPHY

GRADUATE PROGRAM IN COMPUTER SCIENCE

CALGARY, ALBERTA

SEPTEMBER, 2019

© Salim Afra 2019

Abstract

Misusing and benefiting from the development in technology for communication, criminal and terror groups have recently expanded and spread into global organizations and activities. Fortunately, it is possible to benefit from the technology to fight against terror and criminal groups by tracing, identifying, surrendering, and preventing them from executing their bloodily plans. Indeed, it is very affordable to capture various kinds of data which could be analyzed to predict potential criminals and terrorists. Data comes in various formats from text to images, and may become available incrementally due to dynamic sources. This leads to what has been recently classified as big data which has attracted considerable attention from the industry and the research community. Researchers and developers involved in this domain are trying to adapt and integrate existing techniques into customized solutions which could successfully and effectively handle big data with all its distinguishing characteristics. Alternatively, tremendous effort has been invested in developing new techniques to cope with big data for situations where existing techniques neither individually nor as an integrated group could address the shortcomings in this domain. Realizing the need for effective solutions capable of dealing with criminal and terror groups could be mentioned as the main motivation to undertake the study described in this thesis.

The main contribution of this thesis is an early warning system that uses different sources of data to identify potential criminals and terrorists (hereafter both criminals and terrorists will be meant when any of them is mentioned in the text). The process works as follows. Criminal profiles are analyzed and their corresponding criminal networks are derived. This automates and facilitates the work of crime analysts in predicting events that may lead to disaster. We used face images as a data source and performed different studies to determine the accuracy and effectiveness of current face recognition and clustering algorithms in identifying people in uncontrolled environments, which are actually the environments encountered in real situations when dealing with criminals and terrorists. We trained our own face recognition algorithm using convolutional neural networks (CNN) by pre-processing the input

images for better recognition rates. We showed how this is more effective than frontalized profile face images. We designed a queuing system for surveillance camera monitoring to raise an alarm when unknown people who pass through a monitored area turn into potential suspects. We also integrated different data sources such as social media, news, and official criminal documents to extract criminal names. We then generate a criminal profile which includes the activities that a given criminal is involved in. We also linked criminals together to build a criminal network by expanding the coverage and analyzing the collected data. We then proposed several unique criminal network analysis techniques to provide better understanding and knowledge for crime analysts. To achieve this, we added more functions related to criminal network analysis to NetDriller which is a powerful social network analysis tool developed by our research group. We also designed an algorithm for link prediction which better detects if a link between two nodes will exist in the future. All these functionalities have been well integrated into the monitoring system which has been developed and well tested to demonstrate its applicability and effectiveness.

Keywords: criminal networks, terror networks, early warning, link prediction, clustering, classification, face recognition.

Acknowledgements

I would like to express my sincere gratitude, appreciation, and thanks to my great supervisor Dr. Reda Alhajj for his continuous advice, support, guidance, and enthusiasm. I have learnt a lot from him in the past years and this thesis would not be possible without him. It is really so hard to list everything he has done for me. His valuable efforts will always be remembered.

I also want to thank Dr. J. Rokne, Dr. M. Moussavi, Dr. M. Moshirpour, and Dr. B. Tavli, my committee members, for their time and effort in reviewing this work and their feedback. I would like to highly appreciate the help, support, and collegiality of my friends and all the members in the research group of Dr. Alhajj. Thanks for helping me to get through difficult times smoothly and their moral support.

I also would like to thank the Department of Computer Science at the University of Calgary for providing all the necessary resources for successfully completing my thesis.

Finally and most importantly, I am heartily grateful to my family for their love, encouragement, support, patience, and sacrifices throughout my life.

Table of Contents

Abstract	ii
Acknowledgements	iv
Table of Contents	v
List of Figures and Illustrations	viii
List of Tables	xi
1 Introduction	1
1.1 Problem Definition and Motivation	2
1.2 Contributions	4
1.3 Organization of the Thesis	5
2 Background and Relevant Literature	7
2.1 Text Mining and Analysis	7
2.2 Clustering	8
2.3 Criminal Network Construction and Analysis	9
2.4 Image Processing and Face Recognition	11
2.4.1 Face Detection	11
2.4.2 Face Identification	13
3 Combining Feature Extraction and Clustering for Better Face Recognition	16
3.1 Introduction	17
3.2 Methodology	19
3.2.1 Pre-Processing	19
3.2.2 Feature Extraction	20
3.2.3 Applying Clustering Techniques	24
3.3 Datasets	26
3.3.1 JAFFE	26
3.3.2 AT&T	26
3.3.3 LFW	26
3.3.4 Extended YaleB	27
3.4 Experiments & Results	27
3.4.1 Face Pre-Processing Results	27

3.4.2	Feature Extraction Runtime	28
3.4.3	Clustering Results	30
3.5	Conclusions	34
4	Multi-view Face Detection and Recognition for Effective Identification of Social Entities	37
4.1	Introduction	38
4.2	Methodology	39
4.3	Experiments & Results	41
4.4	Conclusion	44
5	Face Reconstruction from Profile to Frontal Evaluation of Face Recognition	46
5.1	Introduction	47
5.2	Related Work	49
5.3	Methodology	51
5.3.1	Face Detection	51
5.3.2	Face Frontalization	53
5.3.3	Feature Extraction	53
5.4	Experiments & Results	55
5.4.1	Datasets	55
5.4.2	Experiments	57
5.4.3	Results	58
5.5	Conclusions	60
6	Link Prediction by Network Analysis	63
6.1	Introduction	64
6.2	Related Work	66
6.3	Methodology	69
6.3.1	The Algorithm	69
6.3.2	Graph Database	71
6.4	Datasets	71
6.5	Experiments and Results	72
6.6	Conclusions	76
7	NetDriller Version 2: A Powerful Social Network Analysis Tool	85
7.1	Introduction	86
7.2	Brief Overview of Existing Similar Tools	87
7.3	The User Interface	88
7.4	New Unique Functionalities	89
7.4.1	Network Construction	90
7.4.2	Incremental Network Modification	94
7.4.3	Temporal Network Visualization	95
7.4.4	Link Prediction	96
7.4.5	Hierarchical Zooming	96

7.5	Conclusions	98
8	Integrated Framework for Criminal Network Extraction from Web Using Text Analysis	100
8.1	Introduction	101
8.2	Related Work	103
8.2.1	Frameworks	104
8.2.2	Text Collection	106
8.2.3	Text Analysis	107
8.2.4	Keyword Extraction	109
8.2.5	Constructing and Visualizing Criminal Network	110
8.2.6	Network Analysis	111
8.2.7	Link Prediction	111
8.3	Methodology	113
8.3.1	Relevant Text Collection	113
8.3.2	Text Analysis	115
8.3.3	Information Extraction	116
8.3.4	Criminal Graph	117
8.3.5	Network Analysis	117
8.4	Framework Implementation and Web Application	119
8.4.1	System Implementation and Validation	119
8.4.2	Web Application	126
8.5	Conclusions	132
9	Early Warning System: From Face Recognition by Surveillance cameras to Social Media Analysis to Detect Suspicious People	134
9.1	Introduction	135
9.2	Methodology	139
9.2.1	Camera System	139
9.2.2	Identify People	140
9.2.3	Alert	142
9.2.4	Queuing System	143
9.2.5	Search for Information	146
9.2.6	Construct and Analyze Person's Network	148
9.3	Dataset	151
9.4	Experiments & Results	153
9.5	Conclusions	155
10	Summary, Conclusions and Future Research Directions	157
10.1	Conclusions	158
10.2	Future Research Plan and Directions	160
	Bibliography	161

List of Figures and Illustrations

3.1	Face clustering overall methodology	19
3.2	Pre-processing steps example	19
3.3	Feature extraction process	23
3.4	(a) presents the JAFFE dataset, as seen all the images are taken in a controlled environment with the difference being the expression shown by the female. (b) presents the AT&T dataset where the face images are also taken in a controlled lab environment with difference in facial features and expressions for each person. (c) presents the LFW dataset, this dataset has a collection for each person images from the web. As seen the images are not related to a scene and the image for a person is taken in different time frames (old/young). It presents a unique challenge to face recognition, as also another people can interfere in the image as shown in the sample. (d) presents the YaleB dataset, the dataset is taken in a controlled environment but the challenge here is with the illumination of each image different such that some images are unrecognizable even for humans.	25
3.5	Figure of all the rankorder results ith varying thresholds	36
4.1	Overall face recognition methodology	40
4.2	Face alignments steps example	40
4.3	The first row shows sample of LFW dataset where faces of the person are taken in an uncontrolled environment in different time instances. The second row shows FEI face database where face images of the person are taken in a controlled environment with a varying rotation of up to 180 degrees	41
4.4	Receiver Operating Characteristic (ROC) curves of the FEI and LFW datasets as mentioned in Section 4.3	44
5.1	The overall methodology for face frontalization from any pose into a frontal one. First face detection is applied to get face boundaries, after that the face is frontalized to get frontal view features of the image which are then used for recognition	51
5.2	Image manipulation for training: (a) an original image from WIDER dataset; (b) a rotated version of the image shown in (a); (c) - (g) versions of the image in (a) using different gamma levels, namely 0.5, 1.5, 2, 2.5, and 3, respectively.	52
5.3	Effective frontalization rotation technique	54

5.4	Sample of the image datasets used; variations in each dataset are illustrated. (a) a sample of FEI dataset where images are taken in a controlled environment. (b) corresponds to FERET dataset where images are taken in controlled environment and at different time sessions. (c) sample from the IJB-A dataset where faces are collected in the wild with different poses and facial appearance.	56
5.5	Sample of the different face frontalization techniques output using two people's set of images	59
6.1	Illustrative example of our prediction algorithm on a simple network. Size of nodes represent eigenvector centrality (larger size means higher value), while color of nodes shows betweenness centrality (more reddish means higher value).	70
6.2	Clustering coefficient analysis of datasets.	73
6.3	Path distance distribution of datasets.	74
6.4	Example of our prediction algorithm on the Karate Network with 30% of the edges removed.	75
7.1	NetDriller Interface	89
7.2	Network Interface	90
7.3	Centrality Measures Calculations	91
7.4	Centrality Measures Calculations	92
7.5	Twitter Data Import and Options	93
7.6	Web Crawler	94
7.7	Import Options	96
7.8	Incremental Network Visualization	97
7.9	Link Prediction Calculation	98
7.10	Hierarchical Zooming Menu	99
7.11	Hierarchical Zooming in Action	99
8.1	System architecture of the integrated framework	112
8.2	False Positives Examples in Crime Articles Prediction	122
8.3	NER Erroneous Examples in Tweets	124
8.4	Example Article Showing People Mentioned Several Times by Their Last Name	125
8.5	Web Application Homepage	126
8.6	Search to analyze people mentioned in articles	127
8.7	Person Analysis Page	128
8.8	Using NetDriller for Analysis	128
8.9	Analyzing the Criminal Graph Using the Web Application	129
8.10	Identifying Key Nodes Using the Web Application	130
8.11	Adjust Network Functionality	131
8.12	Clustering View from the Web Tool	131
8.13	Link Prediction Output	132
9.1	The overall methodology	138
9.2	Interface for security officers to upload or analyze a video stream	140

9.3	Image manipulation for face detection training. (a) shows an original image in the WIDER dataset. (b) shows a rotated image of (a). (c) - (g) shows the image at gamma levels (0.5, 1.5, 2, 2.5, 3)	141
9.4	Person Collected Information Details	147
9.5	Person Collected Information Details	149
9.6	Sample of gallery images (smile and neutral) of the Chokeypoint Dataset . . .	152
9.7	Sample of the video images collected from the Chokeypoint dataset	152
9.8	Sample of the output of our face detector.	153

List of Tables

3.1	Pre-processing face detection accuracy	28
3.2	Feature extraction run time. Every table reports related to a dataset, a comparison of the run time of the three different clustering techniques used in this study. The results are shown in the format of H:MM:ss, where H is hour, M is minutes, and s is seconds. Values marked as ”-” indicate that the clustering algorithm did not finish in a matter of running for 1 day.	29
3.3	Clustering run time. Every table reports results of a dataset, comparing run time of the three clustering techniques used in this study. The results are shown in the format of H:MM:ss where H is hour, M is minutes, and s is seconds. Values marked as ”-” indicate that the clustering algorithm did not finish in a matter of running for 1 day.	31
3.4	Clustering accuracy. Every table reports results of a dataset, comparing the accuracy of the different feature extraction techniques with the three clustering techniques used in this study.	34
5.1	Frontal set accuracy	60
5.2	Profile Accuracy	60
5.3	FEI Angular Results	61
5.4	FERET Angular Results	61
5.5	IJBA verification results	62
6.1	The eleven similarity metrics used in link prediction; set $\Gamma(u)$ represents neighbors of node u in the network, and $ \Gamma(u) $ shows degree of node u	77
6.2	Data Set Networks Size	78
6.3	Karate	79
6.4	AdjNoun	80
6.5	Dolphins	81
6.6	Football	82
6.7	Les Misérables	83
6.8	Polbooks	84
8.1	Sample Words from the Crime Related Gazetteer	121
8.2	Confusion Matrix of the Classified Articles	122
8.3	Classification Accuracy of the Collected Data	122
8.4	Confusion Matrix of Twitter Named Entity Recognition	123
8.5	Confusion Matrix of DailyMail Named Entity Recognition	123

8.6	Confusion Matrix of Reports Named Entity Recognition	123
8.7	Accuracy of Named Entity Recognition on the three datasets	124
9.1	Accuracy results of the known, suspects and unknown people using the P1L-S3-C1 camera sequence	154
9.2	Accuracy results of the known, suspects and unknown people using the P1L-S3-C2 camera sequence	154
9.3	Accuracy results of the known, suspects and unknown people using the P1L-S3-C3 camera sequence	155

Chapter 1

Introduction

Over history, humans felt the need to capture and keep various types and volumes of data ranging from classical text, to images, voice, etc. Always, new sources and domains of data/knowledge emerge as a result of the rapid development in personally or publicly used equipment, e.g., telephone, television, washing machine, car, plane, satellite, hand-held devices, sensors, etc. In response, researchers and practitioners realized the need to catch up by developing techniques capable of maximizing the benefit from the enormous data generated by technology. Improving computing power and storage capacity are not to cope with the rapid change and its associated needs. Thus, improve and powerful algorithms should be seen as the major players who should dig deep in the data to capture and identify all valuable nuggets. In this sense, my research focuses on developing an early warning system capable of collecting data from different sources used to identifying suspects and predicting events which require raising alarm for authorities to take timely action which may lead to avoiding or preventing a disaster. Such a system is severely needed because criminals and terrorists do exist and their number is increasing faster than ever before, whether based on ethnicity, racism, religion, or affected by economic crisis. Criminology and terror are global problems that cannot be ignored nor any kind of tolerance is acceptable.

1.1 Problem Definition and Motivation

Recent development in technology has facilitated effective and robust automated monitoring of specific locations, persons, instruments, etc. Automation is an attractive approach to complement and sometimes replace the human factor in monitoring. For instance, a video captured by a surveillance camera may be directly analyzed by a human expert who may set alert as warning for certain exceptional or outlier cases which need further timely attention. Also, a domain expert may manually trace and analyze a set of social media traces to identify certain incidents or bodies who should receive further consideration; the same applies to traditional media where the process mainly involves document analysis. It is even important to use an image as input and locate as much as possible related data available in different formats from various sources, including social media, traditional media, domain specific archives, etc. It is equally important to use the latter sources to locate existing photos of a specific person who should be investigated further. All these activities may be automated by developing and integrating some advanced computing techniques from machine learning, image processing, text processing and social network analysis.

Fortunately, huge amounts of data are collected and can be analyzed for knowledge discovery. However, existing techniques are not advancing at enough speed to catch data collection rate and volume. As a result, researchers and practitioners are still struggling to develop new methods capable of better analyzing existing data for maximized benefit. In fact, developing new techniques is a continuous ongoing effort which requires either adapting and adjusting existing techniques or developing new techniques to serve new domains. In other words, one technique which is popular and attractive today may become obsolete in the near future. Accordingly, researchers and practitioners should be on duty to provide timely replacement by techniques that satisfy emerging needs.

The objective of this PhD thesis is to develop a system capable of identifying potential criminals/terrorists or suspicious subjects in order to avoid/prevent their actions which may lead to a disaster. Keeping in mind that data is streaming from a variety of sources and

in different formats, a robust system should be capable of integrating various data types, including images captured by surveillance cameras, social media data, traditional media data, data from domain specific archives, and domain experts knowledge. In particular, this research will concentrate on identity resolution based on some initial data available to discover all possible information that will help in understanding, to the best level possible, the character and behavior of a specific person or group. One possible scenario may be described as follows. Given an image of the face of a person who has been witnessed more than expected in a scene, it is required to investigate existing data sources to find more information related to the specific person.

In case of a crime, it will be possible to identify specific suspects with a higher degree of confidence based on the amount of information captured and analyzed. Finding others who are connected to a given suspect is also important for a comprehensive study by authorities. Once a network is constructed, it is possible to incrementally adjust the network as more information becomes available. It is also possible to identify within a network key actors at various levels. Then, authorities may become interested in studying the behavior of the network once key actors at each level are removed from the network. This allows authorities to understand how the network restructures and what could be done to dissolve or disconnect actors in the network as quick as possible.

Combining data from a variety of sources is very attractive and highly contributes to a more robust outcome despite difficulties associated with the process, e.g., heterogeneous domains, different scales, missing values, noise, etc. These obstacles and the like could be overcome by utilizing some advanced machine learning techniques. Once the data is ready for processing, it becomes possible to produce some valuable results which may guide and shape the decision making process. Techniques to be used in the analysis will be borrowed from a variety of disciplines, including machine learning, image processing, text analysis, and social network analysis.

This research has the following components:

1. data collection from various sources, including social media, traditional media, surveillance cameras, etc.
2. image analysis and recognition,
3. intelligent data analysis using clustering, classification and frequent pattern mining,
4. search and matching of images and information,
5. link prediction for possible connections to exist or disappear in future,
6. studying network structure and restructuring after certain actors are removed.

My research focuses on identifying influential actors in a social network and will then investigate how a network will restructure in case first, second, third, etc. level influential actors are removed from the network. This will guide authorities to develop specific strategies to deal with influential actors at each level and within the network as a whole.

Images are captured and normalized by concentrating on faces captured by surveillance cameras. Then, the features of each face are extracted to form a feature vector. All feature vectors of existing faces can then be clustered to help in speeding up the matching of a new face once captured.

Text collected from social media, archives, traditional media, etc. is processed to identify key terms and persons. Clustering and frequent pattern mining based techniques have been developed to analyze text. For instance, we construct an adjacency matrix where each tweet is a row and each term, word or entity is a column. Such a matrix is analyzed using clustering or frequent pattern mining techniques to find the relationship between tweets or between terms. This would reveal important knowledge related to relevant tweets, terms, etc.

1.2 Contributions

This research has the following contributions:

- An early warning system capable of highlighting potential threat and hence avoiding, preventing or at least reducing the possibility of attacks leading to disasters.
- Capturing a face image and locating related information from the available sources, e.g., social media, archive, traditional media, etc.
- Performing a study on applying face clustering with a different clustering method
- Improving the Netdriller tool by adding more capability to analyze criminal networks, the same new features could be used to analyze any other type of network which has some common characteristics with terror networks; these include biological networks, co-workers networks, etc.
- Capturing and analyzing text, whether tweets or traditional documents, to identify some relevant terms, keywords, entities, etc.
- Improving face recognition rate by re-training models with pre-processed face images
- Employing machine learning, image processing, text analysis, and social media analysis techniques in a fully working system to help authorities in handling cases related to terror and criminology.

1.3 Organization of the Thesis

The rest of this thesis is organized as follows. Chapter 2 lists the background and relevant literature related to different parts of the developed system . In Chapter 3, we perform a study on clustering algorithm and feature vector extraction techniques which work the best on the problem of clustering faces. In Chapter 4, we evaluate the performance of using different face detection and recognition techniques; we also evaluate their performance in an uncontrolled multi-view environment. In Chapter 5 we study the effect of applying face rotation to reconstruct faces from profile views to frontal ones using different frontalization

methods to achieve a better face recognition rates on the profile views by training our own CNN model to extract features from faces. In Chapter 6 we developed a link prediction algorithm to find and predict hidden links between entities in a social graph. In Chapter 7 we cover NetDriller which is a social network analysis tool we extended by adding: (1) functionalities to construct social networks from different social media sources (2) more analysis components to help in building networks to guide analysts in visualizing and analyzing criminal networks. In Chapter 8, we developed an early warning system that recognizes people automatically in a surveillance camera environment using a queuing bases system. We also analyzed social media text of a recognized person to then construct the person's network from individuals mentioned with him/her in the text. Further analysis will allow security experts to mark this person as a suspect or innocent. In Chapter 9, we propose a system that extracts criminals' information and their corresponding network using Web sources, such as online newspapers, official reports, and social media. Chapter covers a summary, conclusions, and future research directions of this thesis.

Chapter 2

Background and Relevant Literature

2.1 Text Mining and Analysis

Traditional applications have focused on retrieving and processing raw data or data available from database systems [3]. With the surge of data and information in the web, text mining has been employed for information retrieval to facilitate information access rather than merely analyzing existing data. Moreover, text mining is used to further process the retrieved data [78].

In this thesis, we used text mining to analyze criminal data collected from different sources such as newspapers, online social media, and police reports. The purpose of applying text analysis on these data sources is to extract from the text criminal incidents and criminal personnel involved. The extracted information is then used to create criminal profiles based on their involvements along with their criminal graph.

Several frameworks had been implemented to provide the basic functionality used to analyze text. These frameworks use natural language processing (NLP) which is concerned with using computational learning techniques to understand text [63]. Frameworks such as Stanford CoreNLP [108], GATE [35], NLTK toolkit [17] provide a standard NLP pre-processing pipeline which includes the following:

- Sentence Segmentation: Usually, the first step in NLP processing is identifying sentences and analyze them individually.
- Tokenization: After identifying the sentences, tokenization is used to identify individual words in the sentence.
- Part-Of-Speech (POS) Tagging: For every identified word, POS tagging will label the word as its corresponding POS tag, whether it is a noun, verb, adjective, etc.
- Named Entity Recognition (NER): This is an important step in the NLP toolkit to identify named entities such as people (criminals), places, and organizations.

2.2 Clustering

Clustering is an unsupervised classification of patterns or observations into their corresponding groups (clusters) [73]. Clustering had been addressed and used in many fields where pattern analysis is needed to distribute similar or same data categories into one group without knowing in advance what the labels of the data points are.

Clustering has been applied in pattern recognition and has been successfully used in many different fields. Face clustering analysis has not received enough attention as the clustering of faces not only depends on the clustering algorithm used, but also on the feature extraction technique employed. A variety of feature extraction techniques described in the literature can be applied as a preprocessing step for face clustering, but there is no widely accepted feature representation technique for face representation.

Several studies (e.g., [64], [180]) have evaluated the performance of a feature representation technique in association with a single clustering algorithm. For instance, Hoe et al. [64] used Spectral clustering to evaluate the performance of local gradients with pixel intensity as feature vector for face representation. Zhao et al. [180] used Hierarchical clustering to cluster photos in a personal gallery. They used a combination of features to represent a face

image based on information extracted from faces, bodies and context information. Zhu et al. presented a new clustering algorithm specifically for face clustering [182]; it is called Rank Order distance clustering. Clustering is achieved by measuring the dissimilarity between two faces based on their neighborhood information.

Other studies investigated the effect of using feature representation techniques in combination with clustering algorithms. For instance, Heisele et al. [59] classified faces using Support Vector Machine (SVM) to evaluate three different feature representation techniques, namely component-based method and two global methods for face recognition. In their evaluation, they used the ROC curves of these feature representation techniques for formal and rotated faces. They determined that the component based method outperformed the two global methods. While in [121] they presented analysis similar to our study by checking the performance of different feature extraction techniques and clustering algorithms. They used component based features and compared with a commercial face matcher. After extracting features from a face, they then applied three different clustering algorithms, namely K-means, Spectral clustering, and Rank Order distance which we have used in this study. They used two datasets for their experiments, namely Pinellas County Sheriff's Office (PSCO) and Labeled Faces in the Wild (LFW) dataset. Their results show that commercial face matcher outperformed component based for rank order clustering, but they couldn't run the commercial face matcher on K-means nor Spectral clustering because the feature vectors are not provided by the commercial product. They also showed that Rank Order clustering performs better than K-means and Spectral clustering.

2.3 Criminal Network Construction and Analysis

A criminal network/graph shows interactions between criminals as present in the analyzed domain or text. A criminal graph is defined as $G = (V, E, W)$, where V is the set of vertices representing criminals. For the work described in this thesis, we used text as the main

source to extract names which form vertices in the graph. For every name detected by NER, a vertex is created in the graph with the criminal's name as the label. E is the set of edges representing the existence of a relationship between criminals. An edge is created between two criminal nodes if they are mentioned in the same article or post. Several other works, e.g., [9, 29], used "mentioned in the same document" to create an edge between two nodes. W denotes weights of the edges in the graph. Weights are important to show how strong is the tie between two nodes. Methods such as co-occurrence is used to get the weight between two criminals; it is the number of times two criminals have been mentioned in the same text. Modeling criminal interaction and mentions in text is important because it provides analysts with network visualization. They will see a criminal network and associated interactions in a clear way, and they will further investigate people in the network by applying network analysis techniques.

Anwar et al. [9] proposed a framework to extract criminal information using text mining from chat logs data. In their method, they benefited from chat log data collected from a criminal computer seized by a forensic investigator from a crime scene. After collecting and normalizing the chat data, their framework applies n-gram technique to extract the set of vocabulary in the logs. Then they extracted key information from these n-grams by identifying sets of key terms and key users who have a dominating role in the chat logs. As a result, they built a social graph based on the interaction of users in the chat log, where nodes in the graph are criminal names mentioned in the chat logs and edges in the graph represent whether criminals were mentioned in the same chat session. They analyzed the constructed social graph to determine communities which exist in the criminal network. For this, they applied several clustering techniques, such as k -means, hierarchical agglomerative clustering, and Markov clustering. The challenge with this framework is the assumption that access to criminals computers and their chat log data is possible.

2.4 Image Processing and Face Recognition

Face recognition has been at the center of image processing work in recent years, specifically in the field of biometrics. It involves two procedures, first *face detection* is applied on a given image to search for locations of faces in the image. The second procedure is *face identification*. Given a detected face location, face identification verifies the identity of the specific person.

2.4.1 Face Detection

The last two decades witnessed considerable work on face detection, and a significance progress has been made over the past few years. An early method created by Viola and Jones [163] provides fast and accurate face detection. The method detects a face region by using boosted cascade detectors and simple features (Haar) to determine and extract features from an image. Other face detection methods were also proposed in the literature, such as the Histogram of oriented gradient (HOG) [36], which divides an image into grid cells and then computes the feature vector of the image using the gradient descriptor. While using these methods provides fast and accurate detection rates for frontal images, they fail when it comes to multi-view faces or faces in an uncontrolled environment. This is because of the nature of Haar and gradient features which is weak in describing face features in uncontrolled environments where several challenges may rise under various illumination, pose, and expression conditions.

Many other research efforts focused on enhancing these detectors to be able to detect profile faces by creating multiple face detectors for different poses of the face, e.g., [170], [68], [79].

Due to the recent development in graphic cards, high computation is now available to create complex feature extraction techniques capable of representing objects well even in uncontrolled environments. Recent techniques for face detection make use of deep convolutional neural network as the architecture to detect and extract features of faces. The success

of using deep neural network in speech recognition and image classification shaped and encouraged research efforts on face detection. One of the first efforts to use deep convolutional neural network (DCNN) for face detection was the work done by Zhang et al. [179]. They collected many different face images from different datasets containing different poses of around 120,000 faces, and then trained a DCNN with 4 layers on this dataset. They used this trained model, however, as a post filter for a boosting based multi-view face detector which is used for face identification given an input image.

Other convolutional neural network (CNN) based methods such as the one described in [42] trained a model based on fine tuning AlexNet [90] with face datasets for face identification, and then used a sliding window to detect the region of the face. Their method recorded a receiver operating characteristic (ROC) value of around 80% on the FDDB dataset [74]. In another work, Li et al. [99] created a cascade of six CNNs for face detection. First, regions are extracted from a test image from the first CNN called 12-net using a sliding window of size equal to 12. After this, a CNN is used for alignment. The authors repeated these two procedures for 34 and 48 net CNNs. Their method recorded a ROC value of around 85%.

Girshick et al. [47] proposed a region based CNN called R-CNN. Their work is based on region proposal and object identification using CNN. Given an input image, the first step is to extract regions where each region might contain an object desired to be detected using a region proposal method. This step is not included in the CNN model. Accordingly, any region proposal method can be applied on the test image. Selective Search [162] and EdgeBox [186] methods are widely used for region proposal. Each extracted region will then be warped and fed into the trained CNN model which decides whether the region contains the desired object or not. While this method has proven to be accurate, it has a slow test time because of the need to run full forward pass of CNN for each proposed region. To tackle this problem, Girshick et al. introduced Fast R-CNN [46], where the entire test image is fed only once to CNN. First, the region proposal method is applied on the test image, then the whole image is fed to CNN where the extracted regions are mapped from the original image

to the convolutional feature maps inside CNN. While Fast R-CNN produced good accuracy like R-CNN, using a generic region proposal method, such as Selective Search and EdgeBox still consume a considerable amount of time. These general hand-crafted region proposal methods can be faster and more accurate by using deep learned features to extract regions. For this reason, Faster R-CNN has been proposed [133]. Faster R-CNN uses the same CNN structure already described, but also inserts a Region Proposal Network (RPN) after the last convolutional layer of CNN. RPN is trained to produce region proposals directly; no need for external region proposals.

Faster R-CNN is applied on the face detection problem in [76]. This paper uses a pre-trained ImageNet CNN model, VGG16 [146], to train a Faster R-CNN face detection model on WIDER face dataset [174] which contains 12,880 images and 159,424 faces in the training set. Their model reached almost 90% accuracy on FDDB dataset.

2.4.2 Face Identification

The main process in face identification is to find a representation of the face where faces of the same person have more similar representation than those of other people. Finding a representation of the face is done by a process called feature extraction where a feature vector is computed from a face image in an n-dimension vector. These feature vectors are then used to do the face identification process. Several feature extraction techniques have been developed, they can be divided into two categories, namely hand-crafted and learned features.

Hand-crafted features work by taking an input image and then follow a predefined algorithm to look for key points in the image and extract features based on the location of the key points. Famous hand-crafted feature extraction techniques such as SIFT [103], SURF [13], and LBPH [4] have been described in several papers [44], [15], [41], [27] as effective approaches to represent faces. Based on this, classification is done to verify test images. Other works such as the one described in [122] extract face features by applying Gabor wavelet feature

extraction technique followed by PCA. They then use the SVM classifier model to verify test faces. The authors in [127] compared the usage of two feature techniques on face images. The first technique uses pixel intensity features and normalization then applies blurring to the image with Gaussian filters to describe a face. The other technique is called V1-like features using Gabor wavelets. They applied these features on the LFW dataset and then applied face identification using the multi kernel learning classification technique.

Some recent methods applied face recognition in the wild by either training one model [141, 154] or multiple pose specific models [109] to learn feature extraction models. These representation learning methods make use of neural networks due to their great performance in other object recognition domains [144]. Like face detection neural network models, the model is trained on millions of face images. However, instead of identifying face features for detection, face recognition models train CNN so that different faces of the same person will output similar feature vectors. Masi et al. [109] tackles pose variation by training multiple pose-aware specific models (PAMs). They use deep convolutional neural networks to learn representations of faces at different pose values. First, they apply landmark detection on the input face to classify it into profile or frontal face. Then, they extract corresponding features at different PAMs to fuse the features. Taigman et al. from Facebook introduced a method they called DeepFace [154] which uses a neural network model to learn face representations from large training datasets (order of millions). The innovation of their method is that they implemented 3D modelling for all faces as a pre-processing step to align faces before feeding them to the neural network to learn better face representation. Schroff et al. from Google also created a face feature extraction technique based on neural networks called FaceNet [141]. Like DeepFace, their model is trained on millions of private images where faces are taken in an uncontrolled environment. The difference, however, is that FaceNet doesn't use any kind of 2D or 3D alignment, instead they make use of simple scaling and translation techniques on images. The method, however, uses a triplet loss learning technique on each learning step of the neural network so that the representation is a vector

where the Euclidean distance between the vectors is the distance between the images. Both DeepFace and FaceNet achieved state-of-the-art accuracy results on face identification on LFW dataset $\tilde{97}\%$.

Chapter 3

Combining Feature Extraction and Clustering for Better Face Recognition

In this chapter, we study the performance of face clustering approaches using different feature extraction techniques. This study will highlight best practices for handling faces of terrorists and criminals in an approach which is intended to trace and red flag potential cases. Given as input images containing faces of people, face clustering divides them into K groups/clusters with each group containing images expected to represent almost the same person. Face clustering is very important especially in forensic investigations where millions of images are available in crime scenes to be investigated. We study the performance of face clustering by first choosing different feature extraction techniques to capture information from faces. Feature extraction techniques are employed to check which face representation works better in describing faces as input to clustering algorithms. We also used Rank Order clustering algorithm which is known for its good accuracy when clustering face images along with other traditional clustering techniques. We evaluated the performance of feature extraction techniques and clustering algorithms using four datasets (JAFFE, AT&T, LFW, and YaleB);

each imposing different challenges for face clustering with varying image environment and for datasets of different sizes. These datasets challenge clustering algorithms and feature extraction techniques in run time and clustering accuracy. Experimental results show the effectiveness of Rank Order clustering in terms of accuracy for small datasets while its run time performance degrades for larger datasets. K-means performed poorly on LFW dataset. OpenFace performed the best in describing face images especially on large datasets compared to other feature extraction techniques. The latter method reported high accuracy margin is big and acceptable feature extraction time.

3.1 Introduction

Face recognition involves detecting and verifying persons' identity by processing digital images and frames extracted from videos. Face recognition systems are becoming more popular due to rapidly advancing technology which made it affordable to capture and store large number of images at low cost. They have various applications and benefits, including homeland security where video surveillance systems detect and recognize criminals or intruders. Video surveillance systems that are able to recognize people from a captured video stream are becoming more important especially with incidents related to crimes, e.g., the Boston marathon attack [86]. In such incidents, thousands of images are collected by video surveillance cameras and then inspectors analyze faces residing inside frames.

Clustering of people plays an important role during the investigation of crimes. In crowd areas a large number of persons may pass in a specific location where a video surveillance system will keep on capturing image frames which can be in the order of millions. The same person may appear hundreds of times in frames which are not necessarily all consecutive. Thus, clustering of people will be perfect to apply for the following two main reasons:

- **Filtering Data:** By excluding images where no person is detected in surveillance camera frames. These images should be discarded during the investigation. The remaining

images will be trimmed to concentrate only on persons appearing in frames, mainly their faces.

- **Organizing Data:** Here images of the same person in different location scenes will be identified to belong to the same cluster. This way, an investigator interested in tracking a specific person will only concentrate on a specific cluster and may proceed to identify and investigate other related suspects.

In order to cluster people in video frames, we use face as the identifier because a face is the most distinctive key to person's identity [24]. Clustering faces is a challenging process and dependable not only on the clustering algorithm invoked, but also on the feature extraction technique used. Both have several challenges to cope with. Feature representation challenges are inherited from limitations of visual features due to several factors, including low resolution of face images, changes in illumination between images, capturing a person from different viewpoints, cluttered background, etc. While clustering challenges may attributed to the fact that expected number of people in an input frame is not known in advance. This may cause a problem for some clustering techniques, mainly those which require number of clusters as input. Another issue is that number of images of different people is unbalanced. For instance, some people may appear in a few frames while others may exist in many frames; this aspect is challenging for some clustering techniques as discussed in Section 3.2.3.

Face recognition has recently received considerable attention as evident by the number of face recognition algorithms described in the literature. However, clustering of face images has not received enough attention yet. As a result, existing literature lacks on efforts which investigate appropriate match between feature extraction techniques and clustering algorithms. Motivated by this, the work described in this chapter evaluates the performance of various feature extraction and clustering techniques using a number of datasets of face images. By doing so, we seek to have a better idea on which feature extraction technique works better with a given clustering algorithm with respect to time and clustering accuracy.

The rest of the chapter is organized as follows. Section 3.2 describes the methodology used in face clustering along with feature extraction and clustering algorithms to be used in performance evaluation. Section 3.3 presents the datasets used in the evaluation. Section 3.4 includes the experiments and results. Section 3.5 is conclusions.



Figure 3.1: Face clustering overall methodology

3.2 Methodology

As shown in Figure 3.1, the general methodology of clustering faces consists of four main stages. The first step is to acquire a face dataset from any appropriate source which may be a video surveillance camera. The datasets we used for this purpose is mentioned in Section 3.3. After attaining the image collection, the next step is to pre-process and filter the images to concentrate only on faces of people. Then, feature extraction techniques are applied on the processed faces to get a feature vector of each image/face. The last step, is to cluster the extracted feature vectors. More detail on each step is explained below.

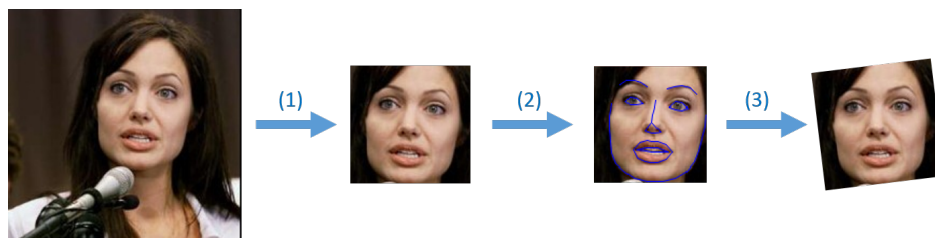


Figure 3.2: Pre-processing steps example

3.2.1 Pre-Processing

Face pre-processing of input images involves three major steps as depicted in Figure 3.2.

1. **Detect Face Region:** The first step is to apply face detection over the image in order to get the face region of a person. By applying face detection, we eliminate extra details in the image and focus only on the face of the person. We have used dlib's implementation [82] of Histograms of oriented Gradients (HOG) face detection method as described in [36]. The output of this method is a face image of size 96×96 pixels.
2. **Face Landmark Detection:** After the face region is extracted, we compute face landmarks as shown in Figure 3.2. We have used dlib's implementation of face pose estimation as presented in [80]. In their work, they have made an ensemble of regression trees to estimate landmark positions of a face from an image. They achieved high quality and fast predictions. The output from this method is 128 points that represent head pose.
3. **Face Alignment:** The last step performed is to align the head position straight with no rotation while keeping same eye, nose and mouth position for all images. This step is important so that all faces are properly aligned because a slight variation in face alignment would be enough to trigger a false positive match with another person in the dataset. A face is aligned by using landmarks detected to put the eyes, mouth and nose at similar location for every image so that the features extracted for every face will have almost same face position. This is done by doing affine transformation of faces with the help of landmarks to normalize and align faces at the same position.

3.2.2 Feature Extraction

After completing the pre-processing stage, face images become ready to extract their features. Extracting features of an image corresponds to building a feature vector which represents its important pixel information. These feature vectors are to be used in the clustering process. There are several feature extraction techniques that can be applied to the face.

Currently, the top feature extraction technique is the one based on convolutional neural networks developed by Google’s FaceNet [141]. In their work, they use up to 200 million private images of people to train a deep neural network to learn a feature vector of a face image and map it to a compact Euclidean space. Using this method, the similarity measure between two faces is simply the squared L2 distance between the two images.

In our analysis, we chose the following image feature extraction techniques. Then, we study the effect of using each of these feature extraction techniques with the clustering models.

- **SIFT** [103]: Scale-invariant feature transform (SIFT) method was developed by D.Lowe in 2004. SIFT extracts key-points of an image and then computes its descriptors. The algorithm to detect key-points involves four major steps: Scale-space extreme detection, key-point localization, orientation assignment, and key-point descriptor generation. Scale-space is found using an approximate Laplacian of Gaussian (LoG) with difference of Gaussian.
- **SURF** [13]: Speeded-Up Robust Features (SURF) method came out in 2006 as a speeded up version of the SIFT algorithm. It does its speed up by using approximation algorithms to improve every step of the sift algorithm.
- **BRISK** [98]: Binary Robust Invariant Scalable Key points (BRISK) was developed in 2011 to make feature extraction effective and faster than previous methods such as SIFT and SURF. BRISK samples patterns out of concentric rings and then applies Gaussian smoothing. Building the descriptor is done by performing intensity comparisons.
- **DAISY** [158]: this method was developed in 2010. It depends on histograms of gradients like SIFT for key-point descriptor, but also uses a Gaussian weighting and circularly symmetrical kernel.

