ABSTRACT

Title of dissertation: MULTI-MODAL ACTIVE AUTHENTICATION OF SMARTPHONE USERS

Upal Mahbub
Doctor of Philosophy, 2018

Dissertation directed by: Professor Rama Chellappa
Department of Electrical and Computer Engineering

With the increasing usage of smartphones not only as communication devices but also as the port of entry for a wide variety of user accounts at different information sensitivity levels, the need for hassle-free authentication is on the rise. Going beyond the traditional one-time authentication concept, active authentication (AA) schemes are emerging which authenticates users periodically in the background without the need for any user interaction. The purpose of this research is to explore different aspects of the AA problem and develop viable solutions by extracting unique biometric traits of the user from the wide variety of usage data obtained from Smartphone sensors. The key aspects of our research are the development of different components of user verification algorithms based on (a) face images from the front camera and (b) data from modalities other than the face.

Since generic face detection algorithms do not perform very well in the mobile domain due to a significant presence of occluded and partially visible faces, we propose facial segment-based face detection technique to handle the challenge of partial faces in the mobile domain. We have developed three increasingly accurate proposal-based face detection methods, namely Facial Segment-based Face Detector (FSFD), SegFace and DeepSegFace, respectively, which perform binary classification on the results of a novel proposal generator that utilizes facial segments to obtain face-proposals. We also propose the Deep Regression-based User Image Detector (DRUID) network which shifts from the classification to the regression paradigm to avoid the need for proposal generation and thereby, achieves better processing speed and accuracy. DeepSegFace and DRUID have unique network architectures with customized loss functions and utilize a novel data augmentation scheme to train on a relatively small amount of data. The proposed methods, especially DRUID show superior performance over other state-of-the-art face detectors in terms of precision-recall and ROC curve on two mobile face datasets.

We extended the concept of facial-segments to facial attribute detection for partially visible faces, a topic rarely addressed in the literature. We developed a deep convolutional neural network-based method named Segment-wise, Partial, Localized Inference in Training Facial Attribute Classification Ensembles (SPLITFACE) to detect attributes reliably from partially occluded faces. Taking several facial segments and the full face as input, SPLITFACE takes a data-driven approach to determine which attributes are localized in which facial segments. The unique architecture of the network allows each attribute to be predicted by multiple segments, which permits the implementation of committee machine techniques for combining local and global decisions to boost performance. Our evaluations on the full CelebA and LFWA datasets and their modified partial-visibility versions show that SPLIT-FACE significantly outperforms other recent attribute detection methods, especially for partial faces and for cross-domain experiments.

We also explored the potentials of two less popular modalities namely, location history and application-usage, for active authentication. Aiming to discover the pattern of life of a user, we processed the location traces into separate state space models for each user and developed the Marginally Smoothed Hidden Markov Model (MSHMM) algorithm to authenticate the current user based on the most recent sequence of observations. The method takes into consideration the sparsity of the available data, the transition phases between states, the timing information and also the unforeseen states. We looked deeper into the impact of unforeseen and unknown states in another research work where we evaluated the feasibility of application usage behavior of the users as a potential solution to the active authentication problem. Our experiments show that it is essential to take unforeseen states into account when designing an authentication system with sparse data and marginal-smoothing techniques are very useful in this regard.

We conclude this dissertation with the description of some ongoing efforts and future directions of research related the topics discussed in addition to a summary of all the contributions and impacts of this research work.

MULTI-MODAL ACTIVE AUTHENTICATION OF SMARTPHONE USERS

by

Upal Mahbub

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2018

Dedication

To my beloved parents & family

For their love, endless support, encouragement & sacrifices

Acknowledgments

First and foremost I'd like to thank my adviser, Professor Rama Chellappa for being an outstanding mentor and for giving me an invaluable opportunity to work on challenging and extremely interesting projects over the past five years. I truly admire his discipline, dedication, and passion towards his research work and his insightful words about life has deeply influenced my view towards it. It has been a pleasure to work with and learn from such an extraordinary individual.

Thanks are due to Professor Larry Davis, Professor K. J. Ray Liu, Professor Min Wu, and Professor Tudor Dumitraş for agreeing to serve on my thesis committee and for sparing their invaluable time reviewing the manuscript and providing their feedback for improving it.

My researches were supported in part by Google Advanced Technology and Projects (ATAP), a cooperative agreement FA8750-13-2-0279 from DARPA and the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA R&D Contract No. 2014-14071600012. I am thankful to the funding organizations for supporting my research endeavors.

My colleagues and friend on the fourth floor of the AVW building have enriched my graduate life in many ways and deserve a special mention. I am thankful to Dr. Vishal Patel for his help and support throughout my Ph.D. program. Dr. Jun-Cheng Cheng has been a great mentor and friend since the beginning of my journey at UMD and helped me numerous times with research. Sayantan has been a wonderful friend, research collaborator and also my mentor in Fussball! I am thankful to Ankan for being an awesome labmate and research collaborator and Rajeev for helping me as a friend with ideas and suggestions. My interactions with my colleagues/labmates/friends over time - Dr. Carlos Castillo, Josh, Andy, Amit, Pouya, Arthita, Navaneeth, Anirudh, Navaneeth, Hui, Swami, Maya, Emily, Jingxiao, Saketh, Prithviraj, Jie Ni, Hongyu, Kota, Raviteja, Mohammed and Henry has been very fruitful and inspiring. I would also like to acknowledge the help and support from Ms. Janice Perrone and Ms. Melanie Prange throughout my time at UMD.

In addition, I thank my colleagues Raghavendra Jillala and Sima Taheri, for a wonderful summer at Digital Signal Corporation in 2014, my managers and co-authors Deepak Chandra and Brandon Barbello for making it an awesome learning experience at Google ATAP Authentication Sprint 2015, and Jan Neumann, Hongcheng Wang, and Md. Mahmudul Hasan for their kind support and mentoring during my summer 2017 internship at Comcast DC Labs. I am grateful to my co-authors Jukka Komulainen and Denzil T. Ferreira from the University of Oulu, Finland for being wonderful mentors and friends during the final years of my Ph.D.

I am forever grateful to Dr. Shaikh Anowarul Fattah and Dr. Celia Shehnaz of Bangladesh University of Engineering and Technology for their guidance and advice throughout my academic life. My heartiest gratitude goes towards Dr. Atiqur Rahman Ahad from the University of Dhaka for his guidance, support and selfless friendship.

This acknowledgment would not be complete without mentioning my dear friends- Towhid, Fahi, Shibani, and Avi. I am grateful for their unconditional love and support and would cherish the wonderful times that we shared together. I also want to thank my close friends in the USA living in different states and the local Bangladeshi student communities for making me a part of various activities.

I owe my deepest thanks to my family - my mother, Saleha Begum, and father, Mahbubul Haque, who have always stood by me and guided me through my career, and have pulled me through against impossible odds at times. Words cannot express the gratitude I owe them. I am also in debt to my kind and supportive parents-in-law, Sajeda Kohinoor and Azam-za Chowdhury for their unconditional blessings and for visiting me every year to make me feel at home. I would like to extend my gratitude towards my elder sister Upoma, her husband Amit and my adorable niece Promiti. Also, I am grateful to my extended family members, Shaikat - Ayesha and little Aydin, and Tanzina - Mujib and their wonderful kids, Evan and Rayan. This last word of acknowledgment I have saved for my dear wife, best friend, and partner in crime, Tasnuva Chowdhury, who has been a constant inspiration for me throughout these years and has made them the best years of my life.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AA	Active Authentication
AA-01-FD	Active Authentication Face Detection Dataset - 01
AFFACT	Alignment-Free Facial Attribute Classification Technique
AFLW	Annotated Facial Landmarks in the Wild dataset
AFW	Annotated Faces in the Wild dataset
ATAP	Google's Advanced Technology and Projects group (ATAP)
CAM	Class Activation Map
CelebA	Celebrity image attribute dataset
CNN	Convolutional Neural Network
CorrD	Correlation Distance
CosD	Cosine Distance
DARPA	Defense Advanced Research Projects Agency
DCNN	Deep Convolutional Neural Network
DMTL	Deep Multi-Task Learning
DP2MFD	Deep Pyramid Deformable Part Model
DPM	Deformable Part-based Model for face detection
DRUID	Deep Regression-based User Image Detector
ED	Edit Distance
EER	Equal Error Rate
EU	Euclidean distance
FD	Face Detection
FDDB	Face Detection Datset and Benchmark
FN	False Negative
FP	False Positive
FPR	False Positive Rate
FSFD	Facial Segment-based Face Detection
GBM	Gradient Boosting Method for classification
GMM	Gaussian Mixture Models
GPS	Global Positioning System
GPU	Graphics Processing Unit
HMM	Hidden Markov Model
HMM-lap	Hidden Markov Model with Laplacian smoothing
HoG	Histogram of oriented Gradients

HRP	Highest Ranked Predictor
HTTP	Hypertext Transfer Protocol
IARPA	Intelligence Advanced Research Projects Activity
KNN	K-Nearest Neighbor
LAEO	Looking At Each Other face detector
LBP	Local Binary Pattern
LDDR	Local Deep Descriptor Regression method
LFW	Labeled Faces in the Wild dataset
LFWA	LFW Attribute Dataset
MC	Markov Chain-based method
MD	Manhattan Distance
M-ED	Modified Edit Distance method
MEEN	Mouth-Eye-Eye-Nose feature extraction method for face verification
MMC	Mobility Markov Chain
MMM	Mixed Markov Chain
MOON	Mixed Object Optimization Network
MSHMM	Marginally Smoothed Hidden Markov Model
NB	Naive Bayes classification method
NPD	Normalized Pixel Difference
NSA	Normalized Score Aggregation
PATH	Person Authentication using Trace Histories
RBF	Radial Basis Function
RCNN	(Regions + CNN) is an object detection method that relies on a external region p
RF	Random Forest estimator
ROC	Receiver Operating Characteristic curve
SM	Sequence Matching method
SPLITFACE	Segment-wise, Partial, Localized Inference in Training Facial Attribute Classificat
SSD	Single-shot multibox detector
SVM	Support Vector Machine
TN	True Negative
TP	True Positive

TPR	True Positive Rate
TZ	Time Zone
UMDAA-02	University of Maryland Active Authentication Dataset - 02
VJ	Viola-Jones face detection method

Chapter 1: Introduction

The recent proliferation of mobile devices like smartphones and tablets has given rise to security concerns about personal information stored in them. Studies show that users are more concerned about the security of their cell phones over laptops [6]. Though over 40% of users in major U.S. cities have lost their phones or have been victims of phone theft [7], industry surveys estimate that 34% of smartphone users in the U.S. do not lock their phones with passwords [8]. This contradictory behavior is due to the time-consuming, cumbersome and error-prone hassles of entering passwords on virtual keyboards or due to users' beliefs that extra passwords are not needed [7]. 76% attacks on smart phones exploit weak passwords [9], but users still prefer those over stronger passwords, as the stronger passwords are difficult to remember and type, especially since the average cell phone user checks their smartphone device 150 times per day [10].

Going beyond traditional passwords and fingerprint-based one-time authentication, the concept of Active Authentication (AA) has emerged recently [11], where the enrolled user is authenticated continuously in the background based on the user's biometrics such as front camera face capture [12], [13], touch screen gesture [14], [15], typing pattern [16] etc. Conceptually, in an AA system users do not password-lock

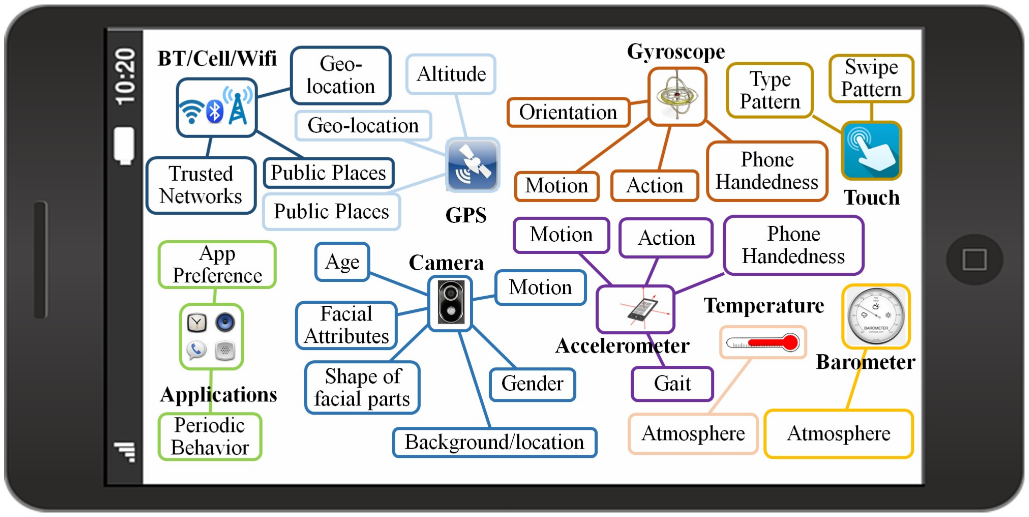


Figure 1.1: Association of smartphone sensors with behavioral and biometric information.

the phone at all. When a user uses the phone, the AA system compares the usage pattern with the enrolled user's pattern of use. If the system deems that the usage patterns are sufficiently similar, the phone's full functionality (including sensitive applications and data) is made available, else it blocks the current user. At present, most of the AA systems are based on face, touch and typing pattern biometrics. Note that the terms active authentication [17], [11], continuous authentication [18], [19], implicit authentication [20], [21], and transparent authentication [22] have been used interchangeably in the literature.

In this introductory chapter we provide an overview of the contributions of this dissertation and, and a guide to its organization by chapters.

1.1 Major Contributions of the Dissertation

The main contributions of this dissertation can be categorized into three broad topics, which are

1. active authentication dataset collection, defining problems and benchmark evaluation,
2. introducing the concept of facial-segment for face-based authentication research, and,
3. exploring the usability of non-face modalities, especially, location history data and application-usage behavior, for active authentication.

The aforementioned contributions are discussed in the following subsections.

1.1.1 Active Authentication Dataset Collection and Benchmarking

As shown in Fig. 1.1, modern smartphones provide multiple sensors associated with a variety of behavioral and physiological biometric information, however research on multi-modal authentication using multi-sensor data has been lagging behind, because of paucity of datasets. One of the biggest contribution of our research on active authentication is the collection of the first non-commercial dataset on unconstrained smartphone usage, namely, the University of Maryland Active Authentication Dataset 02 (UMDAA-02), which contains a wide range of sensor data and is made available for the research community. Unlike task-based data collection schemes, the data collection for UMDAA-02 was passive and hence is representative of the natural, regular smartphone usage by the volunteers. The data collection application ran on the Nexus-5 device, completely in the background, saving sensor data and periodically uploading the data to a secure online location.

Our initial research work included producing benchmark results for the UMDAA-02 dataset on four different experiments, which are, face detection, face-based verification, swipe-based user identification and next location prediction using the temporal geolocation data. The poor benchmark results obtained for state-of-the-art algorithms justified the need for developing more sophisticated algorithms for reliable active authentication.

1.1.2 Face-based Authentication using Facial Segments

1.1.2.1 Facial Segments for Face Detection

The remarkable progress in convolutional neural network (CNN) architectures and the availability of large amount of face data have accelerated the development of efficient and robust face detection techniques in recent years [23] [24] [25]. State-of-the-art face detectors are mostly developed for detecting faces in unconstrained environments with large variations in pose and illumination [26] [27] [28] [29]. However, development of face detectors optimized for detecting occluded and partially visible faces are becoming very essential because of the rapidly increasing usages of cameras mounted on mobile devices [4] [30]. Reliable and fast detection of faces from the front-camera captures of a mobile device is a fundamental step for applications such as active/continuous authentication of the user of a mobile device [11] [31] [15] [32].

Although, face-based authentication systems on mobile devices rely heavily on accurate detection of faces prior to verification, most state-of-the-art techniques are ineffective for mobile devices because of the following reasons:

1. The user's face captured by the front camera of the phone is, in many cases, only partially visible [31]. While most modern face detectors such as Viola-Jones's [33], DPM [26], Hyperface [23], yahoo multiview [25], CUHK [24] etc. work well on detecting multiple frontal or profile faces of various resolutions, they frequently fail to detect single partially visible faces as they do not explicitly model partial faces.

2. For active authentication the recall rate needs to be high at very high precision. Many of the available face detectors have a low recall rate even though the precision is high. When operated at high recall, their precision drops rapidly because of excessive false positive detection [31].

3. The algorithm needs to be simple, fast and customizable to operate in real-time on a cellular device. While in [30] and [34] the authors deploy CNNs on mobile GPUs for face detection and verification, most CNN-based detectors, such as [23] and some generic methods like [26] are too complex to run on the mobile platform.

On the other hand, images captured for active authentication offer certain advantages for the face detection problem because of its semi-constrained nature [4]. Usually there is a single user in the frame, hence there is no need to handle multiple face detection. The face is in close proximity of the camera and of high resolution, thus eliminating the need for detecting at multiple scales or resolution.

In our preliminary research work, we have extensively exploited the idea of facial segment-based face detection. Fig. 1.2 shows a sample decomposition of a

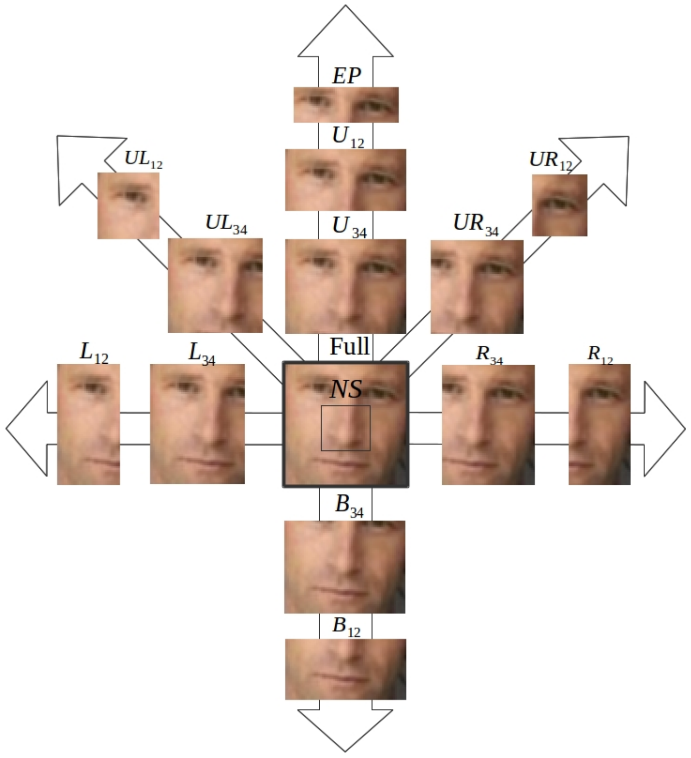


Figure 1.2: Decomposition of a full face into 14 facial segments [4].

full face into facial segments, detection of one or more of which might offer powerful clues about the location of the full face. Partial faces such as the ones present in images captured by the front camera of mobile devices can be handled if the algorithm is able to effectively combine detection of different facial segments into a full face. To address this requirement, we have developed three algorithms, namely, Facial Segment-based Face Detection (FSFD), SegFace and DeepSegFace. These algorithms detect faces from proposals made of face segments by utilizing a fast proposal generation scheme that provides bounding boxes for faces and facial segments. FSFD and SegFace methods use traditional feature extraction techniques and support vector machine (SVM) classifier whereas DeepSegFace is a DCNN-based classifier for differentiating between proposals with and without faces.

The proposal-based approaches, however, suffer from several limitations such as slow speed, upper bound on recall because of the proposal generation step etc. Hence, we developed a regression-based end-to-end trainable face detector for detecting a single user face that does not require any proposal generation at all. This method, named, Deep Regression-based User Image Detector (DRUID), is a deep CNN-based face detector that returns not only the face bounding box, but also the bounding boxes of all the facial segments that are present along with the confidence scores for each segment in a single forward pass. DRUID utilizes a principled data augmentation technique to train using a relatively small number of image and it's throughput is very fast given its architecture and independence from the proposal generation stage. Moreover, training of DRUID is not done on a mobile face dataset similar to the proposal-based approaches, yet, it performs significantly better than other methods mostly due to its unique architecture and data augmentation scheme. The augmentation also makes it robust against scaling of faces and enables to find the bounding boxes for faces of different sizes through regression during training.

1.1.2.2 Facial Segments for Facial Attribute Detection

We also evaluate the suitability of a facial segment-based approach for facial attribute detection, especially when partially visible faces are present. The problem of attribute detection from face images has received much attention from the computer vision community in recent years [35] [36] [37] [2]. Successful detection of facial attributes has numerous practical applications, such as user-verification [38] and image search [39], video surveillance [40], age and gender estimation to assist salutation for HCI [37], and facial expression estimation for mood analysis [37]. Most attribute detection algorithms assume the availability of a full, near frontal and aligned face, and we find that their performance degrades significantly in domains where partially visible faces are frequent. One such domain is front-camera images of smartphones, which are used for continuous active authentication of users [11] [5]. To develop a method that detects attributes from full as well as partial faces, we consider the following key observations:

- Some attributes can be inferred correctly even if the face is partially occluded. For example, it is possible for humans to infer the gender from only the left half or upper half of the face.

- Some attributes are strongly localized in certain part of the face. For example, beard or mustache can only be inferred from the lower half of the face.

Given these observations, it is desirable that a technique for attribute detection be designed, whose performance degrades gracefully with increasing occlusion, rather than suffering catastrophic failures.

In this regard, we present a two-step deep convolutional neural network-based method for facial attribute detection that takes into account the relative strength of different facial segments in detecting different facial attributes. We analyze the detection results obtained in the first step where all facial segments were tasked to decide the attributes. We then present a method to automatically assign selective sets of attributes to different facial regions, resulting in a performance boost in the second step. We also determine appropriate thresholds for deciding on each attribute at each segment based on the detection results obtained from the validation set. Finally, we combine the predictions from different facial segments and produce the final result. Some special features of the proposed algorithm are:

- We have implemented a local to global attribute detection approach that harnesses the strength of different facial segments into determining different attributes. For example, the bottom-half of a face has information about the beard, while the upper-half has information about the hair. Our divide and conquer approach extracts intermediate results from each segment and combines them in the end to boost the overall performance.

- Not all facial segments have to be present for the proposed method to work. The method relies on the whole face and one or more facial segments to estimate all the attributes. The individual facial segments are self-sufficient for estimating the attributes they are assigned to. Hence, the method demonstrates superior performance when the full face is not visible due to partial occlusion or pose variation.

- We analyze the local aspects of facial attributes by associating them with facial segments and develop an automated method to utilized the local information.

- It is a well known fact that an ensemble of networks generally outperform a single network. However training an ensemble is very time-consuming. Our proposed architecture provides us with 16 predictors with only one round of training. We show that a significant increase in the final result is achieved by combining scores from these predictors.

Our proposed network namely, Segmentwise, Partial, Localized Inference in Training Facial Attribute Classification Ensembles (SPLITFACE), is presented in details in this dissertation along with extensive experimentation results.