- **KAZE** [7]: developed in 2012, this method analyzes and describes an image by operating in a nonlinear scale space. The nonlinear scale space is built efficiently by means of Additive Operator Splitting (AOS) schemes, which are stable for any step size and could be parallelized.
- **LBPH** [4]: Local Binary Patterns Histograms (LBPH) feature extraction method can be described with the following steps:
 - Extract local features from images: This is done by not considering the whole image as a high-dimensional vector, instead describe only local features of a face. Features extracted this way will have low dimensions.
 - Summarize the local structure in an image by comparing each pixel with its neighborhood.
 - Take a pixel as center and threshold its neighbors accordingly.
 - Divide the LBP image into m local regions and extract a histogram from each. The corresponding feature vector of the face is obtained by concatenating local histograms. These histograms are called Local Binary Patterns Histograms and the feature vectors of all images have the same size (size of the histogram).
- **OpenFace** [8]: The last feature extraction technique is OpenFace’s implementation of FaceNet from Google. FaceNet yields the highest accuracy reported so far, the model and the data used in training remain private. For this purpose, OpenFace target was to implement the same neural network model of FaceNet and train it with 500k images from public datasets.

All these feature extraction methods, except for LBPH and OpenFace, identify local features in an image and calculate its descriptor as its feature vector. This local features property of images would lead to feature vectors of different sizes. However, to classify faces, all images should have feature vectors of the same size. To overcome this problem, we follow

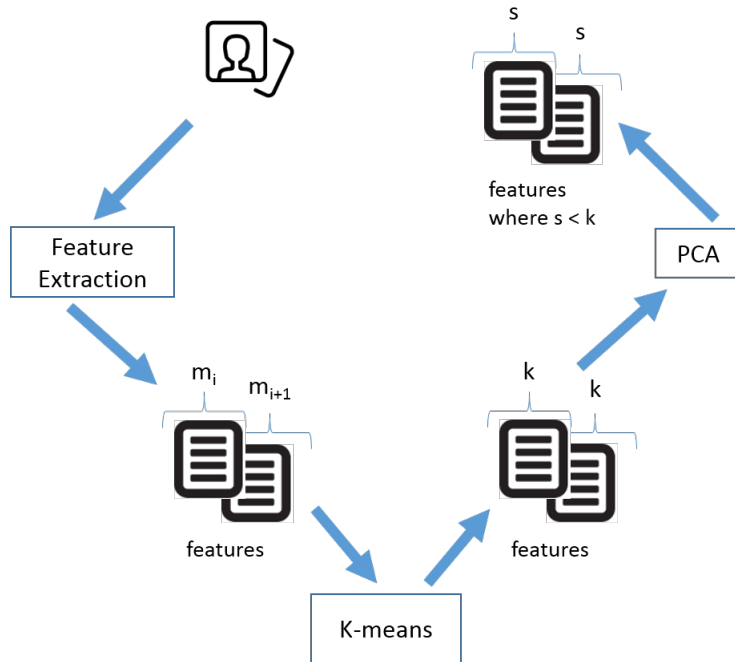


Figure 3.3: Feature extraction process

the feature extraction process proposed in [129] and highlighted in Figure 3.3. This feature extraction process has three main components.

1. **Image Feature Extraction:** The first step of the feature extraction process is to get as input face images and apply one of the general feature extraction methods described above to get corresponding feature vectors. These feature vectors may vary in size across different images.
2. **K-means Clustering:** After having the feature vectors of all face images, the next step is to map them into corresponding feature vectors of equal length. This is achieved by clustering the features using K-means to obtain K bins (where K is set to 200). As a result, every feature from the original feature vectors will be assigned a label of its cluster, in the range 1 to 200. These labels are used to build for each image a feature vector of fixed size K . This is done by iterating over original features of every image and increase the i^{th} entry of its new feature vector where i is the label of a given original feature.

3. **Principle Component Analysis:** The last step in the general feature extraction process is to reduce the dimensionality of the feature space produced based on K-means as described in step 2. Here, principal component analysis (PCA) is applied on the new feature vectors and leads to s features per image. PCA is a statistical method where given a set of feature vectors of possibly correlated variables, it converts correlated variables into a set of linearly uncorrelated variables called principal components. This will eliminate unnecessary features from the feature space and will lead to more efficient clustering of the actual face images.

3.2.3 Applying Clustering Techniques

The last step in the process is to apply clustering algorithms on the produced feature vectors such that images of each person end up in a separate cluster. We have used in this study three of the clustering algorithms described in the literature, namely **K-means** [54], **Spectral clustering** [120], and **Rank-Order clustering** [182].

K-means and Spectral clustering are the most widely used clustering algorithms. Both algorithms require specifying number of clusters as an input parameter. This requires knowing in advance the number of people who appear in the video surveillance system, which is a serious restriction in several applications, e.g., forensic investigation. Moreover, K-means suffers because the final result highly depends on the initial seeds of the clusters. This makes it difficult to handle clusters with varying density, size and shape. Spectral clustering, on the other hand, can handle non-uniform distribution of data, but its complexity is high and usually performs poorly with noisy data [182]. Noise may come from the detection of faces in the various frames and will badly affect the performance of the clustering algorithm.

Rank-Order clustering method successfully tackles the problems associated with K-means and Spectral clustering. This method checks neighborhood of a face to determine its cluster. The method defines a new distance measure based on the dissimilarity in the neighborhood structure. Zhu et al. also claimed that their algorithm can handle non-uniform data

distribution and it is robust to noise [182]. The algorithm has three major steps:

1. **Initialize Clusters:** each face image forms a cluster on its own.
2. **Candidate Merging:** compute the Rank Order distance between every pair of clusters C_i and C_j . If it is less than a threshold t then mark the two clusters as a candidate merging pair.
3. **Transitive Merge:** transitively merge all candidate merging pairs; then update the distance between clusters and loop back to the second step until no further merging is possible.

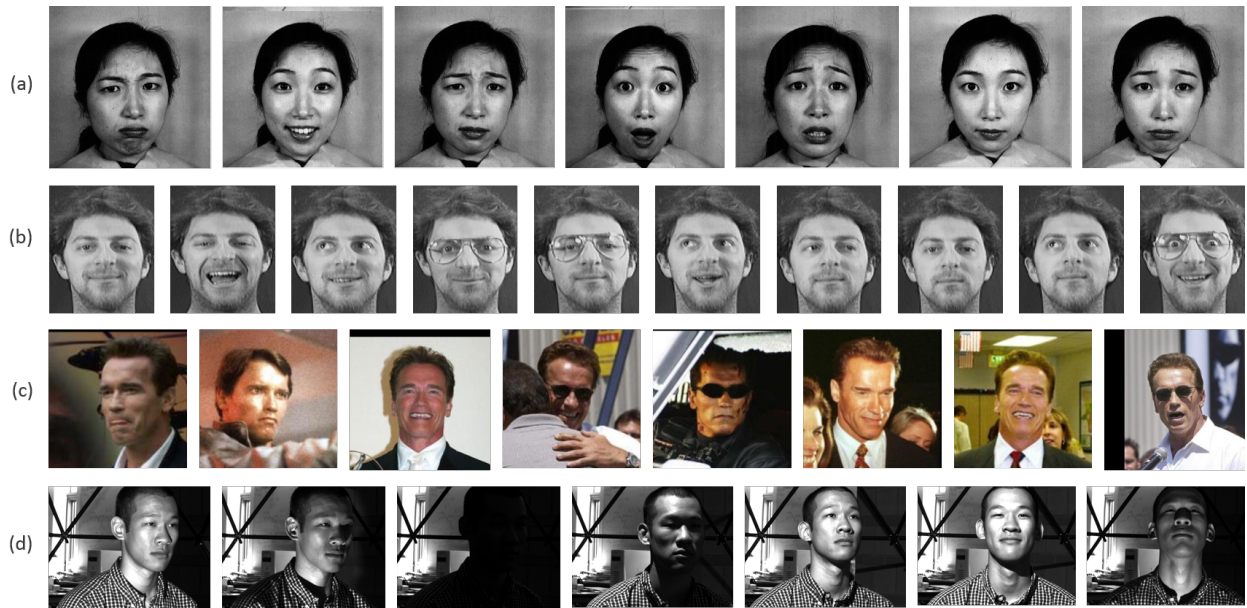


Figure 3.4: (a) presents the JAFFE dataset, as seen all the images are taken in a controlled environment with the difference being the expression shown by the female. (b) presents the AT&T dataset where the face images are also taken in a controlled lab environment with difference in facial features and expressions for each person. (c) presents the LFW dataset, this dataset has a collection for each person images from the web. As seen the images are not related to a scene and the image for a person is taken in different time frames (old/young). It presents a unique challenge to face recognition, as also another people can interfere in the image as shown in the sample. (d) presents the YaleB dataset, the dataset is taken in a controlled environment but the challenge here is with the illumination of each image different such that some images are unrecognizable even for humans.

3.3 Datasets

We have used four datasets to evaluate the feature extraction techniques and the clustering algorithms described in the previous section. Each dataset has a different set of images to cluster, and each has its own challenges exhibited for face recognition.

3.3.1 JAFFE

The Japanese Female Facial Expression (JAFFE) [106] database contains 213 images posted by 10 Japanese females. This dataset challenges face clustering by showing different facial expressions for each female in the dataset, these include happy, sad, angry, disgust, fear, surprise, and neutral. Examples from the JAFFE dataset are shown in Figure 3.4.a.

3.3.2 AT&T

The AT&T dataset [139] contains a set of faces collected at the University of Cambridge. The dataset contains 400 face images of 40 distinct persons, 10 images per person. The challenge for this dataset is to recognize people where each image was captured at different times with varying lighting scenes, different facial expressions, different facial details (glasses/no glasses) and different face alignments. An example of the dataset is shown in Figure 3.4.b.

3.3.3 LFW

The Labeled Faces in the Wild (LFW) dataset [183] contains 13,233 images of faces collected from the Web, representing 5,749 persons. As shown in Figure 3.4.c, faces in this dataset are very challenging for face recognition algorithms as they were captured in the wild with varying conditions, the size of the clusters are varying in size and density because 1,680 of the pictured persons have two or more distinct photos in the dataset, some have tens of images and others have only one image. Given these challenges, this dataset could be classified as the hardest used in this study.

3.3.4 Extended YaleB

The Extended Yale Face Database B (Yale B) [97] is the largest dataset used in this study with 16,128 gray-scale images of 28 individuals. Every person has 9 poses, where each pose has 65 images with a different facial expression or configuration. A sample of the dataset is shown in Figure 3.4.d.

3.4 Experiments & Results

In this section, we evaluate the performance of different clustering algorithms using several feature extraction methods on the four datasets mentioned in the previous section. We first present results of the pre-processing step, then we show performance of the feature extraction techniques and how this affects the clustering algorithms based on running time. Lastly, we show the performance of the clustering algorithms for every feature extraction technique based on clustering accuracy. All experiments were run on a single machine with Intel Core i5-2400 CPU @ 3.1GHz with 8GB of RAM.

3.4.1 Face Pre-Processing Results

Recall that the second step of the methodology described in Section 3.2.1 is to detect the face region in an image using HOG descriptor for face detection. Table 3.1 reports for each dataset, the original number of images the dataset has against the number of faces detected from our HOG face detector. All images from JAFFE dataset were successfully detected. We were able to apply pre-processing steps on them. This is expected with this dataset because it was captured in a controlled environment where the difference between the images is just facial expressions. However, face detection accuracy decreased drastically for AT&T and YaleB datasets, scoring 75% and 55%, respectively. Even though both datasets were captured in a controlled environment, HOG detector failed to identify faces with low illumination where their features can be hardly seen. This is why almost just 55% of YaleB

Table 3.1: Pre-processing face detection accuracy

	Original	Detected	Percentage
JAFFE	213	213	100%
AT&T	400	300	75%
YaleB	16,380	9,0371	55.17%
LFW	13,233	13,176	99.54%

dataset has been detected; almost half the images of this dataset have low illumination. As for LFW dataset which includes faces captured in an uncontrolled environment as shown in Figure 3.4, we were able to detect almost the whole dataset with 99.54% detection accuracy. Having excellent accuracy in detecting LFW dataset is very important because these images were taken in the wild and many different applications such as video surveillance systems capture images in a similar environment.

3.4.2 Feature Extraction Runtime

In this section, we report the running time results of the feature extraction process. It is very important to study the time required to extract the features and cluster the images because there are time critical application where time is essential factor in deciding which method to use clustering results are acceptable.

As mentioned in Section 3.2.2, first each of **SIFT**, **SURF**, **KAZE**, **DAISY**, and **BRISK** extracts features of a face, then K-means and PCA are applied on the result to assign equal number of features for all images. To apply **LBPH** and **OpenFace**, we just have to get features from the given image.

Table 3.2 reports the run time for extracting features by the different extraction techniques for the listed datasets. Table 3.2 includes the following columns. "Extract Feature" time is common to all techniques and refers to the total time needed to extract features. "K-means" and "PCA" reveal the time needed by the feature extraction techniques. "Total" is the total time required by the features extraction techniques, K-means and PCA. As shown in Table 3.2, as the dataset size increases, the run time for the extraction techniques

increases. From these results,

it can be seen that SURF and KAZE are the fastest methods to extract features from images, they are closely followed by LBPH, and then OpenFace. While other methods take significantly more time to extract features for datasets. The difference margin is quite clear in case of **LFW** dataset where LBPH, SURF, DAISY and OpenFace completed the process between 6 to 118 minutes, while KAZE needed almost 20 minutes, SIFT needed almost an hour and a half, and BRISK completed in around 2 and half hours. As detailed in the table, BRISK took so much time because of the extraction technique it uses, where the method needed considerable time to process a single image and get its features. While SIFT is not so slow in the extraction process, it slows down when it moves to the K-means step to produce clusters. The reason behind this is that SIFT generates a huge feature vector describing one image. So, applying K-means on all features of every image requires a lot of time.

Table 3.2: Feature extraction run time. Every table reports related to a dataset, a comparison of the run time of the three different clustering techniques used in this study. The results are shown in the format of H:MM:ss, where H is hour, M is minutes, and s is seconds. Values marked as "-" indicate that the clustering algorithm did not finish in a matter of running for 1 day.

(a)					(b)				
<i>JAFFE</i>	Extract Features	K-means	PCA	Total	<i>AT&T</i>	Extract Features	K-means	PCA	Total
SIFT	0:00:13	0:00:23	0:00:01	0:00:37	SIFT	0:00:19	0:00:55	0:00:02	0:01:15
SURF	0:00:02	0:00:01	0:00:01	0:00:05	SURF	0:00:04	0:00:02	0:00:01	0:00:07
KAZE	0:00:19	0:00:04	0:00:01	0:00:24	KAZE	0:00:26	0:00:04	0:00:02	0:00:32
DAISY	0:00:03	0:00:02	0:00:01	0:00:06	DAISY	0:00:05	0:00:03	0:00:01	0:00:09
BRISK	0:02:22	0:00:03	0:00:01	0:02:26	BRISK	0:03:19	0:00:02	0:00:01	0:03:22
LBPH	0:00:05	-	-	0:00:05	LBPH	0:00:08	-	-	0:00:08
OpenFace	0:00:09	-	-	0:00:09	OpenFace	0:00:19	-	-	0:00:19

(c)					(d)				
<i>YaleB</i>	Extract Features	K-means	PCA	Total	<i>LFW</i>	Extract Features	K-means	PCA	Total
SIFT	0:07:51	0:43:37	0:00:15	0:51:43	SIFT	0:12:37	1:13:25	0:00:22	1:26:14
SURF	0:02:16	0:00:49	0:00:15	0:03:20	SURF	0:06:10	0:01:14	0:00:20	0:07:44
KAZE	0:12:43	0:01:44	0:00:14	0:14:41	KAZE	0:17:30	0:02:27	0:00:20	0:20:17
DAISY	0:02:22	0:01:44	0:00:12	0:04:18	DAISY	0:03:30	0:02:38	0:00:17	0:06:25
BRISK	1:41:21	0:08:02	0:00:16	1:49:39	BRISK	2:29:25	0:06:41	0:00:23	2:36:29
LBPH	0:03:30	-	-	0:03:30	LBPH	0:07:04	-	-	0:07:04
OpenFace	0:07:19	-	-	0:07:19	OpenFace	0:11:52	-	-	0:11:52

3.4.3 Clustering Results

In this section, we first show a study similar to that discussed in the previous section. Here, we analyze clustering time for different feature extraction techniques and clustering tools. As mentioned earlier, the importance of a clustering algorithm should compensate between run time and clustering accuracy.

Clustering run time is reported in Table 3.3. Run time is not reported for some clustering techniques, especially on large datasets. This is because we show only results for clustering algorithms that finished in at most one day time.

For the first 3 datasets described in Section 3.4.1, namely JAFFE and AT&T, corresponding results for all the feature extraction techniques are shown in Table 3.3(a,b,c); these three dataset are the smallest in size in terms of the number of face images. Clustering run time for these three datasets is very fast; it is almost identical for K-means and Spectral clustering, taking almost a second each for all feature extraction techniques. On the other hand, the results show that Rank Order clustering is significantly slower than the other 2 clustering methods.

As reported in Table 3.3-a for the JAFFE dataset, the performance of Rank Order clustering ranges between 12 to 18 seconds for all feature extraction techniques except for the LBPH method. However, it took 49 seconds to complete the clustering for LBPH extraction technique. This is because the number of features generated by LBPH is larger than that of the others. Actually, LBPH constructs a feature histogram for every region in an image. The difference between the performance of LBPH and other methods can be clearly seen in Table 3.3.b. To finish the clustering process, the other methods needed from almost a minute, while LBPH took 2 minutes .

Concerning the two datasets, LFW and YaleB, K-means finished successfully on all the feature extraction techniques. Spectral clustering terminated successfully on YaleB dataset but failed on LFW. On the other hand, Rank Order clustering couldn't finish running for both YaleB and LFW datasets. LBPH was again the slowest taking more than 3 hours to

complete on LFW dataset, while running time of the other feature extraction techniques like k-means ranged from 1 to 2 minutes.

Table 3.3: Clustering run time. Every table reports results of a dataset, comparing run time of the three clustering techniques used in this study. The results are shown in the format of H:MM:ss where H is hour, M is minutes, and s is seconds. Values marked as "-" indicate that the clustering algorithm did not finish in a matter of running for 1 day.

(a)

<i>JAFFE</i>	SIFT	SURF	KAZE	DAISY	BRISK	LBPH	OpenFace
K-means	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01
Spectral	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01
Rank Order	0:00:17	0:00:15	0:00:12	0:00:18	0:00:17	0:00:49	0:00:12

(b)

<i>AT&T</i>	SIFT	SURF	KAZE	DAISY	BRISK	LBPH	OpenFace
K-means	0:00:01	0:00:01	0:00:01	0:00:01	0:00:01	0:00:02	0:00:01
Spectral	0:00:02	0:00:02	0:00:02	0:00:02	0:00:02	0:00:03	0:00:03
Rank Order	0:01:03	0:01:24	0:01:06	0:01:00	0:01:14	0:02:10	0:00:57

(c)

<i>YaleB</i>	SIFT	SURF	KAZE	DAISY	BRISK	LBPH	OpenFace
K-means	0:00:03	0:00:02	0:00:02	0:00:02	0:00:02	0:01:36	0:00:02
Spectral	3:13:32	3:04:31	3:03:43	4:44:03	3:03:20	5:09:30	4:11:15
Rank Order	-	-	-	-	-	-	-

(d)

<i>LFW</i>	SIFT	SURF	KAZE	DAISY	BRISK	LBPH	OpenFace
K-means	0:01:40	0:01:03	0:01:36	0:00:23	0:01:13	3:07:41	0:02:22
Spectral	-	-	-	-	-	-	-
Rank Order	-	-	-	-	-	-	-

Concerning clustering accuracy, our aim is to determine the best feature extraction technique used for face recognition and the best performing clustering technique based on the extracted features. We evaluated the accuracy of the clustering results based on the confusion matrix of the set of class labels predicted by the clustering algorithm for which true values are known from the dataset. To calculate the adjacency matrix, we use external validity indices which were designed to measure the similarity between two partitions (predicted labels vs true labels). This method's confusion matrix as described in [116], represents the count of pairs of points based on whether they belong to the same cluster or not by considering the two partitions. For each pair in the predicted partition, we check whether these

pairs have the same label or not and based on that populate the four entries in the confusion matrix, i.e., true positive, true negative, false positive and false negative counts.

After getting the confusion matrix, we calculate precision and recall by considering images in each cluster produced by the clustering algorithm and compare them with the corresponding ground truth. A given cluster C may contain some face images which are indeed members of C based on the ground truth and some other face images which should have not been included in C . Precision is the proportion of face images that were correctly classified as members of C , i.e., the number of correct faces in C divided by all faces in C . On the other hand, recall considers face images in the ground truth to determine their proportion correctly classified in C , i.e., it is the number of face images correctly classified in C divided by the number of all face images which should have been classified in C according to the ground truth. We also calculate F-measure which is a summary statistic that combines precision and recall as given in Equation 3.1.

$$F = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (3.1)$$

Clustering accuracy results are presented in Table 3.4 for all the datasets and clustering algorithms applied on the feature extraction techniques. Figure 3.5, shows how clustering accuracy of Rank Order clustering varies for different threshold values.

Table 3.4 reports accuracy results obtained by applying the different clustering algorithms on each feature extraction technique mentioned above. Values of best clustering accuracy for each clustering algorithm are shown in bold font. Table 3.4.a shows the results obtained for JAFFE dataset. The best result in this table is obtained by using Rank Order clustering with OpenFace feature extraction technique (82%). This is followed by using Spectral clustering on LBPH method (80%). We can conclude from this table that OpenFace technique has on average the highest accuracy across the three different clustering techniques. Second best performing method is LBPH, while BRISK performed on average the lowest. This conclusion is further supported in Table 3.4.b where the difference margin becomes clear as the dataset

size increases.

The best clustering results belong to OpenFace using Rank Order clustering (94.78%). In this table and all following tables, it is possible to notice that the best results for K-means, Spectral and Rank Order clustering have been obtained using OpenFace technique. Another conclusion which could be noticed from this table is that Rank Order clustering gave on average the best results compared to the other clustering algorithms.

Table 3.4.c shows clustering results for YaleB dataset which has almost 9 thousand images after pre-processing. In the table, we miss the values of Rank Order Clustering which was giving the best results in the previous table. This is because Rank Order clustering could not finish in one day. The results show a big gap when using OpenFace compared to the other feature extraction techniques, reaching 60%.

Spectral clustering with OpenFace reached 75% while the best result shown by the other methods is LBPH (8%). Table 3.4.d reports accuracy results for LFW dataset where only K-means finished in a day. OpenFace reported best results, its accuracy was just 1.41%, while accuracy for the other techniques ranges from 0.01 to 0.24%. This is because LFW dataset has imbalanced cluster sizes, and algorithms like K-means would fail when applied on this kind of datasets.

Figure 3.5 shows Rank Order clustering results in terms of precision, recall, and F-measure for JAFFE and ATT&T image sets. This figure shows that having low threshold will result in high precision almost 100%, while recall is low. And while the threshold of the clustering increases, precision starts to decrease, recall gets higher and F-measure always goes up to a certain threshold then back down with the margin of precision and recall getting high.

This can be explained as follows, for a low threshold most images will be merged together to belong to the same cluster. Having high precision means most retrieved images have the same label. Low recall means smaller percentage of images from a target cluster have been retrieved. As the threshold increases, less images will be considered to belong to the same

Table 3.4: Clustering accuracy. Every table reports results of a dataset, comparing the accuracy of the different feature extraction techniques with the three clustering techniques used in this study.

(a)

<i>JAFFE</i>	SIFT	SURF	KAZE	DAISY	BRISK	LBPH	OpenFace
K-means	78.87%	74.64%	68.3%	58.02%	55.46%	65.4%	78.46%
Spectral	73.88%	70.07%	75.6%	55.5%	50.9%	80.16%	63.28%
Rank Order	64.68% (22)	66.4% (26)	74.14% (22)	63.59% (26)	40.64% (21)	71.61% (25)	82.14% (21)

(b)

<i>AT&T</i>	SIFT	SURF	KAZE	DAISY	BRISK	LBPH	OpenFace
K-means	56.05%	35.16%	40.91%	48.1%	23.71%	37.55%	83.83%
Spectral	50.33%	34.62%	35.82%	39.65%	18.59%	35.53%	77.45%
Rank Order	52.15% (24)	41.03% (19)	44.82% (16)	49.76% (13)	26.22% (15)	57.22% (11)	94.78% (19)

(c)

<i>YaleB</i>	SIFT	SURF	KAZE	DAISY	BRISK	LBPH	OpenFace
K-means	6.63%	4.73%	5.5%	5.49%	4.72%	8.46%	65.84%
Spectral	6.66%	4.83%	5.53%	6.64%	4.29%	6.65%	75.11%
Rank Order	-	-	-	-	-	-	-

(d)

<i>LFW</i>	SIFT	SURF	KAZE	DAISY	BRISK	LBPH	OpenFace
K-means	0.16%	0.01%	0.16%	0.17%	0.05%	0.24%	1.41%
Spectral	-	-	-	-	-	-	-
Rank Order	-	-	-	-	-	-	-

cluster. Number of clusters will increase such that person B will have a cluster with some noise, i.e., recall will increase, while some of person A images will end up in other clusters (due to False Positives) leading to lower precision.

3.5 Conclusions

We have performed a study on different feature extraction techniques to determine which one is better to use in Face clustering. In addition to feature extraction techniques, we also used three clustering algorithms to see which clustering algorithm performs the best on features extracted by the feature extraction techniques.

For this purpose, we have used four datasets, each with its own challenges in the face

recognition problem and with varying sizes. We did our experiments on a single machine and noted down the results based on both run time and clustering accuracy. From the experiments and results, we concluded that the best clustering algorithm is Rank Order clustering, but due to its time complexity we could not manage to run it for large datasets. As for time complexity, K-means run time is massively better than Rank Order clustering and Spectral clustering. Spectral clustering is even faster than Rank Order clustering. Concerning the best feature extraction technique, it was OpenFace which performed the best when used with Rank Order clustering. Not only accuracy levels were great by good margins, also feature extraction time was acceptable and in a good range comparing to the other techniques.

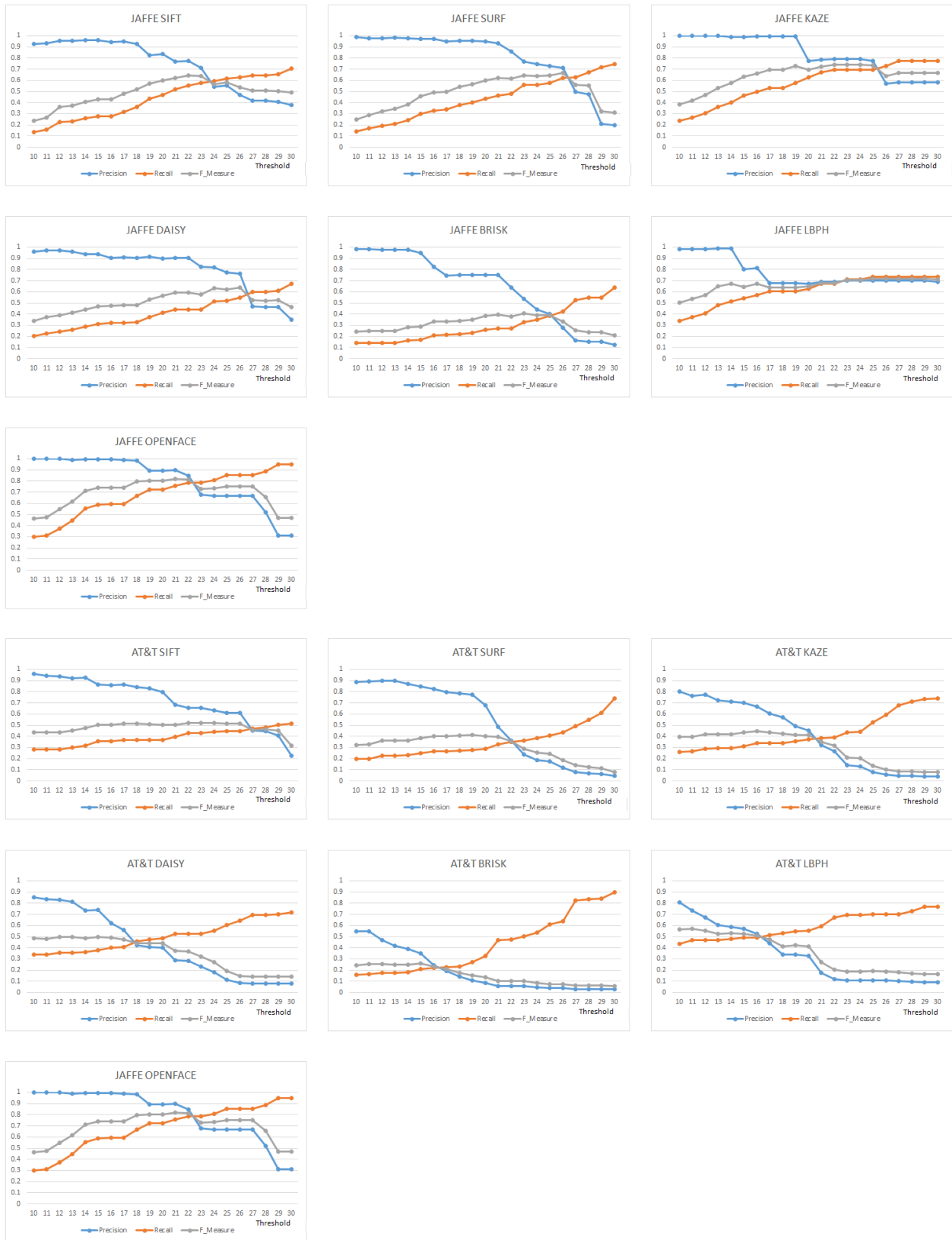


Figure 3.5: Figure of all the rankorder results ith varying thresholds

Chapter 4

Multi-view Face Detection and Recognition for Effective Identification of Social Entities

Traditional face recognition systems are designed to perform well on images captured in a controlled environment. However, images of people are generally taken in an uncontrolled environment with varying illumination, occlusion, face poses, expressions and in cluttered backgrounds. This is especially true for cases when images are captured unintentionally like in a social gathering where capturing frontal faces with good characteristics is almost impossible. Most of the current research on face recognition works on face images taken in an uncontrolled environment where most evaluation tests are performed on the LFW dataset. This dataset, however, consists mainly of frontal faces which may be rarely encountered in real life. This makes testing the performance of face recognition algorithms on profile faces insufficient. In this chapter, we explain the different state-of-the-art algorithms for face recognition including face detection and identification techniques. We then apply state-of-the-art detection and recognition on an uncontrolled environment dataset (LFW) and also on a dataset which consists of multi-view faces with varying poses (FEI). We also conducted

several experiments to test the performance of such techniques in an uncontrolled and multi-view system on two datasets (LFW and FEI). Our results show that face recognition on frontal faces in an uncontrolled and controlled environment performs well with high accuracy ($\tilde{96}\%$). The performance of face recognition on just profile faces, however, reports lower accuracy results (72%), which can be improved when comparing profile to frontal faces as well ($\tilde{80}\%$).

4.1 Introduction

Applications of face recognition span over and could satisfy the needs of a variety of fields, including homeland security, criminal identification, authentication of identity, among others. People in a social gathering may be captured occasionally and hence frontal faces will be rarely encountered in images. Another example where frontal face are rarely encountered is an environment where side views are captured for persons passing by a surveillance camera. These incomplete images may be captured adjusted and recognized to link them to specific entities who may range from famous to suspicious. Completing the phase of a person to the level that allows us to recognize him/her may lead to some serious benefits, including find a suspect and then investigating his/her behavior and network, influence, etc.

Indeed, face recognition is becoming more important with the recent advances in technology where images are analyzed at high frequencies and where millions of images are uploaded and stored daily throughout different social media platforms, surveillance cameras, etc. Due to the new advances in the development of graphic cards, they have been heavily involved in computation phases; where complex feature extraction techniques are now adopted with the use of deep convolutional neural networks. These networks are currently applied on most image processing areas, such as speech recognition, text classification, and image classification.

Applying face recognition is a challenging process for both detection and identification.

Face detection challenges include: (1) having images where faces are occluded and partially visible, (2) changes in illumination, (3) low resolution of face images, (4) changes in pose of faces, and (5) having a cluttered background. While face identification inherits the difficulties of face detection, it also contains challenges such as the lack of face images of one person to represent him/her in several conditions and having different face expressions. While there has been several studies on the area of face recognition in an unconstrained environments, the performance of face recognition where faces are in multi-view environment has not yet received considerable attention. Multi-view face images are images where a person's face is shown in several positions (frontal and profile views). In this chapter we apply state-of-the-art face detection and identification techniques to test the performance of these techniques not only in an uncontrolled environment, but also in a multi-view face scenarios. Test results reported in this chapter are promising.

The rest of this chapter is structured as follows. The methodology for face recognition is presented in Section 4.2. Section 4.3 reports results from the experiments conducted to test the performance of face recognition in uncontrolled and multi-view environments.

4.2 Methodology

The methodology we follow for face recognition is depicted in Figure 4.1. Given any image, first we apply multi-view face detection on these images to get the location of the face in terms of its coordinates x and y , in addition to its height and width, all four dimensions form a bounding box. We use Faster R-CNN model trained on WIDER dataset described in [76]. After the bounding box of a face is extracted from an image, the next step is to do face alignment. This step is necessary due to the way the CNN model for feature extraction is built. We have used OpenFace [8] method as an implementation of FaceNet CNN model.

Since OpenFace uses open source datasets to populate the model, the datasets are small in size compared to Google's training data used to develop that model which is in the order

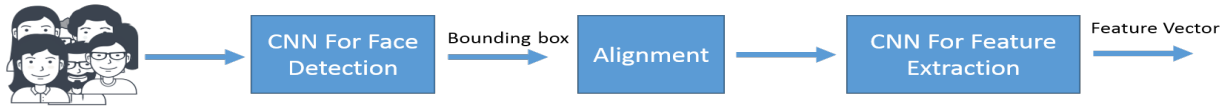


Figure 4.1: Overall face recognition methodology

of millions. Empowered by the size of the training dataset, FaceNet can handle challenges in the input dataset such as localization and face poses. In OpenFace implementation, however, they make use of a heuristic by preparing every image for training in order to have same landmark location. This is why it is important to have the alignment step just before the CNN for feature extraction. The steps for face alignment are shown in Figure 4.2. First landmark detection is applied on the face detected using the method described in [124]. After that, 2D affine transformation is performed to make the landmarks places in the same location for every image.



Figure 4.2: Face alignments steps example

The final step after face alignment is to forward the aligned faces into the CNN model for feature extraction. This process will output a feature vector of the face. The training model of FaceNet applies a CNN architecture to learn features of the face. The most important part of their method is how they train the system as a whole by introducing the triplet loss to the learning phase. After CNN extracts features of the face, the triplet loss method is applied. Triplet loss embeds a face image into a d-dimensional Euclidean space. Given an anchor image with two other images, one belonging to the same person of the anchor image and the other belongs to a different person, the learning phase will update the weights of the architecture such that the Euclidean distance between the images of the same person is less than the distance between the anchor image and the image of the different person. When

comparing two face images with each other, the squared Euclidean distance is calculated between the corresponding two feature vectors to get the distance between these two images as a representative of the similarity measure.

4.3 Experiments & Results

We tested the face detector and identification models on two datasets, namely, the Labeled Faces in the Wild (LFW) [70] and the FEI face database [157]. The LFW dataset contains more than 13,000 images of 5,749 different individuals collected from the Web. The collected faces were taken in an uncontrolled environment, under variations in pose, illumination, age, facial expression, occlusion, etc. The faces were collected using Viola-Jones face detector, i.e., most faces are frontal ones. On the other hand, the FEI face database contains 2,800 face images of 200 individuals. Each person has 14 images taken in a homogeneous background in a frontal position with profile rotation of up to about 180 degrees making this dataset good for testing the multi-view face recognition. Example of the two datasets are shown in Figure 4.3.



Figure 4.3: The first row shows sample of LFW dataset where faces of the person are taken in an uncontrolled environment in different time instances. The second row shows FEI face database where face images of the person are taken in a controlled environment with a varying rotation of up to 180 degrees

First, we run the face detector on the two datasets where the detector successfully de-

tected all the images in the datasets. After that, we take two samples of the images, one with the faces being aligned as discussed in Section 4.2 and another sample without alignment. This is because it is also interesting to note down the difference in the accuracy results if we apply alignment prior to face recognition. Then for each data sample, we extract the feature vector of the face images. For the multi-view face recognition evaluation, we split the experiment data into five sections for the FEI face database:

- *Contains only frontal faces*: to test the performance of the recognition only on frontal faces.
- *Contains only profile faces*: to test the recognition accuracy on just profile faces.
- *Contains all images*: all images of the dataset are included.
- *Contains profile faces vs all*: to compare profile faces on one side with all images in the dataset (profile and frontal), this experiment is done to check if performance of recognition on profile faces is improved when having frontal faces to test on.
- *Contains frontal faces vs all*: to compare frontal faces on one side with all images in the dataset (profile and frontal).

The evaluation is done by computing the squared Euclidean distance between every face image in the data split with every other face image in the split. Then a threshold is used such that if the distance between any two given images is less than the threshold then we predict that these two images are for the same person. Otherwise, if the distance is greater than the threshold then we predict that the two images belong to different people. If the prediction we do is true and these pair of images belong to the same person in the ground truth then the prediction is true, otherwise if the images are for different people then the prediction is false.

This evaluation method is done for the FEI face database because face images contain profile faces. As for the LFW dataset, we only consider the case where all images are

compared with each other to get the overall accuracy with and without alignment.

The evaluation is done by varying the threshold from 0 to 4 with a step size of 0.1. When the threshold value is 0, we predict that all combinations of images in the data split belong to different people, and 4 means that we predict that all pairs of images belong to the same person. By varying the threshold we can generate a set of true positives and false positives and report the ROC curve results. In a ROC curve, the true positive rate is plotted in function of the false positive rate for the mentioned different steps of threshold. The area under the ROC curve is a measure of how well the threshold can distinguish between same and different people images.

The ROC curves for different evaluation sections are shown in Figure 4.4. Each ROC curve in this figure shows one section of the evaluation for both aligned and not aligned faces. As expected, aligned faces in all experiments show better ROC value than the not aligned ones where the margin of difference between aligned and non-aligned faces is larger in frontal faces test compared to profile faces. This is due to the alignment method used which performs better for frontal faces. The first curve shows the ROC for frontal faces of the FEI dataset which has accuracy of 97% when the faces are aligned. On the other hand, when testing only profile faces, we get lower accuracy of 72% as shown in the second ROC curve. The third curve shows the result when comparing profile faces against all other faces in the FEI dataset which shows better result with ROC value of 79%. This shows that using only frontal faces shows high accuracy value, while using only profile faces to recognize a person is not as good. Moreover testing profile faces against frontal and profile faces gives a better result. The fourth curve show the ROC score when frontal faces are tested with all the other ones, the ROC value is 84%. The last two curves show when testing all faces of FEI and LFW datasets. The results show 84% ROC score for FEI while LFW shows a value of 96%. This is because the LFW dataset consists of mostly frontal faces while using profile faces in the FEI dataset makes the ROC score lower.

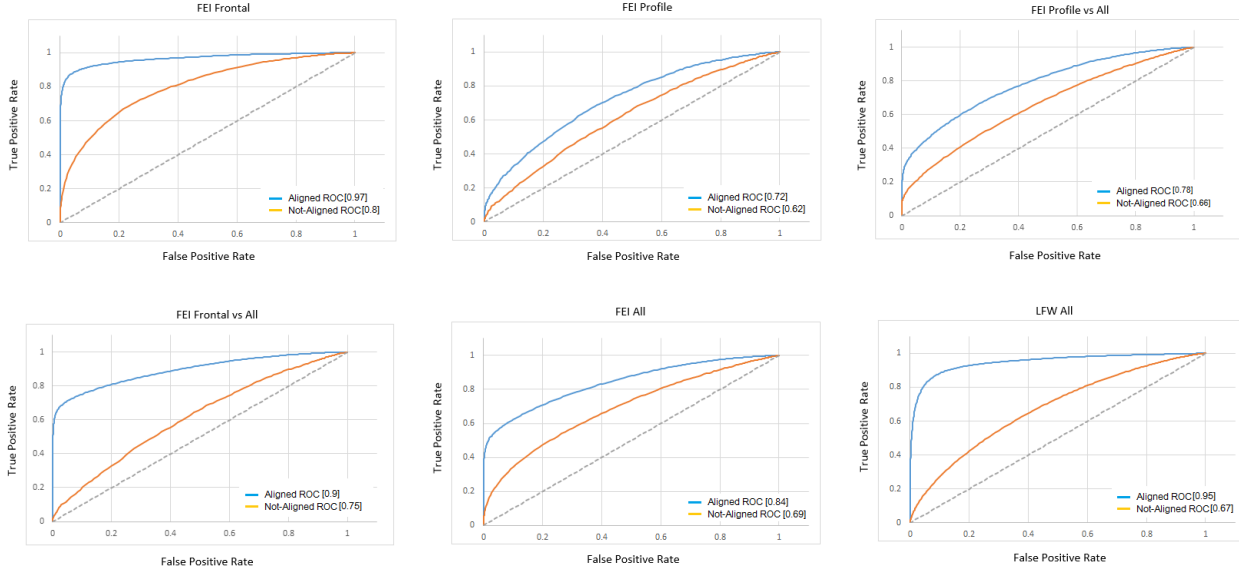


Figure 4.4: Receiver Operating Characteristic (ROC) curves of the FEI and LFW datasets as mentioned in Section 4.3

4.4 Conclusion

In this chapter, we have studied the performance of face recognition on face datasets in an uncontrolled and multi-view environments. We first discussed traditional and new techniques used to apply face detection and identification. After that we used in our face recognition methodology the face detection method discussed in [76] which trains a deep learning model using the faster R-CNN model trained on the WIDER dataset. The latter set contains thousands of face images collected under extreme cases varying scale, pose, occlusion and illumination of faces. After detecting the bounding box of the face, we apply simple affine transformation to align faces which are then fed to the feature extraction CNN model. We also used the state-of-the-art deep learning model technique described in [141]. This was trained on publicly available datasets for extracting features of face images for the purpose of person identification.

For our experiments, we used two datasets that exhibit the challenges of detecting and identifying faces in an uncontrolled and multi-view environments. For the uncontrolled face images, we have used LFW dataset which contains faces in the wild consisting mostly

of frontal faces. The results show high accuracy score on LFW dataset with alignment equal to 95%. To evaluate face recognition in a multi-view environment, we have used FEI face database which contains for each person, profile and frontal faces. We have split the evaluation experiments into five different sections as described in Section 4.3. The accuracy of testing just frontal faces outperforms the accuracy of using just profile faces. The accuracy score increases when comparing profile faces against frontal and profile ones. This shows that while frontal face recognition under difficult conditions perform well, using only profile faces to apply face recognition still needs to be improves by mixing the training data with frontal faces to achieve higher accuracy score.

Chapter 5

Face Reconstruction from Profile to Frontal Evaluation of Face Recognition

One of the main challenges in face recognition is handling extreme variation of poses which may be faced for images collected in labs and in the wild. Recognizing faces in profile view has been shown to perform poorly compared to using frontal view of faces. Indeed, previous approaches failed to capture distinct features of a profile face compared to a frontal one. Approaches to enhance face recognition on profile faces have been recently proposed following two different trends. One trend depends on training a neural network model with big multi-view face datasets to learn features of faces by handling all poses. The second trend generates a frontal face image (face reconstruction) from any given face pose and applies feature extraction and face recognition on the generated face instead of profile faces. Recent methods for face reconstruction use Generative Adversarial Networks (GAN) learning model to train two competing neural networks to generate authentic frontal view of any pose preserving person's identity. For the work described in this chapter, we trained a feature extraction neural network model to learn representation of any face pose which

is then compared with each other using Euclidean distance. We also used two recent face reconstruction techniques to generate frontal faces. We evaluated the performance of using the generated frontal faces against the posed counterparts. In the conducted experiments, we used three face datasets that contain several challenges for face recognition having faces in a variety of poses and in the wild.

5.1 Introduction

Face recognition has received significant interest in the past few decades due to its various important real world applications, including identity verification, video surveillance, monitoring, etc. Advances in face recognition resulted in continuously redefining the problem as new technologies and data become more available. Early face recognition work focused on recognizing people faces in controlled conditions where images are collected in a lab setting with defined parameters such as illumination, face rotation, pose variation, background, occlusion etc. The approaches described in [4, 161] handled face recognition by designing methods for extracting local descriptors from face images. They achieved good results for testing on face images taken in controlled environment.

Research on face recognition then evolved to identify faces in unconstrained environments. For this purpose, several benchmark data sets, such as Labeled Faces in the Wild (LFW) [70], have been proposed for this problem where previous face recognition algorithms performance dropped significantly. Current methods, e.g., [26, 141, 154, 150, 149] used recent development in deep learning, especially convolutional neural networks (CNN), to learn a network model which extracts faces. Deep learning have proved to provide very accurate results in differentiating and identifying people faces in unconstrained environment. They reported around 99% accuracy, which is better than what humans can achieve (97%) [92].

The current challenge facing face recognition is the ability to recognize faces not only in an unconstrained environment, but also with varied and extreme poses of faces. This

problem is referred to as pose-invariant face recognition (PIFR). Though LFW dataset was collected in an unconstrained environment, most faces are near frontal and don't reflect much pose variation. Applying state-of-the-art algorithms resulted in 10% performance loss when applied on frontal to profile faces as shown in [142]; human performance drops slightly. This is due to the nature of profile faces. Having more than half of a face not shown may lead to a dramatic increase in intra-person variances where different people can look the same for the recognizing algorithms.

Recently, several methods, e.g., [55, 176, 160, 71] have been proposed to tackle pose variation in face recognition by applying face frontalization, which refers to the process of synthesizing frontal face images given a face image of any pose in any environment. For the work described in this chapter, we evaluate the performance of using different face frontalization techniques and their effect on face recognition compared to training a CNN model to learn features from any pose. We first train a face recognition model based on [141]. A deep neural model is learned from faces collected from CASIA-WebFace [175] to extract features of faces. Then, we experiment whether any of the newly designed methods for face frontalization could improve the performance of face recognition compared to using profile faces in face verification. We used several challenging data sets for the conducted experiments, namely FEI [157], FERET [126] and IJB-A [84] face data sets which provide pose variance and faces collected in unconstrained environment.

The rest of this chapter is structured as follows. Section 5.2 presents related work on face recognition and face frontalization methods for face identification techniques. The methodology for face recognition using different face frontalization methods is described in Section 5.3. Section 5.4 covers the experiments conducted to test the performance of face recognition in uncontrolled and multi-view environments using three datasets on frontalized and original faces. Section 5.5 concludes the chapter.

5.2 Related Work

Recent methods proposed for pose-invariant face recognition can be classified into two main categories. One category aims at developing a model that learns pose-invariant features given a face image of any pose. The other category applies face frontalization or face rotation to get a frontal view of any pose face image which will then be used for face recognition.

Several methods attempted to apply pose aware face recognition either by training one model, e.g., [141, 150] or multiple pose specific models, e.g., [109]. Methods which use one joint model for training pose aware face recognition handle pose variations and the effect of profile faces by learning a deep neural network where face images in order of tens of millions are used for training. These are private images that the community doesn't have access to. Other methods, such as Masi et al. [109] tackle pose variation by training multiple pose-aware specific models (PAMs). They use deep convolutional neural networks to learn representations of faces for different pose values. First, they apply landmark detection on an input face to classify it into profile or frontal face. Then they extract corresponding features at different PAMs to fuse the features.

Other approaches used face frontalization techniques either by applying face rotation using 2D [69, 167] and 3D [184, 55] alignment techniques, or by learning deep neural network [176, 185]. The work described in [55] used 3D modelling by first extracting location of features on the face and then applying hard frontalization by having a 3d model of a general face and map the extracted features into a face. The method described in [184] meshes a face image into a 3D object and eliminates the pose and expression variations using an identity preserving 3D transformation. Then they apply an inpainting method based on poisson editing to fill the invisible region caused by self occlusion. The work described in [176] applies face rotation by training a deep neural network which takes a face image and a binary code encoding a target pose and generates a face image with the same identity viewed at the target pose. Experiments are within ± 45 poses of faces; they did not consider extreme poses. Frontalized faces generated from these neural network models, however, are

with no significant details in faces, making the generated faces very general. This leads to losing texture information and hence poor performance for using these faces in experiments.

The latest face frontalization methods, .e.g., [160, 71] use variations of the learning model created by Goodfellow et al. [49], called Generative Adversarial Network (GAN), which is used to generate new images by estimating a target data distribution [38]. GAN is implemented by training two neural networks (generator, discriminator) continuously in a min-max two player game fashion. The generator model learns to generate new images by mapping from a latent space to a particular data distribution. The discriminator learns to classify whether an image belongs to the original set of images or it is generated by the generator.

Generator network objective is to increase the error rate of the discriminator such that the generated images are very close to the original ones. While the discriminator network objective is to successfully classify images into either artificial (from generator) or real (original). This is why it is called adversarial learning as both are being trained to compete against each other. Several architectures of GAN have been recently proposed and successfully applied in computer vision tasks, such as image super resolution [94, 177], image synthesis [38, 131], image to image translation [72], etc.

Motivated by the success of GANs in generating realistic images, couple of works [160, 71] have been proposed to apply face frontalization using GAN network architecture. The work described in [71] proposed synthesizing frontal faces by using a two pathway GAN (TF-GAN) to process global and local features. Local features correspond to detecting four landmarks on a face (left eye, right eye, nose tip, mouth) and then aggregating the positions of these landmarks into a general face model. The global path uses an encoder-decoder structure to extract features of all the face. After that, feature map fusion is used to merge the features into one single frontal face, which is then fed to Light CNN [171] for face identification and verification. The work described in [160] proposed a disentangled representation learning GAN (DR-GAN) to perform both face frontalization, and to learn a generative representation of a non-frontal face. They proposed an encoder-decoder GAN

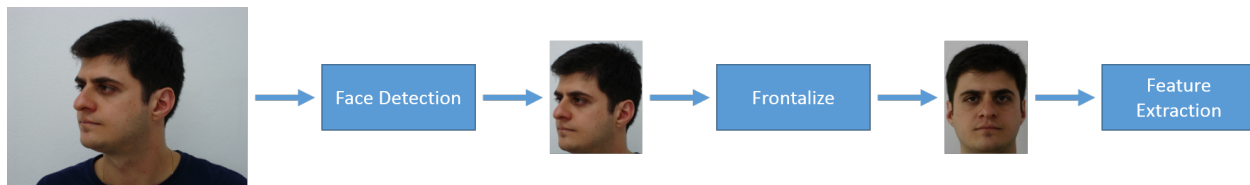


Figure 5.1: The overall methodology for face frontalization from any pose into a frontal one. First face detection is applied to get face boundaries, after that the face is frontalized to get frontal view features of the image which are then used for recognition

structure to learn face representation using the encoder. This representation is used with noise and pose degree to generate a frontalized face. They trained DR-GAN using Multi-PIE [51] and CASISA-Web Face [175]. They evaluated the results on Multi-PIE (faces angled between $\pm 60^\circ$), CFP [142] and IJB-A [84] datasets.

5.3 Methodology

The overall methodology followed for face frontalization and recognition is depicted in Figure 5.1. The methodology consists of three major components.

5.3.1 Face Detection

As our approach is intended to perform face recognition in multi-view face images, a multi-view face detector is required as the first step. We trained a CNN model to perform face detection on WIDER face dataset [174]. WIDER dataset contains thousands of face images collected under extreme cases with varying scale, pose, occlusion and illumination of faces. We used the recently proposed MobileNet-v1 [67] CNN network architecture to train a neural network on WIDER dataset. MobileNet-v1 has been designed using depth-wise separable convolutions to provide drastic decrease in model size and training/evaluation time without affecting detection performance. To get better detection accuracy, we pre-processed every image in the training data of WIDER dataset before training our detector model. We generated four different views for every image in the dataset, and then fed the images to the



Figure 5.2: Image manipulation for training: (a) an original image from WIDER dataset; (b) a rotated version of the image shown in (a); (c) - (g) versions of the image in (a) using different gamma levels, namely 0.5, 1.5, 2, 2.5, and 3, respectively.

learning model. The various image types we generated from every image are listed below. They are illustrated in Figure 5.2.

- Adjust Gamma Degree: The gamma degree controls the lightning effect in an image. For our training, we adjusted the gamma level to these four values: 0.5,1,1.5,2 and 2.5 to have variety of effects ranging from dark to light illumination. This leads to images covering different periods of a day.
- Rotate Image: We also rotated images to get different angles of faces in the training set.

By using these pre-processing steps for training, we improved the performance of the detector on WIDER dataset from 75% to 80%.

5.3.2 Face Frontalization

After detecting the face region, we applied face frontalization to get a frontal and aligned face image from any given face pose. Extracting features of the frontalized face and evaluating its performance will test how realistic the generated frontalized faces are. Different face frontalization techniques were discussed in Section 5.2. For our study described in this chapter, we have chosen DRGAN [160] and Effective Frontalization (EF) [55] techniques which are publicly available.

- DRGAN [160]: This method uses GAN learning architecture to generate frontal face images along with a feature vector extraction method. They modified the original GAN architecture by introducing a generator encoder-decoder structure, not only to learn realistic frontal face generation, but also to learn feature vector representation for faces. The target is to have closer cosine similarity between vectors of various views of the face of the same person. In our study, we use DRGAN for feature extraction and also to generate a frontal face. This generated face is then used as input to our feature extraction technique.
- Effective Frontalization [55]: This method as illustrated in Figure 5.3, rotates any given face into frontal one by first applying landmark detection to locate local features of the face (eyes, mouth, nose, chin). It then estimates a 3D shape of the face, get what parts of the face are invisible, completes the face by using symmetry of visible parts.

5.3.3 Feature Extraction

Before applying person verification which corresponds to check whether two face images belong to the same person or not, feature extraction is needed such that faces of the same person have more similar representation than those of different people. Various feature extraction techniques exist, they can be divided into two categories: hand-crafted and learned features.

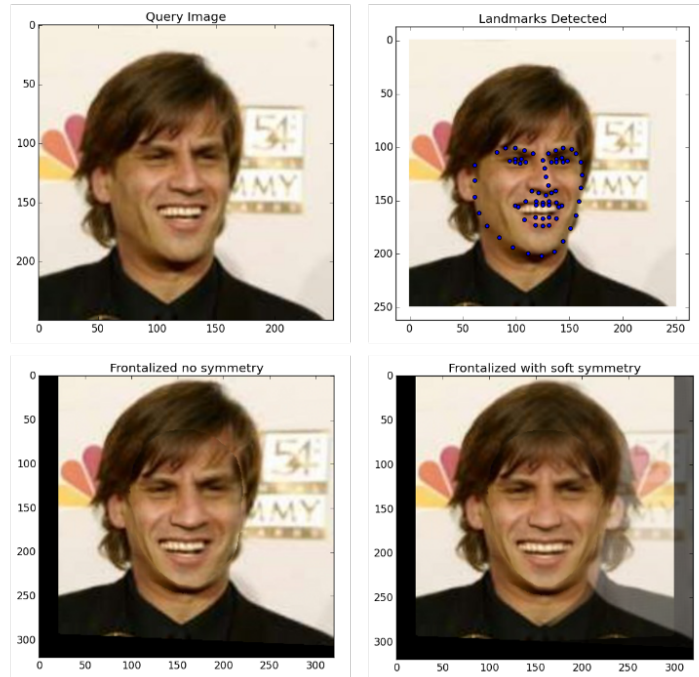


Figure 5.3: Effective frontalization rotation technique

Algorithms for hand-crafted features take an input image of a face and follow some predefined steps to locate key-points in the image. They extract features based on the location of the key-points. Hand-crafted feature extraction techniques such as SIFT [103], SURF [13], and LBPH [4] have been used by several studies, e.g., [44], [15], [41], [27].

For techniques which incorporate learned features, a model is trained based on an image dataset to automatically extract features of a certain object. Recent learned face image representations use neural network model due to its great performance in other object recognition domains [144]. Taigman et al. introduced DeepFace [154] which uses a neural network model to learn face representation from large training datasets (order of millions). The innovation of their method is that they implement 3D modelling for all faces as a pre-processing step to align faces before feeding them to the neural network to learn better face representation. Schroff et al. created a face feature extraction technique based on a neural network model called FaceNet [141]. Like DeepFace, their model is trained on million of private images where faces are taken in an uncontrolled environment. However, the difference is

that FaceNet doesn't use any kind of 2D or 3D alignment. Instead, they use simple scaling and translation techniques on the images. Further, the method uses a triplet loss learning technique on each learning step of the neural network so that the representation is a vector. Then, the Euclidean distance between the vectors is the distance between the images. Both DeepFace and FaceNet achieve a state-of-the-art accuracy of $\tilde{97}\%$ on face identification on LFW dataset.

For this work, we trained our own CNN. In particular, we trained the Inception-Resnet-v1 [152] network architecture on MS-Celeb-1M [52] face dataset. The training implementation follows the method described in FaceNet [141], and using the triple loss learning technique for our training.

5.4 Experiments & Results

In this section, we explain the conducted experiments and show evaluation results of different face frontalization techniques. We report on their effect on face recognition compared to not using any frontalization. In the rest of this section, we will list the datasets used in the evaluation, types of the conducted experiments, and the obtained results.

5.4.1 Datasets

We have used three datasets for the evaluation described below. Samples of these datasets are shown in Figure 5.4.

- FEI [157]: The FEI face dataset contains a set of face images taken between June 2005 and March 2006 in a lab environment. There are 14 images for each of 200 individuals, a total of 2,800 images. Images were collected with a homogeneous background taken in a 180 degrees covering all views of the face, The age of people in the collected images is between 19 and 40 years old with distinct appearance and hairstyle.

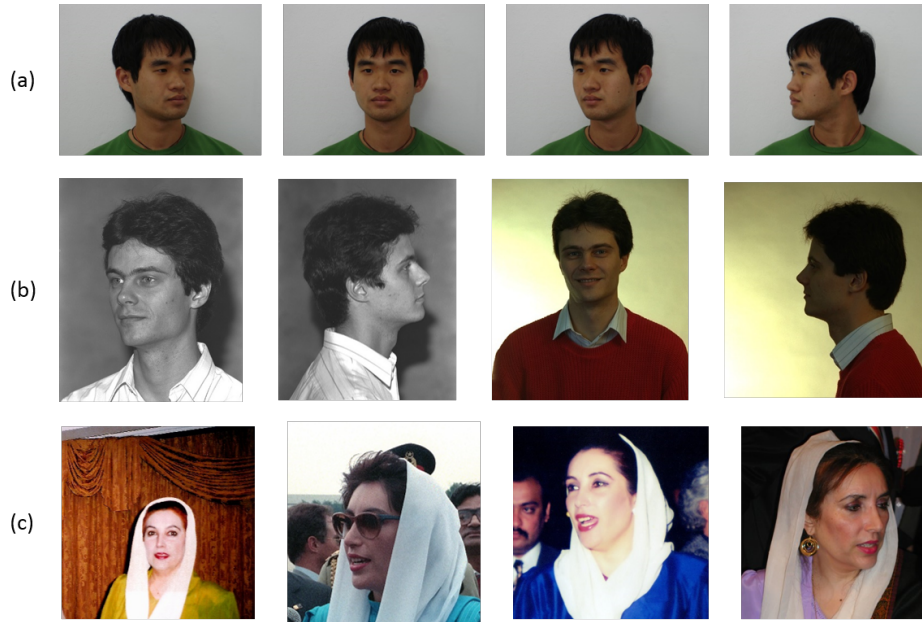


Figure 5.4: Sample of the image datasets used; variations in each dataset are illustrated. (a) a sample of FEI dataset where images are taken in a controlled environment. (b) corresponds to FERET dataset where images are taken in controlled environment and at different time sessions. (c) sample from the IJB-A dataset where faces are collected in the wild with different poses and facial appearance.

- FERET [126]: The FERET database was originally collected to provide large face database to the research community. It was collected between August 1993 and July 1996. The database contains 1,564 sets of images for a total of 14,126 images that includes 1,199 individuals and 365 duplicate sets of images. Every set is collected for a person with several images ranging from pose degree -90 to 90 degrees. A duplicate set is a second set of images of a person already in the database where images in the latter set were taken on a different day. Images of some people are taken over the span of 2 years. This ensures different facial features of the same person to be included in the experiments.
- IJB-A [84]: The IARPA Janus Benchmark-A (IJB-A) contains 25,813 images of 500 subjects collected in an uncontrolled environment. Faces in IJB-A dataset contain several poses with semi-profile faces unlike LFW [70] dataset which contains mostly frontal poses. This dataset is considered one of the most challenging face datasets since

images are collected in an uncontrolled environment (illumination, background, pose variation, etc) with varying pose of individuals and facial appearance.

5.4.2 Experiments

To evaluate the performance of the face frontalization techniques mentioned in Section 5.3.2, for every test face image we apply the following alignment steps:

- **No Alignment:** This step considers the face image output from the face detector without applying any face frontalization technique. It directly extracts features from the image. This step is important to check whether not using any face frontalization technique is actually better.
- **DRGAN:** We use the features extracted (DR-FV) from DRGAN and the frontalized image (DR-Image) to extract its features using our trained feature extraction technique.
- **Effective Frontalization:** We use two rotated images output for a given face as shown in Figure 5.3, namely symmetric (EF-Sym) and non symmetric face (EF-NonSym).

For FEI and FERET datasets, we evaluated the performance of the face frontalization technique by running the following tests.

- **Frontal Faces:** We generate a subset of the datasets using only frontal faces. We evaluated the accuracy of this subset of faces by getting one random frontal face from every person and then testing all these subset images whether they belong to the same person or not. Using this evaluation subset, the performance of the methods is expected to be high since the faces are frontal.
- **Profile Faces:** We followed the same procedure as for frontal faces described above, but this time using profile faces subset from these datasets. The performance in this subset is where most face feature extraction techniques fail and we are trying to improve.

- **Angled Faces:** In this experiment, we split faces into subsets depending on the angle of every face, such that every face with the same angle is in the same subset. We then compare all these subsets of angled faces to a set of one frontal face of every person and calculate the accuracy of every angle set. In this experiment, extreme poses are expected to perform lower than other poses.

As for the IJB-A face dataset, it provides 10 splits of face verification instances. Each split contains tens of thousands of face images pairs. Face verification is done when two images are fed to a system which should decide if these two images belong to the same person or not. From the result, we can derive the confusion matrix of the verification system to calculate the accuracy of the different frontalization techniques on each split.

A sample of original faces with their corresponding generated faces are shown in Figure 5.5. We show samples for two different people taken from the FEI dataset ranging from -90 to +90 degrees of face pose. The effective frontalization method as shown in tFigure 5.5 (Sym and NonSym faces) performs poorly for angles greater than 45 degrees as the 3D modelling fails to complete the missing half of the face. However, it performs reasonably good when most of the face is visible. DRGAN has not reported good results even for frontal faces; the output of the method is mostly a new generated face which sometimes may be very off.

5.4.3 Results

Table 5.1 reports accuracy results for frontal face subsets of FEI and FERET datasets. These accuracy results (100% and $\tilde{97}\%$, respectively) are the best for both datasets when faces are directly used (with no alignment). This is something expected for frontal faces. Using DRGAN, the feature extraction technique (DR-FV) performed second best with accuracy of 92.55% on FERET dataset. On the other hand, using the faces generated from this method reported slightly lower accuracy, 90.14%. However, by using the effective frontalization technique, accuracy levels dropped into the range of 70% for FERET dataset. Accuracy is

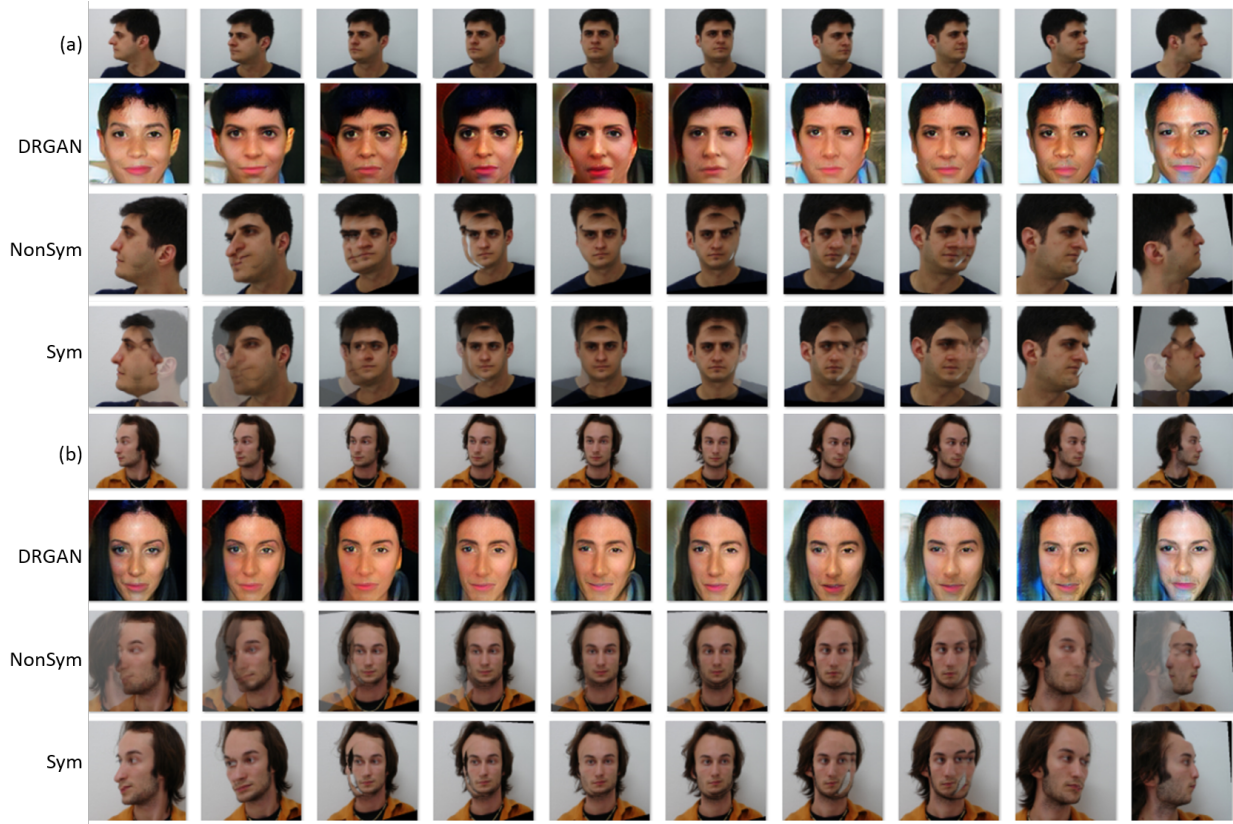


Figure 5.5: Sample of the different face frontalization techniques output using two people's set of images

higher on FEI dataset compared to FERET because the latter dataset contains face images taken at different time instances. This resulted in different facial features, and made it a harder dataset for face recognition.

Table 5.2 shows accuracy results for profile faces from FEI and FERET datasets. As expected, the performance is lower on profile faces than it is on frontal ones. The results reported in Table 5.2 demonstrate how using unaligned faces has produced high accuracy on both datasets comparable to using frontal ones ($\tilde{99}\%$). However, for frontalized faces, the performance dropped significantly to around 80% when DRGAN was used and to around 40% by using effective frontalization method on FERET dataset. From experimenting with FEI and FERET datasets, it can be easily realized that using a neural network feature extraction method on any pose image performs well and better than using frontalization techniques when trained on a large dataset. Although frontalization techniques performed

lower than the original images, DRGAN performed well enough on both datasets, around 98 and 82%, respectively.

Tables 5.3 and 5.4 shows in detail at what angles of faces does different frontalization techniques perform for FEI and FERET datasets, respectively. From the reported results, it can be easily seen that from angle 45 to -45, the results are good for DRGAN and no rotation, while FR performed poorly in general. Going to extreme poses at 90 and -90, the accuracy of no rotation is the best followed by the features extracted from DRGAN, while using frontalized faces performed very poorly.

Table 5.5 shows recognition results of IJB-A across the ten splits. Some interesting results have been reported for this dataset. Using frontalized images, DRGAN performed comparably very good ($\tilde{85}\%$) across all splits compared to using no alignment ($\tilde{88}\%$). On the other hand, feature vector extraction from DRGAN (which performed well for the previous datasets) reported very poor ($\tilde{19}\%$) in an uncontrolled environment (wild). The effective frontalization technique, however, reported on average around ($\tilde{44}\%$) for both symmetric and non symmetric faces output from the method.

Table 5.1: Frontal set accuracy

Frontal	No Alignment	DR-FV	DR-Image	EF-NonSym	EF-Sym
FEI	100	99	99	98	94.5
FERET	96.88	92.55	90.14	78.67	71.23

Table 5.2: Profile Accuracy

Frontal	No Alignment	DR-FV	DR-Image	EF-NonSym	EF-Sym
FEI	99.5	99.5	98	74.5	69
FERET	98.05	93.84	82.38	45.62	36.43

5.5 Conclusions

Interest in face recognition has evolved in recent years to tackle the problem of identifying faces in the wild. Faces in the wild are captured in an uncontrolled environment (public

Table 5.3: FEI Angular Results

Angle	No Alignment	DR-FV	DR-Image	EF-NonSym	EF-Sym
-90	82	71	43	14	8.5
-75	100	94	76.5	28	16
-45	100	98.5	91.96	42.7	45.68
-25	99.5	99	95.5	64.5	49
25	100	99.5	95.5	63.5	63
45	100	99	93.5	44	44.5
75	99	95.98	81.9	18.1	17.59
90	93.5	75.5	58.5	11.5	6.5

Table 5.4: FERET Angular Results

Angle	No Alignment	DR-FV	DR-Image	EF-NonSym	EF-Sym
-90	56.66	40.41	22.46	4.63	2.71
-45	95.39	85.12	62.05	18.22	12.46
-15	98.6	88.82	84.63	54.29	44.11
15	99.2	91.42	84.83	56.29	45.91
45	95.54	83.28	60.76	23.3	14.05
90	55.35	41.05	23.36	7.97	3.6

places) where faces can be captured in an any pose, occluded, cluttered, etc. Early face recognition methods that reported good results for faces captured under certain prespecified conditions are not directly applicable to current datasets which contain faces in uncontrolled environments and in multi-view scenarios. To satisfy this purpose, recent methods are capable of handling the multi-view face recognition problem either by training one neural network model with huge number of faces to extract representative features of every person or by applying face frontalization to get a frontal face of any given input face. A frontal face can be later fed into a feature extraction model. Challenges facing face frontalization techniques include generating a frontal face that has enough details and representation of the human face even in extreme poses where more than half of the face is not visible. Face frontalization methods either use 3D modelling of a face and try to complete a face from existing facial features or use neural networks to generate new frontal face images (GAN).

In this study, we analyzed the performance of recent face frontalization techniques compared to using profile faces as it is the case in different types of challenging datasets. The

Table 5.5: IJBA verification results

Split	No Alignment	DR-FV	DR-Image	EF-NonSym	EF-Sym
1	86.71	21.16	83.63	42.99	44.26
2	88.39	18.81	85.48	45.22	46.69
3	90.26	16.16	87.92	40.74	43.23
4	87.54	20.07	84.78	42.46	44.79
5	89.71	18.58	86.48	42.57	44.03
6	89.05	17.62	86.53	42.43	44.00
7	88.81	17.43	86.53	41.48	43.69
8	89.45	18.02	86.27	42.87	44.54
9	88.13	18.75	85.35	41.78	44.06
10	88.19	18.79	85.17	43.82	45.52

results reported in Section 5.4.3 demonstrated that using frontalized faces leads to better performance in controlled environments and with several pose variations. However, the same process performs poorly when the angle of the face is more than 75 degrees compared to the good accuracy result given while using no alignment for the face. Also, using frontalized faces from uncontrolled environment has produced poor performance. This shows that still for face recognition, training one model to extract features of faces is expected to perform better than current methods for frontalized faces. While frontalized faces can be considered as a visualization technique for analysts to better view of the identity and not for face recognition purposes.

Chapter 6

Link Prediction by Network Analysis

Link prediction refers to the process of mining and determining whether a link between two nodes in a given network may emerge in the future or it is already present but hidden in the network. Link prediction may be categorized under the class of recommendation systems, e.g., finding or predicting link/recommendation between users and items. Thus, efficient link prediction in social networks is the focus of the study described in this chapter. Finding hidden links and extracting missing information in a network will aid in identifying a set of new interactions. We developed a technique for link prediction by exposing the benefits of social network analysis tools and algorithms. We used popular network models commonly used by the research community for testing our algorithm accuracy against well-known algorithms leading to similarity measures. We also decided on using a graph database to model the network for providing better scalability and efficiency compared to storing graph information in a relational database. The experimental results reported in this chapter demonstrate how the proposed algorithm outperforms traditional link prediction algorithms described in the literature.

6.1 Introduction

In the recent surge and evolution of the world wide web, many opportunities arose for analyzing user-generated data, where the term big data became a buzz word and is now used almost everywhere, e.g., high volume of data is available at all times in the Internet. Content of the available web based data, is mostly generated from on-line social networks and e-commerce web applications, among others. Domain specific data encapsulates either homogeneous or heterogeneous actors and the links connecting them leading to a n -mode network, where n is number of heterogeneous groups of actors. For instance, data generated from social networks relates mostly to interactions between users/visitors of the networks, where people are modeled as nodes and a friendship relationship is reflected as links connecting people. On the other hand, data generated from e-commerce websites models items (clothes, food, electronics, etc.) and people as nodes to reflect items viewed and bought by people. Accomplished purchase may suggest linking people to items. This behavior of interaction, whether between people or people and items, may be modeled as a social network. A social network can be viewed as a graph where a vertex represents a person or an item, and an edge corresponds to the underlying relationship between vertexes, e.g., friendship, collaboration, among others.

One of the attractive areas for network analysis is collaborations in research where researchers mostly coauthor papers reporting their findings. Collaboration between authors may last short or long leading to a number of coauthored papers over a period of time. Thus, collaboration may be modeled as a social network, and hence can be represented as a graph $G(V, E)$ where V is set of nodes or vertexes representing authors and E is set of edges or links that exist only between researchers who have coauthored at least one paper. Building such a network or the similar, e.g., whether representing scholars or friends on Facebook or other networking sites, provides the possibility of analyzing and may be predicting or uncovering hidden links in the graph. The latter predictions may highlight a possible fruitful collaboration between potential researchers and hence would lead to a recommendation system (RS) which may bring to the attention of target researchers the

importance of initiating a collaboration.

A recommendation system is an important mechanism which assists people in exploring items of their interest by guiding them into the specific set of available items in a system's directory. This kind of systems do their recommendations and predictions based on users preferences and behavior. A separate profile is built for each user and items previously searched for or preferred by a user would help a recommender system in deciding what similar products to recommend. Recommendation systems are used in different domains and are very common in websites such as Google, Amazon and other e-commerce websites in order to recommend to their users some suggested searches or guide them in buying new items. For example, Amazon recommendation system works by getting a list of items "user A" searched for and viewed, then uses this historical information and checks what other users examined and purchased while also looking at the same set of items. After this step, the recommendation system will use the set of similar items for recommendations to the selected user. Recommendation systems are also used nowadays in social networking platforms such as Facebook and others to help in suggesting friends. This is done by predicting hidden links between actors and use some common features between users of social media. Such information may lead to new friendships between individuals in a social networking platforms such as Facebook, Twitter, etc. In a similar settings, scientists are in need of different collaboration partners, i.e., experts in a specific topic similar to their research field. Indeed, research interests, co-citation and bibliographic coupling have constitute some key metric and measure in searching for potential collaboration within a network.

Link prediction is also extensively used and important in the security domain. Since criminal activities occur in groups, finding a criminal may lead to identifying his/her whole criminal partners. Such that we can build a criminal network where nodes represent criminals and relationships represent an involvement of two criminals in an act. Performing link prediction in this kind of network will help governments, intelligence agencies and other security companies to identify criminals and unveil possible actors involved.

Motivated by the above description, the work described in this chapter tackles the issue of relating nodes in a general social network and then making appropriate suggestions by finding hidden links in the analyzed network. Completing this work will help in:

- Uncovering hidden relationships between nodes
- Categorizing and filtering the network
- Predicting links between nodes

The method described in this chapter has been tested on some benchmark networks. The reported results demonstrate the applicability and effectiveness of the proposed approach.

The rest of the chapter organized as the following. Section 6.2 reviews the most popular previous works. The proposed method is described in Section 6.3. Section 6.5 presents the conducted experiments, the evaluation process and the results. Section 6.6 is conclusions and future research directions.

6.2 Related Work

A considerable amount of research work cover recommendation systems and link prediction, and how they may be applied in different fields. For example, in [164] the authors worked on rating prediction and recommendation of items for users. They carry out the ratings prediction by treating individual user-item ratings as predictors of missing ratings. The final rating is estimated by fusing predictions from the following sources: predictions based on ratings of same item by other users, predictions based on different item ratings made by same user, and ratings predicted based on data from similar users ratings of similar items. Also in [66], the authors built an algorithm FolkRank ranking scheme which generates personalized rankings of items in a folksonomy, and recommends users, tags and resources. The basic idea is to extract communities of interest from folksonomy, which are represented by their top tags and most influential persons and resources. Once these communities are identified,

interested users can find them and participate in. This way community members can more easily get to know each other by using link prediction. Furthermore, [40] proposed two new improved approaches for link prediction: (1) CNGF algorithm based on local information network, and (3) KatzGF algorithm based on global information network.

There are also several efforts that investigate expert recommendation for business institutions, e.g., [66, 58]. [125] developed an expert recommendation system called ICARE, which recommends in an organization experts to work with. The focus of the work does not include relations between authors from their publications and citations, it rather considers organizational level of people, their availability and their reputation. [132] investigated effectiveness of a recommender system for a European industrial association in supporting their knowledge management, foregone a field study, and interviews with the employees. Experts were defined according to their collection of written documents which were automatically analyzed. Additionally, a post-integrated user profile with information about their background and job is used. Using bookmarking services of individual users in building user profiles provides further information about users interests and confirm their recommendations.

Research on link prediction can also be found in [156], where authors proposed a supervised machine learning framework for co-offence prediction. The authors build a network of criminals and offenders first, then they started to find hidden links between known criminals and potential ones by relating offenders to socially related, geographically-related, or experience related criminal cooperation opportunities. Additionally, [14] proposed a new link prediction algorithm to predict links in large-scale two-mode social networks. Based on topological attributes introduced in the chapter, the score (or likelihood) of a link between two nodes can be measured. And they defined link prediction as a two class discrimination problem. Thus, a supervised machine learning approach is applied using these attributes to learn a prediction model. Finally, they validated their results on real datasets which are DBLP bibliographical database and bipartite transaction graph of an on-line music e-commerce site. [5] developed another successful work using supervised learning for prediction; BIOBASE

and DBLP networks are used to validate the model.

Another domain of link prediction in the research domain is to recommend possible future partnership to authors who never worked together before. Using this link prediction, the system will suggest people from other domains to work on similar projects, and this may lead to a fruitful partnership to the benefit of the community. However, the focus of the research is mostly focused on homogeneous networks of authors. [21] modeled a social network of authors for recommending collaborations in academic networks. They presented two new metrics for their social network, namely institutional affiliation aspect and the geographic localization information. They analyzed how these metrics influence the resulting recommendations. [30] proposed a new way for scholar recommendations based on community detection. They used SCHOLST data set in order to build a network of authors who are clustered into communities based on their research fields. then they calculated friendship scores for each community in order to do co-author recommendation based on communities. [37] introduced two approaches for link prediction in heterogeneous networks. In the first algorithm called unsupervised multi-relational link predictor (MRLP), they extended the well-known Adamic/Adar approach. Secondly, they used their previous research based on homogeneous networks in this study by extending for heterogeneous networks. A supervised framework for high performance link prediction (HPLP) show that a supervised approach is superior to others, including MRLP. [155] proposed a methodology based on modularity analysis of heterogeneous YouTube dataset. Finally, [130] analyzed disease-gene networks.

[57] focused on author link prediction, where authors are also modeled as nodes in a social network. What makes this work interesting is the selection of links between authors where bookmarking services are included in edge identification along with author co-citation and bibliographic coupling measurements. They argued how it is important to consider bookmarking along with the other metrics for better link prediction. [151] developed a methodology to predict co-author relationship among authors in heterogeneous bibliographic network. They tested their algorithm on DBLP bibliographic network and according to their

results prediction can be improved using logistic regression-based co-authorship prediction model based on meta path-based topological features. These are the combination of different meta paths and measures.

Discovering new hidden links in a social network is not a trivial task. In [102], when recommending new friendships in a traditional social network, the number of friends in common can be used to estimate the social proximity between users ground model to smooth the rating predictions. [101] showed how to evaluate developed methodologies in order to select the best technique. For more detailed information on this topic, the reader may refer to the review reported in [100, 104].