1.1.3 Usability of Non-Face Modalities for Active Authentication

This work aims at user verification using soft biometric such as touch or swipe dynamics along with acceleration and rotation components of the phone in three dimensional space, application usage, location and connectivity information etc.

1.1.3.1 Location History-based User Authentication

Combining the usability of location traces to model a user's pattern of life with the concept of active authentication, we addressed the Person Authentication using Trace Histories (PATH) problem. The PATH problem assumes that every individual has a pattern of life which heavily regulates the time and sequence of a person's visit to different key places/locations. In PATH, the goal is to perform user verification from historical location data of a user in a continuous manner so that a verification score based on the location information is obtained continuously. This score can be fused with scores returned by other modalities such as touch or face to improve the performance of the overall authentication system. Being motivated by the wide availability of individual GPS data through smartphones, wearable devices etc. and by the discriminating life patterns of different individuals, we developed a unique method for user data clustering for trace state generation and a modified

Hidden Markov Model(HMM)-based user verification method, namely, Marginally Smoothed HMM (MSHMM) which is capable of handling unforeseen and sparse location traces.

1.1.3.2 Application Usage-based User Authentication

With the rapid increase of smartphone users worldwide, the mobile applications are growing both in number and popularity [41]. The number of apps in Google Play store is around 8 million, while in Apple App Store, Windows Store and Amazon Appstore there around 2.2 million, 669 thousand, and 600 thousand applications, respectively[1]. It has been estimated that a total of 197 billion mobile application were downloaded in 2017[2]. A retrospective study in 2016 showed that on average a smartphone user uses over 30 different mobile applications per month and $\sim$ 10 different applications per day [42]. As for usage duration in 2016, in the USA, the smartphone users spend on a daily basis over two hours on mobile applications, *i.e.*, over a month usage of applications in a year [42]. With growing concerns of smartphone security, monitoring the application usage coupled with the diverse pool of applications can help to make a difference in user authentication systems.

Smartphone application usage data can provide several interesting insights on the device users leading to different use cases of such data. There are several research works on user profiling and predicting behavioral patterns using application usage data [43] [41] [44] [45]. Predicting application usage pattern can also

[1]`https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/`

[2]`https://www.statista.com/statistics/271644/worldwide-free-and-paid-mobile-app-store-downloads/`

help optimizing smartphone resources and help simulating realistic usage data for automated smartphone testing [46] [47] [48] [49] [50] [51]. The open foreground application can also work as a context for active authentication using other modalities [52] [53] [54] [55]. For example, when verifying with touch and accelerometer data, the application running in the background can provide useful context for robust authentication. Intuitively, the way a user handles and swipes in a phone for a banking application is very different from that used for a gaming application. The foreground application context can be even more useful for active authentication if some more insightful information about the applications are available as metadata. For example, one key idea of active/continuous authentication is gradually blocking a probable intruder starting from the most sensitive applications, such as banking and social media accounts [56] [31]. If the sensitivity level or the type of application is known as metadata, it would be possible to attain enhanced security. Also, some applications, if permitted, can access the location data and store click information for targeted advertisement and similar applications [57]. A more active use case of application-usage data could be verifying the users solely from the pattern of usage. The different use cases of app-usage data are shown in Fig. 1.3.

In this dissertation, the suitability of application-usage data as a modality for smartphone user verification is thoroughly investigated. Our main contributions in this regard are:

- An innovative formulation of the user verifiaction problem utilizing application usage data pattern as a biometric. The formulation tackles key challenges

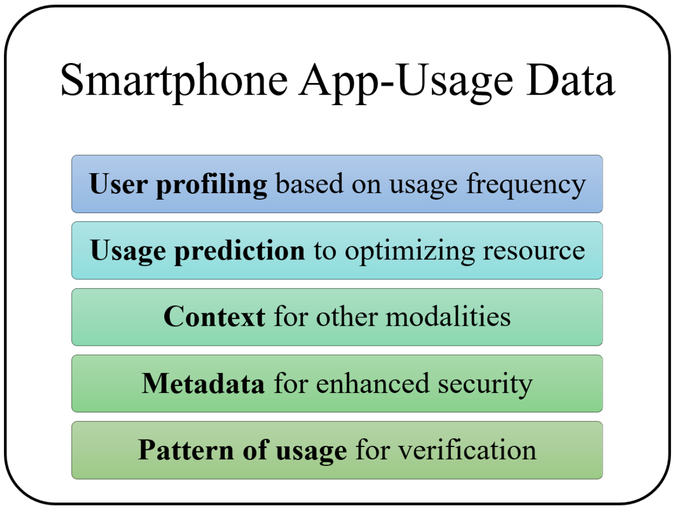

Smartphone App-Usage Data
User profiling based on usage frequency
Usage prediction to optimizing resource
Context for other modalities
Metadata for enhanced security
Pattern of usage for verification

Figure 1.3: Use cases for smartphone app-usage data.

such as data sparsity and accounting for unforeseen test observations. Unlike traditional approaches of using top N-applications for authentication purposes [58], in the proposed formulation the full list of applications is considered for verification models in order to ensure low-latency which is essential for active authentication systems.

- Insight into the application usage similarity among different users and statistics on unforeseen applications.
- A thorough investigation of the impact of unknown applications and unforeseen observations on the verification task.
- A Modified Edit-Distance (M-ED) algorithm and experiments to demonstrate the advantage of including unforeseen events during sequence matching.
- Modeling the Person Authentication using Trace Histories (PATH) problem as a variation of the person authentication using location histories [3].

1.2 Dissertation Outline

The material presented in this dissertation is reasonably self contained. It is organized by chapters as follows.

Chapter 2 : Backgound and related works on the main topics of the dissertation are presented.

Chapter 3 : We introduce the UMDAA-02 dataset, discuss the data collection protocol, provide detailed information on the modalities, define several authentication related problems and provide benchmark evaluation using state-of-the-art methods.

Chapter 4 : The concept of facial-segments for detecting partially visible and occluded faces is explained in detail in this chapter. We propose four increasingly accurate face detectors and compare the performances of those methods with some of the best generic face detectors.

Chapter 5 : The concept of facial segments is applied to the task of facial attribute detection for partially visible faces. We present a method to combine local and global features to achieve more robust attribute estimation. Also, we show the usefulness of facial segment-based approach in improving the cross-domain detection performance.

Chapter 6 : We formulate the active authentication problem as a pattern-of-life verification task using location history data of smartphone users. The marginally-smoothed hidden Markov model (MS-HMM) approach is introduced here to handle the problems originating from the sparsity of data.

Chapter 7 : The usability of application-usage behavior information for active authentication is evaluated in this chapter. We present the results of insightful experiments performed to analyze the impact of unforeseen observations, unknown applications and authentication latency when using app-usage data.

Chapter 8 : We present several future research opportunities that are closely related to the research works presented in this dissertation.

Chapter 9 : Concluding remarks, summarizing the contributions of this dissertation are presented.

Chapter 2: Related Works

Among the AA techniques, the most explored are based on faces [13], [12], touch/swipe signature [59], multi-modal fusion [15], gait [60] and device movement-patterns/accelerometer [61], [18]. Face-based authentication is accurate but it requires heavy computational power and can cause faster battery drain if done continuously. On the other hand, swipe and accelerometer data alone are not discriminative enough. Among the other AA approaches, in [58], the authors fused stylometry with application usage, web browsing data and location information.

2.1 Face-based Authentication

Similar to other face-based authentication systems, in AA, face-based approaches try to tackle two challenge - detection and verification. Face detection is difficult in active authentication because of the high prevalence of partially visible and occluded faces, wide variation in lighting and pose, etc. For face-based verification the hurdle is to verify partially visible faces.

2.1.1 Face Detection

The detection problem in computer vision consists of predicting a bounding box for objects of interest in an image. Broadly, there are two approaches, one based on classification, the other on regression. In classification-based approaches, one generates proposals or considers bounding boxes of multiple sizes at each location exhaustively and then classifies each proposal by deciding if an object is present or not. Popular methods in this category include [62] and [63]. In regression-based approaches, the deep network predicts the bounding box's location and dimensions through regression, without having to go though a proposal generation stage, thus, making them faster than classification-based approaches. Prime examples include YOLO [64] and SSD [65]. Both YOLO and SSD use a single pass, regress the bounding box location and classifies its category. YOLO uses a single activation map, while SSD takes care of varying scales by using multiple activation maps. Note, that in the case of face detection using regression-based methods, the classification of the bounding box is not required, since only one type of 'object', the face, is to be detected.

Face detection, being one of the earliest applications of computer vision dating back several decades [66] [67], was not applicable in real-world settings until 2004 because of poor performance in unconstrained conditions [68]. The first algorithm that made face detection feasible in real-world applications was Viola and Jones's seminal work on boosted cascaded classification-based face detection [33], which is still used widely in digital cameras, smartphones and photo organization software. However, researchers found that the method works reasonably well only for near-frontal faces under normal illumination without occlusion [69] and proposed extensions of the boosted architecture for multi-view face detection, such as [70] [71]. Even these extensions had their shortcomings - they were difficult to train, and did not perform well because of inaccuracies introduced by viewpoint estimation and quantization [69]. A more robust face detector is introduced in [26] that uses facial components or parts to construct a deformable part model (DPM) of a face. Similar geometrical modeling approaches are found in [72] [73]. As support vector machines (SVMs) became effective for classification and robust image features like SURF, local binary pattern (LBP) histogram of oriented gradient (HoG) and their variants were designed, researchers proposed different combinations of features with SVM for robust face detection [68]. In recent year, the performance of the original DPM-based method is greatly improved by researcher in [74] which they denote as DPM Baseline. In the same paper, the authors introduced Headhunter, a new face detector that uses Integral Channel Features (ICF) with boosting to achieve state-of-the-art performance in face detection in the wild. In [75], the authors proposed a fast face detector that uses the scale invariant and bounded Normalized Pixel Difference (NPD) features along with a single soft-cascade classifier to handle unconstrained face detection. The NPD method is claimed to achieve state-of-the-art performance on FDDB, GENKI, and CMU-MIT datasets.

The performance break-through observed after the introduction of Deep Convolutional Neural Networks (DCNN) can be attributed to the availability of large labeled datasets, advancements in GPUs, the hierarchical nature of the deep networks and regularization techniques such as dropout [68]. In [25], the authors introduce a multi-task deep CNN architecture for multiview face detection which achieved state-of-the-art result on the FDDB dataset. Among other recent works, HyperFace [23] is a deep multi-task framework for face detection, landmark localization, pose estimation, and gender recognition. HyperFace exploits the synergy among related tasks by fusing the intermediate layers of a deep CNN using a separate CNN and thereby boosting their individual performances. Another multi-task learning approach is proposed in [76] for simultaneous face detection, face alignment, pose estimation, gender recognition, smile detection, age estimation and face recognition using a single deep CNN. Unlike HyperFace [23], the method in [76] utilizes domain-based regularization by training on multiple datasets in addition to AFLW and employs a robust, domain specific initialization.

Continuous authentication of mobile devices requires detection and verification of the user's face, even if it is partially visible, to operate reliably [31]. [30] and [77] are two known methods that explicitly address the partial face detection problem. Specifically, in [77] the authors achieve state-of-the-arts performance on the FDDB, PASCAL and AFW datasets by generating face parts responses from an attribute-aware deep network and refining the face hypothesis for partial faces.

2.2 Facial Attribute Detection

There has been significant amount of research on attribute extraction starting from learning separate models for each attribute [78] [79] to jointly learning multiple attributes in a multi-task learning fashion [80] [36] [37] [35] [2]. Multi-task optimization is found to improve performance in comparison to training independent models for each attribute detection task [37] [36].

In recent times, the research on attribute detection mostly revolves around two challenging, publicly available datasets namely, CelebA and LFWA [81]. Both datasets have annotations for forty different attributes along with identity information. The CelebA dataset contains $162,770$ images for training, $19,867$ image for validation and $19,962$ more for testing. It is a challenging dataset with wide variations in pose, illumination and image quality. The LFWA dataset is a much smaller dataset with 6263 training and 6880 test images. The datasets are introduced in [81] where the authors proposed a cascaded system of two DCNNs to jointly perform face localization and attribute detection. In [36], the authors addressed the multi-label imbalance problem of the CelebA dataset and proposed a mixed objective optimization network (MOON) that utilizes a unique loss function comprised of a mixed multitask objective with domain adaptive re-weighting.

Some authors, such as in [37] [2], categorized the attributes into different groups to take advantage of their mutual relationships. The authors in [37] suggested an auxiliary network on top of the multi-task DCNN to further exploit the relationships among the attributes. On the other hand, the authors in [2] defined a modified AlexNet with both shared and category-specific feature learning to assist attribute extraction.

Some researchers also implemented the attribute detection task as an auxiliary task of another task. For example, in [35], the authors proposed a DCNN architecture similar to Faster RCNN [82] with additional losses for joint detection of face and associated facial attributes without requiring explicit face alignment. However, the method does not address partial face detection, which is a challenging problem in itself [83]. Other notable attribute detectors for unaligned face are proposed in [84] and [1]. In [84], the authors proposed a cascade network to concurrently localize face regions to different attributes and perform attribute classification. While this method might be suitable for attribute extraction from partially visible faces if trained properly, the authors presented no such extension or analysis. Also, the original network is huge, consisting of separate DCNN branches for each of the 40 attributes and therefore not easily scalable. In [1], the authors introduced a data augmentation technique to assist attribute detection from unaligned faces. They improved the detection performance by augmenting the test data and combining the results. Even though their reported accuracy using an ensemble network of three ResNets is very good on unaligned faces, the architecture does not incorporate any mechanism for partially visible faces and also require combining scores from 162 transformations of the test image to achieve the best performance.

2.2.1 Face Verification

The objective of face verification is to determine whether two face images belong to the same person or not. Being one of the core problems in computer vision, it has been actively researched for over two decades [85]. The first step for a face verification system is to learn invariant and discriminative feature representation [86]. Features can be hand crafted, such as LBP features [87], Gabor wavelets-based encoding [88] etc., or learned from data, such as dictionary-based approaches [89], deep CNN-based approaches [90], [91] etc. The DCNN-based approaches, are tremendously popular now a days because of the superior performances of CNN-based methods and the wide availability of GPUs. For example, in [91], the researchers achieved results that surpass human performance for face verification on the LFW dataset. They used an ensemble of 25 simple DCNN with fewer layers trained on weakly aligned face images. The network is trained on fewer images than other DCNN approaches. Face verification networks proposed in [86] and [92] considered a more realistic scenario of unconstrained face verification on two very challenging datasets. In [93], the authors proposed a similarity model for cross domain face verification and evaluated it for challenging cross-domain matching tasks such as person re-identification under different views and verification of faces from still images and videos, older and younger faces, and sketch and photo portraits etc.

2.3 Authentication using Non-Face Modalities

2.3.1 Touch Gesture-based Authentication

In [19] and [94], the authors extracted behavioral feature vectors from the screen touch data and trained discriminative classifiers for authentication. On the other hand, in [15], the authors proposed kernel sparse representation and kernel dictionary learning-based methods for touch gesture-based active user authentication. These works have demonstrated the feasibility of using touch gestures as a biometric for active user authentication. Apart from touch gesture-based method, other sensors such as the acclerometer, gyroscope, speech, location traces etc. have been used to extract biometric features. For example, in [60] the authors proposed an authentication scheme for AA that recognizes user by gait pattern, whereas, in [61], [95], and [18] the authors train classifiers based on device movement-patterns/accelerometer information along with or without touch gesture. Authentication models that utilizes voice information are proposed in *Voice-Based Authentication*, [96].

2.3.2 Authentication based on Location History

We have performed some priliminary research on user verification based on location traces. Most location-based research reports are focused on data mining to obtain information about an individual's pattern of life. For example, in [97] the authors predict the users movement among the location points and infer user-specific activity at each location. In [98], the authors focus on detecting significant locations of a user and predicting the user's next location or infering the daily movements. In [99], [100] and [101], the authors infer the high-level behavior of the user, such as, the transportation modes on the way to the point locations. Other research efforts on GPS location data include driving behavior mining, finding mode of transportation and the most likely route, learing a Bayesian model of travel through an urban environment etc. [97].

When mining individual life patterns from geo-location data, the problem can be considered as a sequential pattern mining problem, widely used in health-care data processing, web usage analysis, text mining for natural language processing, speech processing, sequential image processing, bioinformatics and in many other domains [102]. In [97], sequential pattern mining has been employed for individual life pattern modeling. Since the geo-location trajectory data are spatio-temporal in nature, the fuzziness of space (usually no two point in the trajectory data are exactly the same) prevents the direct use of traditional frequent pattern mining algorithms. The usual practice is to cluster the geo-location points onto finite number of observation states and then perform sequential pattern mining on the state transition trajectories. In natural language processing, template matching approaches have been employed for matching features from a text sequence with pre-calculated feature vectors using edit distance or some other distance metric. String matching algorithms are also found to be effective in this regard. A different type of approach is based on building state-space models like a Markov Chain or a hidden markov model, from temporal data. Several research works on next place prediction from location history, such as in [98], [103], [104], are based on state-space models like Mobility Markov Chains (MMC), Mixed Markov Chain (MMM) and Hidden Markov Models (HMM).

2.3.3 Application Usage-based Authentication

In recent years, there has been a lot of focus on predicting individual and community-wise application usage patterns [105]. For example, in [41], the authors investigate the ratio of local and global applications in the top usage list, the traffic pattern for different application categories, likelihood of co-occurrence of two different applications and such other patterns in usage. In this work, the authors identify traffic from distinct applications using HTTP signatures. On the other hand, in [106] the authors use mobile in-app advertisements to identify the applications in network traces. Using the ad flow data, the authors tried to analyze the usage behavior of different types of applications. In [43], the authors analyzed the application-usage logs of over $4,000$ smartphone users worldwide to develop an app-usage prediction model that leverages user preferences, historical usage patterns, activities and shared aggregate patterns of application behavior.

From the authentication front, in [53], the authors proposed an application centric decision approach for active or implicit authentication in which applications are used as context to decide what modalities to use to authenticate a user and when to do it. Application usage data has also been used to generate scores for user authentication in [58]. The authors only considered the frequency of occurrence of an application in the training set to determine the likelihood of being a particular user, missing the temporal variation in the the usage pattern.

An interesting use-case of application-usage data is presented in [107]. The authors used a large-scale annotated application-usage dataset to build a predictor that can estimate where a person is (*e.g.*, at home or office) and if he/she is with a close friend or a family member. In [46], the authors used application usage traces along with system status and sensor indicators to predict the battery life of the phones using machine learning techniques.

2.4 Multi-modal Fusion

Various protocols for AA with and without multi-modal fusion have been suggested over the years. In [56], the authors explored the idea of progressive or risk-based authentication by combining multiple verification signals to determine the users level of authenticity. The AA system surfaces only when this level is too low for the content being requested. In [108], the authors proposed context aware protocols for more flexible yet robust authentication. In [109], the authors discuss three possible levels of fusion (a) fusion at feature level, (b) fusion at score level, and (c) fusion at decision level. Different fusion algorithms based on k-Nearest Neighbour classifiers, Support Vector Machines, decision trees, Bayesian methods, Gaussian Mixture Models (GMM) have been employed. [109], [110].

2.5 Active Authentication Datasets

Given the sensitive nature of smartphone usage data, there has been a scarcity of large dataset of front camera images in natural settings. However, the following datasets have been published in recent years that provided a platform to evaluate partial face detection methods in real-life scenarios.

2.5.1 Constrained mobile datasets

The MOBIO dataset [32] is a well-known dataset for face-based AA research. It contains 61 hours of audio-visual data from a NOKIA N93i phone (and a 2008

Mac-book laptop) with 12 distinct sessions of 150 participants spread over several weeks. However, since users were required to position their head inside an elliptical region within the scene while capturing the data, the face images of this dataset are constrained and do not represent real-life acquisition scenarios.

2.5.2 Private datasets

The largest known dataset on smartphone usage is the Google's Project Abacus data set consisting of 27.62 TB of smartphone sensor signals collected from approximately 1500 users for six months on $Nexus5$ phones [111]. Data was collected for the front-facing camera, touchscreen and keyboard, gyroscope, accelerometer, magnetometer, ambient light sensor, GPS, Bluetooth, WiFi, cell antennae, app usage and on time statistics. Google also collected the 114GB Project Move dataset, which consists of smartphone inertial signals collected from 80 volunteers over two months on $LG3$, $Nexus5$, and $Nexus6$ phones. The data collection was passive for both projects. However, neither of these two datasets are available for the research community.

2.5.3 Publicly available unconstrained mobile datasets

The AA-01 dataset [15] is a challenging dataset for front-camera face detection task which contains the front-facing camera face videos for 43 male and 7 female IPhone users under three different ambient lighting conditions. In each session, the users performed five different tasks. To evaluate the face detector, face bounding boxes were annotated in a total of 8036 frames of the 50 users. This dataset, denoted as AA-01-FD, contains 1607 frames without faces and 6429 frames with faces [4], [30]. The images in this are semi-constrained as the subjects perform a set task during the data collection period. However they are not required or encouraged to maintain a certain posture, hence the dataset is sufficiently challenging due to pose variations, occlusions and partial faces.

2.5.3.1 Other AA Datasets

The MIT Reality Dataset [112] consists of call logs, Bluetooth devices in proximity, cell tower IDs, application usage, and phone status (such as charging and idle) information from 100 Nokia-6600 smart phones users collected over $450,000$ hours. Since it focused on analyzing social behavior of the subjects, it does not contain vital biometrics such as face and touch. The Rice Livelab dataset [113] consists of information on application usage, wifi networks, cell towers, GPS readings, battery usage and accelerometer output of 35 users, collected from iPhone 3GS devices over durations ranging from a few days to less than a year.

The Geolife GPS trajectory dataset was collected in Microsoft Research Asia by 182 users over four years (from April 2007 to October 2011). The GPS trajectories of this dataset are represented by sequences of time-stamped points, each containing latitude, longitude and altitude information. The dataset contains $17,621$ trajectories with a total distance of $1,251,654$ kilometers and a total duration of $48,203$ hours. These trajectories were recorded by different GPS loggers and

GPS-phones, and have a variety of sampling rates. 91 percent of the trajectories are logged in a dense representation, e.g. every $1 \sim 5$ seconds or every $5 \sim 10$ meters per point [114], [115]. Apart from the GPS trajectories, the dataset contains information about a broad range of users' outdoor movements, including not only life routines like going to work or home, shopping, hiking etc. but also some entertainments and sports activities, such as shopping, sightseeing, dining, hiking, and cycling. This trajectory dataset has been used in many research fields, such as mobility pattern mining, user activity recognition, location-based social networks, location privacy, and location recommendation [99], [101], [97]

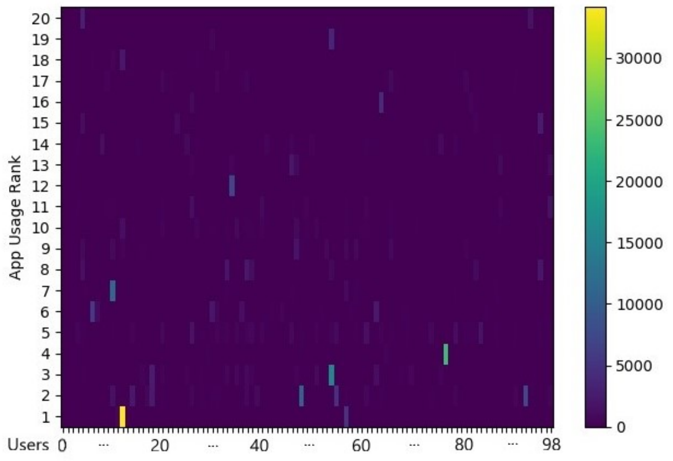


Figure 2.1: Similarity matrix depicting top 20 application-usage rate among users in the training set of the Securacy dataset.