6.3 Methodology

6.3.1 The Algorithm

In our algorithm, we used centrality measures and path information to predict new links between nodes. Eigenvector centrality points popular nodes in a network. However, unpopular nodes (with not many connected links) may be more informative and discover strong links due to rarity in real networks. Betweenness centrality shows whether a certain part of a network is centralized or not. Centralized networks have a higher betweenness value since they have controller nodes to which everyone is connected. This situation will lead to less interaction between nodes because their connections will be over central nodes. Decentralized networks can have more shortest paths and can be more flexible. Also, we are not only looking for common neighbors while predicting new links to get information. The path-passed approach can provide more information compared to locally dealing with nodes in a network. Hence, close nodes will serve more possible connections, we are only considering shortest paths between nodes, meanwhile gaining from the complexity.

$$\sum_{z \in s.paths_{u,v}} \sum_{x \in V(z)} \frac{\exp(-c_{eigen}(x)) \cdot c_{betw}(x)^{-1}}{length(z)} \quad (6.1)$$

where nodes u and v satisfy $\{u, v \in V | e_{u,v} \in E\}$ in a given network $G(V, E)$. (V) and (E) are sets of vertices and edges, respectively. z is a shortest path, denoted $s.paths$, between u and v ; $x \in V(z)$. $c_{eigen}(x)$ is eigenvector centrality of node x , and $c_{betw}(x)$ represents betweenness centrality of node x . $length(z)$ shows number of hops in path z between nodes u and v .

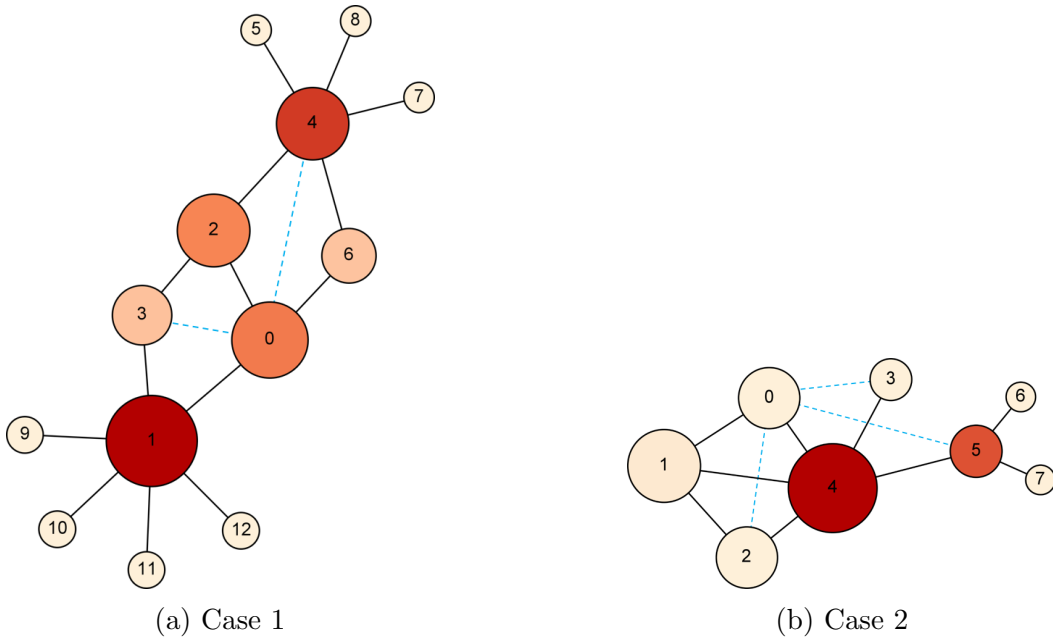


Figure 6.1: Illustrative example of our prediction algorithm on a simple network. Size of nodes represent eigenvector centrality (larger size means higher value), while color of nodes shows betweenness centrality (more reddish means higher value).

Case 1: In Figure 6.1a, shortest paths between 0 and 4 pass via nodes 2 and 6, while shortest paths from 0 to 3 pass via nodes 1 and 2. Node 2 is common for both cases. In this situation, the edge from 0 to 4 is more probable than the edge from 0 to 3 since node 6 has lower eigenvector and betweenness centralities compared to node 1. In our algorithm, we used $\exp(x)$ function to avoid negative values and higher weight for rare neighbors. Also, the difference using \exp is insignificant when the value is small, while it is larger when the

value is large. The number of hops is 1 in this case since there is no shorter path of length 2.

Case 2: In Figure 6.1b, the edge from 0 to 2 is more probable than the edges from 0 to 3 and from 0 to 5 since it has many more shortest paths than the others.

6.3.2 Graph Database

In the implementation of our algorithm, we have used a graph database to store the graph network data. Traditionally storing data in a relational database dominated due to its high performance. But recently data types have changed in the Internet more towards social networking and big data domains; this involves complex interconnected information. Thus, storing and manipulating complex data has become an issue using traditional relational databases. This motivated for the development of several database structures like graph databases. A graph database provides a method or a tool to model and store a graph related data by focusing on the relationship between entities and attributes of the nodes as basic constructs of a data model [145].

We have used Neo4j tool which is an open source graph database based on Java combining graph storage services and a high performance scalable architecture for graph based operations. In our work, we used Java libraries provided by Neo4j to create and store the datasets. We also used the graph based methods in the library in order to get the shortest paths between nodes, eigenvalue and betweenness centralities of each node.

6.4 Datasets

The best way to test our algorithm is to apply it on real-world networks to check if we can successfully predict links between real entities. Accordingly, we have applied our algorithm on six well-known real-world network data sets where the number of nodes and edges are

shown in Table 6.2:

- Zachary Karate Club [178]: is one of the most popular networks in terms of community structure. This network corresponds to members of a karate club at a US university in the 1970s; members are friends. This club has induced fights between its members such that members were split in half. This makes it a perfect real scenario network for link prediction on a member to check to which group he belongs.
- Dolphin Social Network [105]: is an undirected social network of frequent associations between 62 dolphins in a community living off Doubtful Sound.
- Les Misérables [88] :is a network corresponding for co-appearance of characters in the novel Les Misérables. It is interesting to test on this network as there are several communities to apply link prediction on them.
- Books About US Politics (orgnet.com): is a network of books about US politics sold by Amazon where edges between books represent frequent co-purchasing of books by same buyers.
- Word Adjacencies [119]: is an adjacency network of common adjectives and nouns in the novel David Copperfield by Charles Dickens.
- American College Football [48]: is a network of American football games between Division IA colleges during regular season in Fall 2000.

6.5 Experiments and Results

After collecting the data sets related to the various networks, the following steps are applied to run our algorithm which will output the confusion matrix for the evaluation code:

1. Randomly remove δ percentage of edges

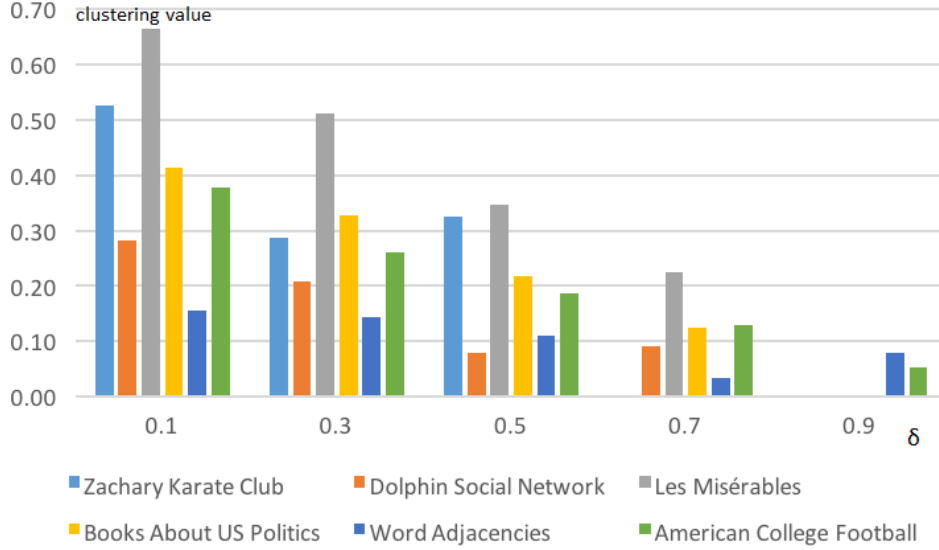


Figure 6.2: Clustering coefficient analysis of datasets.

2. Run the algorithms presented in Section 6.1 on the new network and get the corresponding confusion matrix
3. Calculate eigenvalue and betweenness centralities for all nodes in the network
4. Run our proposed algorithm
5. Select value α , which serves as a threshold for the algorithms predicted results

The results shown in Tables 6.3, 6.4, 6.5, 6.6, 6.7 and 6.8 are average results where for each network we perform Step 1 of removing edges randomly 10 times. Moreover, we chose 3 different values of δ for removing 10%, 30% and 50% of the edges. We decided on these values based on clustering coefficient analysis. This analysis may help in finding missing links between nodes, called structural holes. High degree nodes will have lower local clustering values in this analysis which means more structural holes will exist in the network so that central nodes will collect all the flow of information and reduce alternative paths. In Figure 6.2, we can see that we got more less local clustering values for 0.7 and 0.9 to the edge removal percentage. In addition, we experimentally used $\alpha = 0.1$ in all tests as threshold value for accepting an edge as predicted. We did this testing for the networks by choosing

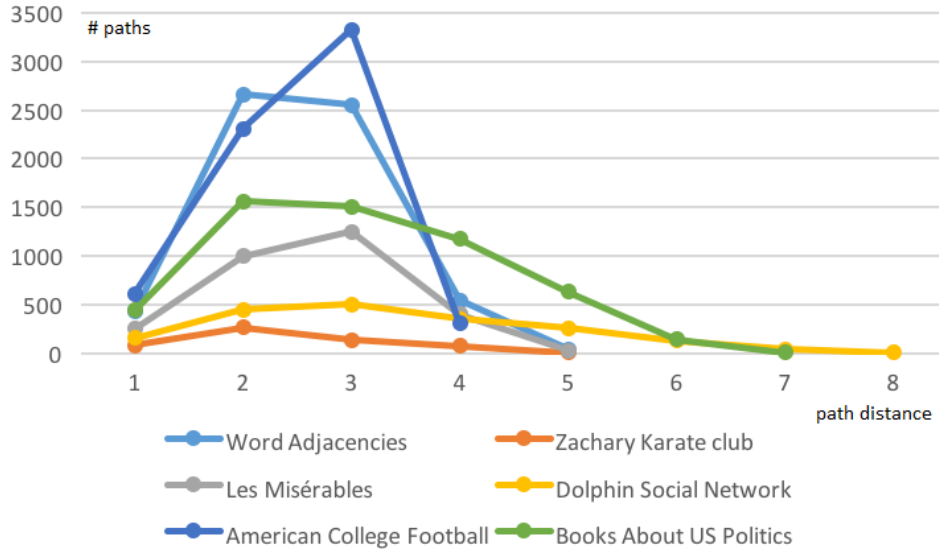


Figure 6.3: Path distance distribution of datasets.

$\ell = 5$, which is the maximum depth used in all algorithms to search for a shortest paths between two nodes. In Figure 6.3, we figured the path distance distribution of the datasets to show why we chose 5 as maximum depth. As shown for all networks, the performance of the algorithms are close to each other but our algorithm most of the times reports better values for the evaluation metrics than the other algorithms. In the various tables, *I* means $\delta = 0.1$, *II* means $\delta = 0.3$, and *III* means $\delta = 0.5$.

According to these tables, our precision and F1-score values are better than others in many cases by considerable margins. However, it is hard to decide which algorithm is better than others from accuracy results since the results are close to each other by small fraction. Also, we can clearly see from specificity results that we are not predicting non-existing links since our results are the higher compared to others. And our dataset is imbalanced which means that the number of negative examples ($FP + TN$, connections not to be predicted) is not close to the number of positive examples ($TP + FN$, connections to be predicted). Because of this, our sensitivity values are low since we have many false negatives by having more positive examples than negative examples.

To further check into how our algorithm is functioning and the advantage of using social network analysis in investigating the results of link prediction, we show in Figure 6.4 a sample

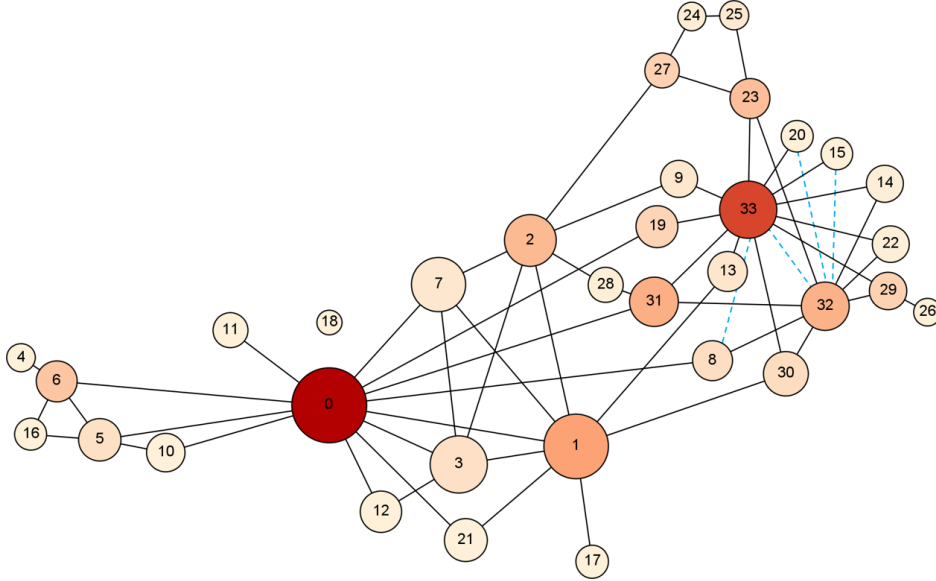


Figure 6.4: Example of our prediction algorithm on the Karate Network with 30% of the edges removed.

run of our algorithm on the Karate network with 30% of its edges removed. After removing edges, we ran the evaluation metrics we presented above to investigate the behavior of different algorithms compared to ours. In order to explain how our algorithm is predicting different than others, we used the color of the nodes to represent betweenness values on a white-red scale where white corresponds to low betweenness while red represent high betweenness. We also used size of the nodes to represent eigenvalues where the size of a node is directly proportional to its eigenvalue.

After running the algorithms on this network, we show edges which were successfully predicted by our algorithm as blue dashed lines; these edges connect nodes 32-33, 8-33, 15-32 and 20-32. All other algorithms have predicted the link between nodes 32 and 32 except Leicht-Holme-Newman algorithm. This is because these nodes have a large number of common neighbors (5) facilitating the prediction of this edge. While none of the other algorithms predicted the existence of the other edges which were successfully predicted by our algorithm, except for Katz which predicted the edge between nodes 8 and 32. This reported result is due to the fact that there is no common neighbors between nodes 8-33, 15-32 and 20-32. Thus, the other algorithms failed to predict these links. While our algorithm

successfully predicts the mentioned links because it does not only use common neighbors between two nodes but also considers the sum of all shortest paths between the two nodes.

6.6 Conclusions

In this chapter, we tackled the problem of predicting the existence of links in a graph by using network analysis. Finding hidden relationships between actors in a network has various advantages in predicting different future partnership, collaboration, etc. that based on the actors properties can help and accomplish a new trend in the research domain. It also provides the ability to unveil already existing links between people. For example, detecting series of related criminals for security reasons. By performing link prediction using social network analysis, we are able to benefit from existing graph theory algorithms that provide good analytical solutions to the problem. We used a combination of shortest path, betweenness and eigenvalue centralities for the link prediction algorithm. We showed with examples how our algorithm can perform better on real-world data sets than other link prediction algorithms which mostly focus on common neighbors for prediction.

Table 6.1: The eleven similarity metrics used in link prediction; set $\Gamma(u)$ represents neighbors of node u in the network, and $|\Gamma(u)|$ shows degree of node u .

	Algorithm	Description
Adamic/Adar	$\sum_{z \in \Gamma(u) \cap \Gamma(v)} \frac{1}{\log \Gamma(z) }$	This index measures similarity with counting of common neighbors z between nodes u and v by weighing the less-connected or rare neighbors more.
Jaccard	$\frac{ \Gamma(u) \cap \Gamma(v) }{ \Gamma(u) \cup \Gamma(v) }$	Common neighbors are divided by total number of neighbors of u and v . It looks for uniqueness in shared neighborhood.
Dice	$\frac{2 \Gamma(u) \cap \Gamma(v) }{ \Gamma(u) + \Gamma(v) }$	Common neighbors are divided by their arithmetic mean. It is a semimetric version of Jaccard.
Katz	$\sum_{\ell=1}^{\infty} \beta^{\ell} \cdot \text{paths}_{u,v}^{\ell} $	This index looks for path lengths and counts by weighting shorter paths between nodes more heavily. Parameter $\beta \in [0, 1]$ controls the contribution of paths. And ℓ represents the length between nodes. Smaller values for β will decrease the contribution of higher values for ℓ .
Common Neighbors	$ \Gamma(u) \cap \Gamma(v) $	This index measures the number of shared neighbors.
Preferential Attachment	$ \Gamma(u) \cdot \Gamma(v) $	New connections are directly correlated with high degree of neighbors.
Salton	$\frac{ \Gamma(u) \cap \Gamma(v) }{\sqrt{ \Gamma(u) \cdot \Gamma(v) }}$	Common neighbors are divided by their geometric mean.
Resource Allocation	$\sum_{z \in \Gamma(u) \cap \Gamma(v)} \frac{1}{ \Gamma(z) }$	It is so similar to Adamic/Adar. While this index takes linear form, Adamic/Adar takes \log form. But, this index is inversely more proportional to higher common neighbors.
Hub Promoted	$\frac{ \Gamma(u) \cap \Gamma(v) }{\min\{ \Gamma(u) , \Gamma(v) \}}$	Common neighbors are divided by minimum degree of neighborhood.
Hub Depressed	$\frac{ \Gamma(u) \cap \Gamma(v) }{\max\{ \Gamma(u) , \Gamma(v) \}}$	Common neighbors are divided by maximum degree of neighborhood.
Leicht-Holme-Newman	$\frac{ \Gamma(u) \cap \Gamma(v) }{ \Gamma(u) \cdot \Gamma(v) }$	Common neighbors are divided by square of their geometric mean.

Table 6.2: Data Set Networks Size

	# nodes	# edges
Zachary Karate Club	34	78
Dolphin Social Network	62	159
Les Misérables	77	254
Books About US Politics	105	441
Word Adjacencies	112	425
American College Football	115	613

Table 6.3: Karate

	Accuracy			Sensitivity			Specificity			Precision			Miss Rate			F1Score		
	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III
Ours	0.99	0.99	0.97	0.11	0.10	0.18	0.99	0.99	0.98	0.04	0.11	0.16	0.89	0.90	0.82	0.06	0.09	0.11
Adamic	0.98	0.96	0.95	0.32	0.32	0.39	0.98	0.97	0.96	0.05	0.09	0.12	0.68	0.68	0.61	0.09	0.13	0.18
Jaccard	0.92	0.94	0.96	0.53	0.37	0.35	0.92	0.94	0.96	0.02	0.05	0.11	0.48	0.63	0.65	0.03	0.08	0.17
Dice	0.90	0.93	0.95	0.81	0.53	0.40	0.90	0.93	0.96	0.02	0.06	0.12	0.19	0.47	0.60	0.04	0.10	0.18
Katz1	0.85	0.87	0.89	0.77	0.69	0.66	0.85	0.88	0.89	0.01	0.04	0.08	0.23	0.31	0.34	0.03	0.08	0.14
Katz2	0.97	0.97	0.97	0.15	0.08	0.01	0.97	0.98	0.99	0.01	0.03	0.01	0.85	0.92	0.99	0.03	0.04	0.01
Katz3	0.97	0.97	0.97	0.00	0.00	0.00	0.98	0.98	0.99	0.00	0.00	0.00	1.00	1.00	1.00	0.00	0.00	0.00
CommonNeighbours	0.91	0.93	0.95	0.74	0.53	0.40	0.91	0.93	0.96	0.02	0.06	0.12	0.26	0.47	0.60	0.04	0.10	0.18
PreferentialAttachment	0.96	0.95	0.95	0.39	0.37	0.43	0.96	0.96	0.96	0.03	0.06	0.12	0.61	0.63	0.57	0.05	0.11	0.19
SaltonIndex	0.90	0.93	0.95	0.81	0.53	0.40	0.90	0.93	0.96	0.02	0.06	0.12	0.19	0.47	0.60	0.04	0.10	0.18
ResourceAllocation	0.99	0.97	0.96	0.25	0.26	0.33	0.99	0.98	0.97	0.05	0.10	0.13	0.75	0.74	0.67	0.09	0.14	0.18
HubPromotedIndex	0.90	0.93	0.95	0.81	0.53	0.40	0.90	0.93	0.96	0.02	0.06	0.12	0.19	0.47	0.60	0.04	0.10	0.18
HubDepressedIndex	0.91	0.94	0.95	0.56	0.38	0.37	0.91	0.94	0.96	0.02	0.05	0.12	0.44	0.62	0.63	0.03	0.08	0.18
LeichtHolmeNewmanIndex	0.93	0.95	0.96	0.56	0.26	0.25	0.93	0.96	0.97	0.02	0.04	0.10	0.44	0.74	0.75	0.04	0.07	0.15

Table 6.4: AdjNoun

	Accuracy			Sensitivity			Specificity			Precision			Miss Rate			F1Score		
	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III
Ours	1.00	1.00	0.99	0.05	0.09	0.09	1.00	1.00	1.00	0.01	0.04	0.05	0.95	0.91	0.91	0.01	0.06	0.06
Adamic	0.99	0.99	0.99	0.16	0.38	0.19	1.00	0.99	0.99	0.02	0.04	0.08	0.84	0.62	0.81	0.03	0.08	0.11
Jaccard	0.99	0.99	0.99	0.23	0.17	0.15	0.99	0.99	0.99	0.01	0.03	0.04	0.77	0.83	0.85	0.02	0.04	0.07
Dice	0.97	0.98	0.99	0.58	0.40	0.32	0.97	0.98	0.99	0.01	0.03	0.06	0.42	0.60	0.68	0.02	0.06	0.10
Katz1	0.98	0.98	0.98	0.60	0.66	0.54	0.98	0.98	0.98	0.01	0.04	0.06	0.40	0.34	0.46	0.02	0.08	0.10
Katz2	0.99	0.99	0.99	0.21	0.21	0.05	0.99	0.99	1.00	0.01	0.05	0.04	0.79	0.79	0.95	0.03	0.08	0.05
Katz3	1.00	1.00	1.00	0.00	0.00	0.00	1.00	1.00	1.00	0.00	0.00	0.00	1.00	1.00	1.00	0.00	0.00	0.00
CommonNeighbours	0.99	0.98	0.98	0.44	0.52	0.38	0.99	0.98	0.99	0.02	0.04	0.06	0.56	0.48	0.62	0.04	0.07	0.10
PreferentialAttachment	1.00	0.99	1.00	0.26	0.38	0.17	1.00	0.99	1.00	0.03	0.09	0.14	0.74	0.63	0.83	0.05	0.15	0.15
SaltonIndex	0.97	0.98	0.99	0.70	0.41	0.36	0.97	0.98	0.99	0.01	0.03	0.06	0.30	0.59	0.64	0.02	0.06	0.10
ResourceAllocation	1.00	0.99	0.99	0.12	0.31	0.15	1.00	0.99	1.00	0.02	0.05	0.09	0.88	0.69	0.85	0.03	0.09	0.11
HubPromotedIndex	0.97	0.98	0.98	0.70	0.49	0.38	0.97	0.98	0.99	0.01	0.03	0.06	0.30	0.51	0.62	0.02	0.07	0.10
HubDepressedIndex	0.98	0.99	0.99	0.40	0.27	0.23	0.98	0.99	0.99	0.01	0.03	0.05	0.60	0.73	0.77	0.02	0.05	0.08
LeichtHolmeNewmanIndex	1.00	1.00	0.99	0.00	0.01	0.05	1.00	1.00	1.00	0.00	0.01	0.03	1.00	0.99	0.95	0.00	0.01	0.04

Table 6.5: Dolphins

	Accuracy			Sensitivity			Specificity			Precision			Miss Rate			F1Score		
	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III
Ours	1.00	0.99	0.99	0.00	0.08	0.05	1.00	0.99	0.99	0.00	0.05	0.06	1.00	0.92	0.95	0.00	0.06	0.05
Adamic	0.96	0.97	0.98	0.63	0.58	0.31	0.96	0.97	0.98	0.02	0.07	0.10	0.38	0.42	0.69	0.04	0.13	0.15
Jaccard	0.97	0.97	0.98	0.50	0.50	0.29	0.97	0.98	0.99	0.02	0.08	0.11	0.50	0.50	0.71	0.04	0.13	0.16
Dice	0.96	0.97	0.98	0.63	0.58	0.31	0.96	0.97	0.98	0.02	0.07	0.10	0.38	0.42	0.69	0.04	0.13	0.15
Katz1	0.94	0.94	0.96	0.63	0.77	0.38	0.94	0.94	0.97	0.01	0.05	0.07	0.38	0.23	0.63	0.03	0.09	0.11
Katz2	0.98	0.99	0.99	0.13	0.10	0.03	0.99	0.99	0.99	0.01	0.04	0.02	0.88	0.90	0.98	0.02	0.06	0.02
Katz3	0.99	0.99	0.99	0.00	0.00	0.00	0.99	0.99	0.99	0.00	0.00	0.00	1.00	1.00	1.00	0.00	0.00	0.00
CommonNeighbours	0.96	0.97	0.98	0.63	0.58	0.31	0.96	0.97	0.98	0.02	0.07	0.10	0.38	0.42	0.69	0.04	0.13	0.15
PreferentialAttachment	0.91	0.93	0.96	0.75	0.69	0.34	0.91	0.93	0.96	0.01	0.04	0.05	0.25	0.31	0.66	0.02	0.07	0.09
SaltonIndex	0.96	0.97	0.98	0.63	0.58	0.31	0.96	0.97	0.98	0.02	0.07	0.10	0.38	0.42	0.69	0.04	0.13	0.15
ResourceAllocation	0.97	0.97	0.98	0.56	0.48	0.31	0.97	0.97	0.98	0.03	0.07	0.10	0.44	0.52	0.69	0.05	0.12	0.15
HubPromotedIndex	0.96	0.97	0.98	0.63	0.58	0.31	0.96	0.97	0.98	0.02	0.07	0.10	0.38	0.42	0.69	0.04	0.13	0.15
HubDepressedIndex	0.96	0.97	0.98	0.56	0.58	0.31	0.96	0.97	0.98	0.02	0.08	0.10	0.44	0.42	0.69	0.04	0.14	0.15
LeichtHolmeNewmanIndex	0.99	0.99	0.99	0.06	0.23	0.13	0.99	0.99	0.99	0.01	0.10	0.11	0.94	0.77	0.88	0.02	0.14	0.12

Table 6.6: Football

	Accuracy			Sensitivity			Specificity			Precision			Miss Rate			F1Score		
	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III
Ours	0.99	1.00	1.00	0.29	0.14	0.11	0.99	1.00	1.00	0.02	0.04	0.09	0.71	0.86	0.89	0.03	0.09	0.11
Adamic	0.99	0.99	0.99	0.82	0.75	0.57	0.99	0.99	0.99	0.02	0.07	0.13	0.18	0.25	0.43	0.04	0.13	0.18
Jaccard	0.99	0.99	0.99	0.72	0.74	0.57	0.99	0.99	0.99	0.04	0.09	0.14	0.28	0.26	0.43	0.08	0.08	0.17
Dice	0.99	0.99	0.99	0.82	0.75	0.57	0.99	0.99	0.99	0.02	0.07	0.13	0.18	0.25	0.43	0.04	0.10	0.18
Katz1	0.97	0.97	0.98	0.99	0.91	0.76	0.97	0.97	0.98	0.01	0.03	0.07	0.01	0.09	0.24	0.02	0.08	0.14
Katz2	1.00	1.00	1.00	0.67	0.41	0.11	1.00	1.00	1.00	0.04	0.11	0.08	0.33	0.59	0.89	0.08	0.04	0.01
Katz3	1.00	1.00	1.00	0.00	0.00	0.00	1.00	1.00	1.00	0.00	0.00	0.00	1.00	1.00	1.00	0.00	0.00	0.00
CommonNeighbours	0.99	0.99	0.99	0.82	0.75	0.57	0.99	0.99	0.99	0.02	0.07	0.13	0.18	0.25	0.43	0.04	0.10	0.18
PreferentialAttachment	0.96	0.97	0.97	1.00	0.99	0.92	0.96	0.97	0.97	0.01	0.03	0.04	0.00	0.01	0.08	0.02	0.11	0.19
SaltonIndex	0.99	0.99	0.99	0.82	0.75	0.57	0.99	0.99	0.99	0.02	0.07	0.13	0.18	0.25	0.43	0.04	0.10	0.18
ResourceAllocation	0.99	0.99	0.99	0.81	0.73	0.57	0.99	0.99	0.99	0.02	0.07	0.13	0.19	0.27	0.43	0.04	0.14	0.18
HubPromotedIndex	0.99	0.99	0.99	0.82	0.75	0.57	0.99	0.99	0.99	0.02	0.07	0.13	0.18	0.25	0.43	0.04	0.10	0.18
HubDepressedIndex	0.99	0.99	0.99	0.82	0.75	0.57	0.99	0.99	0.99	0.02	0.07	0.13	0.18	0.25	0.43	0.04	0.08	0.18
LeichtHolmeNewmanIndex	0.99	0.99	0.99	0.82	0.74	0.53	0.99	0.99	1.00	0.02	0.08	0.18	0.18	0.26	0.47	0.05	0.07	0.15

Table 6.7: Les Misérables

	Accuracy			Sensitivity			Specificity			Precision			Miss Rate			F1Score		
	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III
Ours	1.00	1.00	0.99	0.44	0.05	0.27	1.00	1.00	1.00	0.18	0.44	0.20	0.56	0.95	0.73	0.26	0.09	0.23
Adamic	0.99	0.99	0.99	0.80	0.61	0.60	0.99	0.99	0.99	0.05	0.13	0.20	0.20	0.39	0.40	0.09	0.21	0.30
Jaccard	0.98	0.99	0.99	0.76	0.64	0.47	0.98	0.99	0.99	0.03	0.10	0.15	0.24	0.36	0.53	0.06	0.17	0.23
Dice	0.97	0.98	0.98	0.80	0.72	0.58	0.97	0.98	0.99	0.02	0.07	0.14	0.20	0.28	0.42	0.04	0.13	0.23
Katz1	0.97	0.98	0.98	0.60	0.38	0.50	0.97	0.98	0.99	0.02	0.05	0.12	0.40	0.62	0.50	0.03	0.09	0.19
Katz2	0.99	0.99	0.99	0.60	0.39	0.24	0.99	0.99	0.99	0.04	0.10	0.16	0.40	0.61	0.76	0.08	0.16	0.19
Katz3	0.99	0.99	0.99	0.00	0.00	0.00	0.99	0.99	1.00	0.00	0.00	0.00	1.00	1.00	1.00	0.00	0.00	0.00
CommonNeighbours	0.99	0.99	0.98	0.76	0.55	0.61	0.99	0.99	0.99	0.05	0.13	0.14	0.24	0.45	0.39	0.09	0.21	0.23
PreferentialAttachment	0.98	0.99	0.98	0.52	0.34	0.49	0.98	0.99	0.98	0.02	0.06	0.11	0.48	0.66	0.51	0.05	0.10	0.18
SaltonIndex	0.97	0.98	0.98	0.84	0.78	0.60	0.97	0.98	0.99	0.02	0.08	0.14	0.16	0.22	0.40	0.04	0.14	0.23
ResourceAllocation	0.99	0.99	0.99	0.72	0.58	0.54	0.99	0.99	0.99	0.06	0.15	0.22	0.28	0.42	0.46	0.11	0.24	0.32
HubPromotedIndex	0.97	0.98	0.98	0.88	0.79	0.60	0.97	0.98	0.99	0.02	0.08	0.14	0.12	0.21	0.40	0.04	0.14	0.23
HubDepressedIndex	0.98	0.98	0.98	0.76	0.64	0.56	0.98	0.98	0.99	0.03	0.08	0.14	0.24	0.36	0.44	0.06	0.15	0.23
LeichtHolmeNewmanIndex	0.99	0.99	0.99	0.20	0.28	0.20	0.99	1.00	0.99	0.03	0.12	0.13	0.80	0.72	0.80	0.05	0.17	0.16

Table 6.8: Polbooks

	Accuracy			Sensitivity			Specificity			Precision			Miss Rate			F1Score		
	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III	I	II	III
Ours	1.00	1.00	1.00	0.14	0.05	0.07	1.00	1.00	1.00	0.09	0.19	0.11	0.86	0.95	0.93	0.11	0.08	0.09
Adamic	0.99	0.99	0.99	0.77	0.58	0.55	0.99	0.99	0.99	0.03	0.11	0.13	0.23	0.42	0.45	0.06	0.19	0.20
Jaccard	0.98	0.99	0.99	0.84	0.62	0.33	0.98	0.99	0.99	0.02	0.09	0.11	0.16	0.38	0.67	0.05	0.15	0.17
Dice	0.98	0.99	0.99	0.98	0.80	0.55	0.98	0.99	0.99	0.02	0.08	0.13	0.02	0.20	0.45	0.05	0.14	0.20
Katz1	0.98	0.98	0.99	0.77	0.72	0.62	0.98	0.98	0.99	0.02	0.06	0.10	0.23	0.28	0.38	0.04	0.11	0.17
Katz2	0.99	0.99	1.00	0.55	0.36	0.10	0.99	0.99	1.00	0.03	0.09	0.07	0.45	0.64	0.90	0.06	0.14	0.09
Katz3	1.00	1.00	1.00	0.00	0.00	0.00	1.00	1.00	1.00	0.00	0.00	0.00	1.00	1.00	1.00	0.00	0.00	0.00
CommonNeighbours	0.99	0.99	0.99	0.75	0.84	0.55	0.99	0.99	0.99	0.03	0.08	0.13	0.25	0.16	0.45	0.06	0.14	0.20
PreferentialAttachment	0.98	0.98	0.98	0.59	0.62	0.54	0.98	0.98	0.98	0.01	0.04	0.07	0.41	0.38	0.46	0.03	0.08	0.12
SaltonIndex	0.98	0.99	0.99	0.98	0.83	0.55	0.98	0.99	0.99	0.02	0.08	0.13	0.02	0.17	0.45	0.05	0.14	0.20
ResourceAllocation	0.99	1.00	0.99	0.73	0.47	0.42	0.99	1.00	0.99	0.04	0.14	0.15	0.27	0.53	0.58	0.08	0.21	0.22
HubPromotedIndex	0.98	0.99	0.99	0.98	0.84	0.55	0.98	0.99	0.99	0.02	0.08	0.13	0.02	0.16	0.45	0.05	0.14	0.20
HubDepressedIndex	0.98	0.99	0.99	0.84	0.71	0.40	0.98	0.99	0.99	0.02	0.08	0.11	0.16	0.29	0.60	0.04	0.14	0.18
LeichtHolmeNewmanIndex	0.99	0.99	1.00	0.86	0.31	0.09	0.99	1.00	1.00	0.03	0.08	0.10	0.14	0.69	0.91	0.06	0.13	0.09

Chapter 7

NetDriller Version 2: A Powerful Social Network Analysis Tool

Social network analysis has gained considerable attention since Web 2.0 emerged and provided the ground for two-ways interaction platforms. The immediate outcome is the availability of raw datasets which reflect social interactions between various entities. Indeed, social networking platforms and other communication devices are producing huge amounts of data which form valuable sources for knowledge discovery. Hence the need for automated tools like NetDriller capable of successfully maximizing the benefit from networked data. Most datasets which reflect kind of many to many relationship can be represented as a network which is a graph consisting of actors having relationships among each other. Many tools exist for network analysis inspired to extract knowledge from a constructed network. However, most of these tools require users to prepare as input a dataset that inspires the complete network which is then displayed and analyzed by the tool using the measures supported. A different perspective has been employed to develop NetDriller as a network construction and analysis tool which does some tasks beyond what is normally available in existing tools. NetDriller covers the lack that exists in other tools by constructing a network from raw data using data mining techniques. In this chapter, we describe the second version of NetDriller

which has been recently improved by adding new functions for a richer and more effective network construction and analysis. This keeps the tool up to date and with high potential to handle the huge volume of networks and the different types of raw data available for analysis.

Before diving into the details of this chapter, I would like to thank all students who participated in the development of NetDriller in various capacities. Without their efforts I would not have been able to expand and heavily use NetDriller as part of my research. I especially thank previous PhD and MSc students Keivan Kianmehr, Negar Kockookzadeh and Atieh Sarraf who in addition to their major contribution to the first version of NetDriller, supervised various groups of undergraduate and graduate students who developed various components of NetDriller as part of their course work in CPSC 572/672.

7.1 Introduction

In the first version of NetDriller as described in [89], we presented a social network analysis tool that helps researchers and analysts in creating and analyzing social networks. A social network is a graph where nodes represent actors and edges reflect and inspire a relationship between these actors. With the help of NetDriller, many raw data types such as transactions, scholar, social media, organization structure data, etc. can be converted into social networks and properly analyzed.

The actors in a social media can be homogeneous where all nodes correspond to same entity (people, organizations, roles, etc.) This type of a network is called one-mode social network, while having two entities in the same graph is called two-mode network [165]. The edges in a social network reflect the type of relationship the actors have among each other. In one mode networks, the edges may represent friendship for social media data while edges may represent a works-on relationship in two mode networks where actors form two disjoint sets like employees and projects.

Whether it is a one or two mode network, NetDriller can help an analyst in constructing

the social network from the raw data by using two data mining techniques namely clustering and association rule mining. After constructing the social network, analysis measures are applied on the network to perform knowledge discovery in order to identify key characteristics of this network. Social network analysis (SNA) measures are mostly inspired from graph theory, statistics and linear algebra. These measures are used to study a network at node and network levels, identify communities among nodes, predict what links can exist in the future state of the network, etc.

NetDriller has been developed by our research group in 2009 to cover the social network development and analysis that lacks in most other tools. The project has been under constant updates and new functionalities are always added to the tool to cover the needed areas in the social network analysis domain. We make use of our expertise in machine learning and data mining to provide additional capabilities in network construction. The project has been developed in Java so that it can be platform independent to run on any operating system. The system makes use of the Jung library ¹ for network visualization and Weka tool [53] to apply data mining techniques.

The rest of this chapter is organized as follows. Section 7.3 shows the user interface and the basic social network analysis measures covered in NetDriller. In Section 7.4, we present the new functions that we added to NetDriller 2.0 making it a better social network analysis tool. Section 7.5 is conclusions.

7.2 Brief Overview of Existing Similar Tools

Several tools have been developed for social network analysis both for commercial and research purposes, e.g., [143]. In this section, we briefly cover some of the well known SNA tools. NetMiner ² is a commercial software which visualizes and analyzes networks to extract information from a given network. The tool allows the imports of big networks with

¹Jung: <http://jung.sourceforge.net/> Accessed on 07/09/2017

²NetMiner <http://www.netminer.com> Accessed on 03/10/2017

3D visualizations. Data mining is integrated in the analysis to classify, cluster and apply recommendation for the nodes in the network. Node centrality measures are also used to determine their importance.

Ucinet [20] is a commercial social network analysis tool that treats networks as matrices for data imports and for applying matrix algebra. Analysis techniques such as centrality measures, groups identification and graph theory techniques are provided in the tool.

Gephi [12] is a popular social network analysis tool. It is an open source tool and provides network visualization using several layouts and SNA techniques. The tool offers the common metrics to calculate centrality measures, community detection and shortest paths. The tool also provides the functionality to view the network changes over time.

NodeXL [147] is a free, open-source SNA tool that can be added as a template for Microsoft Excel. This tool is interesting as people who are used to excel can get their data into matrix format in excel and then apply SNA measures on their network using excel. NodeXL provides features to calculate graph metrics, import networks from social media, zoom and scale the network, and dynamic filtering of nodes.

7.3 The User Interface

NetDriller provides a simple user friendly interface that helps any user in importing and generating social networks for further analysis. The program can run on a standard personal computer making it available for any user. With the integration of data mining, any raw dataset can be converted into a social network using the tool.

Figure 7.1 shows the main interface of NetDriller once the application is launched. The "Data" menu is enabled where users can import a network or a raw dataset. If the user imports a network, then all the analysis tools will be enabled. Importing a raw dataset however, will enable the "Network Construction" menu which converts the raw data into a social network using either clustering or association rule mining. We have added two new

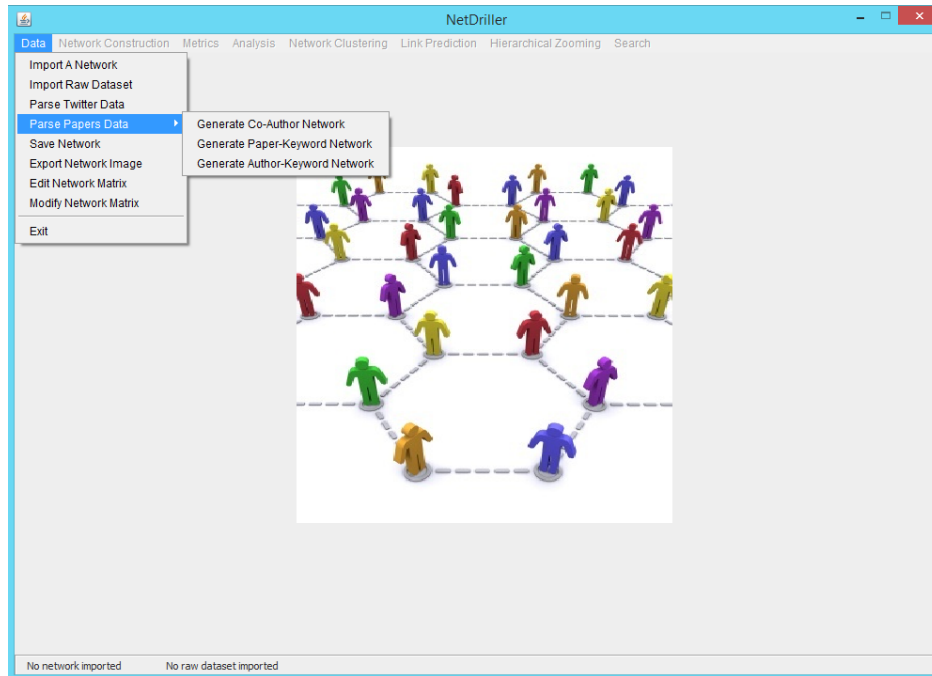


Figure 7.1: NetDriller Interface

import options for the user to import social media data and web search data. We will discuss this further in Section 7.4.

After importing/constructing the network, it is displayed in the main panel as shown in Figure 7.2. The network is shown in terms of the nodes with their corresponding labels, and edges between the nodes with their corresponding weight. After that users can choose to apply SNA techniques on the network to identify the most important nodes using the "Metrics" menu. Other analysis techniques are present in the "Analysis" menu to find bridges and cliques in the loaded network, for folding and inverting the network, etc.

7.4 New Unique Functionalities

In this section, we describe the new main functionalities added to NetDriller. These new functions include adding new network construction techniques from various data sources and also improving the analysis applied on social networks.

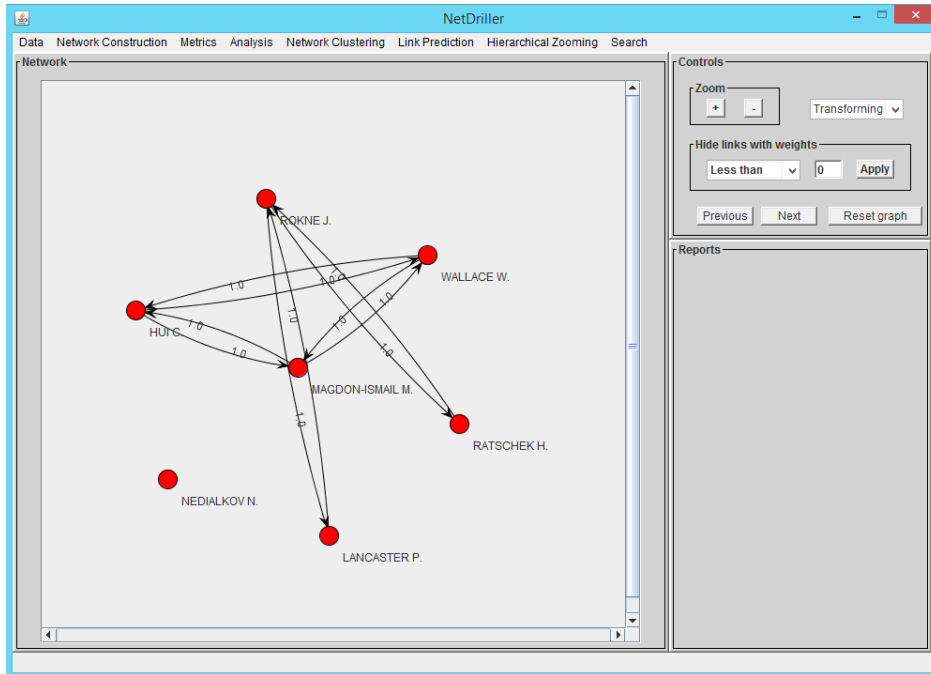


Figure 7.2: Network Interface

7.4.1 Network Construction

In the first version of NetDriller, we have used novel techniques to build a social network of actors from raw data. Techniques such as clustering and association rule mining were used to generate social networks from raw data. In this second version of NetDriller, we have added two more novel ways to construct social networks from social media and the web. One of the most important steps in social network analysis is to first correctly construct the social network from the available data. By providing several sources of data as new options to generate networks, we make our tool more attractive for those who need to apply SNA on these new sources of raw data.

Twitter Data Network Construction

With the ongoing rise of social media, it is now very important to generate social networks from a list of social media posts. Motivated by this, in the second version of NetDriller, we added a new function that allows users to generate several key social network datasets from

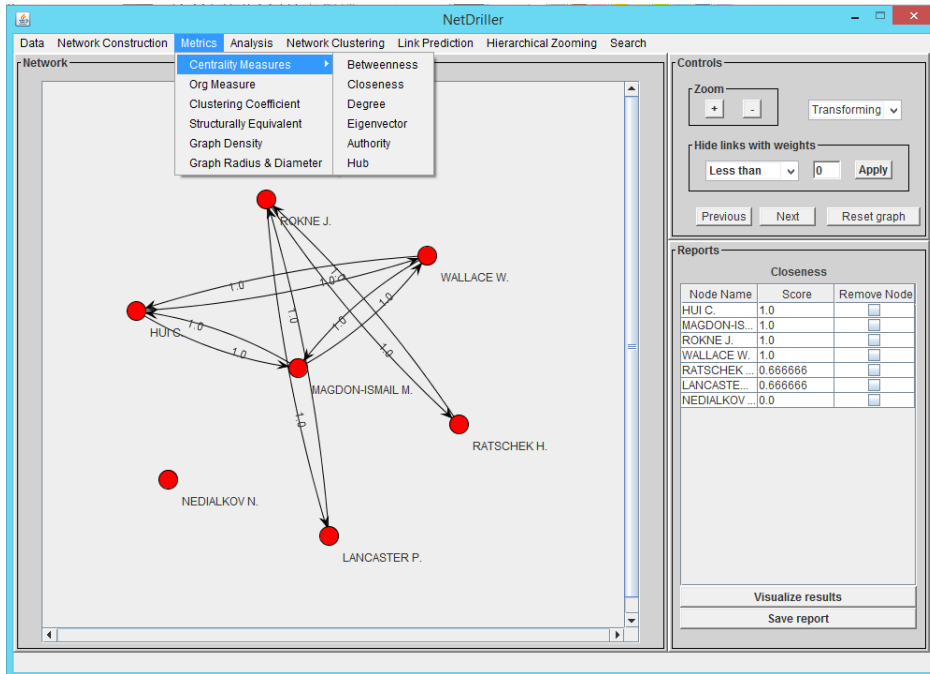


Figure 7.3: Centrality Measures Calculations

a given list of social media posts. Given a set of posts from twitter or online review forums (hereafter called tweets), the data is then processed to produce interrelated datasets which includes:

- a dataset of key terms and the relationship between them based on their co- occurrence in same tweets
- a dataset of tweeters (persons who posted some tweets) and relationships between them based on common terms in their tweets.
- a dataset for two-mode network between tweeters and key terms to show terms written by each tweeter.

The user can import the social media posts by clicking on the "Parse Twitter Data" option in the "Data" menu. This will then open a new dialog box for this purpose as shown in Figure 7.5. As reflected in the figure, the user first uploads the social media posts, then he/she specifies the important keywords that the network is built upon. We also provide the

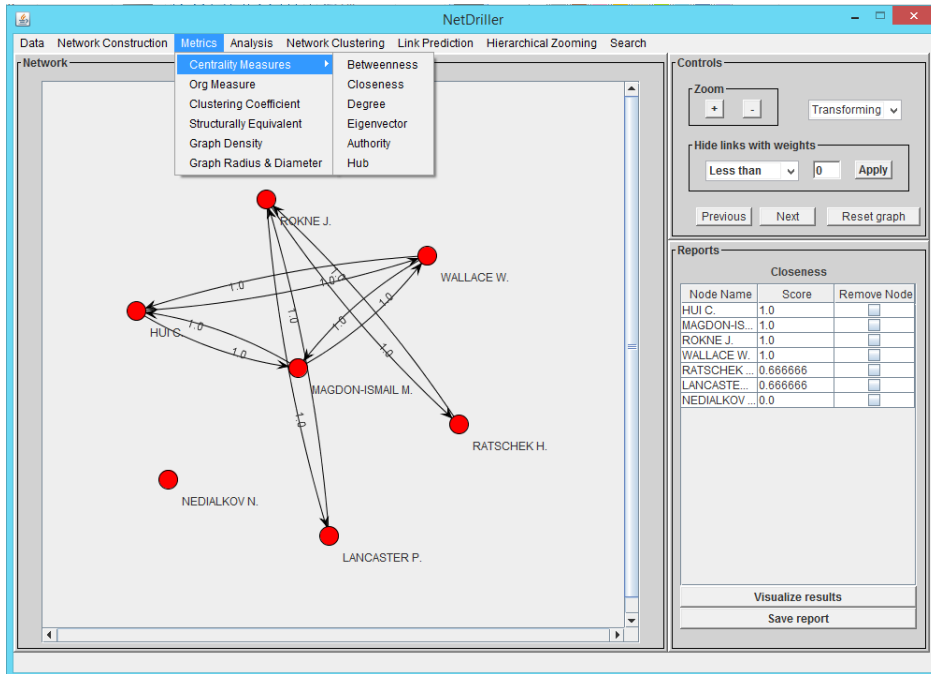


Figure 7.4: Centrality Measures Calculations

functionality for the user to choose start and end dates of the tweets to be included in the posts analysis. This is a very important step to filter tweets while monitoring major events that occur. Usually while analyzing major events, the analysts like to split the analysis of the tweets to before, during and after the occurrence of the event to see the emotions of the people in different times of the event. To include all tweets in the analysis the user should not specify any time entry. The last option is to choose what type of network the user wants to generate from the tweets as mentioned above.

We also implemented a twitter crawler such that given any set of keywords, we go to the twitter API and collect tweets so that analysis can be done on.

Web Network Generator

We designed web crawler such that after importing a labeled network into Netdriller and after ranking nodes using the centrality measures, nodes can then be selected to apply web crawling on them. Every node that is selected starts a new search tree that is processed by the web crawl function. The crawl function takes some root node and a depth value.

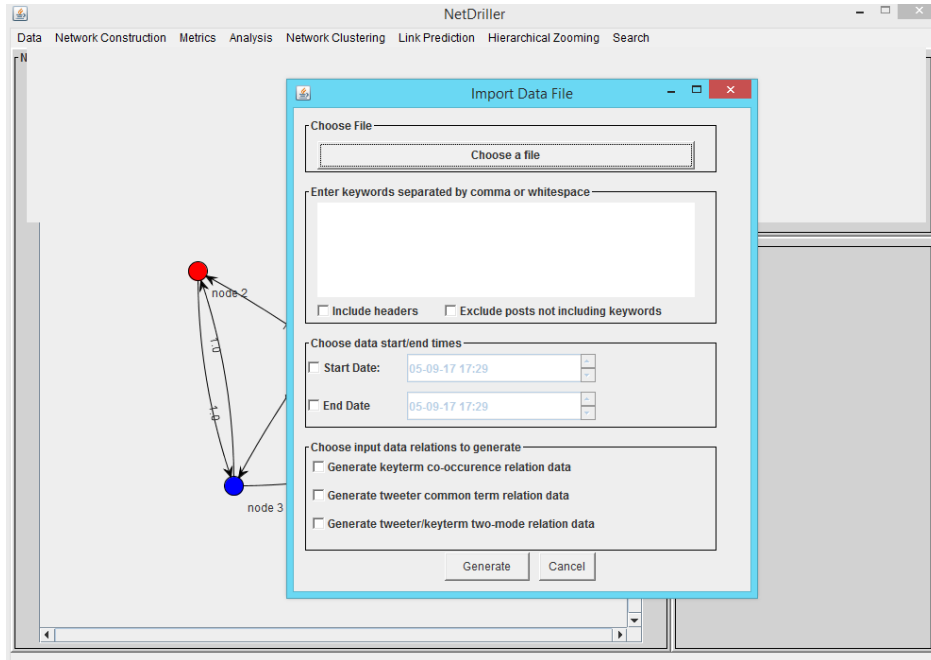


Figure 7.5: Twitter Data Import and Options

When a new Crawler object is instantiated, one should provide a maximum value for the search depth, which is stored in the max depth attribute. This value corresponds with the maximum height of the search tree rooted at the provided node (given by the string start). Thus, given some node, crawl function will recurse for at most max depth layers. On each layer of the search, at most three new nodes are added, and each of these nodes are connected with the provided root node. For instance, suppose the root node is Alice, and max depth = 2. Then the system will find three names most closely-related to Alice and add them to the network. Suppose these names are Bob, Clarence, and Deborah. Then Bob, Clarence, and Deborah will be added to the network graph, and will be connected to Alice with weights given by their frequency values found in documents related to Alice. Then, crawl will be recursively called with roots Bob, Clarence, and Deborah, and the three most closely-related names to each of these nodes will be added to the graph and connected to their respective roots (for a total of nine new nodes for this layer). The search will then terminate, since the maximum depth value has been reached (with respect to the original root node, Alice). This search tree is shown without edge weights below.

We collect these information from the web using Google’s CustomSearch API ³. The term provided (root) is passed to this API, and links are extracted from the returned structure. Each link is processed and entity recognition us applied to extract names from the text. Whenever a name is seen, its frequency is updated and is reflected on the wight of the edge.

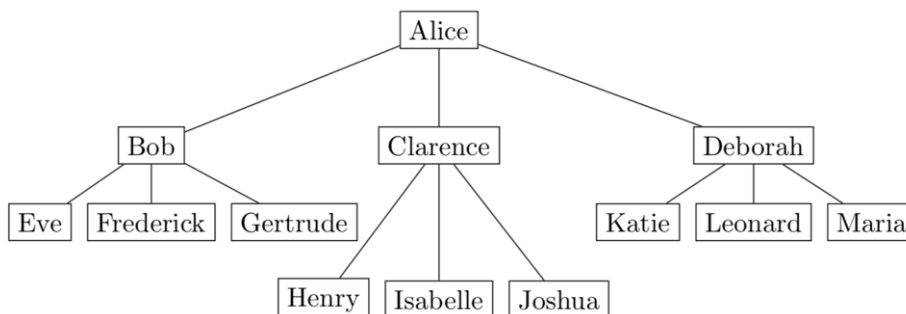


Figure 7.6: Web Crawler

7.4.2 Incremental Network Modification

Most existing network analysis tools do not consider dynamic datasets which are the most commonly encountered in practice. After constructing a social network from an existing dataset, it is possible to have some updates on the dataset. Such updates may be realized and incrementally reflected on the existing network instead of redrawing the network from scratch. New data captured may be used by NetDriller to identify nodes or links to be added or removed from the existing network. This new function added to NetDriller helps in modifying the network incrementally as the network evolves to reflect on the network the changes as inspired from the incremental data. The user will provide a rule file with the changes that occurred on the network which will be then applied on the current network. The structure of the input file which summarizes the various network changes is as follows.

- **Remove a Node:** this rule is made for removing an existing node from the network.

³CustomSearch API: <https://developers.google.com/custom-search/v1/overview>

The format is: "*<node number>,0*" where the former part states the node number to be removed and the latter part is fixed at 0.

- **Add a Node:** to add a new node to the network, the user should add at least one link between existing nodes in the network and the new node. The format is "*<node number>,<new node number>,weight*".
- **Add a Link:** this rule will add a link between two existing nodes in a network with a specified weight of the link. The format is "*<node number>,<node number>,weight*".
- **Remove a Link:** to remove a link between two nodes, the user should use the format "*<node number>,<node number>,-1*".

The user can import the rules file by clicking on the "Modify Network Matrix" option in the "Data" menu which is shown in Figure 7.1.

7.4.3 Temporal Network Visualization

This function allows a user to import several versions of a network that evolves over time. Each version of the network denotes a snapshot of the network at a time-stamp. The network evolution occurs in-terms of nodes/links which are added or deleted. The user can import several network files by going to the "Data" menu, then clicking on "Import a Network" option so that an import dialog is displayed as shown in Figure 7.7. The user can then click on the "Import Multiple Files" button to import the several versions of the network.

After completing the above step, new buttons are added on the right panel which are "Next" and "Previous" as shown in Figure reffig:incremental. These allow the user to go through the different network files imported in order. As nodes/links are added from one version of the network to the other, we show this by highlighting the change by coloring nodes and edges in blue.

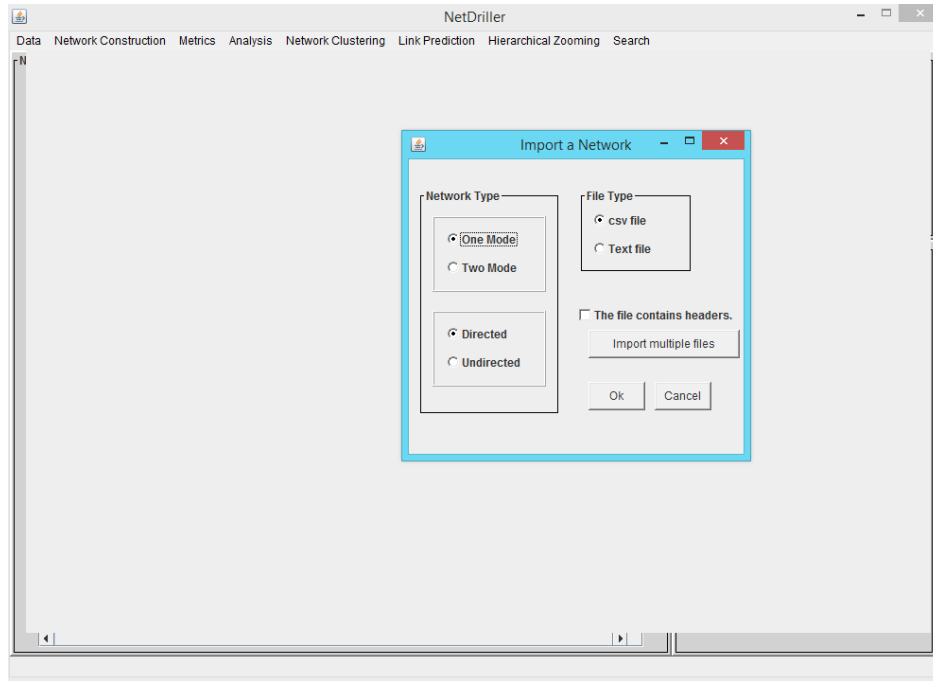


Figure 7.7: Import Options

7.4.4 Link Prediction

Another new social network analysis measure we added to NetDriller is link prediction. Link prediction refers to the problem of anticipating the possibility of what links could be added to the network. This analysis is used in recommendation systems to get which two nodes/people are likely to interact in the future. As shown in Figure 7.9, we added a "Link Prediction" menu which contains four different implementations of existing link prediction techniques, namely Jaccard, Dice, Common Neighbor and Adamic/Adar.

7.4.5 Hierarchical Zooming

Hierarchical Zooming refers to the function of incrementally declustering or clustering the social network into groups in or out, respectively. This analysis technique is required especially in large networks where huge number of nodes and links exist making it difficult to extract useful information from the network by visual analysis. We provide the function to zoom in and out of the network by implementing several community detection algorithms.

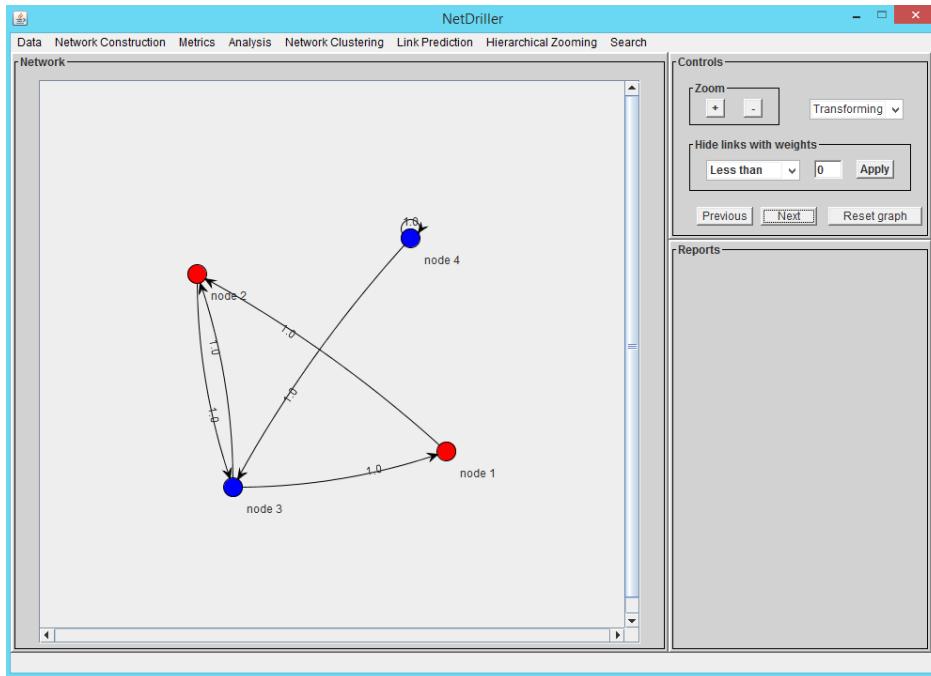


Figure 7.8: Incremental Network Visualization

Each community is displayed as a node in a zoom out mode. Every zoom out step will then reapply the community detection algorithm to reduce each community into a single node. Links across communities become links between nodes representing the communities.

The user can apply this using NetDriller, by clicking on the "Hierarchical Zooming" menu which will open a new window as shown in Figure 7.10. In this figure, the network is shown on the left panel with the hierarchical zooming options present on the right panel. The user can first choose what community detection algorithm he/she wants to use to cluster the nodes. We implemented the Modularity, Edge Betweenness, Voltage, and Minimum Spanning Tree clustering algorithms for the user to choose from. After the user chooses the preferred algorithm, he/she can start zooming in and out of the network using the "+" and "-" buttons. A view of a zoomed in network is shown in Figure 7.11.

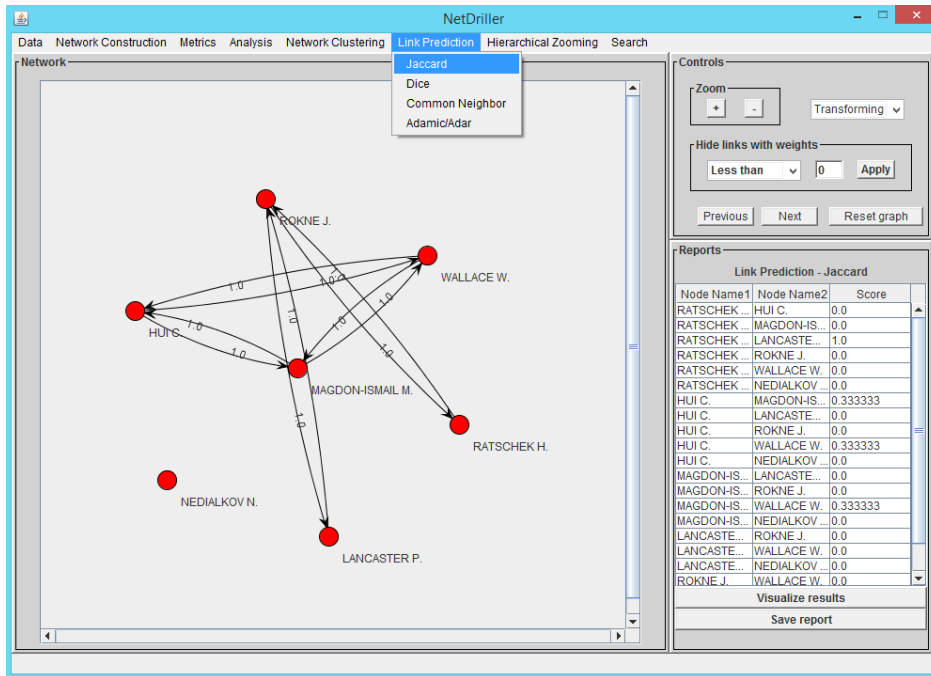


Figure 7.9: Link Prediction Calculation

7.5 Conclusions

We presented in this chapter the second version of NetDriller for social network construction and analysis. NetDriller is a powerful tool to apply analysis on social networks composed from views of social relationships consisting of nodes and links. We added several important social network construction and analysis techniques to NetDriller to work with current and emerging up-to-date datasets. We added a social network construction function which allows users to use social media or web search datasets as input. With the upcoming rise of social media, it is essential for social network analysis tools to generate different networks from social media. We do this using one and two mode networks from terms and tweeters. We also added new analysis measures on the social network to predict hidden relationships in the network using link prediction. Also, since networks can be very large for the analysis, we provide the function for hierarchical zooming of the network. Networks that change over time can be also imported to view the networks at different time-stamps.

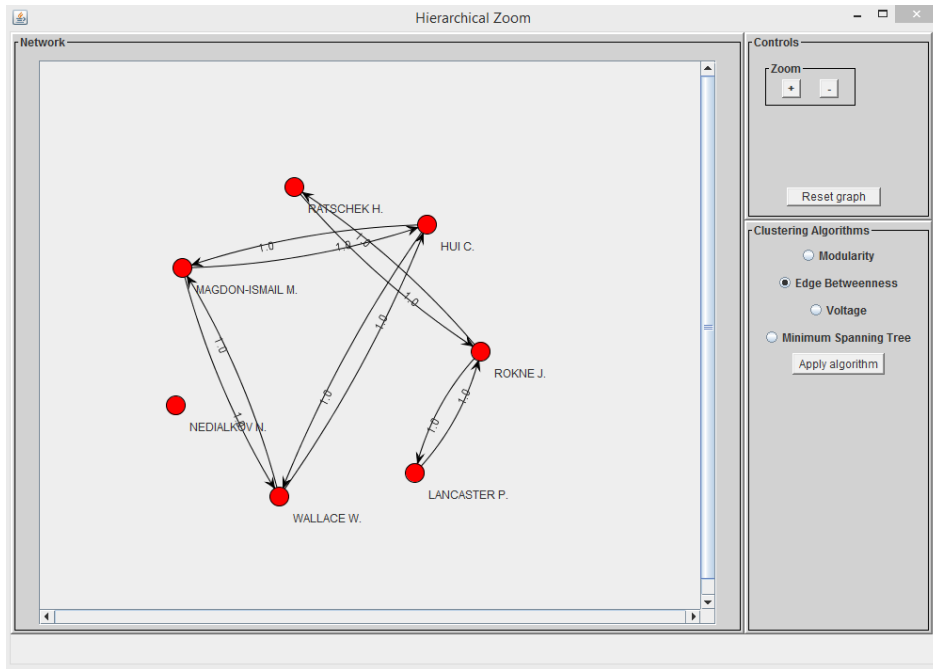


Figure 7.10: Hierarchical Zooming Menu

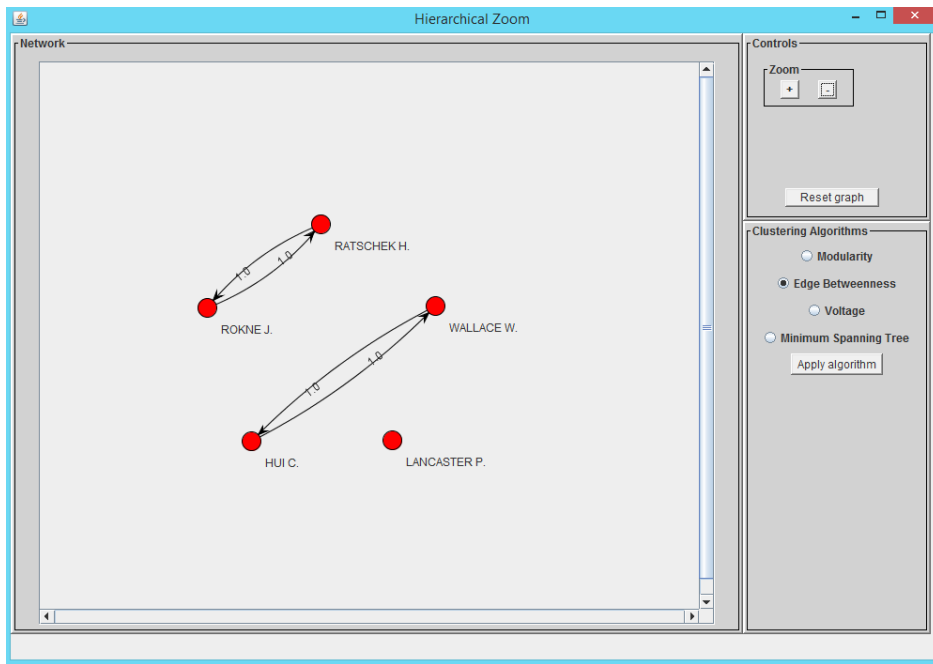


Figure 7.11: Hierarchical Zooming in Action

Chapter 8

Integrated Framework for Criminal Network Extraction from Web Using Text Analysis

Extracting criminals' information and discovering their network are techniques that investigators often rely on to get extra information about criminal incidents and potential criminals. With the recent advances of the Web, a.k.a. Web 2.0, it has become a rich source of data which provides variety of information sources. In this chapter, we propose an integrated framework that combines a variety of available components and makes use of different sources of information provided on the Web to get a better knowledge about criminals or terrorists (we will use criminals to cover all terrorists in the rest of this chapter). Our system extracts criminals' information and their corresponding network using Web sources, such as online newspapers, official reports, and social media. It uses text analysis to identify key persons and topics from crawled Web documents. We build a criminal graph from the analyzed text based on the co-occurrence of mentioning of criminals. Further analysis is applied on the constructed graph to get key people, hidden relationships and interactions between criminals, as well as hierarchical criminal groups within a network. For every process in the

framework, we analyzed various available works and implementations that could be used in the process. While analyzing social media posts, we identified several challenges which show what solutions could be used for that purpose. Finally, we provide a Web application which implements the proposed framework. It also shows how helpful and efficient the system is in extracting and analyzing criminal information.

8.1 Introduction

Over the past decade, the Web has evolved from being a source of information into a rich and more interactive user experience, especially with the development of Web 2.0 based on which social networking platforms emerged. With these advances, huge volumes of data are generated daily from several sources mainly by Web users who upload their personal data and views online, and by providers such as newspapers who always update their websites by posting new news. The massive information available on the Web may serve several purposes one of which is to look into criminal data and analyze criminals interactions.

In this chapter, we propose a framework that focuses on (1) accessing the Web to get criminal incidents used to build criminal profiles, and (2) extract criminal networks using text analysis. The purpose of this framework is to assist a crime investigator by providing access to all information about a criminal and his/her network, and further analyze the criminal network structure. Indeed, the Web hosts several sources of valuable and public information which can be used to analyze criminal networks. Sources such as newspapers have electronic online versions providing authentic and timely data about events. Information related to events covers crimes, suspects and people involved in a criminal incident. Other source of information, such as crime records published by authorities are also available on the Web, providing details about criminal incidents occurring in each country. Social networking websites such as Twitter and Facebook generate most of the data on the Web. In other words, they capture user-generated data, where users have profiles and they add information and

pictures shared with their friends. The type of information extracted from social networking sites is very valuable for crime analysis as criminals post messages and connect with other criminals through social networks.

With the latest criminal and terrorist attacks occurring in the world, analyzing criminals behavior and their connections is the main priority for authorities. The major challenge facing authorities monitoring criminal activities is to accurately and efficiently extract criminals/terrorists information from huge volumes of criminal data available. The large amount of data collected about crimes from the Web are often unstructured, containing duplicate news information. Investigators have to manually go through such data to extract useful information to be used to analyze criminals and study their network. As a result, an analyst faces many difficulties because of the size of the available criminal data as it will be time costly to process; and also it is possible to end up missing useful information inside the collected documents and social media posts. For this reason, text mining can be applied on these documents to extract the required information automatically and efficiently.

The goal of this chapter is to provide an efficient and effective general framework to extract from the Web as much information as possible about criminals/terrorists, their activity involvements, and criminal network to help investigators in analyzing criminal incidents and to provide analytical analysis on criminal networks. To be acceptable, the framework should be characterized by low response time and high accuracy. Unlike other works described in the literature where the focus is on one type of data source (criminal records, chat logs, newspapers, etc.), we provide an integrated framework which combines some well testing and effective components and covers different data sources, including documents, newspapers and social media posts to extract all possible information of a criminal available on the Web. Our framework suggests new techniques to analyze criminal networks to uncover hidden relationships and groups within a network. Implementations of the used components have been integrated into a fully functional and working system. It has been tested using some existing data. The results have been reported in this chapter.

The rest of the chapter is organized as follows. Section 8.2 reviews the current work on criminal analysis and existing frameworks. Section 8.3 shows the general framework, which extracts criminal information and corresponding network from various data sources. We explain in details every process used in the framework and what challenges the processes faces, and we provide several solutions for each process. Section 8.4 provides implementation details related to designing the framework as a Web application. We report results that support the effectiveness of our approach and how the framework is used in real scenarios. In Section 8.5 we conclude the chapter.

8.2 Related Work

Work with criminal data has received considerable attention in the research community, especially post September 11, 2001 attacks in the United States. Researchers are mostly interested to analyze and predict crimes and associated potential criminals. Many research efforts have applied data mining on criminal data. For instance, in [117, 33] criminal data was used to detect patterns to infer the location of criminal incidents that have occurred in the past; the purpose is to highlight such crime locations with the hope to prevent new incidents. Classification and clustering have been applied as well on criminal data in [18] to help in crime related investigations and to summarize crime reports. Chen et al. [29] proposed a criminal data mining framework to better understand the relationship between crime characteristics and analysis capability. Their work is based on data provided by police department criminal database, where they showed an effective use of data mining techniques to extract criminal information and associations using entity extraction, association, prediction, and pattern visualization. They used co-occurrence frequencies between criminals mentioned in the same report to determine the correlation between criminals and to build the social graph.

8.2.1 Frameworks

Anwar et al. [9] proposed a framework to extract criminal information using text mining from chat logs data. In their method, they benefited from chat log data collected from a criminal computer seized by a forensic investigator from a crime scene. After collecting and normalizing the chat data, their framework applies n-gram technique to extract the set of vocabulary in the logs. Then they extracted key information from these n-grams by identifying sets of key terms and key users who have a dominating role in the chat logs. As a result, they built a social graph based on the interaction of users in the chat log, where nodes in the graph are criminal names mentioned in the chat logs and edges in the graph represent whether criminals were mentioned in the same chat session. They analyzed the constructed social graph to determine communities which exist in the criminal network. For this, they applied several clustering techniques, such as k -means, hierarchical agglomerative clustering, and Markov clustering. The challenge with this framework is the assumption that access to criminals computers and their chat log data is possible. In our work, on the other hand, we make use of public data available on the Web to extract criminal information and related networks.

The work described in [91] proposed a system which extracts crime information from police and witness narrative reports. Their system combines information extraction and principles of the cognitive interview to help investigators in collecting and analyzing crime information from witnesses. Many witnesses are often scared or embarrassed to report crime details. To challenge this problem, the authors developed a reporting system which logs witness data and then analyzes the input data by applying entity recognition on crime lexicons to extract relevant information.

Khani et al. [65] proposed a framework to mine criminal data from the Web. Their framework is composed of two main components. First, they developed a crawler system which uses priority queues to fetch unvisited URLs. The second component mines crime information by tagging crime hotspots and categorizing these spots across collected docu-

ments. After collecting documents using crawlers, they extracted prominent communities and constructed their profiles. The authors only discussed the steps needed for the framework without mentioning challenges and implementation issues related to each component.

Yang et al. [172] presented a method to extract criminal information and network from weblogs, which are websites that provide blogging service where users can interact. Their system uses a specific topic exploration from weblogs. Then, they employed Web crawlers to identify blog subscribers who contributed to criminal topics, and accordingly classified bloggers along with their interactions to build a network. After building the bloggers' network, some text classification techniques are used to analyze content of the documents. A visualization of the network has been also suggested for displaying the social network view or the concept network.

Zhou et al. [181] proposed a probabilistic approach which uses email corpus and communication to identify communities of individuals and accordingly provides semantic topic descriptions of these communities. The drawback of this approach is that it is limited to emails, and the constructed network is based on senders and recipients of the emails.

Pramanik et al. [128] proposed a framework that make use of big data resources for criminal pattern detection. Their framework consists of two analytical approaches structure analysis and network mapping. They make use of data from heterogeneous sources to then extract the important nodes from the centrality measures and then visualize the network for investigation purposes.

Sarvari et al. [140] constructed a criminal social graph from a leaked set of email addresses. After collecting these emails, they constructed the social graph by linking these emails to their correspondent Facebook profiles and scraping the public social graph from these profiles. Then they perform social network analysis techniques on the constructed social graph to identify profiles of high rank criminals, criminal organizations and large scale communities of criminals.

Taha et al. [153] proposed a forensic analysis system called SIIMCO that can identify

the influential members of a criminal organization. They do this by collecting crime incident reports which are then turned into a criminal graph using the co-occurrence mentions of two criminals in the same report.

8.2.2 Text Collection

Several works have concentrated on extracting crime information from online newspapers. For instance, the authors in [10] developed a system that extracts crime location information from news articles using named entity recognition, then they used conditional random field (CRF) to classify sentences into crime location sentences. A Web based crime analysis system was proposed in [75] to extract crime information from newspapers. Their analyzer performs crime spot detection, crime comparison and crime pattern visualization from articles available in online newspapers. They first used a focused crawler to extract content of newspaper articles, then they applied document classification to filter out non-crime articles for their analysis. After that, named entity extractor is used to get crime location and date from articles, and then duplicate detector is used to get rid of similar news articles. There are several ways to extract criminal information online using Web crawlers. Crawlers are internet bots that go through web pages for the purpose of Web indexing. They can be also used to fetch content from online newspapers. Herrouz et al. [62] mentioned different tools used for Web content mining and information extraction from web pages. Other research, e.g., [75, 10] used Web crawlers to extract content from newspapers.

After collecting text information from the web crawlers, then text classification is applied to filter relevant text/articles related to crime. For instance, in [10], the authors proposed a method to classify sentences into crime related and non-crime related. They first tokenized sentences mentioned in a text, then they applied named entity recognition to extract location information. After that, they identified ten features for every sentence concerning whether there are different location parts in the sentence and if the sentence contains crime terms. Lastly, they applied CRF (Conditional random fields) algorithm to their classification model

to predict if a sentence is a crime or non-crime related. Choi et al. [31] presented a method to classify news articles from The New York Times newspaper whether they are terrorism related or not. They first extracted context words using WUP similarity from WordNet [115] tree. Bigrams were then built from context words for training on terrorism related articles. They compared these bigrams existing in terrorism related articles with test articles. The comparison is done by calculating a CW (context weight) value such that same bigrams are more frequent if the articles describe the similar subject.

8.2.3 Text Analysis

Researches have applied information extraction (IE) techniques to extract relevant information from crime reports. Various efforts, e.g., [135, 134, 32] have attempted to extract relevant information related to terrorist attacks from a given corpus. These methods rely on using terrorist specific regular expressions to extract information such as how many people were killed, the responsible authority for the attack, and the countries involved. For instance, Conlon et al. [32] performed information extraction from reports on terrorist incidents available online from the National Counter-terrorism Center (NCTC). Their method, called CAINES, relies on lexicons to extract relevant information such as countries and noun phrases which are the most mentioned in the reports. Their system also provides sub-language analysis to extract information related to a crime (Data, Place, Attacker, Attack Type, Killed, and Wounded) using regular expressions on report text.

Other text analysis techniques include text segmentation, tokenization, part of speech tagging and named entity recognition to extract information from text.

Text Segmentation and Tokenization

GATE (General Architecture for Text Engineering) is a framework that has been used by researchers [91] to extract sentences. This method uses a cascade of finite state transducers which split text into individual sentences. In our integrated framework, we employed

NLTK toolkit [16] implementation in python to extract sentences; it uses the PunktTokenizer method described in [83]. CAINES described in [32] invokes the algorithm provided in Lingua POS TAGGER¹ which is a probability based method that assigns POS tags based on a lookup dictionary and a set of probability values. Another method is used in GATE. It benefits from the work described in [61] which uses a decisive list model for POS tag predictions.