The Securacy dataset[1] [116] from the Center of Ubiquitous Computing, University of Oulu was originally created within the context of exploring the privacy and security concerns of a smartphone user by analyzing the location of servers that different applications use and whether secure network connections are used. For

[1]Available at `http://ubicomp.oulu.fi/securacy-understanding-mobile-privacy-and-security-concerns/`

Table 2.1: General information on application-usage data available in the Securacy dataset.

No. of Subjects with ≥ 500 training samples and ≥ 200 test samples for sampling rate of $1/30s^{-1}$ (Train/Test)	201/99
Avg. No. of Sessions/User with App-Usage Data of the 99 selected subjects (train/test)	$\sim 119/\sim 96$
Train/Test split for the experiment	70%/30%
Total Number of Unique Applications Used by the 99 selected subjects (train/test)	1340/554
Average Number of Samples Per User for the 99 selected subjects (train/test)	$\sim 2235/\sim 1745$

a period of approximately six months, the data was collected from 218 anonymous participants who installed the data collection application from the Google Play store. The collected data, 679.90 GB, includes the currently running foreground application, installed, removed or updated applications, application server connections and device location, etc. Out of the 218 users of the original Securacy dataset, 99 are used for experiment who has more than 500 training samples and more than 200 test samples for any sampling rate between $1/5s^-1$ and $1/30s^-1$. The usage rate for the top 20 applications for each of the 99 subjects are show in fig. 2.1. The application usage data for the 99 subjects in the Securacy dataset are summarized in Table 2.1 and the corresponding usage statistics for the top 20 applications are presented in Table 2.2. Note that the top applications ranked 1st, 2nd and 4th in Table 2.2 are actually the same application written in Spanish, English and Finnish, respectively. Similarly, rank 12, ‘Horloge’ is ’Clock’ in French, and therefore is the same application as rank 19. However, these applications are shown separately here because, for the active authentication problem, even the preferred language of the user is a type of biometric metadata and can be used to discriminate between users.

Also, similar to UMDAA-02 dataset usage statistics, there are several applications in the top 20 rank that were actually used by only a few users very frequently (ranked 1, 4, 9, 12, 16). For this dataset, this phenomenon can be attributed to language difference as well because if the language difference were nullified, then rank 1, 2, 4 will collapse at rank 1 and rank 12 and 19 will collapse at 12 - thereby removing three applications from the list (rank 1, 4 and 12) that has very few users. For the user verification research presented here, the language variation is kept unaltered in order to retain the naturalness of the dataset and the algorithms are expected to learn to discriminate between users based on the language as well as on usage pattern.

Table 2.2: App-usage statistics for the top 20 apps for the 99 selected users of the Securacy Dataset.

Rank	App Name	No. of Users	Per User Usage	Overall Usage
1	Sistema Android	4	9972.25	402.92
2	Android System	80	480.44	388.23
3	com.android.keyguard	34	802.79	275.71
4	Android-jrjestelm	5	4820.8	243.47
5	System UI	80	242	195.56
6	Nova Launcher	19	794.79	152.54
7	Maps	38	363.08	139.36
8	Google Search	53	214.3	114.73
9	Launcher	12	650	78.79
10	Chrome	60	128.2	77.7
11	Facebook	49	154.53	76.48
12	Horloge	1	7328	74.02
13	YouTube	49	144.94	71.74
14	TouchWiz home	20	348.3	70.36
15	Securacy	84	75.39	63.97
16	Internet	16	371.25	60
17	WhatsApp	37	154.62	57.79
18	Google Play Store	72	71.83	52.24
19	Clock	44	113.89	50.62
20	Package installer	36	138.69	50.43

Chapter 3: Active Authentication Benchmark Experiments on the UMDAA-02 Dataset

3.1 Description of the UMDAA-02 Dataset

The UMDAA-02 data set [1] consists of 141.14 GB of smartphone sensor signals collected from 48 volunteers on Nexus 5 phones over a period of 2 months (15 Oct. 2015 to 20 Dec. 2015). The data collection sensors include the front-facing camera, touchscreen, gyroscope, accelerometer, magnetometer, light sensor, GPS, Bluetooth, WiFi, proximity sensor, temperature sensor and pressure sensor. The data collection application also stored the timing of screen lock and unlock events, start and end time stamps of calls, currently running foreground application etc. The volunteers used the research phone as their primary device for a week and were given the option to stop data collection at will and review the stored data prior to sharing.

In Table 3.1, the most significant information for each modality associated with the sensor data is presented. Data for most of the modalities are stored when there is significant change in that modality. For example, the GPS data is stored at a rate proportional to the movement speed of the phone. The front camera images are captured only for the first 60 seconds for each session at a rate of 3 fps.

[1] Available at `https://umdaa02.github.io/`

Table 3.1: Significant Information for Each Modality Per Session

Modality	Information
Accelerometer	Event Time, X, Y, Z
Gyroscope	Event Time, X, Y, Z
Image	Shutter Time, Filename
Bluetooth	Developer, Paired/Unpaired Flag
Location	Event Time, Lat., Long., Accuracy
Usage	Event Time, % CPU, % Memory
Magnetic Field	Event Time, X, Y, Z
Gravity	Event Time, X, Y, Z
Connectivity	Capture Time, Flag (Bluetooth, Gps, Wifi, Cell Network), Network Name and Code
Foreground App Info	Start Time, Duration, End Time, App Name, Launched From Home Flag
WiFi	SSID, BSSID, Authentication Type, IP Address, RSSI
Ambient Light	Event Time, Value
Ambient Cells	MCC, CI, MNC, Sig. Strength, TAC
Screen	Event Time, Key
Motion/Touch	Event Time, Type, Pressure, Major-Minor Axis, Position
Call	Event Time, Key
Key	Event Time, Pressure, Type, Key Code
Screen Res	Event Time, X, Y

Table 3.2: Information on UMDAA-02 and UMDAA-02-FD Dataset

Description	UMDAA-02	UMDAA-02-FD
No. of Subjects	36M, 12F	34M, 10F
Age Range (years)	$22 - 31$	$22 - 31$
Avg. Days/User (days)	$\backsim 10$	$\backsim 10$
Avg. Sessions/User	$\backsim 248$	$\backsim 200$
Total Number of Images	600712	33209
No. of Images without Faces	$-$	9060
Avg. Images/User	$\backsim 12515$	$\backsim 755$
Avg. Images/Session	$\backsim 51$	$\backsim 4$
Min. no. of Image for a User	1038	64
Max. no. of Image for a User	49023	2787

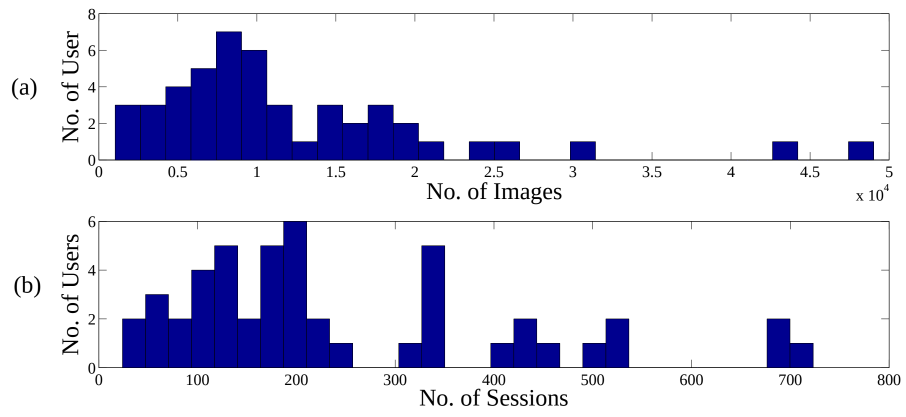


Figure 3.1: (a) Histogram of number of images per user, and (b) histogram of number of sessions per user.

Some general information on the dataset is provided in Table 3.2. The usage information is arranged in 'Sessions' which starts when the user unlocks the phone and ends when the phone goes to the locked state. The data is stored in nested folders with the year, month, day and start time of the session embedded in the folder names.

3.2 Face Detection and User Verification

In this section, we describe face detection and verification tasks from faces captured by the front-facing camera. Fig. 3.1 shows histograms of number of images per user and the number of sessions per user. The number of images varies between 2000 to $50,000$ per user and the number of sessions varies between 25 and 750, thus providing a large number of images for each user and session.

UMDAA-02-FD Face Detection Dataset: State-of-the-art face detection algorithms that perform satisfactorily on datasets like faces-in-the-wild [28], [29] are

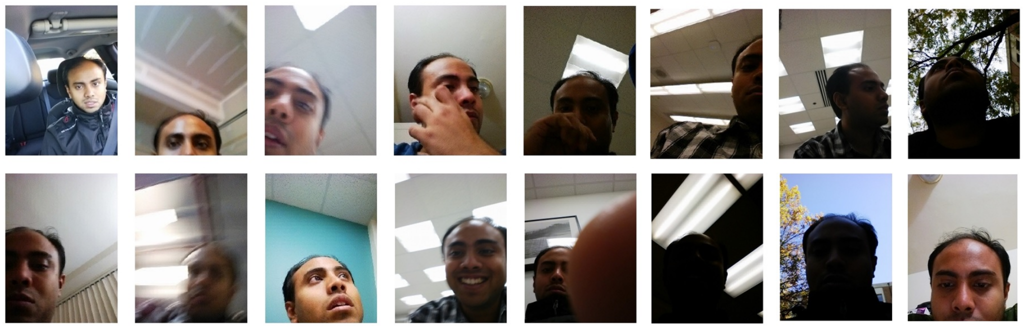


Figure 3.2: Sample images from one of the users showing a wide variety of pose, illumination, occlusion and expression variations.

not suitable for detecting partially visible faces that are typically present in the UMDAA-02 dataset. Moreover, for practical implementation purposes, the algorithm must be very fast and have a high recall rate to ensure continuous authentication [4]. A few sample images are shown in Fig. 3.2 which shows that the faces suffer from partial visibility, illumination changes, occlusion and wide variation in poses and facial expressions.

Excluding the data of 5 users from a phone whose front camera malfunctioned during data collection phases, a set of 33209 images was selected from all sessions of the remaining 43 users at an interval of 7 seconds. The images were manually annotated for ground truth face bounding box, face orientation and five landmarks - left eye, right eye, nose, left and right corners of the mouth to create the UMDAA-02 face detection dataset (UMDAA-02-FD). Some information on the UMDAA-02-FD is provided in Table 3.2. The chronology and session information of all the images are also available. The histogram of face height and width distribution shown in Fig. 3.3 indicates that face widths vary approximately from 400 to 650 pixels, while face heights vary approximately from 300 to 700 pixels. The database contains many

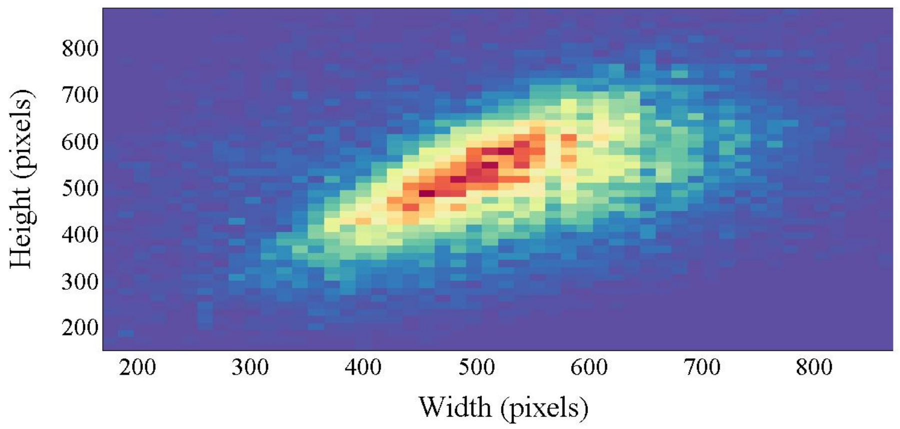


Figure 3.3: Distribution of bounding box width and heights

Table 3.3: Comparison between FD methods at 50% overlap

Method	Accuracy	F1-Score	Time/Image(s)
VJ [33]	60.24	64.50	**0.16**
DPM [26]	62.62	65.50	5.51
LAEO [73]	19.40	32.49	4.57
FSFD(C_{best}) [4]	73.48	79.11	0.68
DP2MFD [117]	**76.15**	**82.83**	15.0(CPU), 0.8(GPU)

partial faces as can be seen from the extremities of the distribution, information from which can help tune the hyper-parameters of face detectors.

Evaluation of Face Detection Performances: Accuracy and F1-score measures are adopted as evaluation metrics for face detection to ensure that both precision and recall performances are taken into consideration. The processing time per image is also measured to analyze the suitability for real-time operations. Prior to face detection, the images are down sampled by 4 to ensure reasonable processing time for all algorithms. 50% intersection-over-union overlap between the detection results and the ground truth bounding box is considered to be the threshold for correct detection.

The performances of four face detection algorithms on the UMDAA-02-FD dataset are presented in Table 3.3. The recently proposed Facial Segment-Based Face Detector (FSFD) algorithm [4] (with number of random subset $\zeta = 20$ and minimum number of segments $c = 2$), which is specifically designed for detecting partial faces, performs better than other popular non-commercial detectors like Viola-Jones (VJ) [33] and Deformable Part-based Model (DPM) [26] and in reasonable processing time. Another recent FD technique, the Deep Pyramid Deformable Part Model (DP2MFD) [117] utilizes normalized convolutional neural network (CNN) features. It outperforms all the other methods in terms of Accuracy and F1-Score but the processing time is quite long (almost 100 times more than VJ) thus making it unattractive for realtime implementation on smartphones. However, the best scores are far from satisfactory and better face detectors for AA are needed.

Face-based User Verification: Face verification is performed on the UMDAA-02-FD dataset. For each annotated face, 68 fiducial landmarks are extracted using the Local Deep Descriptor Regression (LDDR) method trained on Imagenet and FDDB datasets [118]. Feature extraction is performed after alignment, centering and cropping.

Feature Extraction from Faces: Given a face image, pixel intensity, LBP [119] and CNN features using the pre-trained Alexnet network [120] and the DCNN network [121] are extracted. In total, 6 different features are extracted for each face as shown in fig. 3.4.

- F_1: Pre-processed faces are converted to grayscale, rescaled (32×32) and

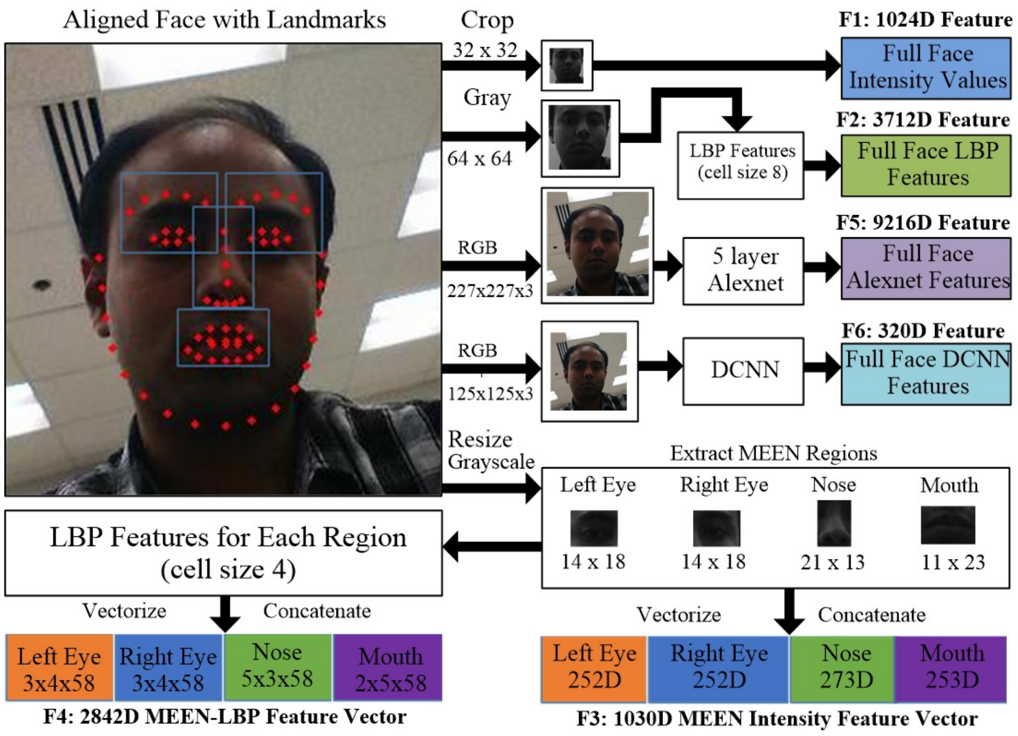


Figure 3.4: Flow diagram for features extraction for face verification.

vectorized (1024 dimensional vector).

- F_2: From the 64×64 rescaled grayscale image, LBP features of size $8 \times 8 \times 58$ (3712 dimensional vector) are extracted for a cell size of 8×8 pixels.

- F_3: Bounding boxes of the eyes, nose and mouth are computed from the landmarks with a 5 pixel margin for each face part from the pre-processed grayscale image. The eyes, nose and mouth bounding boxes are resized to 14×18, 21×13 and 11×23 pixels respectively, then vectorized to a 1030 dimensional MEEN feature [13].

- F_4: LBP features (2842 dimensional) are obtained from each of the resized bounding boxes of MEEN parts (F_3) with a cell size of 4×4 pixels.

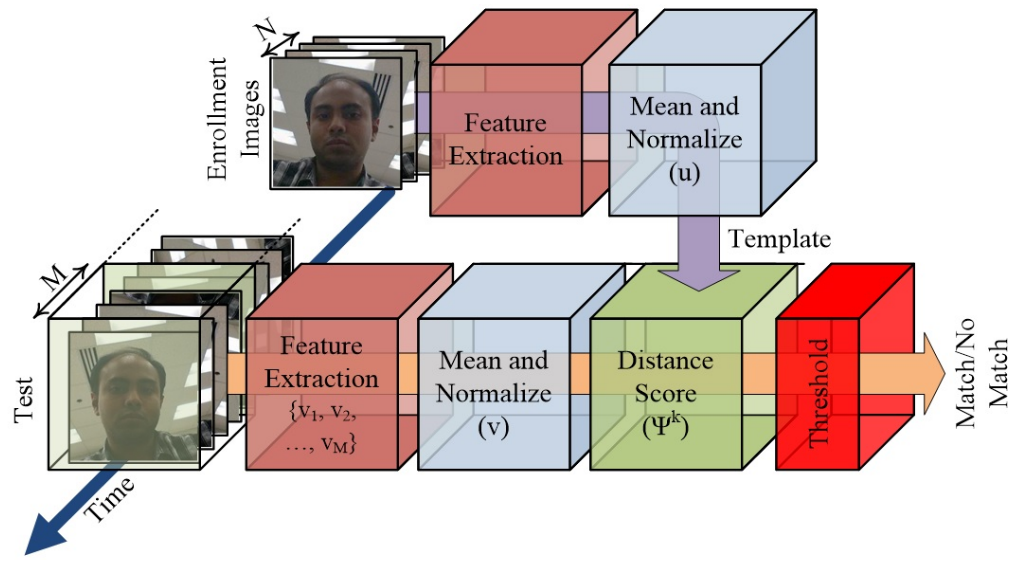


Figure 3.5: Block diagram depicting the face verification protocol.

- F_5: The first five convolutional layers of Alexnet are used to extract features of size $6\times6\times256$ (9216 dimensional) from resized color images of faces (227×227)

- F_6: Landmarks are input to the DCNN based face verification system [121] trained on the CASIA-WebFace dataset [122], which resizes the face to $(125 \times 125 \times 3)$ and then outputs a 320 dimensional feature vector.

Evaluation Protocol: Six types of feature vectors are considered in this experiment. In the absence of any particular enrollment data, to simulate a practical AA scenario, the faces are sorted chronologically for each user and the first N faces are considered for enrollment while the rest are used for verification. The mean of the features of the enrollment set of a user followed by $L2$ normalization of the mean vector is stored as his/her template u.

Fig. 3.5 shows a block diagram of the verification process. A reasonable, practical assumption for robust AA is that the user is verified by the last M faces instead

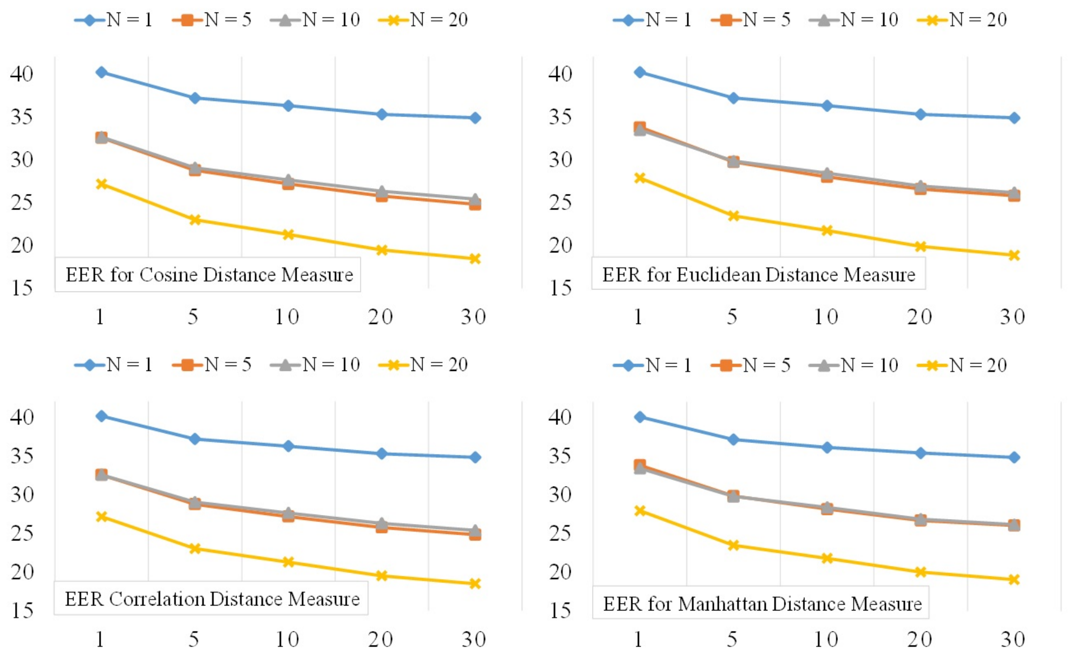


Figure 3.6: EER (%) vs. M for varying N using DCNN features (F_6) and four different metrics.

of a single one. Therefore features v_i $(i = 1, 2, \ldots, M)$ are extracted from each of the M faces for each location of the moving window, then averaged and $L2$-normalized to form the test vector v. The distances between v and u are calculated using four distance measures, namely, Euclidean Distance (EU), Cosine Distance (CosD), Manhattan Distance (MD) and Correlation Distance (CorrD). For the distance measure δ^k of type k the score is $\Psi^k = \frac{1}{\delta^k}$ [12].

Experimental Results: In Fig. 3.6, the equal error rate (EER) (%) produced by using F_6 features are plotted for varying M and N values for the four distance measures. It is evident from the plots that the EER decreases with increasing N and M for all the cases. The lowest EER of 18.44% is achieved for $N = 20$, $M = 30$ using either CorrD or CosD measure.

Fig. 3.7 shows the EER corresponding to different features and distance mea-

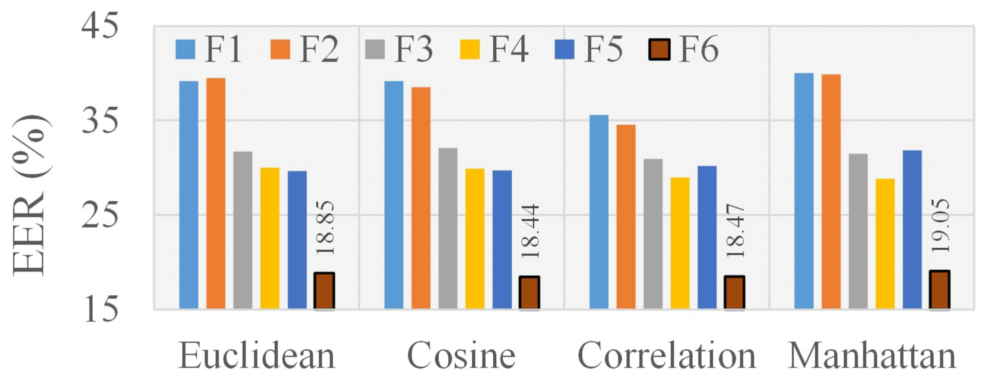


Figure 3.7: EER(%) for 6 feature vectors using four metrics.

sures considering $N = 20$, $M = 30$. The DCNN features (F_6) are found to be the most effective (EER of 18.44% for CosD). Since, for a reliable system the EER is expected to be at least less than 5%, this value is not satisfactory at all. The poor performance may be due to the fact that many faces in the dataset are partially visible and therefore alignment using facial landmarks fails badly for these cases. Also, matching the features from a partial face to the features of the same user's full face may result in a large distance measure. Among the other methods, the Alexnet network does not perform much better than the non-CNN features in this scenario as it is not trained particularly for faces. The LBP of MEEN face (EER of 28.83% for MD) gives the best result among non-CNN features. Note that in practice, the CNN feature extraction step is generally much slower than the non-CNN feature extraction methods without the use of a GPU. Thus, more robust yet fast verification methods are needed to produce satisfactory performance on this dataset.

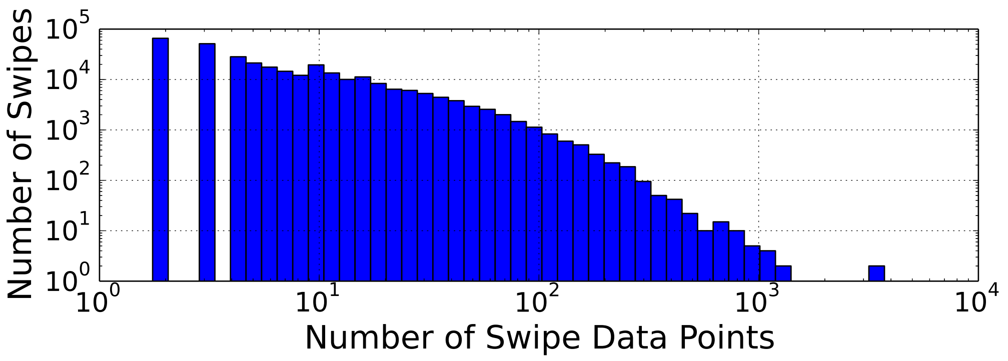


Figure 3.8: Histogram of the number of data points per swipe.

3.3 User Identification Using Swipe Dynamics

In this experiment, single finger touch sequences (swipes) on the screen are studied by considering three types of events - finger down, in-touch and finger up. The length of swipes vary between 1 to 3637 touch data points (Fig. 3.8). For reliable authentication using swipes, longer ones are preferable [19]. Hence, swipes with more than four data points are considered for feature extraction. Table 3.4 summarizes the swipe dataset, shows that it contains a large number of touch and swipe data per user and therefore can serve as a data set for practical experiments on swipe-based authentication. Since the users were not given any particular task to perform, the touch data in AA-02 is representative of how users interact with the phone through touch.

Feature Extraction: Every swipe s is encoded as a sequence of 4-tuples $s_i = (x_i, y_i, p_i, t_i)$ for $i \in 1, \ldots, N_c$ where x_i, y_i is the location coordinates and p_i is the pressure applied at time t_i. N_c is the number of data points captured during the swipe. From each swipe-action data with $N_c \geq 5$, a 24-dimensional feature vector,

Table 3.4: General Information on Swipe Data

No. of subjects	48
Avg. Session/User with swipe data	$\backsim$ 196
Total taps (finger down-finger up)	177417
Total swipes (including taps)	489723
Maximum data points in a swipe	3637
No. of Swipes/User	$\backsim$ 10203
No. of Swipes/Session	$\backsim$ 52
No. of Swipes ($>$ 4 data points)	$\backsim$ 167126
No. of Swipes/User ($>$ 4 data points)	$\backsim$ 3482
No. of Swipes/Session ($>$ 4 data points)	$\backsim$ 18

Table 3.5: Features Extracted From Each Swipe Event

Features	Description
1-2	inter-stroke time, stroke duration
3-6	start x, start y, stop x, stop y
7-8	direct end-to-end distance, mean resultant length
9	up/down/left/right flag
10-12	20%, 50%, 80% -perc. pairwise velocity
13-15	20%, 50%, 80%-perc. pairwise acc
16	median velocity at last 3 pts
17	largest deviation from end-to-end (e-e) line
18-20	20%, 50%, 80%-perc. dev. from e-e line
21	average direction
22	ratio of end-to-end dist and trajectory length
23	median acceleration at first 5 points
24	mid-stroke pressure

listed in Table 3.5, is extracted using the method described in [19] and [15]. Note, in the UMDAA-02 dataset, the measure of area covered by the finger is not present.

Experimental Setup and Evaluation: The swipe data for each user (with $N_c \geq 5$) are sorted chronologically and the first 70% swipes are considered for training-validation while the rest for testing. After extracting the 24-dimensional feature vector from each swipe, the training feature matrix is normalized to zero mean and unit variance. Then individual binary classifiers are trained for each user following the one-vs-all protocol. The classification methods considered for this experiment are k-nearest neighbor (KNN) [19], Gaussian kernel Support Vec-

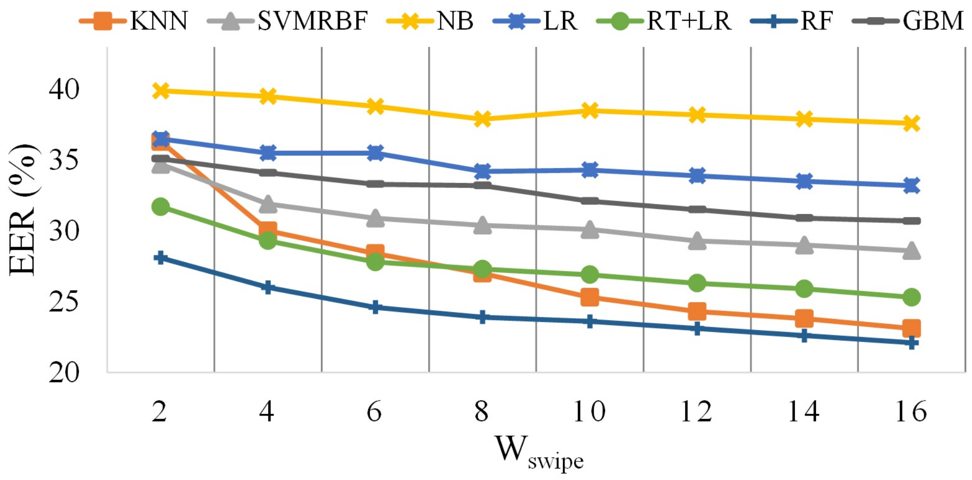


Figure 3.9: EER vs. W_{swipe}.

tor Machine (RBF-SVM) [19], Naive Bayes (NB) [59], Linear Regression (LR) [59], Random Tree estimation followed by Linear Regression (RT+LR), Random Forest estimator (RF) [123], [14], [59] and Gradient Boosting Model (GBM) [124]. The methods are compared based on EER (%).

As proposed in [19], instead of using a single swipe for authentication, the scores of multiple, consecutive W_{swipes} number of swipes are averaged together for robustness. Since all of the methods return confidence probabilities/scores or distance from separating hyper-plane representing confidence, the score fusion is a simple average of individual scores. For the nearest neighbor-based methods, nine neighbors are considered. The parameters of RBF-SVM are tuned by 10 fold cross validation on smaller subsets of the original training data. Since the training data is very large, the SVM is trained on a reduced subset, followed by retraining on the hard negative mined error cases. For the ensemble-based methods, the number of estimators is set to 200 and the maximum tree depth is set to 10. The EER values

Table 3.6: General Information on Geo-location Data

No. of Subjects	45
Avg. No. of Sessions/User with Location Data	∽ 186
Total Number of Location Traces	8303813
Number of Location Traces Per User	∽ 184529
Number of Location Traces Per Session	∽ 993

obtained using different methods for differnet W_{swipe} values are show in Fig. 3.9. The random forest (RF) estimation method outperforms all the other methods and can reach an EER of 22.1%. However, for practical usage, this EER is not satisfactory and therefore achieving a better performance for this dataset is a new research challenge.

3.4 Geo-location Data and Next Place Prediction

The location service of smartphones return geographical location of the user based on GPS and WiFi network. Excluding the users who kept their location service off, geolocation data, stored only if there is significant change in the location, is obtained from 45 users (summarized in Table 3.6). It is possible to reasonably predict the next location that a person might visit based on prior knowledge on the pattern on one's life. In this section, the next place prediction problem is approached using the geolocation data available in the UMDAA-02 dataset.

State Definition for Mobility Markov Chains: Location histories are first clustered into N_i clusters, namely $C_i^1 \dots C_i^N$, for the i-th user using the DBSCAN algorithm [125] based on distances between data points. The maximum distance between a point from the center of the cluster in which that point belongs is set

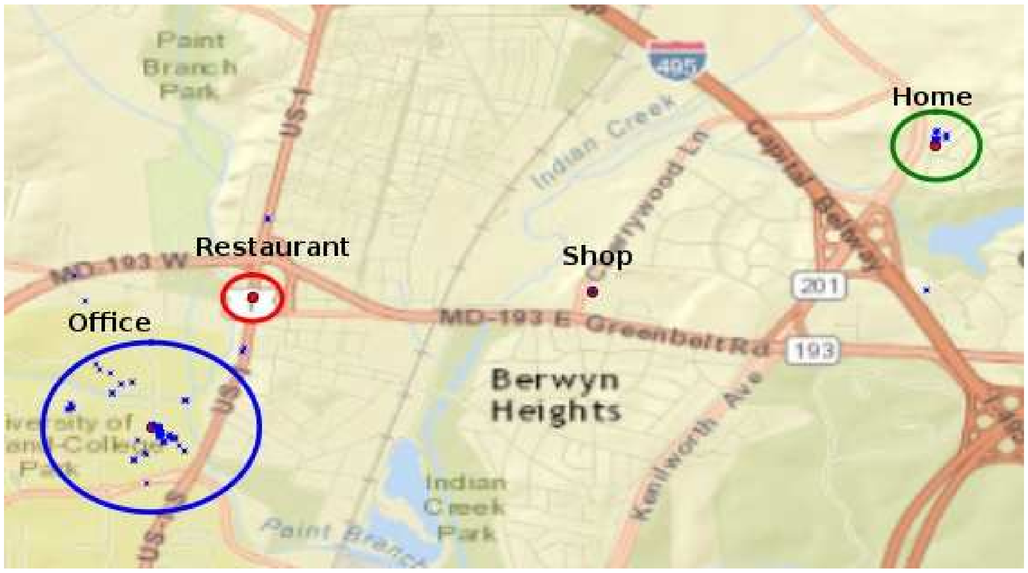


Figure 3.10: Example of Geo-location Data Clustering - Analysis of the clusters reveal states of the user such as 'Home' or 'Work'.

to be below a certain value R meters. Such clustering for a student (shown in Fig. 3.10) reveals the expected dominant regions that the user would visit - home, university, a certain shop and a restaurant. Two additional clusters, Transit (Tr) and Unknown (Unk), are also assigned for each user. If the user is traveling, causing location information to change rapidly ($\geq 2ms^{-1}$), then he/she is assigned to Tr. The remaining data points are denoted as Unk.

Data points at each cluster are assigned to six different observations based on the day and time information. Weekdays and weekend data points are flagged with WD and WE. Also, the whole day is divided into three time zones (TZs) - TZ1 (8:00 am to 4:00 pm), TZ2 (4:00 pm to 10:00 pm) and TZ3 (10:00 pm to 8:00 am). Thus, for the i-th user, there are $(N_i + 2) \times 2 \times 3$ possible observation states. However, since the location service only collects data when the phone is unlocked, there are many gaps in the data and it is possible that many of these observation states are absent in the training phase but present in the test data or vice-versa.

The location service data is utilized for development and evaluation of Mobility Markov Chains (MMC) [98], [103] which is a discrete stochastic process model of the mobility behavior of an individual in which the probability of moving to a state depends only on the last visited state and the transition matrix for all probable states. Thus an MMC is composed of a set of k-states $S = s_1, s_2, \ldots, s_k$, prior probability of entering a state $p_1, p_2, \ldots, p_k$ and a set of transitions $t_{i,j}$ where $t_{i,j} = Prob(X_n = s_j | X_{n-1} = s_i)$.

Experimental Setup and Evaluation: From the chronological organization of a user's mobility traces, the first 70% are used for training while the rest for testing. Each trace of the training set is tagged with a unique tag identifying the state it belongs to. The prior and transition probabilities of each state are calculated from the chronological traces. Since, the number of states for a subject depends upon the maximum radius parameter R for the clusters, nearby small clusters get merged into bigger ones with increasing R causing a reduction in the number of states. In the training set, the average number of states per user drops to 35 from 144 if the maximum radius is increased to 500 m from 20 m.

MMC-based next location prediction results in terms of $Accuracy$ and $Accuracy3$ (percentage of times the correct next location was among the top 3 most probable locations) metrics are presented in Fig. 3.11. The horizontal axis represents the number of previous observations. Considering n previous observations, the MMC algorithm returns probabilities of each state to be the next. Since the day and time zone of the next location are known, states that do not belong to that day and

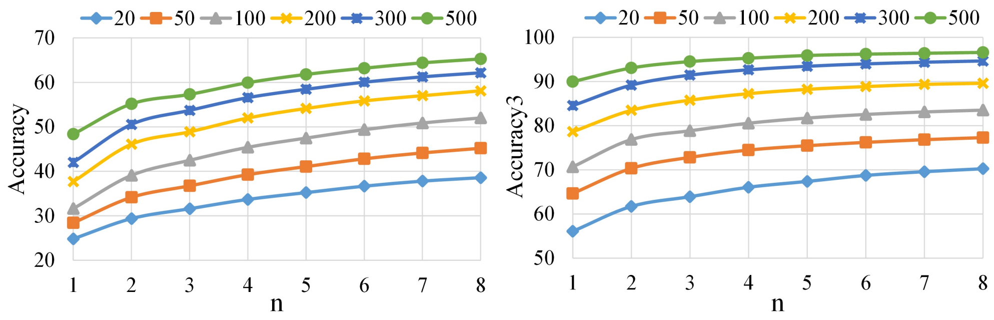


Figure 3.11: Next location prediction *Accuracy* (left) and *Accuracy*3 (right) measures for increasing number of previous observations for MMC at different R.

Table 3.7: General information on application-usage data available in the UMDAA-02 dataset.

No. of Subjects with ≥ 500 training samples and ≥ 200 test samples for sampling rate of $1/30s^{-1}$ (Train/Test)	32/26
Avg. No. of Sessions/User with App-Usage Data of the 26 selected subjects (train/test)	$\sim 582/\sim 197$
Train/Test split for the experiment	70%/30%
Total Number of Unique Applications Used by the 26 selected subjects (train/test)	119/67
Average Number of Samples Per User for the 26 selected subjects (train/test)	$\sim 4307/\sim 1399$

time zone are dropped. The most probable state among the rest of the states is picked as the next predicted location. Fig. 3.11 indicates that knowing more prior states increases the accuracy. The accuracy also increases with increasing maximum radius R (from 20 meters to 500 meters) at the cost of localization capability. Between the two measures, *Accuracy*3 is can go much higher (*Accuracy* = 65.3% and *Accuracy*3 = 96.6% for $R = 500$ meters, $n = 8$) indicating the feasibility of location prediction.

3.5 Application-Usage Data in UMDAA-02 Dataset

The UMDAA-02 dataset is specifically designed for evaluating active authentication systems in the wild. The dataset consists of 141.14 GB of smartphone sensor data collected from 45 volunteers who were using Nexus 5 phones in their regular daily activities over a period of two months. The data collection application ran completely in the background and the collected data includes the front-facing camera, touchscreen, gyroscope, accelerometer, magnetometer, light sensor, GPS, Bluetooth, WiFi, proximity sensor, temperature sensor and pressure sensor among with the timing of screen unlock and lock events, start and end timestamps of calls and currently running foreground application, etc. The application usage data from 45 users is summarized in Table 3.7. However, not all the users have adequate amount of usage data. For all the experiments in this dissertation, a total of 26 users are used who has more that 500 training samples and more than 200 test samples for any sampling rate between $1/5s^{-1}$ to $1/30s^{-1}$. The usage statistics for the top 20 applications for the selected 26 subjects is presented in Table 3.8. The usage rate for the top 20 applications for each user is shown in fig. 3.12(a). From the table and the figure, it is readily seen that the applications ranked 6th, 8th, 12th and 20th are in the top list because of excessive usage by very few users, where as, the remaining applications are genuinely popular among the users.

Table 3.8: App-usage statistics for the top 20 apps for the 26 selected users of the UMDAA-02 dataset.

Rank	App Name	No. of Users	Per User Usage	Overall Usage
1	com.google.android. googlequick-searchbox	26	283.27	283.27
2	com.android.dialer	25	255.24	245.42
3	com.whatsapp	15	303.6	175.15
4	com.android.chrome	26	141.42	141.42
5	com.facebook.katana	11	308.18	130.38
6	com.nextwave.wcc2	1	2366	91
7	com.google.android.youtube	16	144.38	88.85
8	com.ea.game.pvzfree	2	872.5	67.12
9	com.google.android.gm	24	51.04	47.12
10	com.android.mms	22	52.09	44.08
11	com.google.android.talk	18	62.28	43.12
12	com.andrewshu.android.reddit	1	842	32.38
13	com.nextbus.mobile	19	41.89	30.62
14	com.google.android.apps.docs	24	33	30.46
15	com.android.settings	24	27.71	25.58
16	com.google.android.apps.maps	14	44	23.69
17	com.android.camera2	22	20.5	17.35
18	com.google.android.gallery3d	17	24.94	16.31
19	com.android.vending	21	20.1	16.23
20	com.viber.voip	5	74.6	14.35

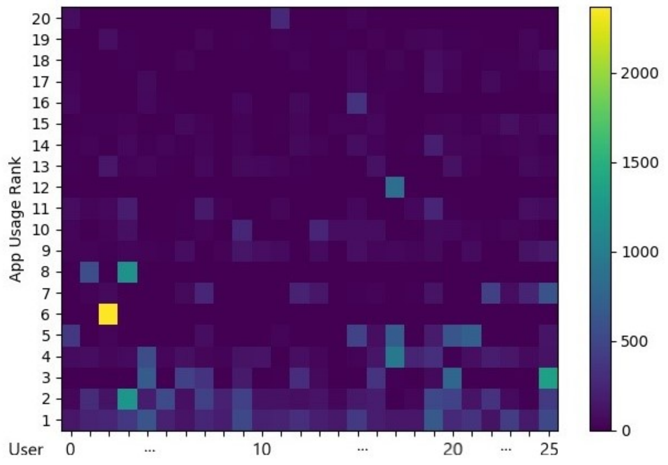


Figure 3.12: Similarity matrix depicting top 20 application-usage rate among users in the training set of the UMDAA-02 dataset.

Chapter 4: Facial Segments-based Face Detection for Active Authentication

One of our research goals is to perform face-based user authentication from the front camera images of mobile devices. The first step of such a system is to reliably detect the user's face from the image. As we mentioned earlier, face detection in the mobile domain is a very challenging task itself and we have proposed a novel facial segment-based approach to face detection considering the frequent occurrence of partially visible faces in the mobile domain face datasets. We have developed several face detection algorithms which can be divided into two general approaches - proposal-based classification methods and regression-based end-to-end detection methods. The approaches and the corresponding face detection methods are described in this chapter.

4.1 Proposal-based Methods for Face Detection in Mobile Domain

In proposal-based detection methods, bounding boxes are generated as face proposals based on facial segments. The detectors are trained on the proposals and corresponding facial segment information to classify each segment as face/non-face with confidence scores. Three detectors of this type are presented along with a fast

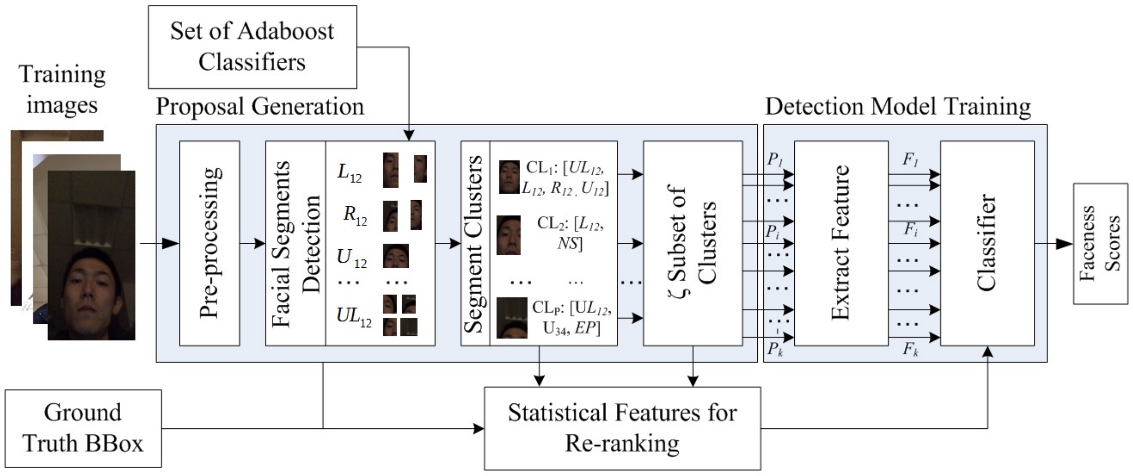


Figure 4.1: Block diagram showing the general architecture of a face segment to face detector, with components such as facial segments-based proposal generator, feature extractor, classifier and re-ranking based on prior probabilities of segments [5].

proposal generation mechanism. The detectors are

1. Facial Segment-based Face Detection (FSFD) [4]

2. SegFace [5], and

3. DeepSegFace [5].

Here, FSFD is a simple yet effective approach, SegFace has a more robust formulation and DeepSegFace is a deep CNN-based approach that is even more robust and accurate. A general pipeline for the proposal-based approach is shown in Fig. 4.1. As can be seen from the figure, the whole pipeline can be divided into two major parts - proposal generation and detection model training.