POS Tagging

Stanford NLP toolkit [108] uses the method described in [159] for POS tagging. Its reported accuracy is around 97%. Further, another system that could be suggested is that of Ritter et al. (<http://www.aclweb.org/anthology/D11-1141>) which does POS tagging and named entity recognition of tweets. Research on POS tagging has received considerable attention recently with the rise of social media where users' posts are usually unstructured, have grammatical mistakes, contain abbreviations, and are short. These challenges in social media posts motivated researchers to develop techniques with improved POS tagging for social media posts. As shown by the study described in [39], the POS tagger reaches around 98% in normal text, while it shows a drop of accuracy to around 75% on social media posts. As described in [19], GATE includes a new information extractor specifically adapted to micro-blog content called *TwitIE*; it modifies the NLP pipeline to adjust to micro-blog data. First, normalization is applied on text to correct spelling mistakes. Instead of a fixed list of variations, they used a heuristic to suggest correct spellings. Then, they trained their POS tagger with the same method used by Stanford tagger. But, they trained it on millions of annotated twitter posts, reaching accuracy of 90%. Another POS tagger for social media was developed by Gimpel et al. [45]. They used CRF classification model trained on Twitter posts for POS tagging, reaching accuracy of around 90%. The tagger is available online².

¹<http://search.cpan.org/~acoburn>

²<http://www.ark.cs.cmu.edu/TweetNLP>

Named Entity Recognition

Stanford NLP toolkit uses the method described in [43] to train a NER model using a combination of CRF sequence taggers trained on various text. GATE provides several implementations for NER purposes as well. However, like POS tagging, recognizing entities from social media text has challenges that are not addressed in older NER methods that perform well on text articles.

Work on NER for micro-blog content is described in [45, 19]; models are trained on Twitter data to cope with user generated text. Ritter et al. [136] addressed the issue of building a robust named-entity recognition for twitter data by rebuilding the NER pipeline. To overcome the difficulty in the style and vocabulary of twitter data for POS tagging and NER, they manually annotated 800 tweets to build a CRF model to identify POS tags. As for NER, they manually identified 34,000 tokens from 2,400 tweets and used CRF for learning and inference. Trained models of Ritter et al. [136] are available online³.

8.2.4 Keyword Extraction

There exist several keyword extraction techniques described in the literature. For instance, Mihalcea et al. [114] developed a term extractor called TextRank. It is a graph based method where a ranking algorithm is applied on vertices to determine the importance of words. Medelyan et al. [111] proposed a method called Kea to extract key-terms from documents by training a supervised model with the following four main features for annotated training.

- Tf-idf [138]: is a count measure used to get key terms, formed by counting the number of occurrences of each word in a document and getting the top n words.
- First occurrence: is the percentage of documents in which the term is the first occurrence.
- Length: is the number of words in the key-phrase.

³Twitter NLP: https://github.com/aritter/twitter_nlp

- Node degree: is the number of phrases in the candidate set which are semantically related to the key phrase

Implementation of this method is available online⁴. Medelyan et al. [112] extended the work done in Kea by providing functionality to use Wikipedia for key phrase extraction. It is available online⁵. Many other types of topic extractors from documents can be found in [148].

Extracting topics from social media posts however is more challenging as posts are short and mostly composed of just one sentence. Extracting topics from social media has attracted the attention of several researchers, e.g., [137, 110, 173]; they mainly used a stream of posts to identify topics on Twitter. Criminal posts however don't flow as stream, and hence the analysis should be done on individual tweets. For this reason, we propose the usage of hashtags and verb phrase extraction on each post to get topics of individual posts.

8.2.5 Constructing and Visualizing Criminal Network

A criminal network/graph shows interactions between criminals as mentioned in the analyzed text. A criminal graph is defined as $G = (V, E, W)$, where V is the set of vertices representing criminals. For every name detected by NER, a vertex is created in the graph with the criminal's name as the label. E is the set of edges representing the existence of relationship between criminals. An edge is created between two criminal nodes if they are mentioned in the same article or post. Several other works, e.g., [9, 29], used "mentioned in the same document" to create an edge between two nodes. W denotes weights of the edges in the graph. Weights are important to show how strong is the tie between two nodes. Methods such as co-occurrence is used to get the weight between two criminals; it is the number of times two criminals have been mentioned in the same text. Modeling criminal interaction and mentions in text is important because it provides analysts with network visualization.

⁴Kea Term Extractor: <http://www.nzdl.org/Kea/>

⁵<https://code.google.com/archive/p/maui-indexer>

They will see a criminal network and associated interactions in a clear way and they will further investigate people in the network by applying network analysis techniques.

Several tools can be used for visualizing a social network. Open source software tools such as Gephi [12] and IGraph [34] provide the functionality to create and analyze graphs. They have features to apply social network analysis (SNA) measures on a graph. They also apply link analysis which is used to reveal underlying associations between different nodes. Another software tool called NetDriller [89] has been developed by our research group. It is a powerful social network analysis tool used to visualize and analyze social networks. It provides functionality to extract important nodes and relationships between nodes; it can identify communities. These functionalities make NetDriller a good choice for our criminal analysis, and hence has been integrated as part of the developed framework.

8.2.6 Network Analysis

Al-Zaidy et al. [6] focused on mining criminal networks from unstructured text documents. The documents under study were collected from a criminal's device. First, their method extracts personal names from these documents. Then, normalization is applied to eliminate duplicate names that refer to the same person. They discovered prominent criminal communities from the extracted names by applying a method similar to association rule mining (ARM) [107] which looks into people mentioned in documents to get prominent communities of people using a specific threshold. They suggested a way to extract indirect relationship between people specified in the analyzed documents. Their method also extracts for each person a profile which contains his/her network, cities mentioned in the documents, key topics, and text summary of the list of documents in which the person is mentioned.

8.2.7 Link Prediction

Link prediction refers to the problem of anticipating links (interactions between criminals in our case) which might disappear from a network and new interactions which may emerge in

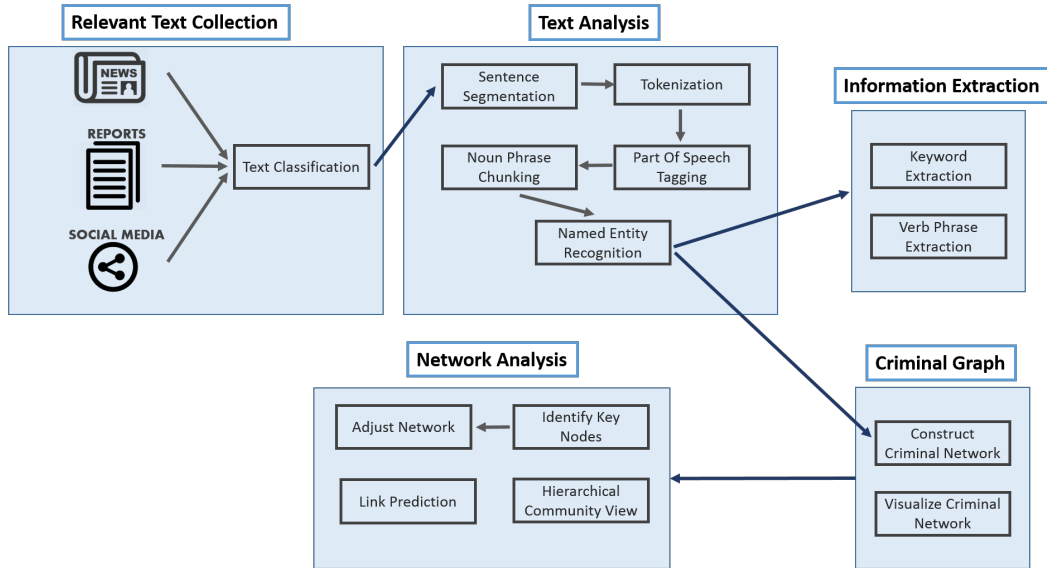


Figure 8.1: System architecture of the integrated framework

the near or far future. Finding hidden links in a criminal graph is very important as we can predict an interaction between two criminals by analyzing the graph structure using social network analysis measures.

Many researchers have tackled this link prediction problem. In general, a connection score is assigned to every pair of nodes in a network to indicate the possibility of link existence. The higher the score is for a given pair, the larger is the chance to have a link between the two nodes. Newman [118] suggested the use of the number of common neighbors between two nodes as a similarity measure to predict a link between these nodes. Adamic/Adar [1] developed a link prediction method which produces a similarity score between nodes depending on their proximity. Afra et al. [2] proposed a method for link prediction where link score between two nodes is calculated using number of shortest paths and length of the shortest path connecting them, as well as their eigenvector and betweenness centrality measures. They reported that their method performed better than other link prediction algorithms described in the literature.

8.3 Methodology

The objective of this work is to extract and predict criminal information and communities to help a forensic investigator by using available public and private resources. The proposed system consists of five major components as depicted in Figure 8.1 which shows the system architecture with all the interactions between the five different components. The first component is used to extract from the Web relevant text information about criminals; then text analysis techniques are applied on the collected text to get people mentioned in the criminal incidents. The extracted names are used by two different components. In the first component, information extraction is applied on the collected documents to extract important keywords and verb phrases. These are linked with people from text analysis to build a criminal profile. In the second component, the names extracted from text analysis are used to construct a criminal graph network. Finally, the criminal graph is used by the network analysis component to identify key criminals, to extract hidden relationships between criminals, and to cluster criminals into different groups.

8.3.1 Relevant Text Collection

The first step in the proposed framework is to extract and collect relevant text from sources available on the Web. To complete this step, the collected text is filtered to keep information related to crimes only.

Text Collection

As shown in Figure 8.1, we use three sources for the information retrieval process.

- Online Newspapers: provide timely updates on events happening around the world, where many stories are published daily by newspapers regarding criminal incidents. For the proposed framework, we suggest using RSS (Rich Site Summary) feeds and newspapers' APIs (application programming interface) for this purpose. Several news-

papers and news sites, such as *The New York Times*⁶, *Daily Mail*⁷, *CNN*⁸, *Google News*⁹ provide APIs which allow users to programmatically access their news articles.

- Reports: refer to official news documents collected from authorities. This type of documents provide reliable and accurate news about criminal incidents that occur in a city, country, continent, or even incidents from around the world. Reports are published monthly or yearly. They form a type of news source different from news articles, but they contain more accurate and detailed information. Reports can be collected from online or private national agencies. Public reports provided by sources, such as the United States Department of State Country Reports on Terrorism¹⁰ contain terrorism incidents from around the world. They are published on annual bases since 2001. Private reports have similar information and text structure as public ones, but they are more specific to countries and cities; collaboration with local authorities is needed to get access to these reports.
- Social Media: provides a very rich source of crime information on which it is possible to capitalize. Crime posts might come from an official source, people witnessing crimes, or even from criminals themselves as it has been shown that criminals use these social media platforms to communicate and share their ideas [11]. Twitter provides an API¹¹ which allows users to search for real-time tweets by a hashtag, keyword, or username. While real-time data is interesting, the API doesn't provide historical data which is important to build a criminal profile over time. For this reason, several tools such as Gnip¹² provide historical twitter dataset.

⁶<https://developer.nytimes.com/>

⁷<https://newsapi.org/daily-mail-api>

⁸<https://developer.cnn.com/docs/read/api>

⁹<https://newsapi.org/google-news-api>

¹⁰US Reports on Terrorism: <https://www.state.gov/j/ct/rls/crt/>

¹¹Twitter API: <https://dev.twitter.com/overview/api>

¹²GNIP: <https://gnip.com/>

Text Classification

After collecting documents and social media posts, the next step is to apply text classification. In our integrated framework, this refers to the process of assigning to articles and posts whether they contain crime related information or not. This process is used to filter out posts or news articles that don't contain crime related information.

8.3.2 Text Analysis

After relevant documents and posts are collected by the first module, the second module applies text analysis techniques to extract phrases and names entities from the text. The following sub-processes are invoked in this module.

Sentence Segmentation

The first process takes an article and splits it into sentences. This step is important because the next sub-processes of this module deal with individual sentences

Tokenization

Tokenization refers to the process of splitting each sentence into tokens such as words, numbers, punctuation, and symbols.

Part Of Speech Tagging

Each token extracted from the previous process is then annotated with its corresponding part-of-speech (POS) tag. This step is very important to extract nouns, verbs, and adjectives so that the machine can later interpret the meaning of each sentence and get semantic information about the text.

Noun Phrase Chunking

Noun phrase chunking is a technique used to identify entities. It segments and labels nouns extracted from POS tagging into noun phrases.

Named Entity Recognition (NER)

After extracting noun phrases, named entity recognition is applied to classify noun phrases to check whether they are people or not. This method is useful to extract who is mentioned in a crime article or post in order to further analyze each mentioned person.

8.3.3 Information Extraction

The third module aims to extract key information about the extracted criminal names. Sub-processes of this module are used to develop a profile for criminals by extracting key terms and actions that criminals participated in using the output from the *Text Analysis* module.

Keyword Extraction

This process aims to extract main key-terms that a criminal is mentioned with across all collected documents and posts. For every article or post, we extract the main keywords from its text and then assign them to people mentioned in the text as extracted by the NER process. The output of this process is a weighted bipartite graph linking criminals to sets of keywords they are mentioned with.

Verb Phrase Extraction

Verb phrases extraction is important in order to analyze and get the meaning of the text. Verbs describe set of actions (killing, stealing, and escaping) in which noun phrases (criminals in our case) are mentioned. Verb phrases are used to extract relevant information about associated noun phrases. In our integrated framework, for every person entity extracted by the NER method, we get the verb phrase in the same sentence using the output of the POS

tagger and then map the verb phrase to the person. At the end, we show a weighted network between this person and all his/her mentioned verb phrases related to criminal incidents.

8.3.4 Criminal Graph

The fourth module handles the extraction and visualization of a criminal network based on input from the text analysis module.

8.3.5 Network Analysis

Network analysis measures provide additional useful information for an analyst while investigating a criminal and his/her network. We integrated four network analysis measures in our framework to provide enough information to analysts in their investigation. These measures have been implemented as part of NetDriller [89].

Identifying Key Nodes

This process aims to identify major and most influential criminals/nodes in a criminal graph. In graph theory, centrality measures identify the most important vertices in a network. There are four main centrality measures implemented in NetDriller and can be calculated for every node in a graph.

- Degree Centrality: is defined as the number of links/edges connecting a node. A high degree centrality means a criminal is mentioned and involved with many other criminals.
- Closeness Centrality: is defined as the average length of the shortest path between a node and all other nodes in a graph. A high value of closeness centrality means a criminal is at the center of the network, i.e., he/she can easily reach all other criminals in the network.

- Betweenness Centrality: is defined as the number of times a node acts as a bridge along the shortest paths between all pairs of other nodes available in the network. Nodes with high betweenness centrality measure are the ones who have more control over information passing between other nodes. Removing these criminals will cut the linkage of the graph. This is true because the latter criminals fall mostly along shortest path(s) and hence may be considered as key nodes for some criminals who try to reach other nodes.
- Eigenvector Centrality: is a measure that identifies the most important and influential nodes in a network. The importance of a node is based on whether it is connected to other important nodes.

Adjusting a Network

After identifying key nodes in a network, it is essential to provide analysts with the option to view a criminal network and modify some existing nodes depending on their importance. This will allow analysts to realize the effect of deleted/modified nodes in the network. By removing criminals from the network, analysts can look into how to disrupt the criminal network structure so that they can arrest these criminals. Recursively applying the later step will greatly help in dissolving or diminishing the network.

Hierarchical Community View

Hierarchical community view aims to show a criminal network as a set of communities. It is essential for an analyst to view a network as a set of communities because this generally infers criminal groups residing within a network. Many community detection algorithms have been created; they are described in the literature; for more details, see [93], which provides a benchmark for community detection algorithms. We suggest, however, a hierarchical community view as a functionality of the analysis supported by our framework. We provide the option to zoom in and out of the network using community detection algorithms. This

way, it is possible to display each community as a node in a zoom out mode and, on the other hand, display each node as a community in a zoom in mode. Investigators will benefit from the flexibility of our approach to concentrate better on details of any component of a network by zooming in only for the specific component.

Link Prediction

Applying link prediction and identifying hidden relationships between criminals is an important step as interactions between two criminals can be revealed from this type of analysis based on the graph structure.

8.4 Framework Implementation and Web Application

The objective of implementing the proposed framework is to demonstrate its usage and effectiveness in crawling data from different sources to extract and analyze criminal information and associated networks. In this section, we report for every component of the system architecture the methods used in its implementation and their effectiveness. We also show the implementation of the framework as a Web application and this will maximize its availability for usage by authorized investigators.

8.4.1 System Implementation and Validation

Relevant Text Collection

The first task in the system as discussed in Section 8.3.1 is to collect and classify articles and social media posts. For testing, we randomly collected 91 articles from the Web. These articles may be classified as follows:

- *News Articles*: We collected 21 articles from DailyMail API. To build a diverse data for classification, the selected articles are related to world news and some to sport news.
- *Social Media*: We collected 50 posts from Twitter API. These were taken from random user accounts.
- *Official Reports*: The remaining 20 articles were taken from 2007 Terrorism report published under United States Department of State Country Reports. Paragraphs were randomly selected for analysis.

The collected data was then manually annotated to extract people names mentioned in each snippet and to mark whether the post/article is crime related or not. We applied this manual annotation to test the accuracy and effectiveness of our text classification and NER methods.

We utilized these five different metrics to evaluate the crime classification and person entity recognition tasks:

- Confusion Matrix: shows the performance of a classifier compared to a test data by highlighting potential sources of errors. Columns of the matrix represent prediction results and rows represent actual classes. Four cases (matrix entries) are possible in a two-classes classification problem. True positive (tp) means a post was manually annotated as a crime and it was successfully predicted as crime related. False negative (fn) means the latter post was predicted as non-crime related. True negative (tn) means a post was manually annotated as non-crime and it was successfully predicted as not crime related. False positive (fp) means the latter post was predicted as crime related.
- Accuracy: is the ratio of correctly classified results, i.e., the fraction of tp and tn.

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn} \quad (8.1)$$

- Precision: is the number of positive predictions divided by the total number of positive class values predicted.

$$Precision = \frac{tp}{tp + fp} \quad (8.2)$$

- Recall: is the number of positive predictions divided by the number of positive class values in the test data.

$$Recall = \frac{tp}{tp + fn} \quad (8.3)$$

- F-Measure: is the harmonic mean of precision and recall.

$$F - Measure = 2 * \frac{precision * recall}{precision + recall} \quad (8.4)$$

For text classification, we compiled a crime gazetteer (dictionary) which contains 84 crime related words chosen from “Crime Vocabulary”: <https://myvocabulary.com/word-list/crime-vocabulary/>. A sample of crime related keywords in gazetteer is shown in Table 8.1.

Table 8.1: Sample Words from the Crime Related Gazetteer

Crime Gazetteer Sample
”shooting”, ”slaying”, ”smuggling”, ”stabbing”, ”suspect”
”terrorist”, ”theft”, ”trafficking”, ”weapon”, ”crime”
”captivity”, ”imprisonment”, ”jihadi”, ”sentenced to”, ”terror”

We then chose a simple technique for text classification by checking if individual posts/articles contain a word from gazetteer. An article that contains a word listed in gazetteer is classified as crime related. Although this approach is simple, it was very effective when applied on the collected data. Table 8.2 shows the confusion matrix of the classified documents, and Table 8.3 reports accuracy results of using the gazetteer to predict whether articles/posts

are crime related or not. As shown in Table 8.3, text classification on social media, news articles and posts reported high accuracy scores, 96%.


Table 8.2: Confusion Matrix of the Classified Articles

Reports	Predicted: NO	Predicted: Yes
Actual: NO	34	2
Actual: Yes	2	53

Table 8.3: Classification Accuracy of the Collected Data

	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)
Twitter	96	96	96	96
Reports	95	100	95	97.44
News	95.24	91	100	95.24
Total	95.6	96.36	96.36	96.36

Figure 8.2 shows two examples where a post was predicted as crime related but in fact it is not. The first post was classified as crime because it contains the word “crime”, while the second post is a football article which contains the word “illegal”.

Source	Title
 Twitter	Recent research finds that counties with sanctuary policies experience less crime and stronger economies than those without the policies


Source	Title	Text
 DailyMail	Southampton warn Liverpool and Chelsea off defender Virgil van Dijk: 'He is not for sale in this window'	Southampton chairman Ralph Krueger insists the club's new Chinese owners are fully behind their refusal to sell star centre-back Virgil Van Dijk. The Dutch defender is desperate to leave and has handed in a transfer request in the knowledge that Liverpool, Chelsea and Manchester City are all interested in signing him. Southampton have stood firm this summer, reporting Liverpool to the Premier League for making an illegal approach and warning other rivals away.

Figure 8.2: False Positives Examples in Crime Articles Prediction

Text Analysis

The two main objectives of text analysis are (1) to extract POS tags of the words in the collected text, and (2) to extract people mentioned in the text. Sentence segmentation, tokenization and POS tagging have been implemented in python using NLTK toolkit [16]. We used Stanford’s NER technique [43] for person entity recognition.

As mentioned earlier, we have manually annotated the collected documents to extract people mentioned in them. After extracting person entities using Sanford NER technique, we compared the manually labeled names with the extracted ones. Tables 8.4, 8.5, and 8.6 report the performance of Stanford NER for person detection with the confusion matrices of Twitter, DailyMail, and US reports, respectively. True negatives (Prediction= False, Actual= False) of these confusion matrices correspond to the number of posts without any name mentioned in them, and the prediction is that there is no name in the whole post. Accuracy performance of the person entity recognition process is reported in Table 8.7 where the F-Measure value for all three datasets is around 96%.

Table 8.4: Confusion Matrix of Twitter Named Entity Recognition

Twitter	Predicted: NO	Predicted: Yes
Actual: NO	12	1
Actual: Yes	3	26

Table 8.5: Confusion Matrix of DailyMail Named Entity Recognition

DailyMail	Predicted: NO	Predicted: Yes
Actual: NO	0	5
Actual: Yes	4	89

Table 8.6: Confusion Matrix of Reports Named Entity Recognition

Reports	Predicted: NO	Predicted: Yes
Actual: NO	6	3
Actual: Yes	0	47

Figure 8.3 shows two misclassification examples of the person entity recognition process on the collected tweets. In the first tweet, the word “Lemme” is classified as a person

Table 8.7: Accuracy of Named Entity Recognition on the three datasets

	Precision (%)	Recall (%)	F-Measure (%)
Twitter	98.41	95.38	96.88
DailyMail	94.68	95.7	95.19
Report	94	100	96.91
ALL	95.7	97.03	96.32



Source	Title
 Twitter	Lemme be blunt about this, anybody who thinks Lebron is better then Kobe OR Jordan needs to be checked into a mental hospital immediately.
 Twitter	Trump should be arrested and tried for the crime of aiding & abetting in the murder of #HeatherHeyer. @Rosie #resisthate @chelseahandler

Figure 8.3: NER Erroneous Examples in Tweets

name while in fact it is a slang of “let me”. In the second tweet, we manually annotated “#HeatherHeyer” as a person name, but it was not classified correctly by the person entity recognition process. These two examples demonstrate why Twitter text analysis is challenging as users have no structure in spelling names. Still average accuracy of 97% was reported for the person entity recognition process on tweets, and this shows that such tweets are rare.


Information Extraction

There are two processes in information extraction that help analysts in understanding a specific criminals by providing analysis on text in which they are mentioned. Keyword extraction has been accomplished by implementing TextRank [114] keyword extraction technique. As mentioned in Section 8.3.3, TextRank is a graph based ranking where words are considered as vertices to determine their importance. For tweets, however, we also used the hashtag as a topic of the tweet because social media posts are limited in number of words. As for verb phrase extraction, we used the output of the POS tagger from the “Text Analysis”

component to get verb phrases representing actions.

Criminal Graph

A criminal graph is built using co-occurrence of people mentioned in the collected text. Persons mentioned in text are represented as nodes and edges between nodes exist when names are mentioned in the same article/post. We used two alternatives to view a graph. In the first alternative we made a graph compatible with already existing graph visualization tools, such as NetDriller. This allows users to import a network using these tools and view the corresponding graph. In the second alternative users may utilize our Web application to view a criminal graph on the Web.



LONDON (AP) - Three British men convicted of planning a knife and bomb attack on troops or police inspired by Islamic extremism were sentenced Thursday to at least 20 years in prison. An accomplice received a minimum 15-year-term. Naweed Ali, Khobaib Hussain, Mohibur Rahman and Tahir Aziz were convicted in a London court on Wednesday of preparing terrorist acts after a trial that was partly held in secret for national security reasons. Ali, Hussain and Rahman met while serving prison terms for terrorism offenses, and later set up a group called the "Three Musketeers" on a messaging app. The men were arrested in August 2016 after weapons were found in Ali's car, including a partial pipe bomb and a meat cleaver with "kaffir" - infidel in Arabic - on the blade. Prosecutors say they intended to attack police or military targets. Prosecutor Bill Emlyn Jones said the defendants probably intended to use their cars as weapons in an attack, as well as knives and the pipe bomb. Judge Henry Globe sentenced Ali, Hussain and Rahman to life with no chance of parole for 20 years. He said Aziz, a late recruit to the plot, must serve at least 15 years before being considered for parole.

Figure 8.4: Example Article Showing People Mentioned Several Times by Their Last Name

Figure 8.4 shows a sample news article where multiple people are mentioned. Names like Naweed Ali and Khobaib Hussain were successfully extracted by the person entity extractor. The same article mentions again the same people but referring to them by their last names instead. This type of reference was encountered in several articles. For this reason, while building a criminal network, names mentioned in the same article and at the same time as part of other names already encountered in the same article are considered to belong to the same person.

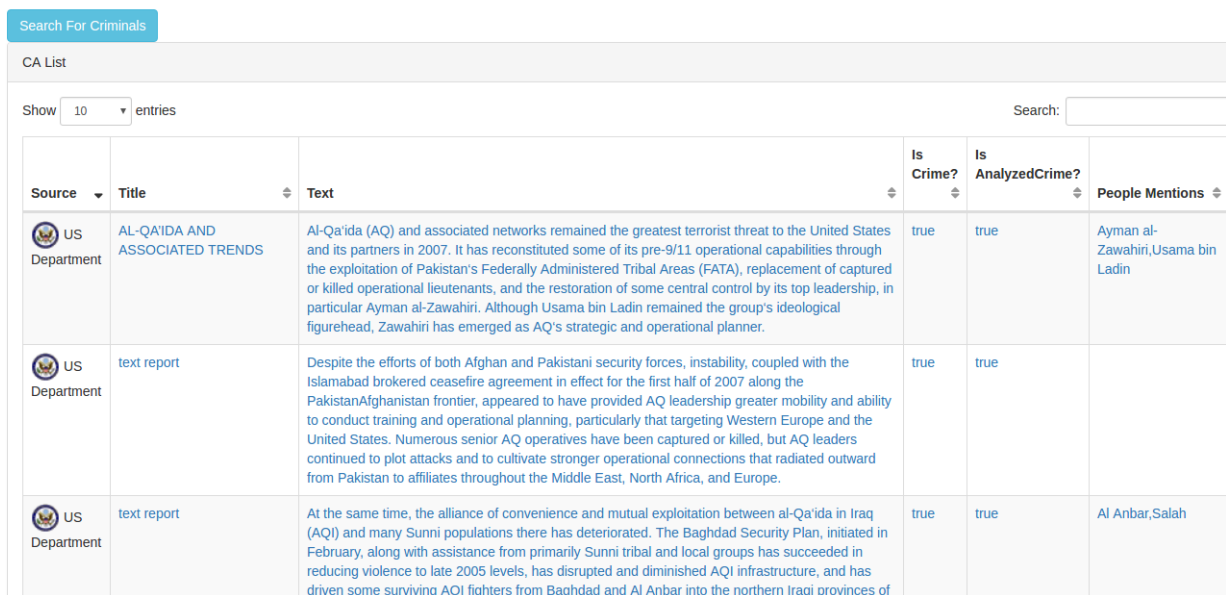
Network Analysis

We provide two ways for analysts to apply the proposed network analysis techniques. The first alternative is based on their preferred graph analysis tool where we provide a criminal graph as a csv file to be imported. The second alternative is using our Web application which analysts can use to identify key nodes, apply link prediction, clustering and adjust the network, all through the Web application discussed in the next section.

8.4.2 Web Application

In this section, we explain how our proposed framework may be invoked as a Web application. We will refer to system implementation details as discussed in Section 8.4.1.

Collected Articles



The screenshot shows a web application interface for 'Collected Articles'. At the top, there is a search bar labeled 'Search For Criminals'. Below it, a 'CA List' section contains a 'Show' dropdown set to '10' and a 'Search:' input field. The main content is a table with the following columns: 'Source', 'Title', 'Text', 'Is Crime?', 'Is AnalyzedCrime?', and 'People Mentions'. Three rows of data are visible, each representing an article from the 'US Department'.

Source	Title	Text	Is Crime?	Is AnalyzedCrime?	People Mentions
US Department	AL-QA'IDA AND ASSOCIATED TRENDS	Al-Qa'ida (AQ) and associated networks remained the greatest terrorist threat to the United States and its partners in 2007. It has reconstituted some of its pre-9/11 operational capabilities through the exploitation of Pakistan's Federally Administered Tribal Areas (FATA), replacement of captured or killed operational lieutenants, and the restoration of some central control by its top leadership, in particular Ayman al-Zawahiri. Although Usama bin Ladin remained the group's ideological figurehead, Zawahiri has emerged as AQ's strategic and operational planner.	true	true	Ayman al-Zawahiri, Usama bin Ladin
US Department	text report	Despite the efforts of both Afghan and Pakistani security forces, instability, coupled with the Islamabad brokered ceasefire agreement in effect for the first half of 2007 along the PakistanAfghanistan frontier, appeared to have provided AQ leadership greater mobility and ability to conduct training and operational planning, particularly that targeting Western Europe and the United States. Numerous senior AQ operatives have been captured or killed, but AQ leaders continued to plot attacks and to cultivate stronger operational connections that radiated outward from Pakistan to affiliates throughout the Middle East, North Africa, and Europe.	true	true	
US Department	text report	At the same time, the alliance of convenience and mutual exploitation between al-Qa'ida in Iraq (AQI) and many Sunni populations there has deteriorated. The Baghdad Security Plan, initiated in February, along with assistance from primarily Sunni tribal and local groups has succeeded in reducing violence to late 2005 levels, has disrupted and diminished AQI infrastructure, and has driven some surviving AQI fighters from Baghdad and Al Anbar into the northern Iraqi provinces of	true	true	Al Anbar, Salah

Figure 8.5: Web Application Homepage

Figure 8.5 shows the homepage of our application where the collected news are displayed in table format. For every article/post, we display its manually annotated class (crime/non-crime), predicted class, and people extracted from the text using the person entity recognizer.

The homepage also provides a search button on the top of the page to make it possible to search for specific person(s).

If the “Search for Criminals” button is clicked, users will be redirected to the search page where they can enter names to search for. The system stores any entered name as a panel for later usage as shown in Figure 8.6.

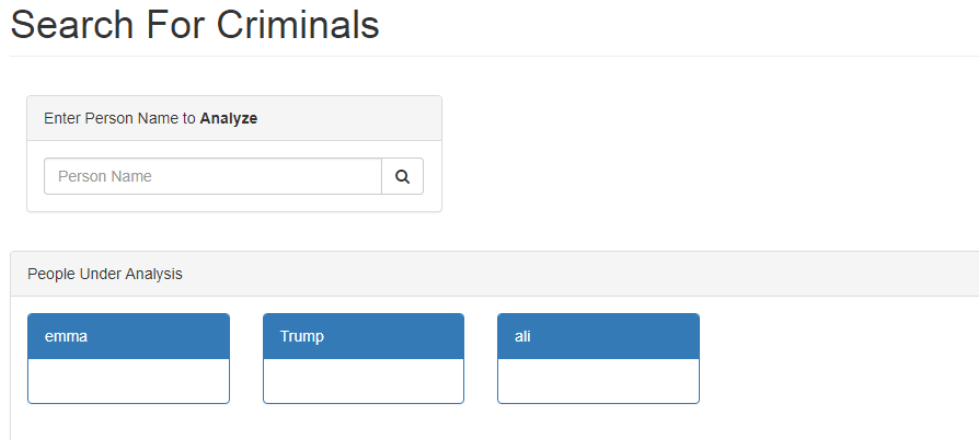


Figure 8.6: Search to analyze people mentioned in articles

Users can then click on the searched name panel to be redirected to a specific person’s analysis page. Figure 8.7 shows the analysis page when we searched for the name ”ali”. As shown in Figure 8.7, the analysis page is composed of five components.

The first component shows text documents in which this person’s name is mentioned. For this step, we used the output of the person entity recognition to get articles that this person is mentioned in.

The second component shows keywords extracted from articles where this person is mentioned. Size of a keyword is directly related to the number of times it is extracted from the articles.

The third component shows verb phrases extracted from the articles and like the keyword extraction panel, the size of a verb phrase is directly related to its number of occurrences. Words in the keyword extraction and verb phrase panels are clickable. Once a user clicks on any word in the chart, he/she will be directed to a page which shows articles that contain

(ali) - Person Investigation

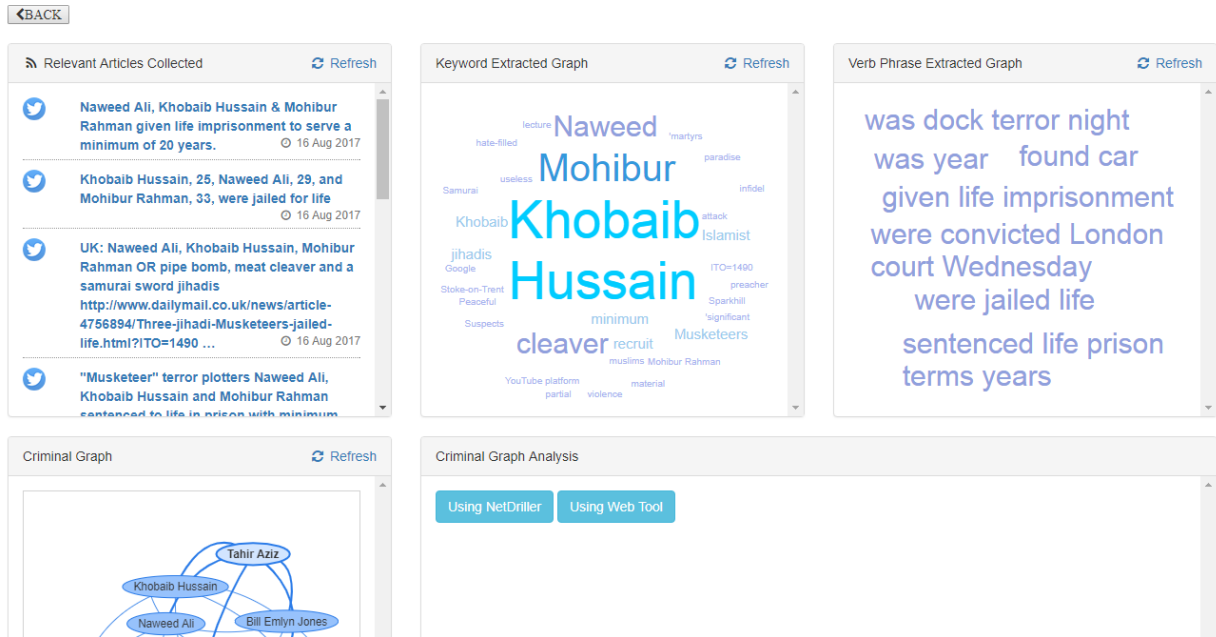


Figure 8.7: Person Analysis Page

the word and associated phrases for further analysis.

The fourth component shows the criminal graph extracted from the co-occurrence between the person searched for and other people mentioned in the same articles.

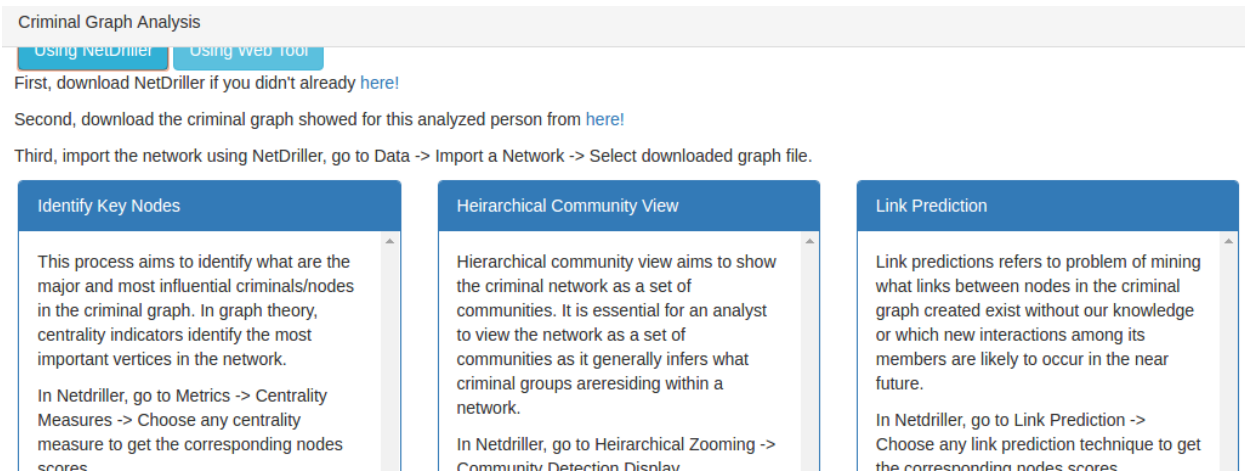


Figure 8.8: Using NetDriller for Analysis

The last component provides two options for applying the analysis techniques discussed in Section 8.3.5 on the criminal graph. The first option is using NetDriller when a user clicks

on the “Use NetDriller” button. Details on downloading a criminal network, then importing and analyzing it in NetDriller are shown in Figure 8.8. The other option is clicking on the “Using Web Tool” button for graph analysis using the Web application. When a user clicks on the latter button, he/she will be redirected to the Web tool analysis page shown in Figure 8.9. Here, the network of the person is shown with edit options as in the first component. Buttons on the right of the network allow users to export the network as a csv file and to re-initialize the network after editing. Buttons under the network are dedicated for the rest of the analysis. Next, we will show how users can benefit from this page to apply the four different analysis measures proposed in Section 8.3.5:

(ali) - Graph Analysis

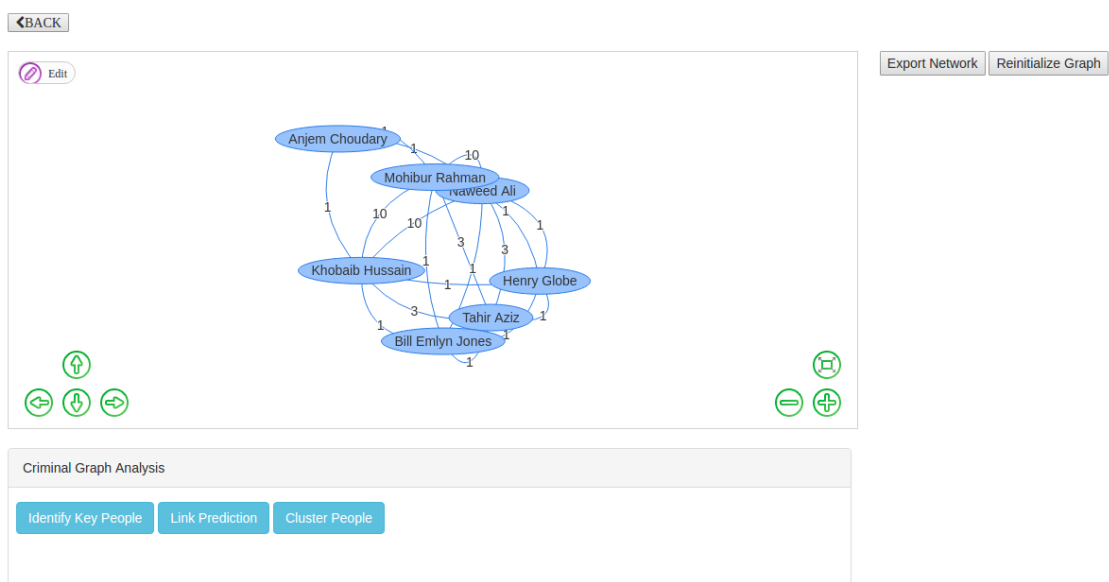


Figure 8.9: Analyzing the Criminal Graph Using the Web Application

1. Identify Key People: this process is done by assigning to every node in the network a score which reflects its importance. We used NetDriller API to calculate scores of the nodes based on degree centrality, closeness centrality, or eigenvector centrality. This step is achieved in the Web Application by clicking on "Identify Key People" button in the analysis page. Figure 8.10 shows the output of the key people identification step

using degree centrality measure. To calculate the scores using different measures, users can choose their desired method from the drop-down list.