4.1.1 Proposal generation

The basis of our approach is dividing the task of face detection among different segments of the face. By segments, we mean portions of the face such as left half, right half, upper half, bottom half, nose segment etc. We divide a face into 14 such facial segments (as shown in Fig. 1.2) using 21 fiducial keypoints (as shown in Fig. 4.2) which are upper-left-half (UL12), upper-half (U12), upper-right-half (UR12), upper-left-three-fourth (UL34), upper-three-forth (U34), upper-right-three-fourth (UR34), left-half (L12), left-three-fourth (L34), eye-pair (EP), nose region (NS), right-half (R12), right-three-fourth (R34), bottom-three-fourth (B34), and bottom-half (B12). Let us denote the points shown in Fig. 4.2 with (x_k, y_k) where x_k and y_k are the horizontal and vertical pixel distances from the $(0, 0)$ pixel coordinate (top-left corner) of an image image of width W and height H, and $k \in \{\text{TL, BR}, 1, 2, \ldots, 21\}$. TL and BR corresponds to Top-Left and Bottom-Right coordinate of the full face bounding box. The fiducials and full face bounding boxes are obtained from All-in-One Face [126] along with visibility scores v_k where $k \in \{\text{TL, BR}, 1, 2, \ldots, 21\}$. Now, $\forall v_j|_{j=1}^{21} \geq \tau$ where τ is a visibility threshold, the bounding boxes of segments $\boxdot_L$, were $L \in \{\text{UL12, UR12}, \ldots, \text{B12}\}$are defined as

$$\begin{aligned} \boxdot_{\text{EP}} \;=\; & [max(x_{\text{TL}}, min(x_i|_{i=1}^{12})), max(y_{\text{TL}}, min(y_i|_{i=1}^{12}) - \Delta_{\text{EP}}), \\ & min(x_{\text{BR}}, max(x_i|_{i=1}^{12})), min(y_{\text{BR}}, max(y_i|_{i=1}^{12}) + \Delta_{\text{EP}})] \end{aligned}$$

$$\begin{aligned} \boxdot_{\text{NS}} \;=\; & [max(x_{\text{TL}}, min(x_i|_{i\in\{8,14,15,16,18\}})), max(y_{\text{TL}}, max(0, \frac{1}{3}\sum_{i=14}^{16} y_i - 2\Delta_{\text{NS}})), \\ & min(x_{\text{BR}}, max(x_i|_{i\in\{11,14,15,16,20\}})), min(y_{\text{BR}}, max(H, \frac{1}{3}\sum_{i=14}^{16} y_i + 2\Delta_{\text{NS}}))] \end{aligned}$$

$$\boxdot_{\text{UL12}} = [x_{\text{TL}}, y_{\text{TL}}, max(x_i|_{i\in\{3,9,14,15,19\}}), max(y_i|_{i=14}^{16})]$$

$$\boxdot_{\text{U12}} = [x_{\text{TL}}, y_{\text{TL}}, x_{\text{BR}}, max(y_i|_{i=14}^{16})]$$

$$\boxdot_{\text{UR12}} = [min(x_i|_{i\in\{4,10,15,16,19\}}), y_{\text{TL}}, x_{\text{BR}}, max(y_i|_{i=14}^{16})]$$

$$\boxdot_{\text{UL34}} = [x_{\text{TL}}, y_{\text{TL}}, max(x_i|_{i\in\{5,11,16,20\}}), max(y_i|_{i=18}^{20})]$$

$$\boxdot_{\text{U34}} = [x_{\text{TL}}, y_{\text{TL}}, x_{\text{BR}}, max(y_i|_{i=18}^{20})]$$

$$\boxdot_{\text{UR34}} = [min(x_i|_{i\in\{2,8,14,18\}}), y_{\text{TL}}, x_{\text{BR}}, max(y_i|_{i=18}^{20})]$$

$$\boxdot_{\text{L12}} = [x_{\text{TL}}, y_{\text{TL}}, max(x_i|_{i\in\{3,15,19\}}), y_{\text{BR}}]$$

$$\boxdot_{\text{L34}} = [x_{\text{TL}}, y_{\text{TL}}, max(x_i|_{i\in\{5,11,16,20\}}), y_{\text{BR}}]$$

$$\boxdot_{\text{R34}} = [min(x_i|_{i\in\{2,8,14,18\}}), y_{\text{TL}}, x_{\text{BR}}, y_{\text{BR}}]$$

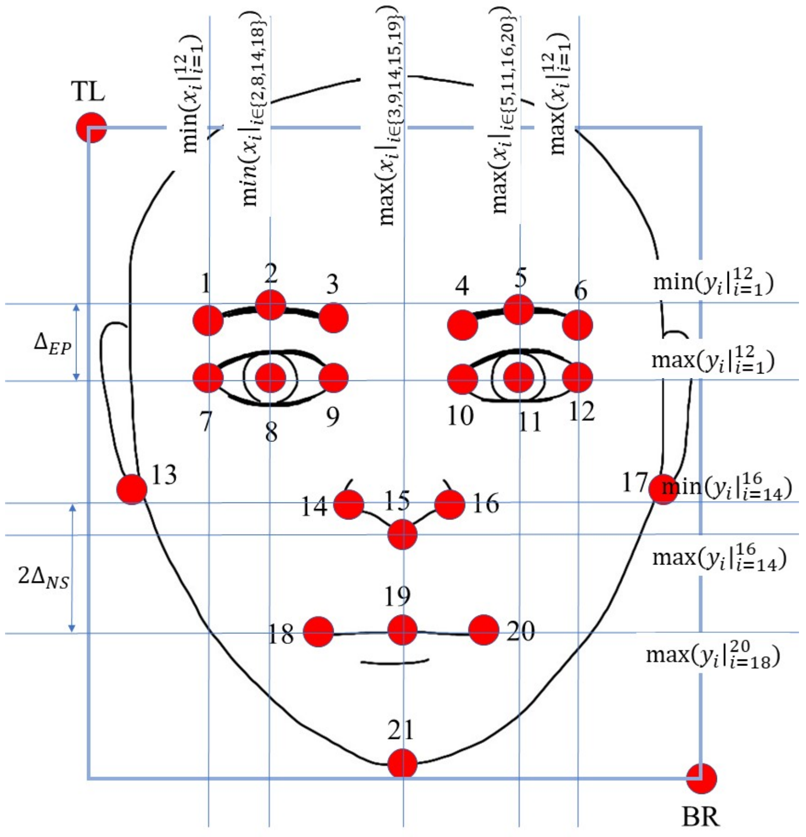


Figure 4.2: The 21 fiducial key points and the full face bounding box.

$$\boxdot_{\text{R12}} = [min(x_i|_{i\in\{4,10,15,16,19\}}), y_{\text{TL}}, x_{\text{BR}}, y_{\text{BR}}]$$

$$\boxdot_{\text{B12}} = [x_{\text{TL}}, min(y_i|_{i=14}^{16}), x_{\text{BR}}, y_{\text{BR}}]$$

$$\boxdot_{\text{B34}} = [x_{\text{TL}}, min(y_i|_{i=7}^{12}), x_{\text{BR}}, y_{\text{BR}}]. \tag{4.1}$$

Here, $\Delta_{\text{EP}} = max(|y_i - y_{i-6}|_{i=7}^{12})$ and $\Delta_{\text{NS}} = 0.5 * (max(y_i|_{i=18}^{20}) - min(y_i|_{i=14}^{16}))$, $\forall v_j \geq \tau$, where $j \in \{1, 2, \ldots 21\}$.

The set of facial segments is denoted by $S = \{FS_k \mid k = 1, 2, \ldots M\}$, where FS_k is the k-th facial segment. M weak Adaboost facial segment detectors are trained to detect each of the segments in S.

Given an image, all the segment detectors are employed. Each detector may return one or more facial segments for the same image. For each facial segment, the bounding box of the full face is estimated according to the ideal position of that segment relative to the whole face. For example, if the top left and bottom right corners of the bounding box obtained for segment L_{12} are (x_1^{L12}, y_1^{L12}) and (x_2^{L12}, y_2^{L12}), respectively, then those for the estimated full face are (x_1^{L12}, y_1^{L12}) and $(min(w_{img}, x_2^{L12} + (x_2^{L12} - x_1^{L12})), y_2^{L12})$, where w_{img} is the width of the image. The estimated face center from this segment is $(x_2^{L12}, y_1^{L12} + (y_2^{L12} - y_1^{L12})/2)$. For each estimated face center j, a cluster of segments CL_j that depicts the full face is formed where, the other segments of that cluster have estimated face centers within a certain radius r pixels from the center. Here, $j = \{1, 2, \ldots c_I\}$ and c_I is the number of clusters formed for image I.

A bounding box for the whole face B_{CL_j} is calculated based on the constituent segments. For the generation of a proposal set, duplicate clusters that yield exactly same bounding boxes are eliminated and at most ζ face proposals are generated from each cluster by selecting random subsets of face segments constituting that cluster. Therefore, each proposal P is composed of a set of face segments S_P, where $S_P \in \mathcal{P}(S) - \{\varnothing\}$ and $\mathcal{P}$ denotes the power set. To get better proposals, one can impose extra requirements such as $|S_P| > c$, where $|\cdot|$ denotes cardinality and c is a threshold. Each proposal is also associated with a bounding box for the whole

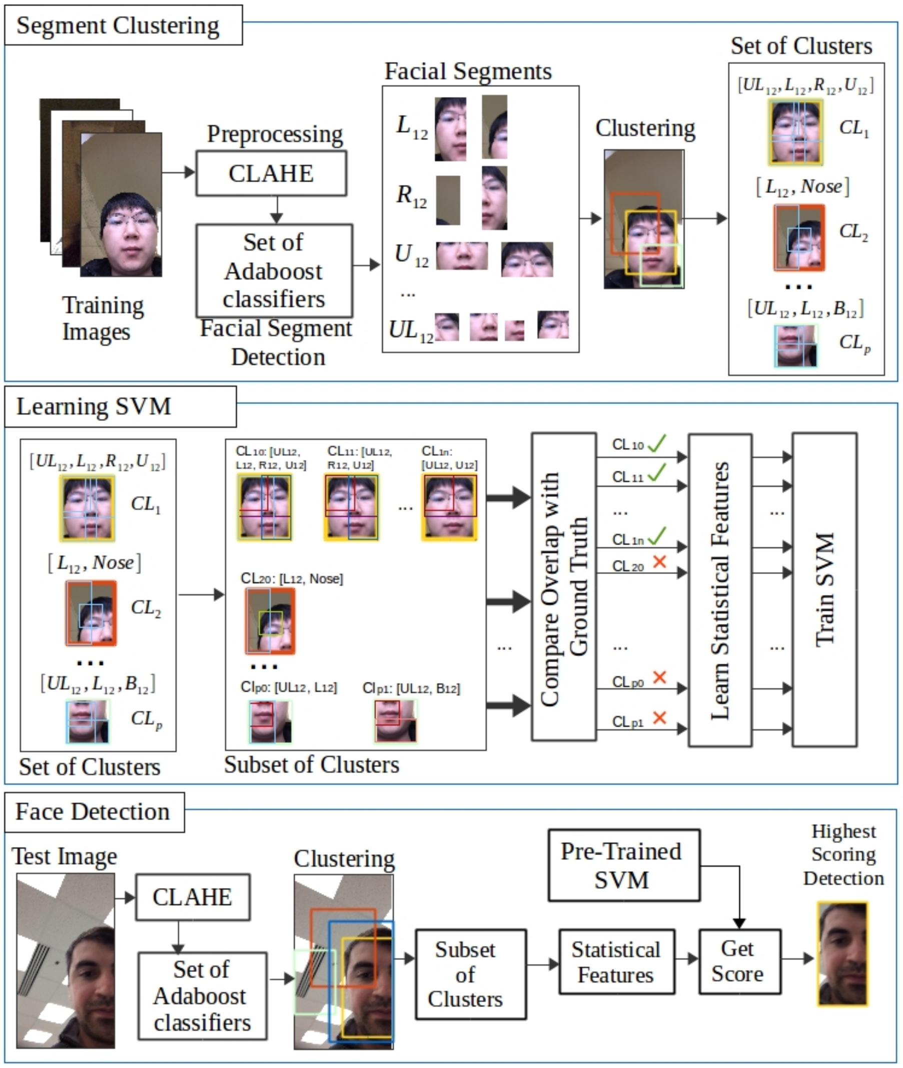


Figure 4.3: The block diagram of the overall system of FSFD [4].

face, which is the smallest bounding box that encapsulates all the segments in that proposal.

Fig. 4.1 depicts the integration of the proposal generation block into the face detection pipeline.

4.1.2 Facial Segment-based Face Detection (FSFD)

The system block diagram for FSFD is shown in Fig. 4.3. Segment clustering is done at first. Then, in the SVM learning phase, the first ζ subsets of the total set of facial segments from each cluster are regarded as proposed faces. Consider the k-th segment that was detected in an image. If there are m segments that are within a certain distance from it and $m+1 \geq c$, then those $m+1$ segments are considered to be part of a single proposal. The value c is the threshold that decides the minimum number of colocated segments that are required if they are to be declared to be a face proposal.

For example, if the 4 segments U_{34}, B_{12}, L_{12} and UL_{12} form a cluster around NS and $c = 2$, then the viable subsets are $[[NS, U_{34}], [NS, B_{12}], \ldots, , [NS, U_{34}, B_{12}, L_{12}, UR_{34}]]$. The total number of subset here is $\sum_{j=1}^{4} \binom{4}{j} = 15$ including the complete set. Keeping the k-th segment fixed, ζ random subsets are considered for face proposals. In this example, ζ can vary from 1 to 14. Since for $m + 1$ segments, the number of subsets is in the order of 2^{m+1}, the number of subset is limited to $\zeta << 2^{m+1}$.

The bounding box of the face proposal is the smallest bounding box that contains all the estimated faces from all the facial segments in that proposal. Intuitively, the greater the number of facial segments with better detection accuracy in a proposal, the higher the probability of that proposal being a face. Further, experimentally, it is found that some particular sets of facial segments are more likely to return a face than others, while some sets of segments provide more accurate bounding boxes with greater consistency than such other sets.

A linear SVM classifier is trained on the proposed faces using the following prior probability values from the training proposal set that represents the likeliness of certain segments and certain combinations. These are

- Fraction of total true faces constituted by proposal P, i.e.

$$\frac{|P \in \Theta^F|}{|\Theta^F|},$$

where Θ^F is the set of all proposals that return a true face.

- The fraction of total mistakes constituted by proposal P, i.e.

$$\frac{|P \in \Theta^{\overline{F}}|}{|\Theta^{\overline{F}}|},$$

, where, $\Theta^{\overline{F}}$ is the set of all proposals that are not faces.

- For each of the M facial segment $s_k \in S$, the fraction of total true face proposals of which s_k is a part of, i.e.

$$\frac{|s_k \in S_p; S_p \in \Theta^F|}{|\Theta^F|}, \text{ where, } k = 1, 2, \ldots, M.$$

- For each of the M facial segment $s_k \in S$, the fraction of total false face proposals of which s_k is a part of, i.e.

$$\frac{|s_k \in S_p; S_p \in \Theta^{\overline{F}}|}{|\Theta^{\overline{F}}|}, \text{ where, } k = 1, 2, \ldots, M.$$

Experimentally, it is found that the nose detector is the most accurate of all the detectors, while B_{12} is the least accurate. For M facial segments, the size of this feature vector is $2M+2$ for each proposal. There are $2M$ values corresponding to the face and non-face probabilities of each of the M segment and the rest 2 values are the probabilities of the cluster being and not-being a face. Among the $2M$ values, only those corresponding to the segments present in the proposal are non-zero.

For each pre-processed test image, the proposed faces are obtained in a similar manner as the training faces. Thus, there are ζ face proposals from each face and feature vectors of size $2M + 2$ for each proposal. The SVM classifier returns a confidence score for each proposal. The proposal that has the highest confidence score above a threshold is chosen as the detected face.

4.1.3 SegFace

SegFace is a fast and shallow face detector built from segments proposal. For each segment in $s_k \in S$, a classifier C_{s_k} is trained to accept features $f(s_k)$ from the segment and generate a score denoting if a face is present. Output scores of C_{s_k} are stored in an M dimensional feature vector F_C, where, elements in F_C corresponding to segments that are not present in a proposal are set to 0.

Another feature vector F_S of size $2M+2$ is constructed using prior probability values as features as described in section 4.1.2. F_C and F_S are appended together to form the full feature vector F of length $3M + 2$. Then a master classifier C is trained on the training set of such labeled vectors $\{F_i, Y_i\}$, where Y_i denotes the label (face or no-face). Thus, C learns how to assign relative importance to different segments and likeliness of certain combinations of segments occurring in deciding if a face is present in a proposal. Thus, SegFace extends the face detection from

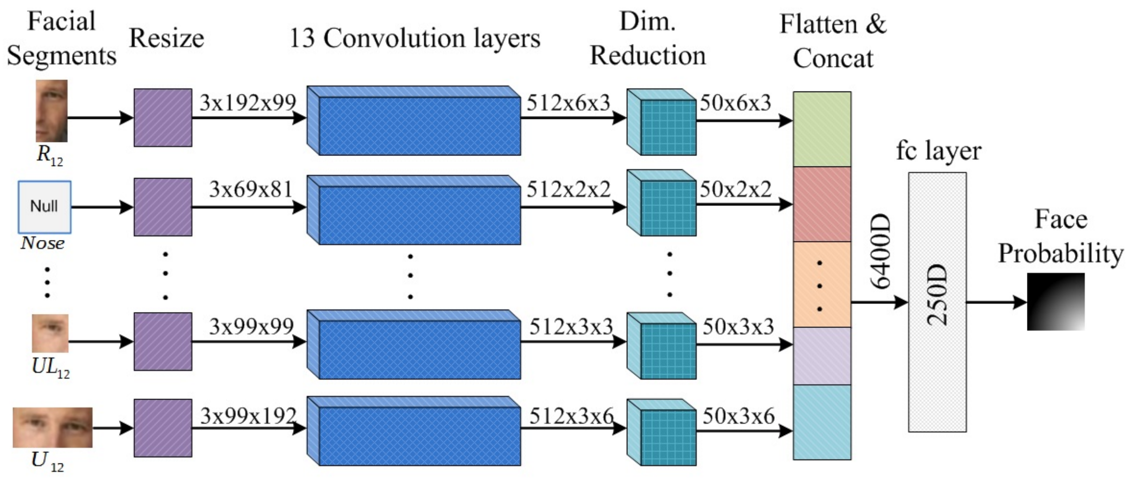


Figure 4.4: Block diagram showing DeepSegFace architecture.

segments concept in [4] using traditional methods to obtain reasonably accurate results. In our implementation of SegFace, HoG [127] features are used as f and Support Vector Machine (SVM) classifiers [128] are used as both C_{s_k} and C for generating segment-wise scores, as well as the final detection score, respectively.

4.1.4 DeepSegFace

DeepSegFace is an architecture to integrate deep CNNs and segments-based face detection from proposals. At first, proposals, consisting of subsets of the $M = 9$ parts as discussed earlier, are generated for each image. DeepSegFace is then trained to calculate the probability values of the proposal being a face. Finally, a re-ranking step adjusts the probability values from the network. The proposal with the maximum re-ranked score is deemed as the detection for that image.

The architecture of DeepSegFace is arranged according to the classic paradigm in pattern recognition: feature extraction, dimensionality reduction followed by a classifier. A simple block diagram of the architecture is shown in Fig. 4.4. Different components of the figure are discussed here.

Convolutional Feature Extraction: There are nine convolutional network columns, structurally similar to VGG16 [129] and initialized with its pretrained weights, for each of the nine segments. Thus each network has thirteen convolution layers arranged in five blocks. Each segment in the proposal is resized to standard dimensions for that segment, then the VGG mean value is subtracted from each channel. For segments not present in the proposal, zero-input is fed into the networks corresponding to those segments, as shown for the $Nose$ segment in the figure.

Dimensionality reduction: The last convolutional feature map has 512 channels, hence naively concatenating them results in a very large feature vector. Hence, a randomly initialized convolutional layer with filter size 1×1 and 50 feature maps is appended to provide a learnable dimension reduction.

Classifier: The outputs from the dimensionality reduction block for each segment-network are vectorized and concatenated to yield a 6400 dimensional feature vector, constituents of which can be seen in Table 4.1. A fully connected layer of 250 nodes, followed by a softmax layer of two nodes (both randomly initialized) is added on top of the feature vector. The two outputs of the softmax layer sum to one and correspond to the probability of the presence or absence of a face.

Re-ranking: The DeepSegFace network outputs the face detection probabilities for each proposal in an image, which can be used to rank the proposals and then declare the highest probability proposal as the face in that image. However there is some prior knowledge that some segments are more effective at detecting the presence of faces than others. This information is available from the prior probability

Table 4.1: Structure of DeepSegFace's Convolutional layers (feature extraction and dimensionality reduction)

Segment	Input	Feature	Dim. Reduce	Flatten
$Nose$	$3 \times 69 \times 81$	$512 \times 2 \times 2$	$50 \times 2 \times 2$	200
Eye	$3 \times 54 \times 162$	$512 \times 1 \times 5$	$50 \times 1 \times 5$	250
UL_{34}	$3 \times 147 \times 147$	$512 \times 4 \times 4$	$50 \times 4 \times 4$	800
UR_{34}	$3 \times 147 \times 147$	$512 \times 4 \times 4$	$50 \times 4 \times 4$	800
U_{12}	$3 \times 99 \times 192$	$512 \times 3 \times 6$	$50 \times 3 \times 6$	900
L_{34}	$3 \times 192 \times 147$	$512 \times 6 \times 4$	$50 \times 6 \times 4$	1200
UL_{12}	$3 \times 99 \times 99$	$512 \times 3 \times 3$	$50 \times 3 \times 3$	450
R_{12}	$3 \times 192 \times 99$	$512 \times 6 \times 3$	$50 \times 6 \times 3$	900
L_{12}	$3 \times 192 \times 99$	$512 \times 6 \times 3$	$50 \times 6 \times 3$	900

values discussed in section 4.1.3. For DeepSegFace, these values are used to re-rank the final score by multiplying it with the mean of the statistical features.

Facial Segment Drop-out for Regularization and Data Augmentation: As mentioned in the proposal generation scheme, subsets of face segments in a cluster are used to generate new proposals. For example, if a cluster of face segments contains n segments and each proposal must contain atleast c segments, then it is possible to generate $\sum\limits_{k=t}^{n} \binom{n}{k}$ proposals. Now, if all the facial segments are present, the network's task is easier. However, all the nine parts are redundant for detecting a face, because of significant overlaps. Also often many segments are not detected by the weak segment detectors. Thus, one can interpret the missing segments as 'dropped-out', i.e. some of the input signals are randomly missing (they are set to zero). Thus the network must be robust to face segments 'dropping out' and generalize better to be able to identify faces. Training with subsets of detected proposals also has the additional effect of augmenting the data. It has been observed that around sixteen proposals are generated per image. Many of these proposals are actually training the network to detect the same face using different combination of segments.

4.2 Face Detection for Mobile Domain by End to End Regression

While the three algorithms (FSFD, SegFace, DeepSegFace) presented here are fast and efficient, they have a lot of scope for improvement. A primary bottleneck of these algorithms is the proposal generation stage, which suffers from the following problems when trading off quality and speed:

4.2.0.1 Slow Speed

The proposal generator can generate many proposals to ensure high recall, but it makes the pipeline slow as the detector must evaluate each of them. For example, [23] uses Selective Search [130], which generates around 2000 proposals per image. On the other hand, a faster version of the All-In-One network [76] uses the Single Shot Multibox Detector (SSD) [65] for generating proposals. These methods are neither end-to-end trainable nor suitable for real-time applications.

4.2.0.2 Upper bound on Recall

If high recall rate is traded for speed, then one can use weak proposal generators which generate less number of proposals. However in this regime the detector is bound by the performance of the proposal generator and cannot detect faces in images where the proposal generator did not return any results. FSFD, SegFace and DeepSegFace use fast proposal generators (around 16 proposals per image), but have no way of recuperating when the proposal generator fails.

4.2.0.3 Special Training

One way to generate a small number of proposals, yet have high recall, is to train specific proposal generators that identify faces and facial segments. However most off-the-shelf proposal generators detect generic objects, and one would have to retrain them for detecting faces.

In this regard, we propose, a regression-based novel deep CNN method with a customized loss function, namely, Deep Regression-based User Image Detector (DRUID), that generates facial segments as well as the full face bounding box with confidence scores without requiring any proposal. DRUID is completely end-to-end trainable and utilizes a unique data augmentation technique that allows it to train on non-mobile domain faces, and still achieve superior performance on unconstrained mobile domain faces.

4.2.1 DRUID: An End-To-End Network For Facial Segment-Based Face Detection

The DRUID is designed to detect a single face from an image along with all the 14 facial segments (Fig. 1.2) without any proposal generation. The algorithm takes a resized image as input and returns the location of each facial segment and the full face along with a visibility/confidence value. The network architecture of DRUID is shown in Fig.4.5. The network is made of Resnet units [131]. Resnet like structures

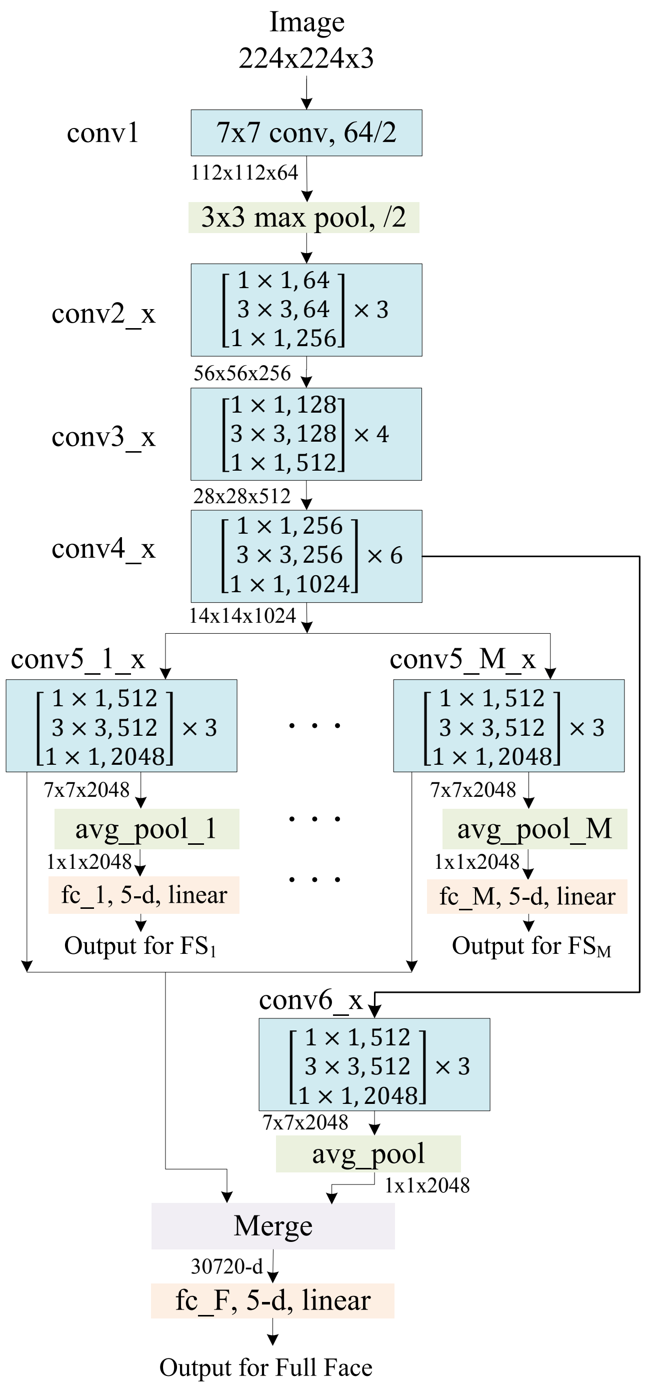


Image
224x224x3
conv1
7x7 conv, 64/2
112x112x64
3x3 max pool, /2
conv2_x
$\begin{bmatrix} 1\times1,64 \\ 3\times3,64 \\ 1\times1,256 \end{bmatrix}\times 3$
56x56x256
conv3_x
$\begin{bmatrix} 1\times1,128 \\ 3\times3,128 \\ 1\times1,512 \end{bmatrix}\times 4$
28x28x512
conv4_x
$\begin{bmatrix} 1\times1,256 \\ 3\times3,256 \\ 1\times1,1024 \end{bmatrix}\times 6$
14x14x1024
conv5_1_x
$\begin{bmatrix} 1\times1,512 \\ 3\times3,512 \\ 1\times1,2048 \end{bmatrix}\times 3$
7x7x2048
avg_pool_1
1x1x2048
fc_1, 5-d, linear
Output for FS_1
conv5_M_x
$\begin{bmatrix} 1\times1,512 \\ 3\times3,512 \\ 1\times1,2048 \end{bmatrix}\times 3$
7x7x2048
avg_pool_M
1x1x2048
fc_M, 5-d, linear
Output for FS_M
conv6_x
$\begin{bmatrix} 1\times1,512 \\ 3\times3,512 \\ 1\times1,2048 \end{bmatrix}\times 3$
7x7x2048
avg_pool
1x1x2048
Merge
30720-d
fc_F, 5-d, linear
Output for Full Face

Figure 4.5: Network Architecture for DRUID.

are built of basic blocks that have two parallel paths, one that learns the residual and the other is is either an identity transformation or a single convolution. For the residual path, a 'bottleneck' architecture is chosen, which consists of three sets of convolutional transformations (convolution, batch normalization, ReLU activation) of sizes 1×1, 3×3 and 1×1. These basic blocks are stacked into groups. Within a group, there is no downsampling. Let $convi\_x$ denote the x^{th} block of the i^{th} group. Each group starts with an block that downsamples by strided convolution (in both paths of the block), followed by other blocks with identity connections.

DRUID's architecture is shown in Fig. 4.5. It has a common network trunk consisting of 3 Resnet groups, containing 3, 4 and 6 blocks respectively. Then the network is expanded to $M = 14$ auxiliary branches, each containing the $conv5\_x$ unit of Resnet-50. Each of these auxiliary branches has 7×7 average pooling on top followed by a fully connected network ($fc\_1 - fc\_M$) with linear activation that gives a 10-dimensional output.

As for the full face detection output, an additional auxiliary branch is created without the fully connected layer (the convolution unit is denoted as $conv6\_x$ for this) and the flattened output of its average pooling block is merged with the intermediate flattened outputs of all the $conv5\_i\_x$ units (where $i = \{1, 2, \ldots, M\}$). The output of this merge block goes into another fully connected network fc_F with linear activation that outputs a 5-dimensional vector. A comparison of the three proposal-based methods and DRUID is provided in Table 4.2.

Table 4.2: Comparison of the proposed methods

Component	FSFD	SegFace	DeepSegFace	DRUID
Proposal Generation	Clustering detections from cascade classifiers for facial segments	Clustering detections from cascade classifiers for facial segments	Clustering detections from cascade classifiers for facial segments	Not required
Low level features	Prior probabilities	HoG and Priors	Deep CNN features (VGG)	Deep CNN features (Resnet)
Intermediate stage	none	SVM for segment i outputs a score on HoG features of segment i	Dimension reduction and concatenation to single 6400D vector	Facial segment bounding box regression.
Final Stage	Classification: SVM trained on priors	Classification: SVM trained on scores from part SVMs and priors	Classification: Fully connected layer, followed by a softmax layer	Regression: Full face bounding box.
Using priors	Used as features in the final SVM	Used as features in the final SVM	Used for re-ranking of face probabilities in post processing	Not used
Trade-offs	Very fast but less accurate	Fast but less accurate	Slower but more accurate	Faster and more accurate than DeepSegFace

4.2.1.1 Loss Function for Training

For the i-th facial segment, denote the estimated bounding box as $\widehat{\mathbf{b_i}} = \{\widehat{x_{1i}}, \widehat{y_{1i}}, \widehat{x_2 i}, \widehat{y_2 i}\}$ and predicted visibility as $\widehat{v_i}$. Here, $\{\widehat{x_{1i}}, \widehat{y_{1i}}\}$ denotes the top-left and $\{\widehat{x_{2i}}, \widehat{y_{2i}}\}$ denotes the bottom right coordinates of the i-th facial segment. The ground truth bounding box for the i-th facial segment is $\mathbf{b_i} = \{x_{1i}, y_{1i}, x_{2i}, y_{2i}\}$ and the ground-truth visibility is v_i. Also, the ground truth for the full face bounding box is $\mathbf{b^F} = \{x_1^F, y_1^F, x_2^F, y_2^F\}$. Now, the 5-dimensional vector that each of the fully connected layers predict by regression is $\{\widehat{x_{1i}}, \widehat{y_{1i}}, \widehat{x_2 i}, \widehat{y_2 i}, \widehat{v_i}\}$.

The loss terms defined for this regression are

$$
\begin{aligned}
L_b(i) &= \| \widehat{\mathbf{b_i}} - \mathbf{b_i} \|_2^2 && (4.2)\\
L_v(i) &= (\widehat{v_i} - v_i)^2 && (4.3)\\
L_x(i) &= (v_i \mathbb{1}[\widehat{x_{1i}} - \widehat{x_{2i}} > 0])^2 && (4.4)\\
L_y(i) &= (v_i \mathbb{1}[\widehat{y_{1i}} - \widehat{y_{2i}} > 0])_2^2 && (4.5)\\
L_{x_1}(i) &= (v_i (x_1^F - \widehat{x_{1i}}) \mathbb{1}[x_1^F - \widehat{x_{1i}} > 0])^2 && (4.6)\\
L_{x_2}(i) &= (v_i (\widehat{x_{2i}} - x_2^F) \mathbb{1}[\widehat{x_{2i}} - x_2^F > 0])^2 && (4.7)\\
L_{y_1}(i) &= (v_i (y_1^F - \widehat{y_{1i}}) \mathbb{1}[y_1^F - \widehat{y_{1i}} > 0])^2 && (4.8)\\
L_{y_2}(i) &= (v_i (\widehat{y_{2i}} - y_2^F) \mathbb{1}[\widehat{y_{2i}} - y_2^F > 0])^2 && (4.9)\\
L_O(i) &= \| (1 - IOU(\widehat{\mathbf{b_i}}, \mathbf{b_i}) \mathbb{1}[IOU(\mathbf{b_i}, \widehat{\mathbf{b_i}}) \leq 0]) \times \\
&\quad \mathbb{1}[v_i \neq 0])^2 && (4.10)
\end{aligned}
$$

where, $\mathbb{1}$ denotes an indicator function. The total loss for each of the auxiliary network corresponding to facial segments and the loss for the full face detection network on top is defined as

$$L_i = \sum_k \lambda_k L_k \qquad (4.11)$$

where, $k \in \{v, x, y, x_1, x_2, y_1, y_2, O\}$ and $i \in S \cup \{F\}$. Here, F denotes the full face and corresponds to the topmost network. The components L_b and L_v are the ℓ_2 losses for the predicted bounding box and visibility/confidence, respectively. Losses

L_x and L_y imply that the coordinates of the bottom-right points are larger than those of the top-left points. L_{x_1}, L_{x_2}, L_{y_1} and L_{y_2} dictate that the coordinates of a facial segments are contained within the coordinates of the full face. Finally, L_O implies that the overlap between the predicted bounding box and the ground truth bounding box for each segment should be as high as possible. Note that the ground truth visibility v_i is a binary value for all $i \in S$ based on the visibility/non-visibility of corresponding segments in the image. However, v_F is a floating point number which is calculated as $v_F = \frac{\sum_{i \in S} v_i}{|S|}$, i.e. the fraction of total facial segments that is visible. Thus, a smaller value of v_F will mean that only a few parts can be seen and indicate a partially visible face. v_F is zero when no facial segments are visible, i.e. when there is no face. The 10-dimensional output for each auxiliary branch indicates that each of the i facial segments tries not only to predict the bounding box and visibility for that block itself, but also the bounding box and visibility of the full face in a manner that the corresponding loss is minimized.

4.2.1.2 Data Augmentation

In order to ensure that the network is trained robustly for detecting facial segments, a novel data augmentation technique is adopted. Apart from flipping the input image and the corresponding bounding box coordinates, the images are cropped according to a pre-defined mechanism to ensure partial visibility in the training set. Some examples of such augmented data are shown in Fig. 4.6. For the first row, since both the left, right and top portion are visible, there can be

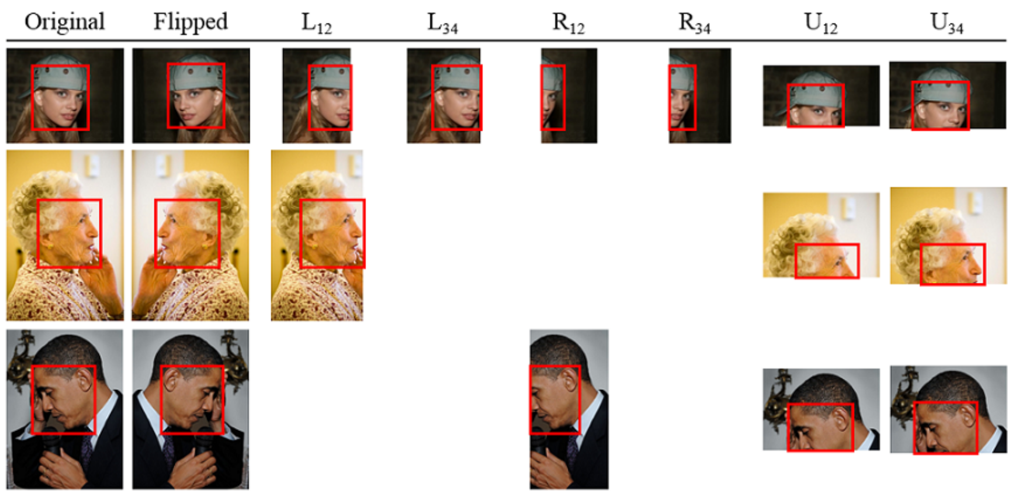


Figure 4.6: Image processing for generating training data for DRUID ensuring partial visibility of faces.

a maximum of six more augmented versions generated by cropping the image in a manner that only upto the L_{12} or L_34 or R_{12} or R_{34} or U_{12} or U_{34} segment of the face is visible. Since, only left half and right half faces are visible in the original image in the second and third rows of Fig. 4.6, respectively, they can produce less number of augmented images. The proposed data augmentation technique enables the network to train robustly for detecting partially visible faces with a few training images even if the original samples do not have many partially visible faces. In addition, the presence of certain segments dictate the presence of some other segments (e.g. if L_{34} is visible then L_{12} and UL_{12} must be visible) and training with more partially visible faces enables the network to learn these interrelations as well.

Apart from the augmentation, Gaussian blur is also applied on the original images with a probability of 0.7 and a random radius between 0 and 5 pixels in order to improve the capability of the network to detect blurry faces. Moreover, a gamma correction in the form of $I_o(x,y) = AI_i(x,y)^\gamma$ was applied to each pixel $I_i(x,y)$ of the normalized image, where, $A = 1$ and $\gamma = 2^s$ was set. For an image, the value of s was obtained from a zero mean, unit variance Gaussian distribution. Thus, when s is near the 0, the transformed image is similar to the original, whereas, a positive value of s would make the image darker and a negative value will make it brighter. Since there are frequent very dark and very bright faces in the mobile face datasets, it is expected that this sort of transformation would make DRUID robust against extreme illumination variations.

4.2.1.3 Salient points of DRUID

DRUID's architecture provides certain innate advantages which are discussed below:

Training Data: DRUID is trained with images from AFLW, but for the task of mobile image detection. Therefore DRUID is able to handle the domain shift gracefully, without the need of specialized data from the mobile domain for training or fine-tuning unlike the other methods like FSFD, SegFace and DeepSegFace.

Robustness: DRUID is robust to occlusion due to its ability to detect partial faces which is ingrained in its architecture. DRUID is also very robust to variations in scale, if multi-scale data is present during training. DRUID does not need to scan the image at multiple scales to ensure it finds faces at all scales, since it does direct regression for the face location. Thus, as it has seen multi-scale faces during training, it became scale invariant to some degree. Also, because of the application of random gamma transformation and Gaussian blur during training, DRUID achieved better capabilities of detecting blurry and dark/bright faces, which are very frequent in mobile face domain, compared to other methods. It has the added advantage of multiple facial segment-based detection, which allows it to estimate the face bounding box with the help of on or more of the segments even when all the other segment detectors fail. Hence, it is very robust against occlusion, illumination and pose variation.

4.3 Experimental Results

In this section, the experimental results of the facial segment-based detectors on the AA-01-FD and UMDAA-02-FD datasets are compared with a) Normalized Pixel Difference (NPD)-based detector [75], b) Hyperface detector [23], c) Deep Pyramid Deformable Part Model detector [117], and d) DPM baseline detector [74].

4.3.1 Experimental Setup

FSFD, SegFace and DeepSegFace are trained on 3964 images from AA-01-FD and trained models are validated using 1238 images. The data augmentation process produces $57,756$ proposals from the training set, that is around 14.5 proposals per image. The remaining 2835 images of AA-01-FD are used for testing. For UMDAA-02-FD, $32,642$ images are used for testing. In all experiments with FSFD, SegFace and DeepSegFace, $c = 2$ and $\zeta = 10$ are considered.

DRUID is trained using the Annotated Facial Landmarks in the Wild (AFLW) [29] dataset which provides a large-scale collection of annotated face images gathered from Flickr, exhibiting a large variety in appearance (e.g., pose, expression, ethnicity, age, gender) as well as general imaging and environmental conditions. From a total of 21123 faces that are manually annotated up to 21 landmarks per image, 18958 images contain single faces, which are considered for training DRUID. The total number of training samples is 233029 after data augmentation and negative sample extraction of which 100000 are randomly chosen at the start of each epoch for training. Data is augmented by flipping (18958 images), cropping everything beyond L_{12} (15619 images), L_{34} (13583 images), R_{12} (16228 images), R_{34} (13493 images), U_{12} (15362 images) and U_{34} (15237 images). 105591 Negative samples are generated from the backgrounds of the training images with sizes equal to the size of the face in the corresponding image.

The results are evaluated by comparing the ROC curve and precision-recall curves of these detectors since all of them return a confidence score for detection. The goal is to achieve high True Acceptance Rate (TAR) at a very low False Acceptance Rate (FAR) and also a high recall at a very high precision. Hence, numerically, the value of TAR at 1% FPR and recall achieved by a detector at 99% precision are the two metrics that are used to compare different methods.

4.3.2 Selection of Parameter Values

In our proposal generation scheme, $M = 9$ is used. The nine parts under consideration are nose ($Nose$), eye-pair (Eye), upper-left three-fourth (UL_{34}), upper-right three-fourth (UR_{34}), upper-half (U_{12}), left three-fourth (L_{34}), upper-left-half

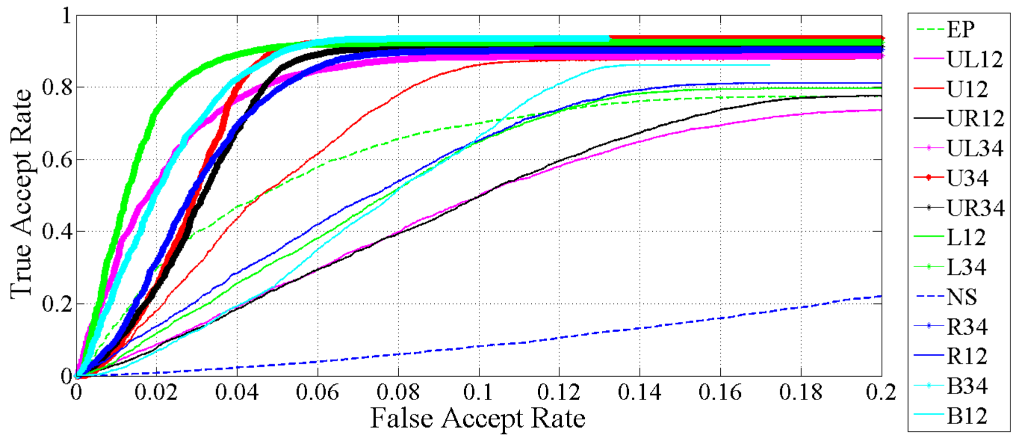


Figure 4.7: TAR vs. FAR ROC plots for different segment prediction results obtained for DRUID on 18958 images of the AFLW dataset.

(UL_{12}), right-half (R_{12}) and left-half (L_{12}. These nine parts, constituting the best combination C_{best} [4] according to the analysis of effectiveness of each part in detecting faces, are considered in this experiment since the same adaboost classifiers are adapted in this work for proposal generation. The threshold c is set to 2. A small value is chosen to get high recall, at the cost of low precision. This lets one generate a large number of proposals, so that any face is not missed in this stage. ζ is set to 10.