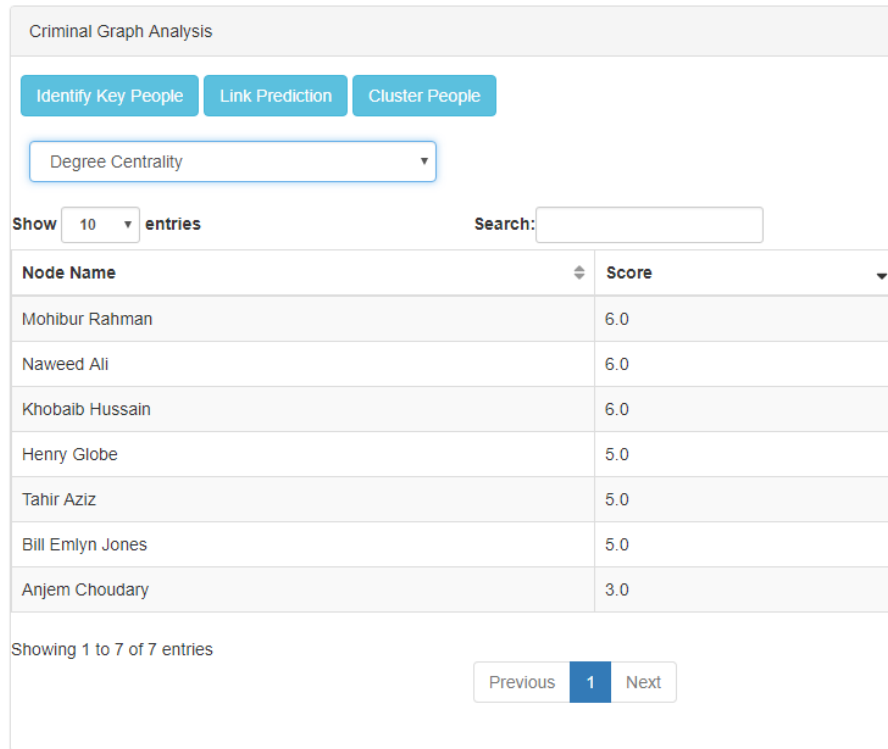


Figure 8.10: Identifying Key Nodes Using the Web Application

2. Adjust Network: After identifying key nodes, users have the ability to modify the network by removing these important nodes and watch how the network will restructure. This is done by applying the edit functionality on the network. Figure 8.11 shows the different options for users to modify the network. Users can delete nodes, add nodes, add links, and delete links. Users can use the export button to download the modified network as a csv file.
3. Hierarchical Community View: This functionality is activated by clicking “Cluster People” button. Clustering is done using NetDriller API; the result is displayed on the screen as shown in Figure 8.12. Users can then click the same button again to re-cluster the clusters. We also provide the functionality for users to open up clusters by double clicking inside them to view their internal nodes.

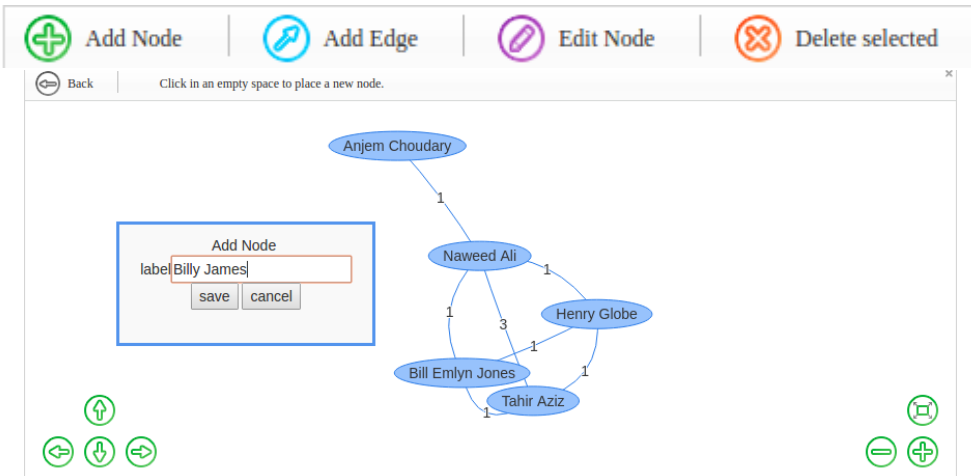


Figure 8.11: Adjust Network Functionality

(ali) - Graph Analysis

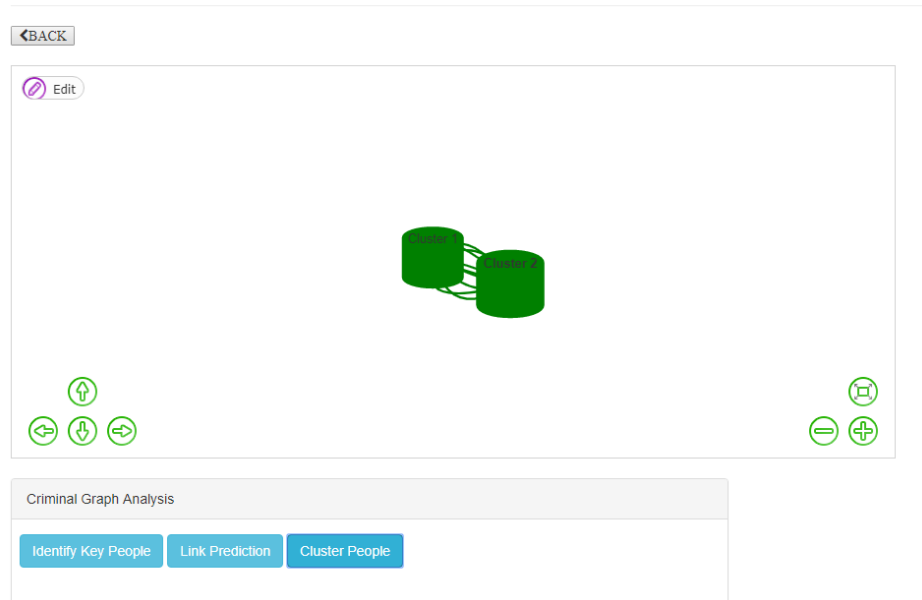


Figure 8.12: Clustering View from the Web Tool

4. Link Prediction: this process is done by calculating a score value for all possible links that may exist in a graph. After users click on the “Link Prediction” button, link scores are displayed as shown in Figure 8.13. Users can also select their desired link prediction method (Jaccard, Dice, Common Neighbor, and Adamic/Adar) to execute. Results from each method are taken from NetDriller API.

Criminal Graph Analysis

Identify Key People Link Prediction Cluster People

Dice

Show 10 entries Search:

Node1 Name	Node2 Name	Score
Khobaib Hussain	Mohibur Rahman	0.8333333134651184
Naweed Ali	Khobaib Hussain	0.8333333134651184
Naweed Ali	Mohibur Rahman	0.8333333134651184
Henry Globe	Bill Emlyn Jones	0.800000011920929
Tahir Aziz	Henry Globe	0.800000011920929
Tahir Aziz	Bill Emlyn Jones	0.800000011920929
Anjem Choudary	Bill Emlyn Jones	0.75
Henry Globe	Anjem Choudary	0.75
Tahir Aziz	Anjem Choudary	0.75
Bill Emlyn Jones	Khobaib Hussain	0.7272727489471436

Showing 1 to 10 of 21 entries

Previous 1 2 3 Next

Figure 8.13: Link Prediction Output

8.5 Conclusions

In this chapter, we presented a new criminal analysis framework for analysts to use in investigations. The framework crawls and captures all possible information needed about a criminal from available Web sources such as newspapers, official reports, and social media which provide a variety of timely and important information about criminals and their activities. After collecting information from Web sources, text classification is applied as a preprocessing step to filter crime news. We then discussed various text analysis processes which are applied on the collected documents and posts to tag text with POS tags and to extract criminal names. We described challenges of text analysis techniques when applied on social media posts generated by users. We also described methods used to tackle these challenges. A profile is built for each criminal by extracting keywords and his/her involve-

ments in crime incidents. Another important module we proposed is extracting and viewing a criminal graph which reflects interactions and mentions of criminals with each other. This helps analysts to view and handle criminals network. Finally, after creating a criminal graph, network analysis techniques are applied to identify key members, communities and hidden links. These provide analysts with extra needed information about who is leading crime groups and people who should be watched. Our criminal analysis framework is unique as it extracts online information from multiple Web based sources to get and analyze criminals and their networks. We also provided an implementation of the framework by building a Web application that analysts can use for criminal information extraction and network analysis.

Chapter 9

Early Warning System: From Face Recognition by Surveillance cameras to Social Media Analysis to Detect Suspicious People

Surveillance security cameras are increasingly deployed in almost every location for monitoring purposes, including watching people and their actions for security purposes. For criminology, images collected from these cameras are usually used after an incident occurs to analyze who could be the people involved. While this usage of the cameras is important for a post crime action, there exists the need for real time monitoring to act as an early warning to prevent or avoid an incident before it occurs. In this chapter, we describe the development and implementation of an early warning system that recognizes people automatically in a surveillance camera environment. We train a feature extraction model for face recognition using convolutional neural networks to get a good recognition rate on the Chokepoint dataset collected using surveillance cameras. The system also provides the function to record people appearance in a location, such that unknown people passing through a scene excessive num-

ber of times (above a threshold decided by a security expert) will then be further analyzed to collect information about them. We implemented a queue based system to record people entrance. We try to avoid missing relevant individuals passing through as in some cases it is not possible to add every passing person to the queue which is maintained using some cache handling techniques. We collect and analyze information about unknown people by comparing their images from the cameras to a list of social media profiles collected from Facebook and intelligent services archives. After locating the profile of a person, traditional news and other social media platforms are crawled to collect and analyze more information about the identified person. The analyzed information is then presented to the analyst where a list of keywords and verb phrases are shown. We also construct the person's network from individuals mentioned with him/her in the text. Further analysis will allow security experts to mark this person as a suspect or safe.

9.1 Introduction

Video surveillance systems are installed and used almost everywhere nowadays for the purpose of recording, monitoring and reviewing incidents that may happen around from permitting only certain persons to enter a building to identifying potential suspicious criminals as early and preventive as possible. Several applications are associated with video surveillance systems such as traffic monitoring, security systems, incident recording, etc. Images from surveillance security cameras/closed-circuit television (CCTV) are used as an important evidence during crime investigations to identify key persons who are involved in the crime. In theory, using CCTV images to identify people involved in a crime scene and compare these collected face images to gallery images of criminals should be a straightforward process for police officers and crime forensic experts. This might be true and affordable for limited cases. However, the current era of globalism and the associated big data turns manual analysis unfeasible and hence pushes hard towards more effective automated systems capable of

supporting investigators in their duties.

Many researchers, e.g., [22, 60, 23, 113], however, have shown that identifying unfamiliar faces and comparing CCTV images with mugshot gallery images is very difficult and challenging even for humans and police officers. Bruce et al. [22] performed several experiments aimed at testing the ability of people to identify faces in mugshot images. People were shown a person's face and then shown 10 other target images of the same person and other people for the purpose of matching the shown face with one of the 10 candidate images. The subjects were then asked to decide whether or not the shown face was present in the 10 other images; and if it is present, to pick the correct match. The results of this experiment showed that people performed poorly. They picked the correct person only about 70% of the occasions. The department of psychology at University of Glasgow [25] did a research work on the ability of individuals and police officers to identify target people captured by a surveillance security camera. They performed experiments to answer questions about the performance of people to identify familiar and unfamiliar people in a video surveillance environment. The first experiment examined whether personal familiarity with people in the video affects recognition rate. They did the experiments with 20 students who knew people in videos, 20 other students unfamiliar, and 20 police officers experienced in the field of forensic investigations, but are unfamiliar with the subjects. They concluded through their experiments that individuals who are familiar with the targets performed very well at identifying them, while individuals unfamiliar with the targets performed very poorly along with police officers who performed as poorly as unfamiliar students.

Due to recent advances in technology and machine learning models being proposed, face recognition using a machine outperforms in many cases the performance of humans in the ability to identify people using face images [123]. This certainly helps in automating the identification process in recognizing face images collected using surveillance cameras and solves a problem that many current surveillance systems have, i.e., they are mostly used as recording machines. Such that if an incident occurs, cameras are used for analysis after

and not as part of an integrated warning system if an unusual behavior occurs in the image frames. A modern surveillance system is expected to do real time analysis on images it get and not just do basic object detection and tracking. But also to interpret object behavior and warn security officials of any security breach on the spot, and hence avoiding any more danger.

Research in video surveillance systems took a step towards making these systems automated in analyzing and processing video images in real time, overcoming the manual monitoring of security personnel process. Identifying people in a surveillance video cameras is an important task of face recognition where many institutes need systems for the purpose of access control, security monitoring, etc. Identifying a person's identity using surveillance cameras is challenging due to the variety of factors involved in the identification process, including the background environment, person's motion, variable lightening, and face visibility and detail exposure.

Face recognition has been extensively studied over the past decade to improve the performance and applicability of face recognition. Where early research efforts on face recognition [161, 4] focused on identifying people in frontal face images taken in a controlled environment where the background, pose, illumination are all pre-defined and set. These methods extract local descriptors of the image based on pixel intensity. Several other methods [81, 28, 56] were later designed to improve performance accuracy of face recognition in frontal controlled environment setting. The problem of face recognition then shifted to identifying people in uncontrolled environments (wild) where face images are collected outside a lab environment. Recent research work [141, 166, 149, 123] applied face recognition by first extracting features of given faces using convolutional neural networks (CNN) and then applied distance measures to compare face images for identification.

Identifying people's images to check if the person is suspect or not is one of the most important tasks for an automated surveillance system that applies face recognition. However, in real world scenarios such as at airports, military areas, diplomatic and official regions,

street blocks, etc. people with criminal intent can pass and perform criminal activities in a location where the criminal’s face may be detected in the surveillance cameras but authorities don’t have this person marked as a suspect. It is important for security surveillance systems to identify unknown people and try to get more information about a given person to try and predict if he/she is safe or dangerous to take appropriate precautions, as many crime incidents happen where authorities don’t have the person marked as a suspect.

In this work, we propose a security surveillance system that acts as an early warning system to detect from camera images not only suspects and known people, but also to apply further investigation on unknown people passing through a location that might be dangerous. In our system, we collect more information about unknown people passing through a scene by matching the person under investigation with his/her social media profile and then applying further analysis on his/her posts to conclude if the person is a potential suspect or not. The goal of this system is to help security officers in crime forensic to get more information about people to assist them so that they can take decisions on the spot.

The rest of this chapter is organized as follows. Section 9.2, covers our proposed security surveillance system with detailed information about how each part works so that we can perform person identification on images and identify potential suspects who were unknown before. We describe the dataset we used to evaluate the performance of the system to correctly identify people in Section 9.3. Section 9.4, provides experimental results. Section 9.5 concludes the chapter.

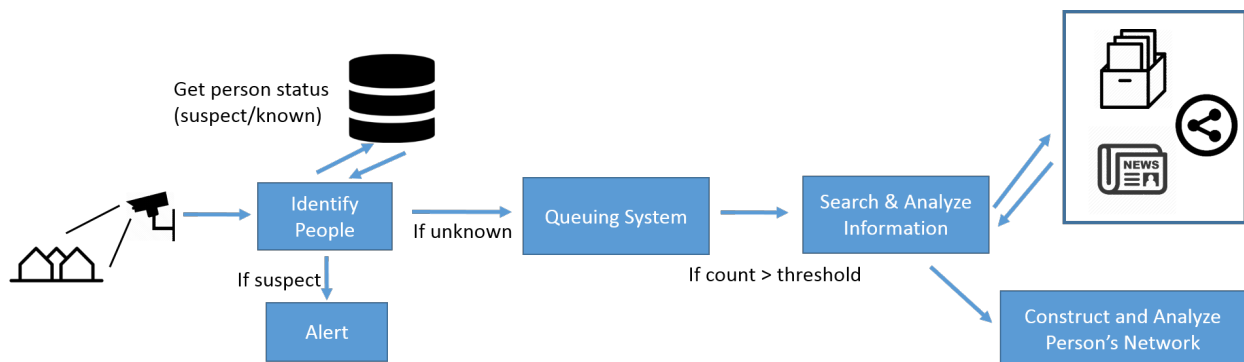


Figure 9.1: The overall methodology

9.2 Methodology

The overall methodology of the proposed surveillance system is shown in Figure 9.1. First, security cameras are used to monitor an area of interest whether it is a campus location, military base, street, etc. The collected images from these cameras are then fed to the "Identify People" process which applies machine learning techniques to extract face images which will then be compared with a database to check whether the person in the image is known or is a suspect. If the person's identity was inferred to be a suspect by the face recognition model, an alert will be raised by the system for a security officer to be warned in order to take a corresponding action. If the person is identified as known (safe) then no further action is taken by the system. Whereas if the person's face image was not matched with the known/suspect database, the face image is added to the "Queue". The idea behind the queue system is that not for every person passing through a scene the system should apply further investigation to collect information about him/her. Instead, an image is added to the queue. Every person in the queue has a counter which is incremented every time the person passes through the scene. More details about the implementation of the queue will be explained in Section X. The "Search for Information" module will be used to further investigate every person whose counter passes a certain threshold. This module takes the person's image and compares it with existing social media profiles in order to match the person with his/her profile. If the social media of the person is successfully matched, analysis of the social media posts and the person network is applied to classify this person as a potential suspect or as a safe person. Details related to the implementation and functionality of each module are explained below.

9.2.1 Camera System

In our system, we make it possible to connect a local security camera on the network for live analysis. We also provide a functionality to get already recorded camera feed and import it

to our system so that we can apply the same analysis on a crime incident.

Upload & Analyze Images

The interface is divided into two main sections. The top section, 'Add New Dataset', features a text input field containing 'ChokePoint Sample', a larger text area with 'ChokePoint Dataset sample for validation', a 'Choose Files' button, and a camera icon. The bottom section, 'Uploaded Datasets | Under Analysis: (1) datasets', displays two dataset cards. The first card, 'Stream1 (120)', is green and shows '13/9/2018 stream on portal 3'. The second card, 'Stream2 (200)', is blue and shows '12/9/2018 stream on portal 3'.

Figure 9.2: Interface for security officers to upload or analyze a video stream

Figure 9.2 shows how the user can use the system to upload the content of a camera or connect to a live camera. The officer can upload several video sets or connect to a live camera feed from the network. The officer can also go back and check previously analyzed video stream where the video sequence is shown and people passing are labeled with their face image and detected identity. For our evaluation purposes we use the Chokepoint dataset which contains images collected using surveillance cameras.

9.2.2 Identify People

After connecting a camera to the system or uploading camera images to the system, the system will then run through the images to identify people present in the frames. There are two processes in identifying people, first the face image should be extracted and located in the frames, this process is referred to as face detection. The second process is to use the extracted face to match with other faces of suspects and known people available in the

gallery. Matching face images is known as face recognition. Details on what is used for both face detection and recognition is described in the sub sections below.

Face Detection

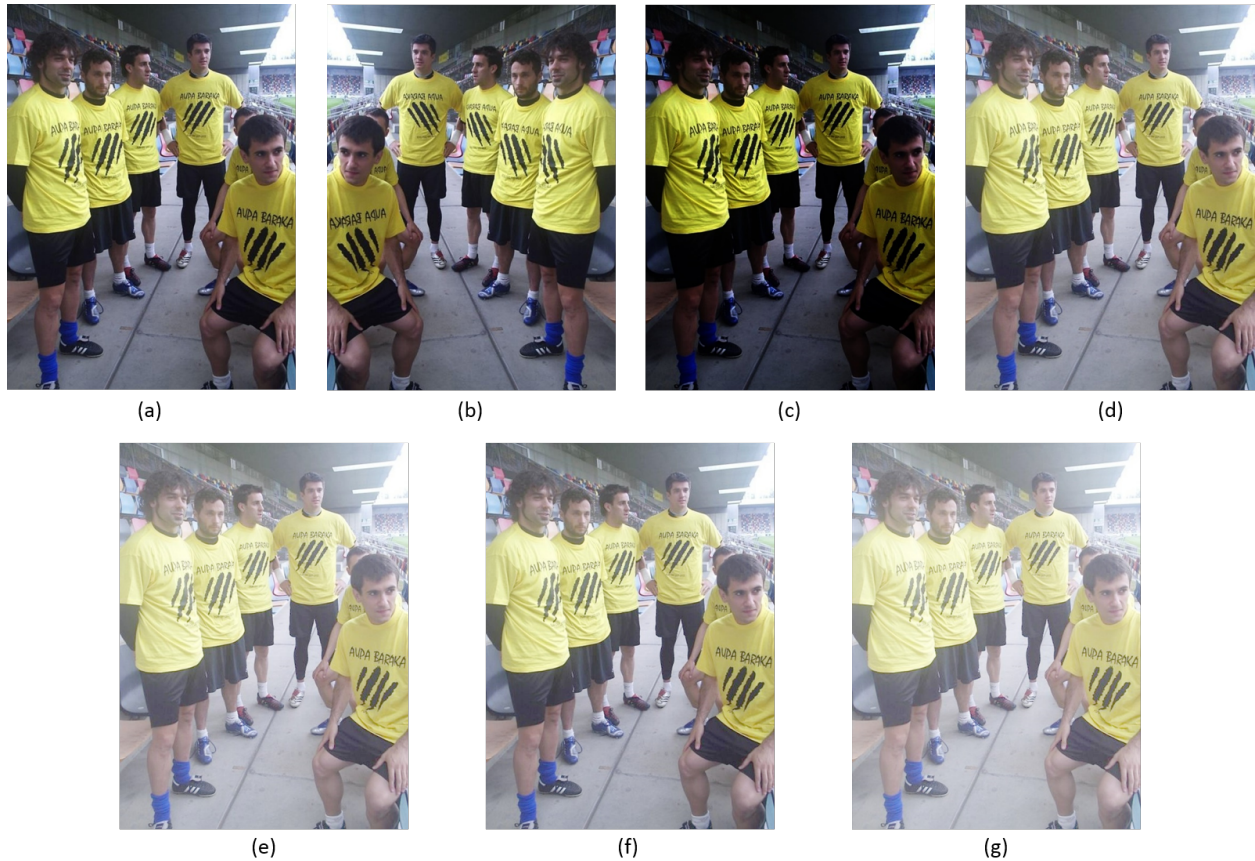


Figure 9.3: Image manipulation for face detection training. (a) shows an original image in the WIDER dataset. (b) shows a rotated image of (a). (c) - (g) shows the image at gamma levels (0.5, 1.5, 2, 2.5, 3)

For our system, face detection should be a fast process such that it can be applied in real time analysis on a camera stream. It should be also accurate in a multi-view environment as people passing through can have their faces in different poses. For this purpose, we trained a CNN model to perform face detection on the WIDER face dataset [174]. The WIDER dataset contains thousands of face images collected under extreme cases varying scale, pose, occlusion and illumination of faces. For the CNN model, we used the recently proposed

MobileNet-v1 [67] network architecture for training using the WIDER dataset. MobileNet-v1 is designed using depth-wise separable convolutions providing drastic decrease in model size and training/evaluation times while performing better in detection making it a perfect architecture for our purpose. To get better detection accuracy, we pre-process every image in the training data of the WIDER dataset before we train our detector model. We generate four different pictures for every image in the dataset and then feed the images to the learning model. The four different image types we generate from every image are shown in Figure 9.3.

Face Recognition

For this work, we opted to use two different feature extraction techniques based on training CNN and compare which method is better for face recognition in surveillance camera type of images. The first extraction technique we use is from the popular OpenFace's [8] implementation of the FaceNet feature extraction technique. While FaceNet have trained their neural network model with over 200 million private images not available for the public, OpenFace trained their model with around 500 thousand images from public datasets and they provide their trained model for research purposes. OpenFace implements the triplet loss learning suggested by the FaceNet work in the feature learning process. The second feature extraction technique we use is the one we created by training our own CNN. The model architecture we used is the Inception-Resnet-v1 [152] network architecture and trained on the MS-Celeb-1M [52] face dataset. The training implementation also follows the method as in FaceNet [141] using the triple loss learning technique for our training.

9.2.3 Alert

After the person face is identified from the previous step, then the person will be classified as either known, unknown or a suspect. If the person is known then the system does no further action. But, if the person is unknown then the face image is sent to the Queue System for further analysis., An alert is generated by our system to the security personnel to act on the

spot in case the person is identified as a suspect for being listed in the database.

9.2.4 Queuing System

When a person face image is classified as unknown, his/her face image is added to our queuing system. The purpose of the queue is to monitor the trend in which random people are arriving to the queue. For a random/unknown person who passes a number of times in a scene, we should apply further investigation to gather more information about the person to check if he/she is safe or not. The number of passes of the same person is deemed suspicious depends on the situation and specific circumstance where the surveillance camera is set. For-example. if the surveillance camera is deployed in a military base, then the threshold for the number of times a random person passes should be lower than when a camera is deployed on a public street. In general, this threshold is set by security experts depending on the location.

Not every unknown person will have his/her face analyzed because in real life scenarios many people pass via a scene and never appear again. That is why a threshold is defined by a security expert to set a reasonable number of times that a person has to pass in order for our system to apply further analysis. Also, the queue cannot store every person face image that passes especially if the surveillance system is deployed in a busy area where large number of people pass. In such cases most people in the queue only pass one time and take unnecessary space in the queue. The queue size should have an upper bound set by the security expert depending on the monitoring location.

When the queue is full, a face replacement policy should be implemented such that the new face can enter the queue and one face instance will be removed from the queue. The decision of which queue element to discard is up to the replacement policy to decide on. Many replacement policies have been proposed for queuing systems coming from web cache replacement policies which are mainly used to manage cache content for web pages. Cache is an important aspect of the web to reduce loading times for web pages. A cache server

stores Web objects such as HTML pages, images, and other files locally to be used for future requests. As the cache size of a browser is finite, a cache replacement policy is needed to manage cache content. The goal of the replacement policy is to make the best use of available resources such that we don't want popular items to leave the queue. Even recently added items might become popular in the near future. In our case, we treat face images like cache objects of a web page. Traditional replacement policies such as least recently used (LRU) and least frequently used (LFU) were proposed. More recent proposed solutions [77, 96, 95] provide only slight improvements and variations of these early methods. But, actually there is no single policy that performs best in all environments. It depends on the application in place [168]. The LFU method is a frequency-based policy which uses the count of an object solely to decide where the item will rank in the queue. The higher the count of an item is the higher it is in the queue. Items with the lowest count will leave the queue when new items arrive. The other type of cache replacement policy is LRU which where items that have been used least recently will be removed from queue regardless of how popular they were.

There are problems with both LRU and LFU. LRU doesn't take into account the usability of the item where the most accessed object can be evacuated from the queue. While the problem with LFU is that it ignores the latest item accessed which can be evacuated right after its addition because of its low frequency and may not take the chance to increase its value. A better approach will be combining both the frequency and the recency of an item for the removal policy. For this purpose, we implemented our own replacement policy for our queuing system to consider both the recency and the frequency of a person passing. This way, every time a person passes we increase his/her frequency and note down the time of the passage. When the queue is full, the removal policy is not only based on the person with least frequency score but based on the time period the item has been in the queue unreferenced. For every x mns passes (ex:30 to be set by security expert), the item loses one frequency score so that the most recent item won't be removed.

The algorithm to add a new face item to the queuing system is shown in Algorithm 1. We describe the variables and functions in the list below.

- x : Face image of the unknown person.
- $element$: An element in the queue refers to a face image of an unknown person who already has a count and time of arrival in the queue.
- $getDist(image, element)$: A method that takes as input a face image and an element from the queue to calculate the distance between the two feature vectors of the face images.
- $distThreshold$: A variable that decides whether a face image belongs to the same person or not. If the distance between two face images is less than $distThreshold$, then the two images belong to the same person. (we set the threshold to 1.1 in our experiments)
- $getCurrentTime()$: A method that gets the current time, used to record the entry of a face image or to update the last time an element got referenced.
- $addCount(element, time)$: A method that takes as input an element and increments the hit counter of that element. The method also takes as input the current time to update the last time this element was referenced.
- $maxCount$: A variable defined by the security expert depending on the environment where the surveillance camera is installed. Further investigation is applied on an element if its count is greater than this variable.
- $investigate(element)$: A method that applies further investigation to collect information about the input element. The element is also removed from the queue to make space for new elements. Further details on the investigation process can be found in Sections 9.2.5, 9.2.6.

- *removeElement()*: Removes an element from the queue to make space for a new item. The process of removal takes into account the count of an element
- *addElement(image,time)*: A method that adds a new element to the queue with count equal to 1 and time equal to the current time.

Algorithm 1 Add Face to Queue

```

1: procedure ADDFACE(x)
2:   for element in queue do
3:      $dist \leftarrow getDist(x, element)$ .
4:     if  $dist < distThreshold$  then
5:        $currentTime \leftarrow getCurrentTime()$ .
6:        $count \leftarrow addCount(element, currentTime)$ 
7:       if  $count = maxCount$  then
8:          $investigate(element)$ .
9:       end if
10:      return
11:    end if
12:  end for
13:   $currentTime \leftarrow getCurrentTime()$ .
14:  if queue is full then
15:     $removeElement()$ .
16:  end if
17:   $addElement(x, currentTime)$ 
18: end procedure

```

9.2.5 Search for Information

If an unknown person in the queue reached maximum hits, his/her image is then used to identify the person using the social media. Terrorists and criminals has been shown over several studies [85, 87] to use social media accounts to plan or to discuss criminal activities. For our system, we have collected thousands of Facebook profiles associated with people's profile picture. Facebook's Graph API ¹ provides several functions to access Facebook's social graph. We used the API and generated random Facebook user IDs which gave us access to information of random people from Facebook. From this data of social network

¹Facebook Graph API: <https://developers.facebook.com/docs/graph-api/Accessedon9/6/2018>

(khobaib hussain) - Person Investigation

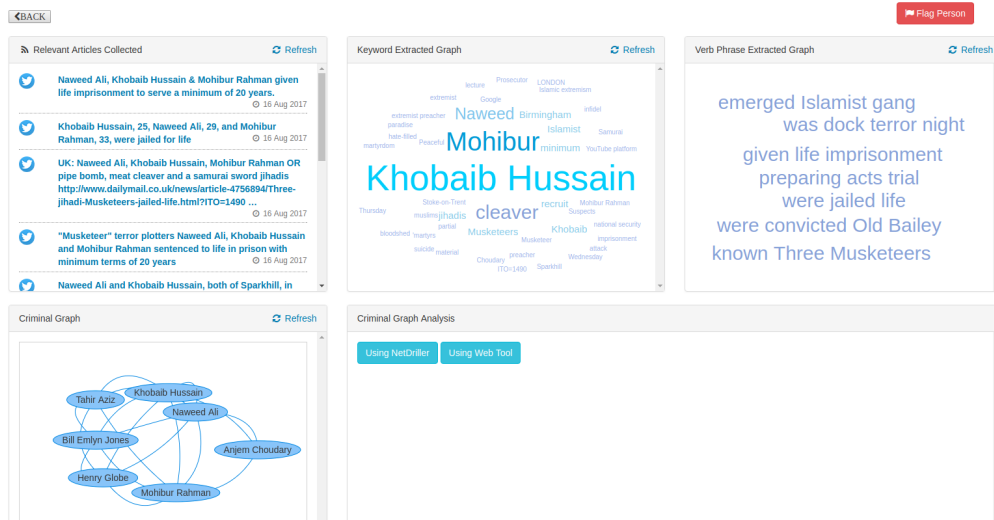


Figure 9.4: Person Collected Information Details

profiles who we collected, we built an SVM classifier model based on the collected images. Thus when an unknown person image is collected from the surveillance camera and passed thorough the queue system, we compare the collected face with the social media profiles we have. If the face is matched with the social media profile, then we use the name of the social media profile to search more about the person using Twitter and traditional news. Figure 9.4 shows an example of the page that an analyst is given once an unknown person has exceeded the pass number. We then get the person name from the social media profiles collected and then extract tweets, posts and news articles that mention this person name. We then automatically apply text analysis on the collected information to provide the analysts with relevant keywords and verb phrases that this person is mentioned in.

Keywords are extracted from the text using a method proposed by Mihalcea et al. [114] where they developed a term extractor called TextRank, which is a graph ranking based method applied on words as vertices in order to determine the importance of the words. For the tweets, however, we also used the hashtag as a topic of the tweet because social media posts are limited in number of words. After providing the analyst with the person name, articles mentioned with links, keywords, verb phrases and criminal network, the analyst can

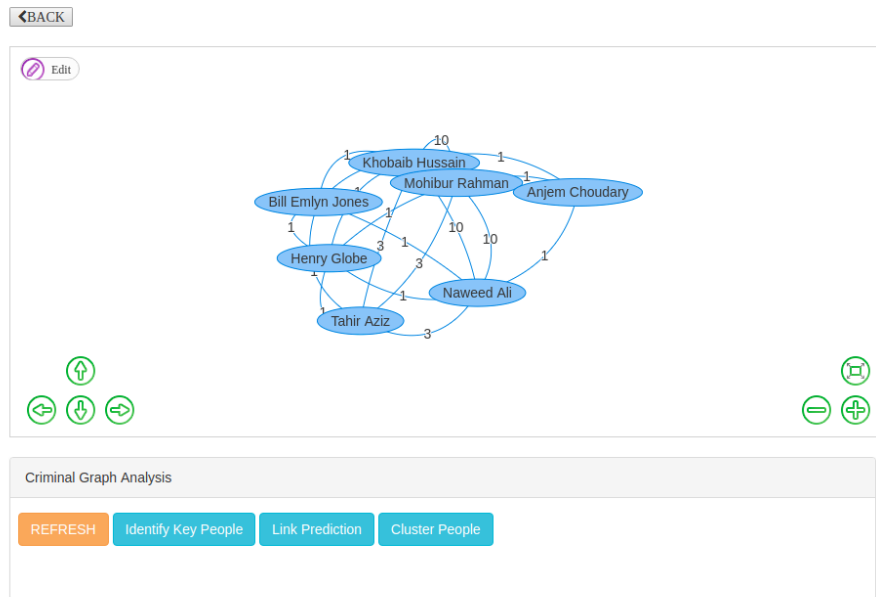
finally decide whether to flag the person. This will lead to the person being added to the suspect list if deemed so.

9.2.6 Construct and Analyze Person's Network

After deducing the name of the unknown person from the search for information process, we can build a social graph of the person to check his/her network if it is safe or not. A social graph is defined such that nodes in the graph represent people and edges between these people indicate an interaction or relation between them. Research efforts, e.g., [9, 29], have been done to populate a criminal graph of a person from his/her name using news articles. The procedure is done by first searching for articles in which the person is mentioned and then apply named entity recognition (NER) to detect all other people names mentioned in the same articles. For every person name existing in the article, a vertex is added to the person social graph. For all people mentioned in the same article, an edge is added between them representing an interaction. If two people are mentioned in more than one article, then a weight is added to their edge to show the number of articles they were mentioned in. Modeling the social interactions and mentions in the text is an important mechanism for analysts as it allows network visuals to see a criminal network and different interactions in a clear way. This leads further investigation of other people in a network by applying network analysis techniques. In case a person has links with suspects then the analyst will cluster this person as a potential suspect and later appearances of this person in the surveillance cameras will raise an alert.

Figure 9.5 shows the network graph of "Khobaib Hossain" as shown to the analysts. First, the names of the related people are extracted from the collected text where "Khobaib Hossain" is mentioned. We use Stanford NLP toolkit to extract names from the text, the tool uses the method described in [43] to train a NER model using a combination of CRF sequence taggers trained on various text. The graph is then populated from these names and mentions in same articles.

(khobaib hussain) - Graph Analysis



We provide several graph analysis techniques for the analyst to apply further investigation on the graph. These are useful especially in large networks. We chose four different network analysis techniques in our system. These will provide enough information for analysts in their investigation.

- *Identify Key Nodes*: This process aims to identify what are the major and most influential criminals/nodes in the criminal graph. In graph theory, centrality indicators identify the most important vertices in the network. There are three main centrality measures that are calculated for every node in the graph. Degree Centrality which is defined as the number of links/edges a node has. A high degree centrality means that a criminal is mentioned and involved with many other criminals. Closeness Centrality is defined as the average length of the shortest path between the a node and all other nodes in the graph. A high value of the closeness centrality refers to a criminal who is at the center of the network where he can easily reach all other criminals in the network. Betweenness Centrality is defined as the number of times a node acts as a

bridge along the shortest path between two other nodes. Nodes with high betweenness value are the ones who have more control over information passing between other nodes. Removing these types of criminals will cutoff the linkage of the graph because other nodes rely on these criminals to reach other nodes. Eigenvector Centrality is a measure that identifies the most important and influential nodes in the network. The importance of the node comes if this node is linked to by other important nodes. To calculate such network measures, the analyst can click on the identify key people button to select his/her metric to calculate.

- *Adjust Network*: After identifying key nodes in the network using the previous process, it is essential to give an analyst the option to view the criminal network and modify the existence of some nodes depending on their importance to see what effect they have on the network. By removing criminals from the network, the analyst can look into how to disrupt the criminal network structure so that they can arrest these persons to possibly collapse the network. This can be done using the edit button on the graph to add/delete/update nodes and edges.
- *Cluster Nodes*: We provide hierarchical clustering which aims to show the criminal network as a set of communities. It is essential for an analyst to view the network as a set of communities because it generally infers what criminal groups reside within a network. We provide a hierarchical community view as a functionality for the analyst using our framework which provides the option to zoom in and out of the network using community detection algorithms and display each community as a node in a zoom out mode and display each node as a community in a zoom in mode. This is provided by the cluster nodes button.
- *Link Prediction*: Link prediction refers to the problem of mining what links between nodes in the criminal graph created may exist without our knowledge or which new interactions among its members are likely to occur in the near future. Finding hid-

den links in a criminal graph is very important because we can predict an interaction between two criminals by analyzing the graph structure using social network analysis techniques. Many link prediction methods have been proposed for this purpose, e.g., [118, 1, 2]. This analysis technique is provided using the link prediction button.

9.3 Dataset

To evaluate our system to identify and classify people into suspects, known, and unknown, we chose to use the Chokepoint dataset [169] for our experiments. The Chokepoint dataset is a video based dataset designed for experiments on identifying and verifying people's identity under real-world surveillance conditions. The dataset is collected using an array of 3 cameras above several portals to capture people walking through each portal in different face views (frontal/profile). The dataset consists of 25 subjects (19 male and 6 female) in portal 1 and 29 subjects (23 male and 6 female) in portal 2. In total, it consists of 48 video sequences and 64,204 face images. Each set has variations in terms of illumination conditions, pose, sharpness and has been taken in different times of the day to make the dataset more challenging. Sequence names are unique and correspond to the recording conditions, where P, S, and C stand for portal, sequence and camera, respectively. Further, E and L indicate subjects either entering or leaving the portal, respectively. The dataset environment of surveillance cameras is similar to those observed at airports [50] where individuals pass in a natural free-flow way in a narrow corridor.

A sample of the dataset is shown in Figures 9.6 and 9.7. Figure 9.6 shows gallery images of photos taken for every subject who passes through the portals. There are two images taken for every person, one of them with neutral face while the other one with a smile. These images are used as database images in our system to specify people as suspects or known from these images. Figure 9.7 shows a sample of the video images collected from different ports and the three different camera angles for every sequence; it is used for validating the

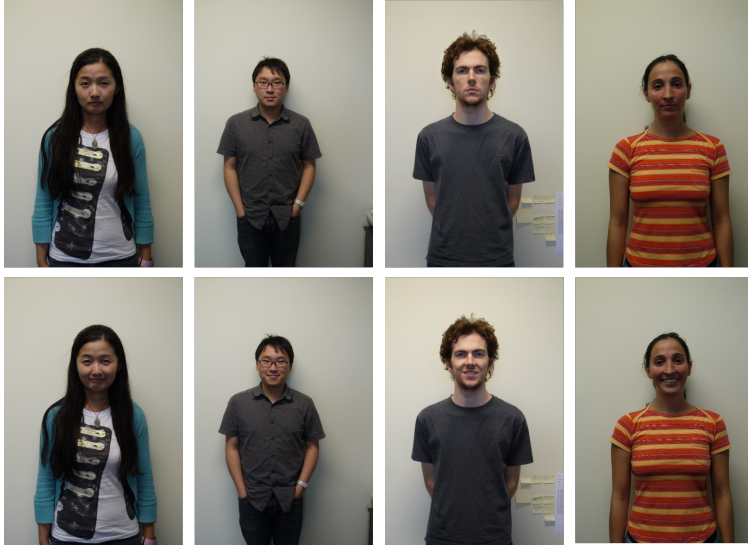


Figure 9.6: Sample of gallery images (smile and neutral) of the Chokeypoint Dataset



Figure 9.7: Sample of the video images collected from the Chokeypoint dataset

system.