For DRUID, all the 14 segments are considered. The parameters for training DRUID are set as follows: optimizer - adam, learning rate 0.0001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1e-08$, decay=0.0, total epoch 25 and batch size 32.

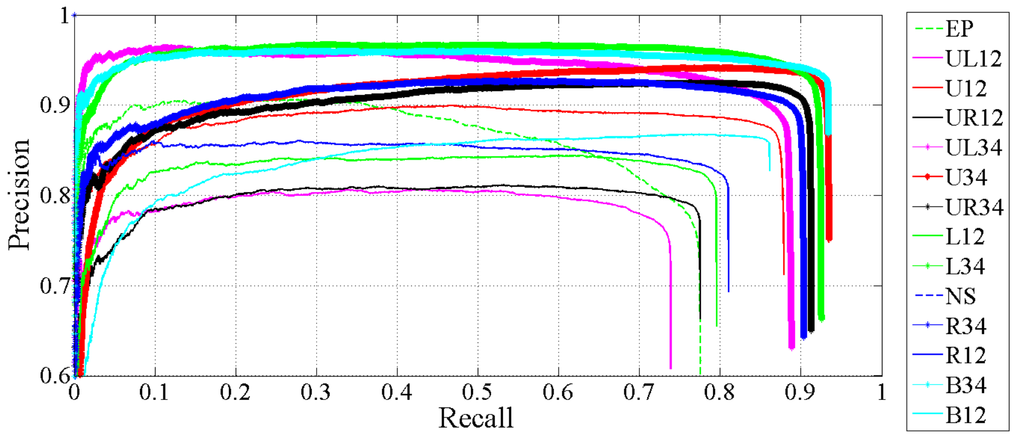


Figure 4.8: Precision vs. Recall curves for different segment prediction results obtained for DRUID on 18958 images of the AFLW dataset.

Table 4.3: Comparison at 50% overlap on AA-01-FD and UMDAA-02-FD datasets

Methods	AA-01		UMDAA-02	
	TAR at 1% FAR	Recall at 99% Prec.	TAR at 1% FAR	Recall at 99% Prec.
NPD [75]	29.51	38.53	33.47	26.80
SSD [132]	62.68	58.77	51.90	46.41
DPMBaseline [74]	85.08	83.25	78.44	72.92
DeepPyramid [117]	66.17	41.19	71.11	66.16
HyperFace [23]	90.52	90.32	73.19	71.17
All-In-One [76]	90.94	89.73	81.71	78.84
FSFD C_{best} [4]	59.06	56.23	55.72	26.87
SegFace [5]	67.00	62.88	66.39	61.48
DeepSegFace [5]	86.87	86.49	82.19	76.36
DRUID No Aug	84.46	82.42	85.86	84.08
DRUID	**91.98**	**91.52**	**88.47**	**86.90**

4.3.3 Quantitative Analysis

4.3.3.1 Performance

In Table 4.3, the performance of DRUID is compared with other facial segment-based and state-of-the-arts methods for both datasets in terms of TAR at 1% FAR and recall at 99% precision. From the measures on the AA-01-FD and UMDAA-

02-FD datasets, it can be seen that DRUID outperforms all the other detectors on both metrics. In fact, on UMDAA-02-FD datset, DRUID improves the TAR by almost 6% and recall by 10% compared to the second best DeepSegFace. Among the proposal-based methods, SegFace, in spite of being a traditional feature based algorithm, outperforms DCNN-based algorithms such as NPD, SSD and DeepPyramid on the AA-01-FD dataset and NPD and SSD on the UMDAA-02 dataset. Note that the SSD-based face detector evaluated here is adopted from [132] uses a MobileNet architecture. The table also shows comparative results for DRUID with and without the proposed augmentation method. It can be seen that DRUID without augmentation (DRUID No Aug) performs poorly compared to the proposed DRUID with augmentation. The improvement in TAR values and precision is approximately 3% for using the proposed augmentation on the UMDAA-02 dataset. Note that, even without augmentation, DRUID achieves competitive performance on both datasets, specially on UMDAA-02, in comparison to other state-of-the-art methods.

The TAR-FAR and precision-recall curve for UMDAA-02-FD dataset is shown in Fig. 4.12 and 4.11. The TAR-FAR and Precision-Recall curves for the AA-01-FD dataset are shown in Figs. 4.13 and 4.14, respectively. DRUID outperforms all the other methods in terms of TAR at 1% FAR and recall at 99% precision.

From fig. 4.11, the ROC for UMDAA-02-FD dataset, it can be seen that DeepSegFace gives the second best performance even with the bottleneck of proposal generation (the curve flattens around 87.5%). This is because all the traditional methods suffer so much more when detecting mobile faces in truly unconstrained settings that a true acceptance rate of even 87% is hard to achieve. The DRUID

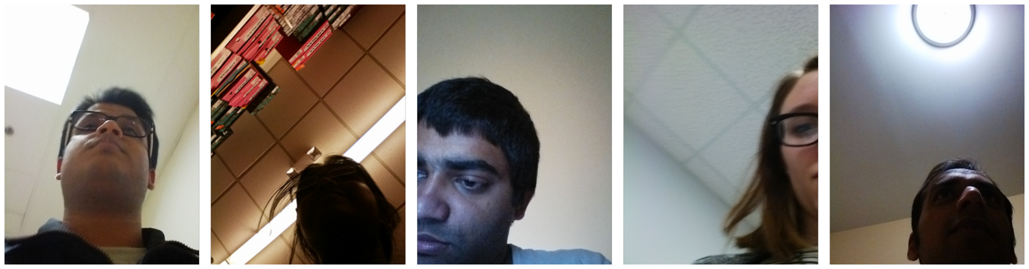


Figure 4.9: Images without even one good proposal returned by the proposal generation mechanism. This bottleneck can be removed by using better proposal generation schemes.

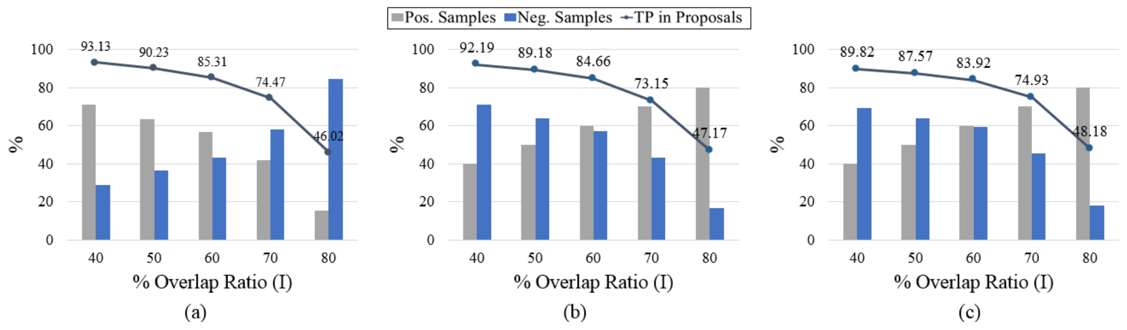


Figure 4.10: (a) for 57756 Train Proposals from AA-01-FD Dataset, (b) 39168 Test Proposals from AA-01-FD Dataset, and (c) 410138 Test Proposals from UMDAA-02-FD dataset. In all cases $c = 2$ and $\zeta = 10$.

network, being free from proposals, achieves 87.45% TAR at 1% FAR.

4.3.3.2 Performance bottleneck in proposal-based techniques

In Fig. 4.9, some images are shown for which the proposal generator did not return any proposals or returned proposals without sufficient overlap, even though there are somewhat good, visible faces or facial segments in them. The percentage of true faces that are represented by at least one proposal in the list of proposals for the training and test sets are counted. The result of this analysis is shown in Fig. 4.10. The bar graphs denote the percentage of positive samples and negative samples present in the proposal list generated for a certain overlap ratio. For example, out of the 55, 756 proposals generated for training, there are approximately 62% positive samples and 35% negative samples at an overlap ratio of 50%. Considering the overlap ratio fixed to 50% for this experiment, it can be seen from the line plot in Fig. 4.10(b), corresponding to the AA-01-FD test set, that the proposal generator actually represent 89.18% of the true faces successfully and fails to generate a single good proposal for the rest of the images. Hence, the performance of the proposed detectors are upper-bounded by this number on this dataset, a constraint that can be mitigated by using advanced proposal schemes like selective search which generates around 2000 proposals per image for Hyperface, compared to just around sixteen proposals that are generated by the fast proposal generator employed by DeepSegFace. Another walk-around to these bottleneck imposed by the proposal generation is modeling the task as a regression problem which is done in the DRUID network. When considering the UMDAA-02-FD test set, which is completely unconstrained and has almost ten times more images than AA-01-FD test set, this upper bound might not be so bad. From Fig. 4.10(c) it can be seen that the upper bound for UMDAA-02-FD is 87.57% true positive value for the proposal generator and from Table 4.3, it can be seen that DeepSegFace was able to achieve 82.26% TAR. However, this proves the argument about proposal generation being the bottleneck for this sort of methods, while DRUID, being free from such constraints, was able to demonstrate much better performance in terms of both evaluation measures.

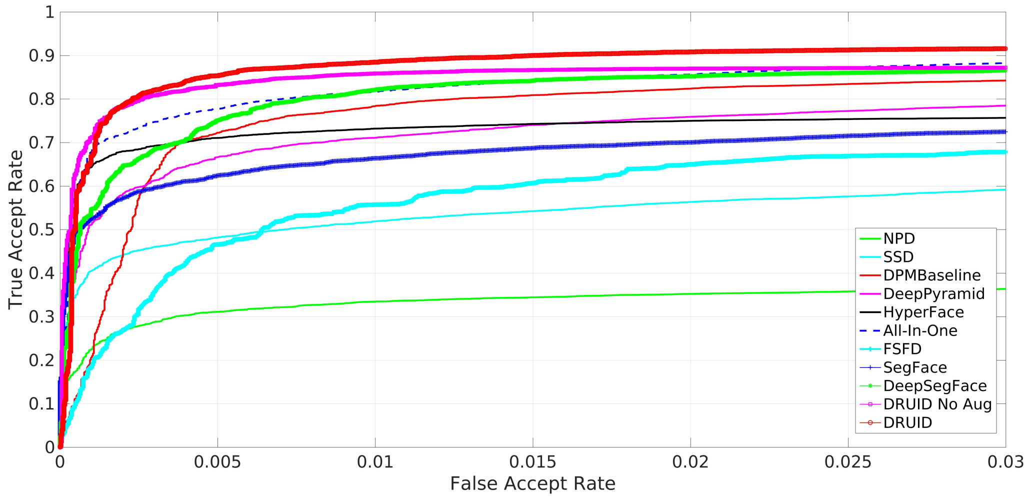


Figure 4.11: ROC curve for comparison of different face detection methods on the UMDAA-02-FD dataset

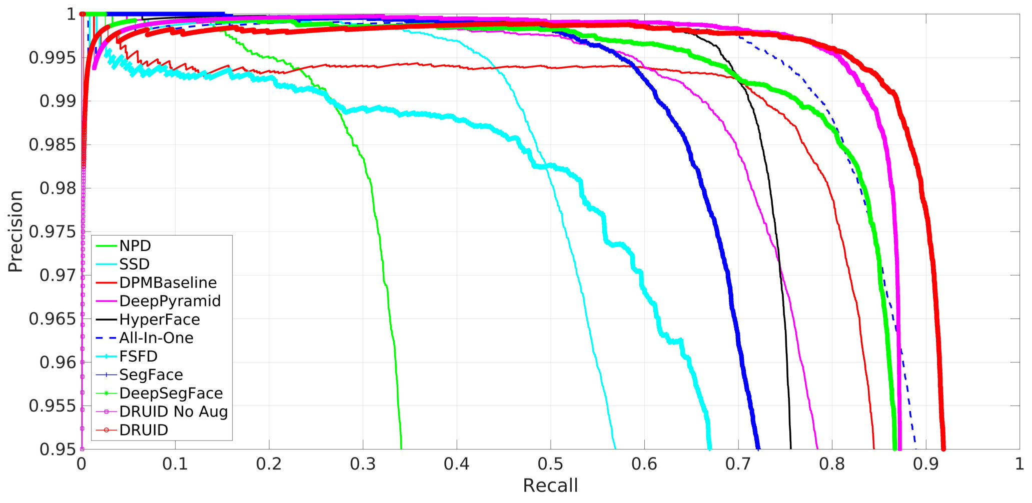


Figure 4.12: Precision-Recall curve for comparison of different face detection methods on the UMDAA-02-FD dataset.

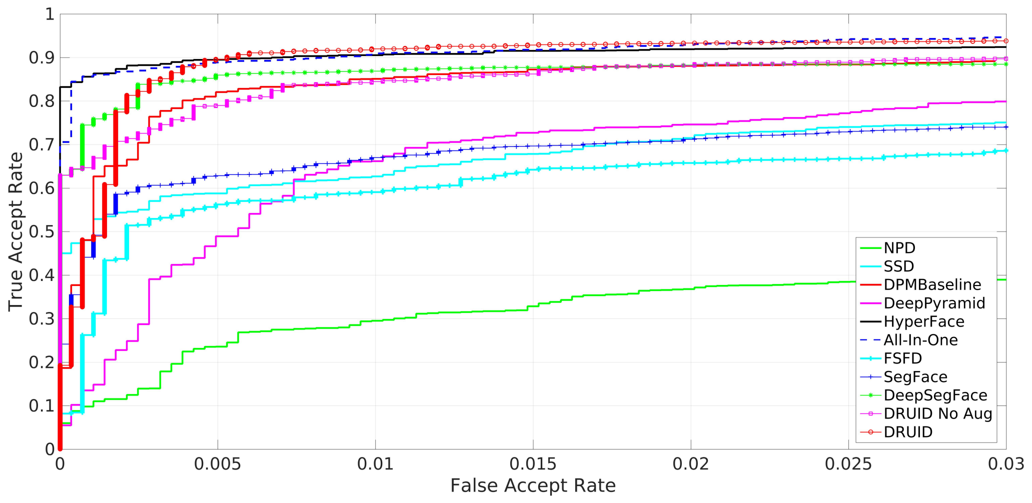


Figure 4.13: ROC curve for comparison of different face detection methods on the AA-01-FD dataset

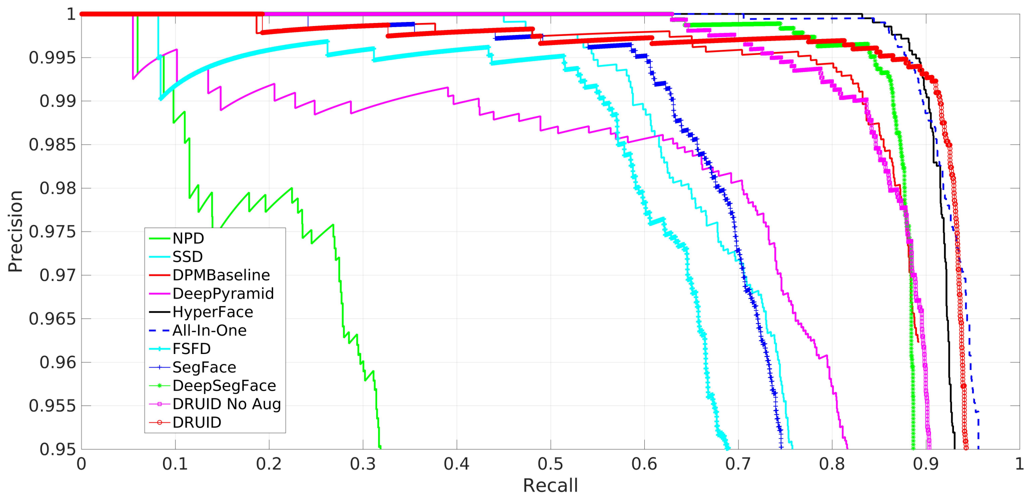


Figure 4.14: Precision-Recall curve for comparison of different face detection methods on the AA-01-FD dataset.

4.3.3.3 Performance of auxiliary networks in DRUID

To investigate the training of the auxiliary networks that predict different facial segments, the TAR-FAR and Precision-Recall curves for each facial segment are plotted in Figs. 4.7 and 4.8, respectively, for the AFLW dataset. Interestingly, it can be seen from these figures that for DRUID the ranking of segments is not similar to the proposal-based methods. The plots reveal that the facial segments that represents a bigger portion of the face are trained better. In fact, the nose segment NS which represents the smallest facial segment, performs the poorest, while all the big segments such as L_{34}, R_{34}, B_{34} etc. are much better trained.

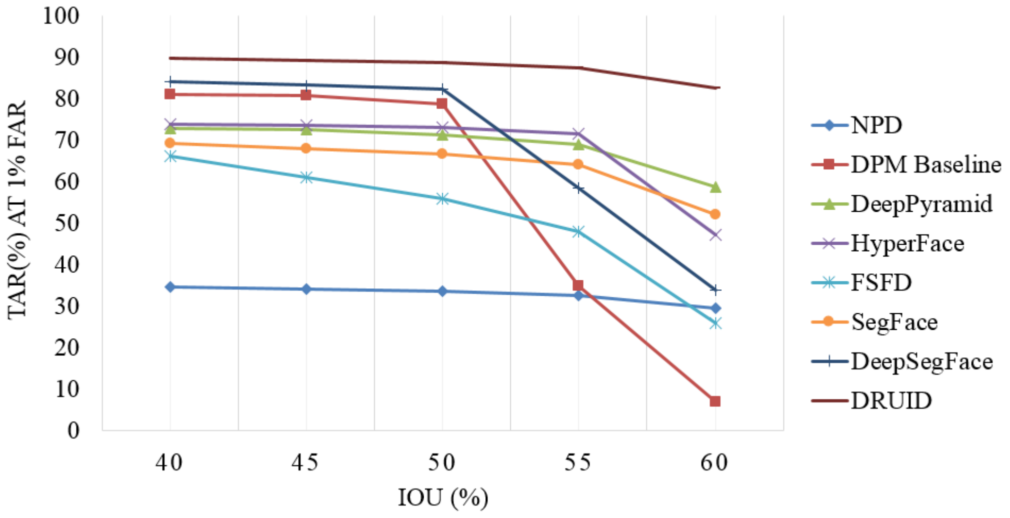


Figure 4.15: TAR at 1% FAR for different IOU for different methods.

4.3.3.4 Timing information

The processing time per image for FSFD on an Intel Xeon(R) CPU E5506 operating at 2.13GHz with 5.8GB memory is 0.52 seconds without any multi-threading

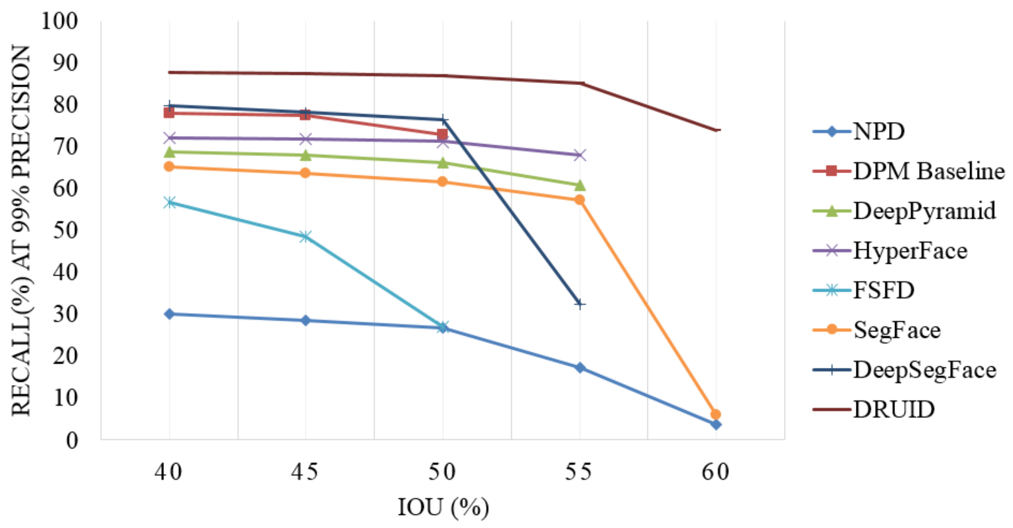


Figure 4.16: Recall at 99% Precision for different IOU for different methods.

or special optimization. SegFace takes around 0.49 seconds when running on an Intel Xeon CPU E-2623 v4 (2.604 GHz) machine with 32GB Memory without multi-threading, hence it is possible to optimize it to run on mobile devices in reasonable time without requiring a specialized hardware. When forwarding proposals in batch sizes of 256, DeepSegFace takes around 0.02 seconds per proposal on a GTX Titan-X GPU. The end to end throughput for DRUID on a Titan-X GPU is 0.008 seconds (batch size 32), which is architecturally faster that DeepSegFace since it does not require time for proposal generation. For DeepSegFace, the proposals are analyzed to reveal that on an average only three segments per proposal are present for both datasets. Thus, while there are nine convolutional networks in the architecture, only three of them need to fire on an average for generating scores from the proposals.

Our experiments show that the CPU throughput for DRUID is 0.99 seconds/image when batch size is 32. The latency on CPU for DRUID (batch size set to 1) is 2.19 seconds. In comparison, the CPU running time for hyperface is 15 seconds per image. Hence, DRUID runs quite fast on a CPU. On a single Titan-X GPU, the latency of DRUID is 0.0816 seconds, which low in comparison to other state-of-the-art methods. For example, on the same GPU, we found that the latency of All-In-One face detector is 0.2 seconds (including proposal generation time of 0.05 seconds) and for mobilenet-SSD it is 0.996 seconds. In addition, DRUID has the additional option to become faster by pruning less important branches. Hence, DRUID could be customized and compressed to develop a high-end face detector for active authentication on smartphones.

4.3.3.5 Sample detection results

Some sample face detection results of the DRUID network along with DPM Baseline, HyperFace, FSFD, SegFace and DeepSegFace are shown in Fig. 4.17. It can be seen that, the proposed network performs much better than the others for extreme illumination and pose variation and partial visibility of faces. More sample images depicting the detection performance of DRUID at various illumination, pose and occlusion are provided in Figs. 4.18 and 4.19 for the UMDAA-02-FD and AA-01-FD datasets, respectively. Fig. 4.20 shows the bounding boxes of facial segments in some images from AFLW. DRUID returned correct bounding boxes with high visibility scores located spatially in appropriate positions. It can be seen that if a face is in profile, then parts that are not visible have very low scores. For example in Fig. 4.20 the top left face's left half is not visible hence left segments like L_{12}, UL_{34} and UL_{12} have zero scores.