9.4 Experiments & Results

For our experiments, we used face images collected in the gallery settings from the Chokepoint dataset as our database images. We did two separate experiments, in the first one we used gallery images consisting of only neutral face images. In the another experiment, for every person there are two gallery images (neutral and smile images). For our experiments, we considered people (1, 2, 3, 4, 5, 6) as known while people (7, 9, 10, 11, 12, 13) as suspects. The rest of the people are deemed to be unknown and gallery images of these people were not used.

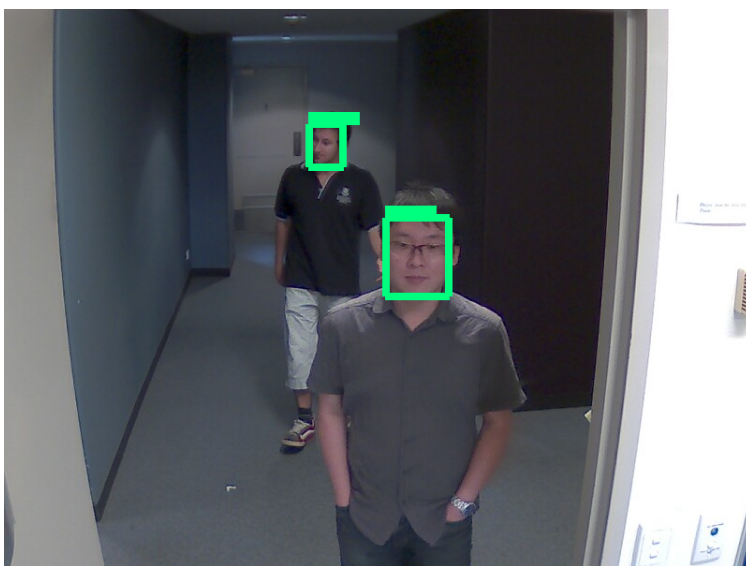


Figure 9.8: Sample of the output of our face detector.

For video images in the dataset, we didn't apply our face detection on images. The reason is that in the ground truth of the dataset, they don't include face images of far people. Thus, we couldn't evaluate our recognition accuracy with the ground truth as shown in Figure 9.8. Figure 9.8 shows a sample of the face detection process of our model; it reports two faces detected in the image. The ground truth of the dataset however, defines only one face, namely the frontal one. Instead, for every image frame in the ground truth that shows a

person, we applied our different feature extraction technique on the face.

First, we extracted feature vectors of all face images from camera feed and gallery images. We used OpenFace model and our trained model to collect two separate feature vectors for every image to compare which feature extraction model works better in a surveillance environment. Recall that we had specified people with IDs (1,2,3,4,5,6) as known, people (7,9,10,11,12,13) as suspects and the rest as unknown. We then compare the accuracy of recognition in two different settings, one with only one face of a person in the gallery (neutral face) compared to when we have two faces in the gallery for every person (neutral and smiling face). After setting up the database gallery, we ran our tests on each portal and sequence with the three different cameras available. Tables 9.1, 9.2, 9.3 show accuracy results of using the different recognition models to classify people passing in portal 1 sequence 3 into known, suspects and unknown people. On each row, we have the type of the model we used for accuracy; it is either OpenFace or our model with each having neutral face or neutral and smile face in the database gallery.

Table 9.1: Accuracy results of the known, suspects and unknown people using the P1L-S3-C1 camera sequence

P1L_S3_C1	KnownAcc	SuspectAcc	UnkownAcc
openFaceNeutral	64.07	72.57	65.10
openFaceNeutralSmile	66.12	72.93	67.96
ourNeutral	81.82	80.70	78.11
ourNeutralSmile	84.85	82.67	80.47

Table 9.2: Accuracy results of the known, suspects and unknown people using the P1L-S3-C2 camera sequence

P1L_S3_C2	KnownAcc	SuspectAcc	UnkownAcc
openFaceNeutral	64.53	66.30	66.32
openFaceNeutralSmile	69.19	67.00	71.28
ourNeutral	83.05	81.40	78.34
ourNeutralSmile	86.44	83.60	80.41

We only show accuracy results of camera sequence of P1L_S3 because we got similar accuracy results for all other camera sequences. As shown in Tables 9.1, 9.2, 9.3, accuracy results

Table 9.3: Accuracy results of the known, suspects and unknown people using the P1L-S3-C3 camera sequence

P1L_S3_C3	KnownAcc	SuspectAcc	UnkownAcc
openFaceNeutral	64.85	66.74	66.28
openFaceNeutralSmile	75.25	67.95	70.00
ourNeutral	81.01	80.30	78.06
ourNeutralSmile	82.28	81.20	80.59

across different camera angles for the same sequence don't hugely affect the recognition rate of our trained neural network model. Camera 2 shows slightly better results. Also we show that our approach works better than OpenFace feature extraction technique by almost 15%. Further, using two face images in the gallery slightly enhances recognition rate. This means that even using one face image in the gallery produces good result.

9.5 Conclusions

We present in this chapter an early warning system that integrates face recognition, social media and text analysis for recognizing people in surveillance camera environments. Monitoring people in surveillance systems is being used for security purposes where security officers have to manually watch suspicious people or activities. Many scenarios happen when security officers can't recognize well people in a surveillance environment as shown in many previous research efforts, e.g., [22, 25, 60, 23, 113]. Even a person who passed in front of a camera might be a potential suspect who the system doesn't know about. We propose a system that first takes as input image frames from surveillance cameras. These images are then used to locate and recognize people based on their faces. The system maintains a database of known and suspicious people to raise an alarm for security officials when a suspect is shown in a scene. When a person identity is unknown, his/her image is added to a queuing system. The same people passing a number of times will be then forwarded for further investigation to know who they possibly are and if they are dangerous. A person face image that has been forwarded by the queuing system will then have his/her face com-

pared with social media profile images collected from Facebook. If the social media profile is found in the database, the name of the person is used to collect more information and text from news and other social media profiles. The result is used for text analysis which is applied to get important sentences and people mentioned with a given person. This leads to construct social graph of the person. Using our tool, the analysts can then use a variety of network analysis tools to identify important people in the network and check if this person is suspicious or not. We show by the conducted experiments that using our trained neural network provides good accuracy levels in recognizing people compared to other approaches in a surveillance camera environment.

Chapter 10

Summary, Conclusions and Future Research Directions

Criminology and terror received considerable global attention post September 11 attacks which hit severely in the United States, though many regions and countries in the Middle East, Africa, Europe and Asia have been suffering from organized criminal and terror attacks for decades. Even the United States suffered from several attacks which were at small scale compared to September 11. These include attacking embassies, killing government officials, innocent citizens, destroying the economy, etc. Unfortunately, yet there is no globally agreed upon identification of criminals and terrorists. In many occasions some called the attackers freedom fighters, while others classified them as dangerous terrorists. This is because they play role in the global conflicts between countries, ideologies and blocks. Thus, it is not an easy task to find a global remedy. However, as has been demonstrated by the research conducted for this dissertation, advanced technology could be utilized to develop some preventive solutions capable of identifying potential criminals and terrorists. This is achievable by employing sophisticated computing techniques to gather and analyze various types of interrelated data captured from available sources and in a wide variety of formats ranging from plain text to images. Additionally, it is necessary to cope with dynamic data which

may become available incrementally and from a variety of sources.

Identifying criminals and detecting any potential acts they may be involved in is very important and critical to ensure safety of humans, the infrastructure, the economy and the environment. Fortunately, this thesis achieved the objective of developing an early warning system which tries to detect and identify criminals before they commit criminal/terror acts. The availability of data is essential to accurately detect and recognize criminals. In the developed system, we make use of a wide range of data sources (e.g., social media, news, police reports, surveillance cameras) to detect criminals. The outcome successfully guides the construction of a criminal profile together with the individual network to facilitate the work of criminal analysts and raise timely alarm when necessary.

Our system identifies and recognizes criminals or potential suspects using video surveillance cameras by implementing a queuing system which employs face recognition. The system also utilizes text mining techniques to extract a criminal profile and link criminals involved in the same incidents to build a criminal graph. After building a criminal network/graph, several graph analysis techniques are applied to identify key figures in an incident and to guide investigators in their effort to analyze and make sense of the captured networks. We also provide the ability of clustering images which are collected from a crime scene or event to classify each person's images into one group for analysts to view all people involved and their individual involvements.

10.1 Conclusions

There are several lessons learned from the study described in this thesis. The domain tackled in this dissertation is complex and could not be handled in one step and based on one perspective. Instead, to produce a successful solution the problem has been well investigated and divided into components. Identifying the particulars of each component individually could produce a corresponding sub-solution. Then all sub-solutions could be integrated into a uni-

fied comprehensive system which satisfies the target of avoiding or preventing terror/criminal incidents by issuing timely alerts. Indeed, the findings described in this dissertation continue to be at the core of current research in homeland security. I am delighted to have developed a fully working early warning system which will be greatly beneficial for Canada and beyond.

We contributed to the early warning criminal research by designing a criminal framework capable of handling data which could be captured from a wide range of sources which are directly or indirectly related to the investigated domain. After the analysis, we get criminal profiles and the corresponding network for further analysis. Our framework also does its early warning by implementing a queuing system to detect people passing in an area covered by surveillance cameras.

Clustering is included as a a major component of the system to analyze people and their involvement in a crime scene. Clustering helps us in focusing the investigation further to concentrate on specific persons directly related to a suspicious incident.

Images captured by surveillance cameras are most of time not easy to process directly. To overcome this, we contributed by enhancing face recognition rate on images taken in an uncontrolled environment. We also conducted studies on how face recognition rates differ from frontal to profile views. Capturing a face is not beneficial unless it is well utilized in as part of a comprehensive analysis. For this purpose, we improved the functionalities of NetDriller which is a social network analysis tool under development by our research group for over a decade. We are proud to have implemented new functions to further analyze criminal networks. We demonstrated how it is possible to study the influence of network leaders at different levels, how the network will restructure after removing certain leader, identifying leaders who should be excluded from the network to destroy the whole network by splitting it into small isolated islands who may not be effective at all. We designed a new link prediction algorithm that improves the detection of hidden/future links for criminal network analysis. All these contributions combined have produced a sophisticated and powerful system which could help in informative decision making.

10.2 Future Research Plan and Directions

This research project has been an entertaining experience. Every time I considered a new perspective I got more excited. However, the process had to be terminated at a certain point to allow me to submit and defend my dissertation leading to Ph.D. in Computer Science. However, this research domain may be described as an area which need tremendous effort and attention for improved contributions that may help in realizing a fully preventive system, though it is not an easy target to hit. What has been achieved in this dissertation has paved a roadmap for future research directions which can build on our discoveries to advance the research and development components further in a number of directions.

It is important to deploy the system in a real world environment and work with crime institutions and governments to analyze large networks and criminal records to help in preventing potential attacks from occurring. Unifying identity across various social media platforms is a vital issue which has recently received increased attention. Developing a robust identity unification mechanism and integrating it into the developed system will improve its power. It is also essential to turn the system into language independent by considering social media postings in various languages and unifying them into English as the common language understandable by most people. This requires some rigorous translation models which may directly utilize one of the available successful translators, like Google Translator. The outcome from a translator may need some fixing by using some linguistic rules. Another area to investigate is conflict resolution in case one perspective reports a suspect as a potential terrorist or criminal while some other perspectives reflect a different perspective. This conflict might be due to data collected from the archives of different countries. Each country has its own considerations in classifying people, though there are some common factors agreed upon globally. However, some countries might have their own hidden agendas which are hard to discover and this adds a higher degree of complexity to the planned extensions to the developed system.

Bibliography

- [1] Lada A Adamic and Eytan Adar. Friends and neighbors on the web. *Social networks*, 25(3):211–230, 2003.
- [2] Salim Afra, Alper Aksaç, Tansel Özyer, and Reda Alhajj. Link prediction by network analysis. In *Prediction and Inference from Social Networks and Social Media*, pages 97–114. Lecture Notes on Social Networks Series, Springer, 2017.
- [3] Charu C Aggarwal and ChengXiang Zhai. *Mining text data*. Springer Science & Business Media, 2012.
- [4] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. Face description with local binary patterns: Application to face recognition. *IEEE transactions on pattern analysis and machine intelligence*, 28(12):2037–2041, 2006.
- [5] Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed Zaki. Link prediction using supervised learning. In *SDM'06: Workshop on Link Analysis, Counter-terrorism and Security*, 2006.
- [6] Rabeah Al-Zaidy, Benjamin CM Fung, Amr M Youssef, and Francis Fortin. Mining criminal networks from unstructured text documents. *Digital Investigation*, 8(3):147–160, 2012.
- [7] Pablo Fernández Alcantarilla, Adrien Bartoli, and Andrew J Davison. Kaze features. In *Computer Vision–ECCV 2012*, pages 214–227. Springer, 2012.

- [8] Brandon Amos, Bartosz Ludwiczuk, and Mahadev Satyanarayanan. Openface: A general-purpose face recognition library with mobile applications. Technical report, CMU-CS-16-118, CMU School of Computer Science, 2016.
- [9] Tarique Anwar and Muhammad Abulaish. A social graph based text mining framework for chat log investigation. *Digital Investigation*, 11(4):349–362, 2014.
- [10] Remy Arulanandam, Bastin Tony Roy Savarimuthu, and Maryam A Purvis. Extracting crime information from online newspaper articles. In *Proceedings of the Second Australasian Web Conference-Volume 155*, pages 31–38. Australian Computer Society, Inc., 2014.
- [11] Stephanie Alice Baker. From the criminal crowd to the “mediated crowd”: the impact of social media on the 2011 english riots. *Safer communities*, 11(1):40–49, 2012.
- [12] Mathieu Bastian, Sebastien Heymann, Mathieu Jacomy, et al. Gephi: an open source software for exploring and manipulating networks. *Icwsn*, 8:361–362, 2009.
- [13] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. *Computer vision–ECCV 2006*, pages 404–417, 2006.
- [14] Nesserine Benchettara, Rushed Kanawati, and Celine Rouveirol. Supervised machine learning applied to link prediction in bipartite social networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2010 International Conference on*, pages 326–330. IEEE, 2010.
- [15] Manuele Bicego, Andrea Lagorio, Enrico Grosso, and Massimo Tistarelli. On the use of sift features for face authentication. In *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW’06. Conference on*, pages 35–35. IEEE, 2006.
- [16] Steven Bird. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL*

- on Interactive presentation sessions*, pages 69–72. Association for Computational Linguistics, 2006.
- [17] Steven Bird and Edward Loper. Nltk: the natural language toolkit. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, page 31. Association for Computational Linguistics, 2004.
- [18] Kaumalee Bogahawatte and Shalinda Adikari. Intelligent criminal identification system. In *Computer Science & Education (ICCSE), 2013 8th International Conference on*, pages 633–638. IEEE, 2013.
- [19] Kalina Bontcheva, Leon Derczynski, Adam Funk, Mark A Greenwood, Diana Maynard, and Niraj Aswani. Twitie: An open-source information extraction pipeline for microblog text. In *RANLP*, pages 83–90, 2013.
- [20] Stephen P Borgatti, Martin G Everett, and Linton C Freeman. Ucinet for windows: Software for social network analysis. 2002.
- [21] Michele A Brandão, Mirella M Moro, Giseli Rabello Lopes, and José PM Oliveira. Using link semantics to recommend collaborations in academic social networks. In *Proceedings of the 22nd international conference on World Wide Web companion*, pages 833–840. International World Wide Web Conferences Steering Committee, 2013.
- [22] Vicki Bruce, Zoë Henderson, Karen Greenwood, Peter JB Hancock, A Mike Burton, and Paul Miller. Verification of face identities from images captured on video. *Journal of Experimental Psychology: Applied*, 5(4):339, 1999.
- [23] Vicki Bruce, Zoë Henderson, Craig Newman, and A Mike Burton. Matching identities of familiar and unfamiliar faces caught on cctv images. *Journal of Experimental Psychology: Applied*, 7(3):207, 2001.

- [24] Vicki Bruce and Andy Young. Understanding face recognition. *British journal of psychology*, 77(3):305–327, 1986.
- [25] A Mike Burton, Stephen Wilson, Michelle Cowan, and Vicki Bruce. Face recognition in poor-quality video: Evidence from security surveillance. *Psychological Science*, 10(3):243–248, 1999.
- [26] Zhimin Cao, Qi Yin, Xiaoou Tang, and Jian Sun. Face recognition with learning-based descriptor. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2707–2714. IEEE, 2010.
- [27] Chi-Ho Chan, Josef Kittler, and Kieron Messer. Multi-scale local binary pattern histograms for face recognition. *Advances in biometrics*, pages 809–818, 2007.
- [28] Rama Chellappa, Charles L Wilson, and Saad Sirohey. Human and machine recognition of faces: A survey. *Proceedings of the IEEE*, 83(5):705–741, 1995.
- [29] Hsinchun Chen, Wingyan Chung, Jennifer Jie Xu, Gang Wang, Yi Qin, and Michael Chau. Crime data mining: a general framework and some examples. *computer*, 37(4):50–56, 2004.
- [30] JieMin Chen, Yong Tang, JianGuo Li, ChengJie Mao, and Jing Xiao. Community-based scholar recommendation modeling in academic social network sites. In *Web Information Systems Engineering–WISE 2013 Workshops*, pages 325–334. Springer, 2014.
- [31] Dongjin Choi, Byeongkyu Ko, Heesun Kim, and Pankoo Kim. Text analysis for detecting terrorism-related articles on the web. *Journal of Network and Computer Applications*, 38:16–21, 2014.
- [32] Sumali J Conlon, Alan S Abrahams, and Lakisha L Simmons. Terrorism information

- extraction from online reports. *Journal of Computer Information Systems*, 55(3):20–28, 2015.
- [33] Massimo Craglia, Robert Haining, and Paul Wiles. A comparative evaluation of approaches to urban crime pattern analysis. *Urban Studies*, 37(4):711–729, 2000.
- [34] Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research. *InterJournal, Complex Systems*, 1695(5):1–9, 2006.
- [35] Hamish Cunningham. Gate, a general architecture for text engineering. *Computers and the Humanities*, 36(2):223–254, 2002.
- [36] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, pages 886–893. IEEE, 2005.
- [37] Darcy Davis, Ryan Lichtenwalter, and Nitesh V Chawla. Multi-relational link prediction in heterogeneous information networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 281–288. IEEE, 2011.
- [38] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, pages 1486–1494, 2015.
- [39] Leon Derczynski, Diana Maynard, Niraj Aswani, and Kalina Bontcheva. Microblog-genre noise and impact on semantic annotation accuracy. In *Proceedings of the 24th ACM Conference on Hypertext and Social Media*, pages 21–30. ACM, 2013.
- [40] Liyan Dong, Yongli Li, Han Yin, Huang Le, and Mao Rui. The algorithm of link prediction on social network. *Mathematical Problems in Engineering*, 2013, 2013.

- [41] Geng Du, Fei Su, and Anni Cai. Face recognition using surf features. In *Sixth International Symposium on Multispectral Image Processing and Pattern Recognition*, pages 749628–749628. International Society for Optics and Photonics, 2009.
- [42] Sachin Sudhakar Farfade, Mohammad J Saberian, and Li-Jia Li. Multi-view face detection using deep convolutional neural networks. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, pages 643–650. ACM, 2015.
- [43] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 363–370. Association for Computational Linguistics, 2005.
- [44] Cong Geng and Xudong Jiang. Face recognition using sift features. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 3313–3316. IEEE, 2009.
- [45] Kevin Gimpel, Nathan Schneider, Brendan O’Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A Smith. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 42–47. Association for Computational Linguistics, 2011.
- [46] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015.
- [47] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.
- [48] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 99(12):7821–7826, 2002.

- [49] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.
- [50] Eric Granger and D Gorodnichy. *Evaluation methodology for face recognition technology in video surveillance applications*. Defence R & D Canada, 2014.
- [51] Ralph Gross, Iain Matthews, Jeffrey Cohn, Takeo Kanade, and Simon Baker. Multiple. *Image and Vision Computing*, 28(5):807–813, 2010.
- [52] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. MS-Celeb-1M: A dataset and benchmark for large scale face recognition. In *European Conference on Computer Vision*. Springer, 2016.
- [53] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.
- [54] John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.
- [55] Tal Hassner, Shai Harel, Eran Paz, and Roei Enbar. Effective face frontalization in unconstrained images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4295–4304, 2015.
- [56] Xiaofei He, Shuicheng Yan, Yuxiao Hu, Partha Niyogi, and Hong-Jiang Zhang. Face recognition using laplacianfaces. *IEEE transactions on pattern analysis and machine intelligence*, 27(3):328–340, 2005.
- [57] Tamara Heck. Combining social information for academic networking. In *Proceedings*

- of the 2013 conference on Computer supported cooperative work, pages 1387–1398. ACM, 2013.
- [58] Tamara Heck, Isabella Peters, and Wolfgang G Stock. Testing collaborative filtering against co-citation analysis and bibliographic coupling for academic author recommendation. In *Proceedings of the 3rd ACM RecSys’ 11 Workshop on Recommender Systems and the Social Web*, pages 16–23, 2011.
- [59] Bernd Heisele, Purdy Ho, and Tomaso Poggio. Face recognition with support vector machines: Global versus component-based approach. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 688–694. IEEE, 2001.
- [60] Zoe Henderson, Vicki Bruce, and A Mike Burton. Matching the faces of robbers captured on video. *Applied Cognitive Psychology*, 15(4):445–464, 2001.
- [61] Mark Hepple. Independence and commitment: Assumptions for rapid training and execution of rule-based pos taggers. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 278–277. Association for Computational Linguistics, 2000.
- [62] Abdelhakim Herrouz, Chabane Khentout, and Mahieddine Djoudi. Overview of web content mining tools. *arXiv preprint arXiv:1307.1024*, 2013.
- [63] Julia Hirschberg and Christopher D Manning. Advances in natural language processing. *Science*, 349(6245):261–266, 2015.
- [64] Jeffrey Ho, Ming-Husang Yang, Jongwoo Lim, Kuang-Chih Lee, and David Kriegman. Clustering appearances of objects under varying illumination conditions. In *Computer vision and pattern recognition, 2003. Proceedings. 2003 IEEE computer society conference on*, volume 1, pages I–11. IEEE, 2003.

- [65] Javad Hosseinkhani, Suriayati Chaprut, and Hamed Taherdoost. Criminal network mining by web structure and content mining. In *International Conference on Information Security and Privacy, Advances in Remote Sensing, Finite Differences and Information Security, Prague, Czech Republic*, pages 24–26, 2012.
- [66] Andreas Hotho, Robert Jäschke, Christoph Schmitz, and Gerd Stumme. *Information retrieval in folksonomies: Search and ranking*. Springer, 2006.
- [67] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.
- [68] Chang Huang, Haizhou Ai, Yuan Li, and Shihong Lao. Vector boosting for rotation invariant multi-view face detection. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 446–453. IEEE, 2005.
- [69] Gary Huang, Marwan Mattar, Honglak Lee, and Erik G Learned-Miller. Learning to align from scratch. In *Advances in neural information processing systems*, pages 764–772, 2012.
- [70] Gary B Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, Technical Report 07-49, University of Massachusetts, Amherst, 2007.
- [71] Rui Huang, Shu Zhang, Tianyu Li, Ran He, et al. Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis. *arXiv preprint arXiv:1704.04086*, 2017.
- [72] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint*, 2017.

- [73] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.
- [74] Vidit Jain and Erik G Learned-Miller. Fddb: A benchmark for face detection in unconstrained settings. *UMass Amherst Technical Report*, 2010.
- [75] Isuru Jayaweera, Chamath Sajeewa, Sampath Liyanage, Tharindu Wijewardane, Indika Perera, and Adeesha Wijayasiri. Crime analytics: Analysis of crimes through newspaper articles. In *Moratuwa Engineering Research Conference (MERCon), 2015*, pages 277–282. IEEE, 2015.
- [76] Huaizu Jiang and Erik Learned-Miller. Face detection with the faster r-cnn. *arXiv preprint arXiv:1606.03473*, 2016.
- [77] Song Jiang and Xiaodong Zhang. Lirs: an efficient low inter-reference recency set replacement policy to improve buffer cache performance. *ACM SIGMETRICS Performance Evaluation Review*, 30(1):31–42, 2002.
- [78] Karen Sparck Jones. *Readings in information retrieval*. Morgan Kaufmann, 1997.
- [79] Michael Jones and Paul Viola. Fast multi-view face detection. *Mitsubishi Electric Research Lab TR-20003-96*, 3:14, 2003.
- [80] Vahid Kazemi and Josephine Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1867–1874, 2014.
- [81] Yan Ke and Rahul Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–II. IEEE, 2004.

- [82] Davis E King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10(Jul):1755–1758, 2009.
- [83] Tibor Kiss and Jan Strunk. Unsupervised multilingual sentence boundary detection. *Computational Linguistics*, 32(4):485–525, 2006.
- [84] Brendan F Klare, Ben Klein, Emma Taborsky, Austin Blanton, Jordan Cheney, Kristen Allen, Patrick Grother, Alan Mah, and Anil K Jain. Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark a. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1931–1939, 2015.
- [85] Jytte Klausen. Tweeting the jihad: Social media networks of western foreign fighters in syria and iraq. *Studies in Conflict & Terrorism*, 38(1):1–22, 2015.
- [86] Joshua C Klontz and Anil K Jain. A case study of automated face recognition: The boston marathon bombings suspects. *Computer*, 46(11):91–94, 2013.
- [87] Emily Goldberg Knox. The slippery slope of material support prosecutions: Social media support to terrorists. *Hastings LJ*, 66:295, 2014.
- [88] Donald Ervin Knuth, Donald Ervin Knuth, and Donald Ervin Knuth. *The Stanford GraphBase: a platform for combinatorial computing*, volume 37. Addison-Wesley Reading, 1993.
- [89] Negar Koochakzadeh, Atieh Sarraf, Keivan Kianmehr, Jon Rokne, and Reda Alhajj. Netdriller: A powerful social network analysis tool. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*, pages 1235–1238. IEEE, 2011.
- [90] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with

- deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [91] Chih Hao Ku, Alicia Iriberri, and Gondy Leroy. Crime information extraction from police and witness narrative reports. In *Technologies for Homeland Security, 2008 IEEE Conference on*, pages 193–198. IEEE, 2008.
- [92] Neeraj Kumar, Alexander C Berg, Peter N Belhumeur, and Shree K Nayar. Attribute and simile classifiers for face verification. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 365–372. IEEE, 2009.
- [93] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical review E*, 78(4):046110, 2008.
- [94] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. *arXiv preprint*, 2016.
- [95] Donghee Lee, Jongmoo Choi, Jong-Hun Kim, Sam H Noh, Sang Lyul Min, Yookun Cho, and Chong Sang Kim. On the existence of a spectrum of policies that subsumes the least recently used (lru) and least frequently used (lfu) policies. In *ACM SIGMETRICS Performance Evaluation Review*, volume 27, pages 134–143. ACM, 1999.
- [96] Donghee Lee, Jongmoo Choi, Jong-Hun Kim, Sam H Noh, Sang Lyul Min, Yookun Cho, and Chong Sang Kim. Lrfu: A spectrum of policies that subsumes the least recently used and least frequently used policies. *IEEE transactions on Computers*, 50(12):1352–1361, 2001.
- [97] Kuang-Chih Lee, Jeffrey Ho, and David J Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions on pattern analysis and machine intelligence*, 27(5):684–698, 2005.

- [98] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555. IEEE, 2011.
- [99] Haoxiang Li, Zhe Lin, Xiaohui Shen, Jonathan Brandt, and Gang Hua. A convolutional neural network cascade for face detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5325–5334, 2015.
- [100] David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031, 2007.
- [101] Ryan Lichtnwalter and Nitesh V Chawla. Link prediction: fair and effective evaluation. In *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, pages 376–383. IEEE Computer Society, 2012.
- [102] Giseli Rabello Lopes, Mirella M Moro, Leandro Krug Wives, and José Palazzo Moreira De Oliveira. Collaboration recommendation on academic social networks. In *Advances in Conceptual Modeling—Applications and Challenges*, pages 190–199. Springer, 2010.
- [103] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.
- [104] Linyuan Lü and Tao Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150–1170, 2011.
- [105] David Lusseau, Karsten Schneider, Oliver J Boisseau, Patti Haase, Elisabeth Slooten, and Steve M Dawson. The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology*, 54(4):396–405, 2003.

- [106] Michael J Lyons, Shigeru Akamatsu, Miyuki Kamachi, Jiro Gyoba, and Julien Budynnek. The japanese female facial expression (jaffe) database, 1998.
- [107] Bing Liu Wynne Hsu Yiming Ma and Bing Liu. Integrating classification and association rule mining. In *Proceedings of the fourth international conference on knowledge discovery and data mining*, 1998.
- [108] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60, 2014.
- [109] Iacopo Masi, Stephen Rawls, Gérard Medioni, and Prem Natarajan. Pose-aware face recognition in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4838–4846, 2016.
- [110] Michael Mathioudakis and Nick Koudas. Twittermonitor: trend detection over the twitter stream. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 1155–1158. ACM, 2010.
- [111] Olena Medelyan and Ian H Witten. Thesaurus based automatic keyphrase indexing. In *Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, pages 296–297. ACM, 2006.
- [112] Olena Medelyan, Ian H Witten, and David Milne. Topic indexing with wikipedia. In *Proceedings of the AAAI WikiAI workshop*, volume 1, pages 19–24, 2008.
- [113] Ahmed M Megreya and A Mike Burton. Unfamiliar faces are not faces: Evidence from a matching task. *Memory & cognition*, 34(4):865–876, 2006.
- [114] Rada Mihalcea and Paul Tarau. Texttrank: Bringing order into text. In *EMNLP*, volume 4, pages 404–411, 2004.

- [115] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.
- [116] Glenn W Milligan and Martha C Cooper. A study of the comparability of external criteria for hierarchical cluster analysis. *Multivariate Behavioral Research*, 21(4):441–458, 1986.
- [117] Shyam Varan Nath. Crime pattern detection using data mining. In *Web intelligence and intelligent agent technology workshops, 2006. wi-iat 2006 workshops. 2006 iee/wic/acm international conference on*, pages 41–44. IEEE, 2006.
- [118] Mark EJ Newman. Clustering and preferential attachment in growing networks. *Physical review E*, 64(2):025102, 2001.
- [119] Mark EJ Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical review E*, 74(3):036104, 2006.
- [120] Andrew Y Ng, Michael I Jordan, Yair Weiss, et al. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 2:849–856, 2002.
- [121] Charles Otto, Brendan Klare, and Anil K Jain. An efficient approach for clustering face images. In *2015 International Conference on Biometrics (ICB)*, pages 243–250. IEEE, 2015.
- [122] Shaoning Pang, Daijin Kim, and Sung Yang Bang. Membership authentication in the dynamic group by face classification using svm ensemble. *Pattern Recognition Letters*, 24(1):215–225, 2003.
- [123] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, et al. Deep face recognition. In *BMVC*, volume 1, page 6, 2015.
- [124] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent

- Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.
- [125] Helô Petry, Patricia Tedesco, Vaninha Vieira, and Ana Carolina Salgado. Icare. a context-sensitive expert recommendation system. *ECAI'08*, pages 53–58, 2008.
- [126] P Jonathon Phillips, Hyeonjoon Moon, Syed A Rizvi, and Patrick J Rauss. The feret evaluation methodology for face-recognition algorithms. *IEEE Transactions on pattern analysis and machine intelligence*, 22(10):1090–1104, 2000.
- [127] Nicolas Pinto, James J DiCarlo, and David D Cox. How far can you get with a modern face recognition test set using only simple features? In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2591–2598. IEEE, 2009.
- [128] Md Ileas Pramanik, Raymond YK Lau, and Md Kamal Hossain Chowdhury. Automatic crime detector: A framework for criminal pattern detection in big data era. In *PACIS*, page 311, 2016.
- [129] Steven Puttemans, Joseph Howse, Quan Hua, and Utkarsh Sinha. Opencv 3 blueprints: Expand your knowledge of computer vision by building amazing projects with opencv 3. 2015.
- [130] Predrag Radivojac, Kang Peng, Wyatt T Clark, Brandon J Peters, Amrita Mohan, Sean M Boyle, and Sean D Mooney. An integrated approach to inferring gene–disease associations in humans. *Proteins: Structure, Function, and Bioinformatics*, 72(3):1030–1037, 2008.
- [131] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*, 2016.

- [132] Tim Reichling and Volker Wulf. Expert recommender systems in practice: evaluating semi-automatic profile generation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 59–68. ACM, 2009.
- [133] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.
- [134] Ellen Riloff. Little words can make a big difference for text classification. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 130–136. ACM, 1995.
- [135] Ellen Riloff et al. Automatically constructing a dictionary for information extraction tasks. In *AAAI*, pages 811–816, 1993.
- [136] Alan Ritter, Sam Clark, Oren Etzioni, et al. Named entity recognition in tweets: an experimental study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534. Association for Computational Linguistics, 2011.
- [137] Hassan Saif, Yulan He, and Harith Alani. Semantic sentiment analysis of twitter. *The Semantic Web–ISWC 2012*, pages 508–524, 2012.
- [138] Gerard Salton and Michael J McGill. Introduction to modern information retrieval. 1986.
- [139] Ferdinando S Samaria and Andy C Harter. Parameterisation of a stochastic model for human face identification. In *Applications of Computer Vision, 1994., Proceedings of the Second IEEE Workshop on*, pages 138–142. IEEE, 1994.
- [140] Hamed Sarvari, Ehab Abozinadah, Alex Mbaziira, and Damon Mccoy. Constructing

- and analyzing criminal networks. In *Security and Privacy Workshops (SPW), 2014 IEEE*, pages 84–91. IEEE, 2014.
- [141] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.
- [142] Soumyadip Sengupta, Jun-Cheng Chen, Carlos Castillo, Vishal M Patel, Rama Chellappa, and David W Jacobs. Frontal to profile face verification in the wild. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*, pages 1–9. IEEE, 2016.
- [143] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S. Baliga, Jonathan T. Wang, Daniel Ramage, Nada Amin, Benno Schwikowski, and Trey Ideker. Cytoscape: A software environment for integrated models of biomolecular interaction networks. *Genome Research*, 13:2498–2504, 2009.
- [144] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 806–813, 2014.
- [145] Adrian Silvescu, Doina Caragea, and Anna Atramentov. Graph databases, 2002.
- [146] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [147] Marc Smith, Natasa Milic-Frayling, Ben Shneiderman, E Mendes Rodrigues, Jure Leskovec, and Cody Dunne. Nodexl: a free and open network overview, discovery and exploration add-in for excel 2007/2010, 2010.

- [148] Mark Steyvers and Tom Griffiths. Probabilistic topic models. *Handbook of latent semantic analysis*, 427(7):424–440, 2007.
- [149] Yi Sun, Ding Liang, Xiaogang Wang, and Xiaoou Tang. Deepid3: Face recognition with very deep neural networks. *arXiv preprint arXiv:1502.00873*, 2015.
- [150] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation from predicting 10,000 classes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1891–1898, 2014.
- [151] Yizhou Sun, Rick Barber, Manish Gupta, Charu C Aggarwal, and Jiawei Han. Co-author relationship prediction in heterogeneous bibliographic networks. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*, pages 121–128. IEEE, 2011.
- [152] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12, 2017.
- [153] Kamal Taha and Paul D Yoo. Siimco: A forensic investigation tool for identifying the influential members of a criminal organization. *IEEE Transactions on Information Forensics and Security*, 11(4):811–822, 2016.
- [154] Yaniv Taigman, Ming Yang, Marc’Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014.
- [155] Lei Tang, Xufei Wang, and Huan Liu. Uncovering groups via heterogeneous interaction analysis. In *Data Mining, 2009. ICDM’09. Ninth IEEE International Conference on*, pages 503–512. IEEE, 2009.

- [156] Mohammad A Tayebi, Martin Ester, Uwe Glässer, and Patricia L Brantingham. Spatially embedded co-offence prediction using supervised learning. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1789–1798. ACM, 2014.
- [157] Carlos Eduardo Thomaz. Fei face database. *online]* <http://fei.edu.br/~cet/facedatabase.html> (accessed 2 October 2012), 2012.
- [158] Engin Tola, Vincent Lepetit, and Pascal Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(5):815–830, 2010.
- [159] Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics, 2003.
- [160] Luan Tran, Xi Yin, and Xiaoming Liu. Disentangled representation learning gan for pose-invariant face recognition. In *CVPR*, volume 3, page 7, 2017.
- [161] Matthew A Turk and Alex P Pentland. Face recognition using eigenfaces. In *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on*, pages 586–591. IEEE, 1991.
- [162] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013.
- [163] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511. IEEE, 2001.

- [164] Jun Wang, Arjen P De Vries, and Marcel JT Reinders. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 501–508. ACM, 2006.
- [165] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994.
- [166] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European Conference on Computer Vision*, pages 499–515. Springer, 2016.
- [167] Lior Wolf, Tal Hassner, and Yaniv Taigman. Similarity scores based on background samples. In *Asian Conference on Computer Vision*, pages 88–97. Springer, 2009.
- [168] Kin-Yeung Wong. Web cache replacement policies: a pragmatic approach. *IEEE Network*, 20(1):28–34, 2006.
- [169] Yongkang Wong, Shaokang Chen, Sandra Mau, Conrad Sanderson, and Brian C Lovell. Patch-based probabilistic image quality assessment for face selection and improved video-based face recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*, pages 74–81. IEEE, 2011.
- [170] Bo Wu, Haizhou Ai, Chang Huang, and Shihong Lao. Fast rotation invariant multi-view face detection based on real adaboost. In *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*, pages 79–84. IEEE, 2004.
- [171] Xiang Wu, Ran He, Zhenan Sun, and Tieniu Tan. A light cnn for deep face representation with noisy labels. *arXiv preprint arXiv:1511.02683*, 2015.

- [172] Christopher C Yang and Tobun D Ng. Terrorism and crime related weblog social network: Link, content analysis and information visualization. In *Intelligence and Security Informatics, 2007 IEEE*, pages 55–58. IEEE, 2007.
- [173] Shuang-Hong Yang, Alek Kolcz, Andy Schlaikjer, and Pankaj Gupta. Large-scale high-precision topic modeling on twitter. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1907–1916. ACM, 2014.
- [174] Shuo Yang, Ping Luo, Chen-Change Loy, and Xiaoou Tang. Wider face: A face detection benchmark. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5525–5533, 2016.
- [175] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. Learning face representation from scratch. *arXiv preprint arXiv:1411.7923*, 2014.
- [176] Junho Yim, Heechul Jung, ByungIn Yoo, Changkyu Choi, Dusik Park, and Junmo Kim. Rotating your face using multi-task deep neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 676–684, 2015.
- [177] Xin Yu and Fatih Porikli. Ultra-resolving face images by discriminative generative networks. In *European Conference on Computer Vision*, pages 318–333. Springer, 2016.
- [178] Wayne W Zachary. An information flow model for conflict and fission in small groups. *Journal of anthropological research*, pages 452–473, 1977.
- [179] Cha Zhang and Zhengyou Zhang. Improving multiview face detection with multi-task deep convolutional neural networks. In *Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on*, pages 1036–1041. IEEE, 2014.

- [180] Ming Zhao, Yong Wei Teo, Siliang Liu, Tat-Seng Chua, and Ramesh Jain. Automatic person annotation of family photo album. In *International Conference on Image and Video Retrieval*, pages 163–172. Springer, 2006.
- [181] Ding Zhou, Eren Manavoglu, Jia Li, C Lee Giles, and Hongyuan Zha. Probabilistic models for discovering e-communities. In *Proceedings of the 15th international conference on World Wide Web*, pages 173–182. ACM, 2006.
- [182] Chunhui Zhu, Fang Wen, and Jian Sun. A rank-order distance based clustering algorithm for face tagging. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 481–488. IEEE, 2011.
- [183] Xiangxin Zhu and Deva Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2879–2886. IEEE, 2012.
- [184] Xiangyu Zhu, Zhen Lei, Junjie Yan, Dong Yi, and Stan Z Li. High-fidelity pose and expression normalization for face recognition in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 787–796, 2015.
- [185] Zhenyao Zhu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Multi-view perceptron: a deep model for learning face identity and view representations. In *Advances in Neural Information Processing Systems*, pages 217–225, 2014.
- [186] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *European Conference on Computer Vision*, pages 391–405. Springer, 2014.