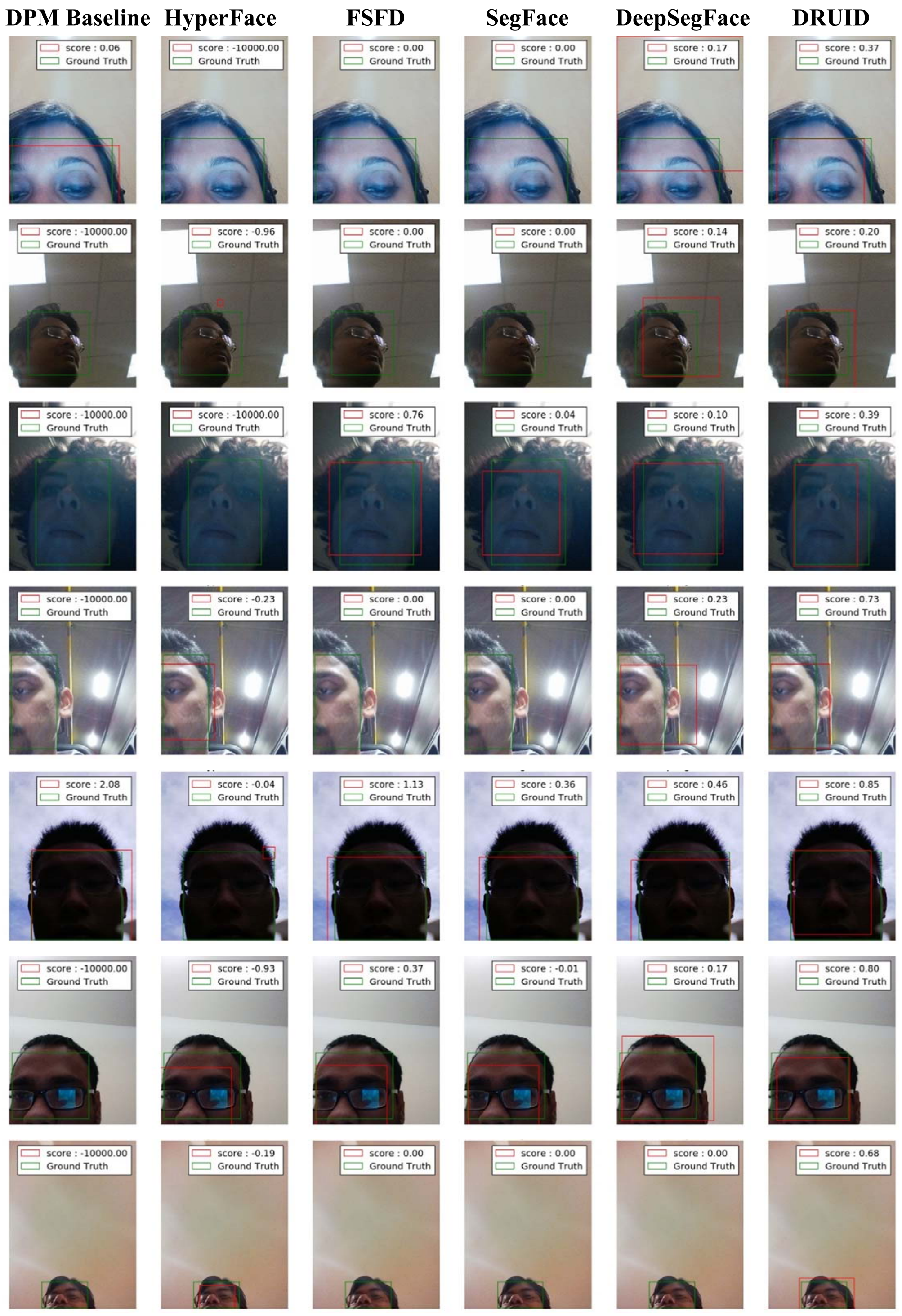


Figure 4.17: Face detection performance comparison of the four facial segment-based detectors with DPM Baseline and HyperFace.

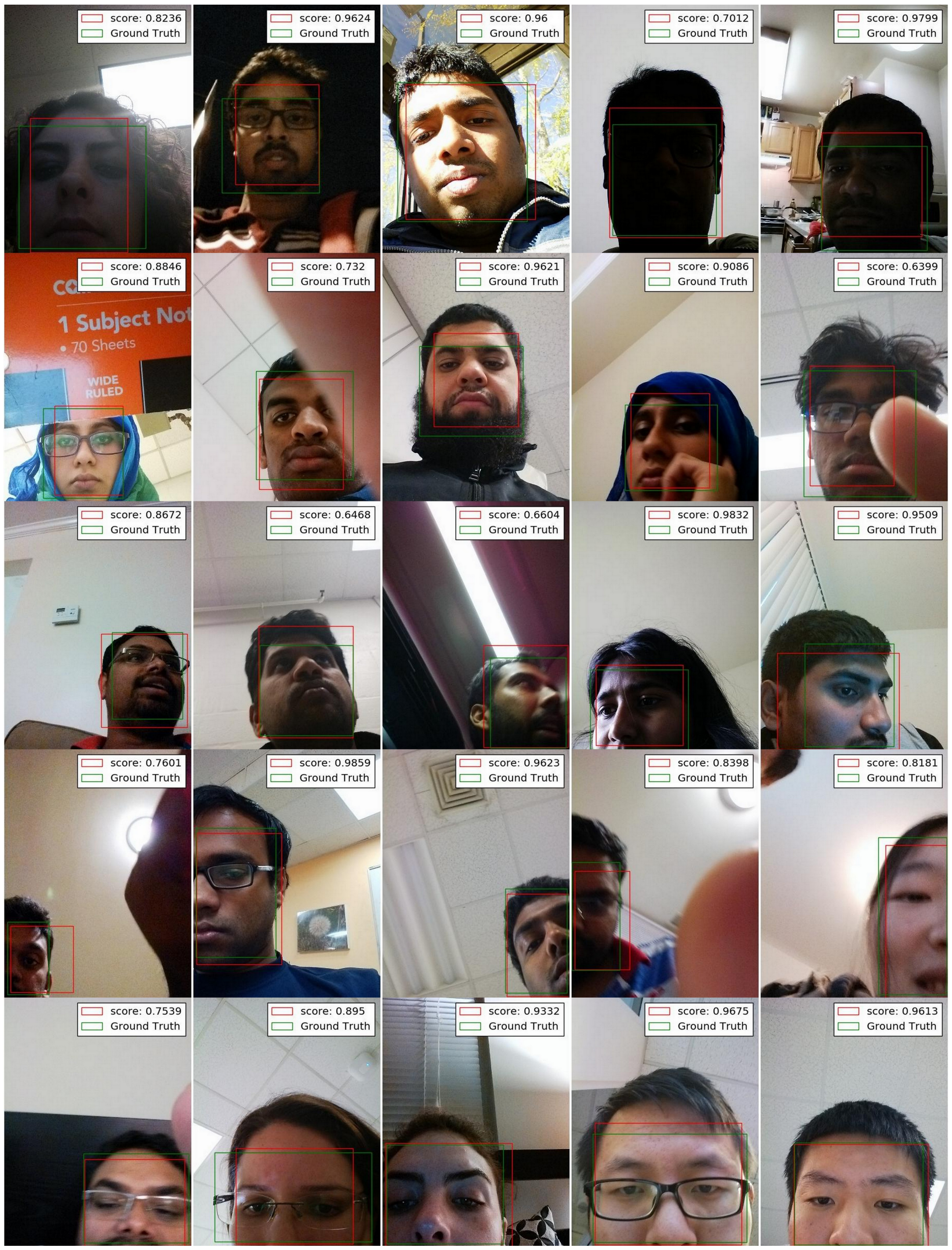


Figure 4.18: Sample face detection outcome of the proposed DRUID method on the UMDAA-02-FD dataset. The five rows (top to bottom) show detection at different illumination, occlusion. pose variation, partial visible side faces and partially visible upper portion of faces, respectively.

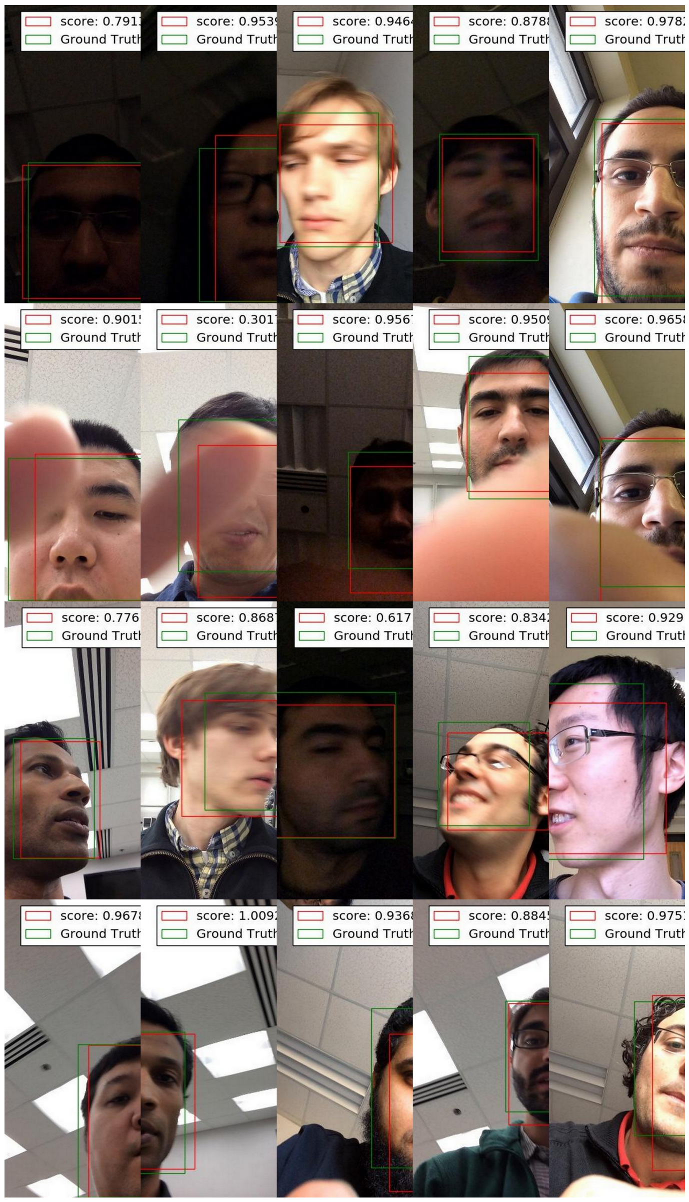


Figure 4.19: Sample face detection outcome of the proposed DRUID method on the AA-01-FD dataset. The four rows (top to bottom) show detection performance at various illumination, occlusion, pose and partial visibility.

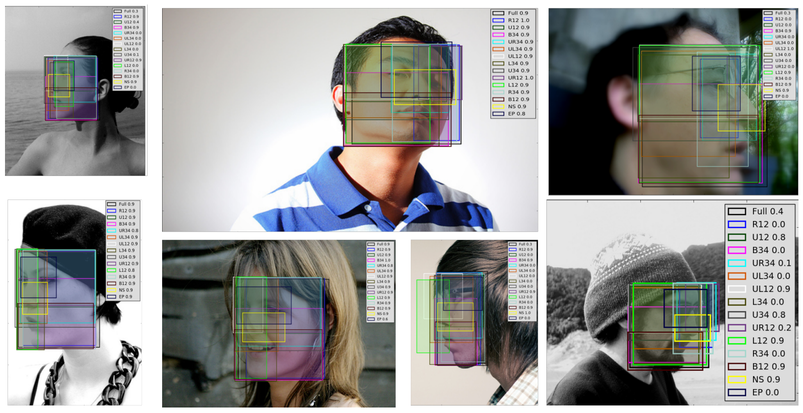


Figure 4.20: Some images from AFLW showing face detections from DRUID. The facial segments bounding boxes and their scores are shown too.

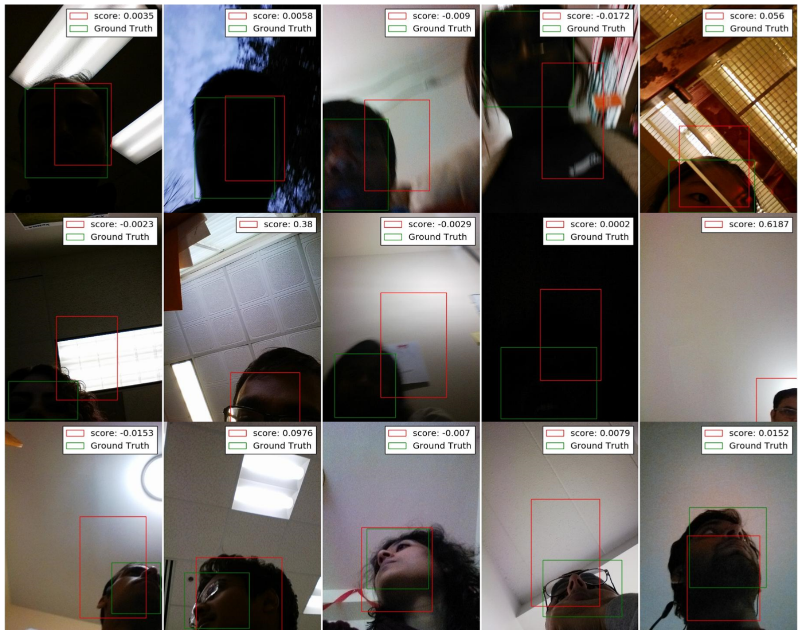


Figure 4.21: Some failure cases of the DRUID network for the UMDAA-02-FD dataset. The first row fails as the images are very blurry, low contrast and difficult in general. The second row shows failures due to very stringent ground-truth annotations that mark near invisible faces or due to cases when the detector detects a partial face that is unannotated in the groundtruth. The third row shows failures due to extreme poses.

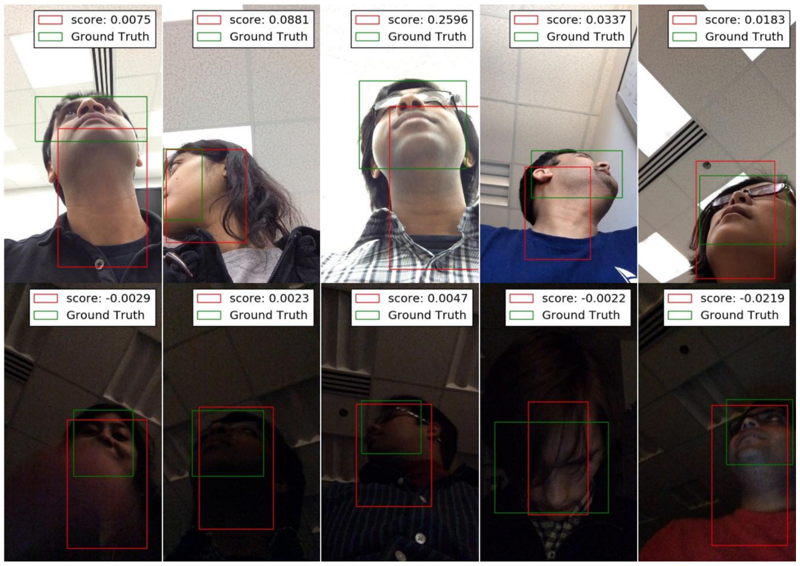


Figure 4.22: Some failure cases of the DRUID network for the AA-01-FD dataset. Most failures can be attributed to extreme poses

Some failure cases are accumulated in Figs. 4.21 and 4.22. It can be seen from these figures, that the network failed mostly in extremely difficult scenarios. In some cases, the ground truth (green) is questionable, for example, in Fig. 4.21, image 2 and 4 of row 2. The first one does not have a proper ground truth while in the second one, the face is not visible at all but a ground truth is marked.

Chapter 5: Facial Segments for Attribute Detection

In this chapter, we introduce a unique deep neural network architecture that utilizes facial segments to estimate the facial attributes in a robust manner. The proposed network showed excellent performance in cross domain settings and also outperformed state-of-the-art attribute detectors both for fully visible and partially occluded faces. Experimental results for the proposed methods and comparisons with state-of-the-art methods along with an analysis of the results and discussions are provided in this chapter.

5.1 Proposed Method

Intuitively, certain facial segments are more effective at predicting a subset of attributes than others. For example, we can expect that segments related to the upper part of the face (e.g. $U12$, $U34$ etc.) would contain information about the person being bald or having certain types and color of hair. Therefore, even if some other part of the face is occluded (e.g. $B12$, EP, or NS being not visible), by looking at the upper portion of the face, one can still predict attributes related to hair. Thus detecting attributes from parts as opposed to the whole face, has the advantage of allowing graceful degradation of performance rather than catastrophic failures with increasing occlusion.

While some attributes can be easily predicted from facial segments, some attributes reflect more global characteristics. For example, one can get hints if a person is young or not from multiple parts of the face, but youth is a global attribute. Therefore it is important to combine the segment predictions into a global prediction, so that multiple parts can contribute to the final prediction. Naturally, the following questions arise:

- *Global vs local attributes:* How does one decide if an attribute is better predicted by facial segments or by a global predictor?
- *Optimal segment selection:* How does one decide which facial segment is more suitable for predicting a particular attribute?
- *Combining results multiple networks:* Given that each attribute is predicted by multiple segments of the proposed network, how does one combine the results optimally?
- *Handling occlusion:* If a certain facial segment responsible for predicting a certain attribute is not visible, how does one get a reasonable prediction?

We can summarize the solutions to these problems as follows.

- *Network architecture:* The first problem is solved by choosing an architecture of a DCNN that is not only able to predict attributes from facial segments, but also performs feature-level fusion of intermediate features through a global prediction network to produce accurate global predictions. Also, the submodules of the network have a Global Average Pooling which endows the networks with localization ability [133].

- *Output pruning:* The second question is answered by the two-stage training approach that is adopted in this work, where the first stage primarily is used to prune the outputs of the segment networks by deciding which segments are good at predicting which attributes.

- *Committee Machines:* For the third problem, we use two-committee machines to perform score level fusion of the multiple predictors to significantly improve the performance of any single constituent network.

- *Hierarchy of best predictors and Segment Dropout:* Finally, to address the fourth problem, we keep track of a hierarchy of segments, which are good at predicting that attribute. Therefore, even if the best segment for an attribute is not present, one can fall back on other segments that are known to do somewhat well for that attribute. We also train our network with 'Segment Dropout' [134] to make it more robust to partial faces.

These ideas are core to our proposed method: **S**egmentwise, **P**artial, **L**ocalized **I**nference in **T**raining **F**acial **A**ttribute **C**lassification **E**nsembles (SPLITFACE). The algorithm looks at facial segments and learns to infer the local attributes, to better handle partial faces. The next four subsections expand on these ideas.

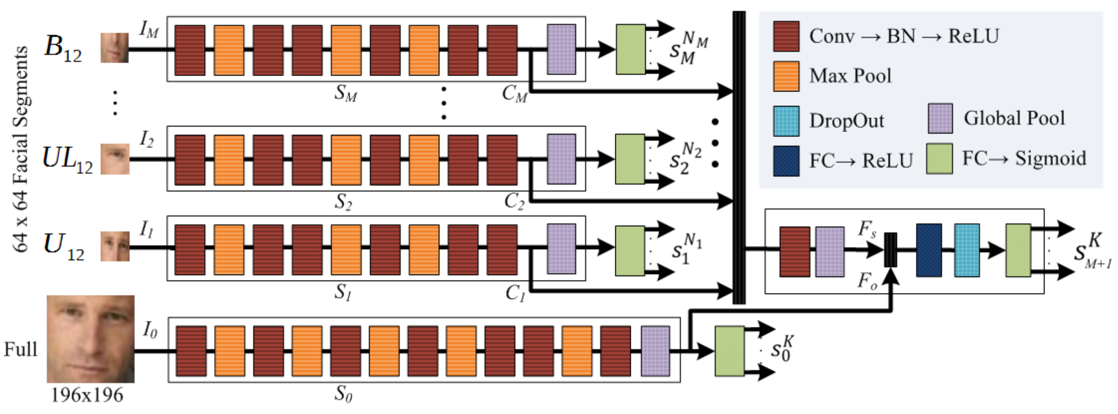


Figure 5.1: SPLITFACE network architecture showing the Facial Segment Networks and the Full Face Network, which are culminate in the Global Prediction Network.

5.1.1 Local to Global Network Architecture

The three constituents of the proposed network namely, the Full Face Network, the Facial Segment Networks and the Global Predictor Network, and their training losses are described in this subsection:

Facial Segment Networks: Let $I_1, I_2, \ldots, I_M$ denote the face regions for the aforementioned $M = 14$ facial segments. Each segment has some predictive power which is unknown initially. In the next section, we describe our data-driven approach to find which attributes are predicted more accurately by each segment. For now, let us say that segment i predicts a set of attributes N_i, where the number of attributes predicted by each segment $|N_i| \leq K$, where $i \in \{1, 2, \ldots, M\}$. Initially, all segments predict all $K = 40$ attributes, but later each segment is allowed to specialize, as described in the next section. We denote these Segment Networks S_i, where $i \in \{1, 2, \ldots, M\}$. When the facial segment I_i is passed through its corresponding segment network S_i, it yields attribute scores s_i for each attribute in

N_i and a feature for that segment C_i, i.e.

$$s_i, C_i = S_i(I_i) \tag{5.1}$$

where C_i is tapped from the last convolutional layer of S_i. The architectures of all the segment networks S_i are same, as described in Table 5.1, and each of these segment networks is independent of one another.

Full Face Network: Let I_0 represent the full-face region, which is passed through a DCNN S_0. We have adopted a seven layer deep convolutional network as S_0. Details on the network architecture are provided in Table 5.1. The full face region is expected to always predict all the K attributes ($|N_0| = K$). Hence, it outputs a vector s_0 of length K, and also a compact feature representation F_0 after global pooling the last convolutional feature, i.e.

$$s_0, F_0 = S_0(I_0) \tag{5.2}$$

Global Prediction Network for local feature fusion: In the Global Prediction Network, we combine the results from the local segment networks and the full face network to produce predictions for all the K attributes. To do so, we first concatenate the convolutional features C_i from the M segment networks, convolve them, then apply global pooling to get a flattened feature from the segments, F_s. This is concatenated with F_0, the flattened features from the Full Face Network, and passed through a few fully connected layers, to finally yield predictions for all the

$K = 40$ attributes. The Global Prediction Network can be thought of as performing a feature level fusion of the different segments, as opposed to the score level fusion of committee machines described in the next section. The output of the Global Predictor is

$$s_{M+1} = S_{M+1}(F_0, C_1, \ldots, C_M). \tag{5.3}$$

The color-coded network architecture for the SPLITFACE network is shown in Fig. 5.1. It shows the above mentioned architectural choices, namely predictions from segments, predictions from the full face and fusion of segment and full face features to provide a global prediction. In what follows, we shall use the word 'predictor' to mean any of the $M+2$ sub-networks $S_i, i \in \{0, 1, \ldots, M+1\}$ described above, that is, any of the M Facial Segment Networks, the Full Face Network or the Global Prediction Network.

Localization using Global Average Pooling: It has been shown in [133] that Global Average Pooling (GAP) introduced in [135] has remarkable localization properties. Since we are aiming to predict localized attributes well from partial segments, we use a GAP layer in the architecture to transition from convolutional to fully connected layers. Using Class Activation Maps (CAM) in section 5.2.3 we observe that this provides the network with the desired property of being able to focus on regions of interest, thus making the process more interpretable.

Loss: We use binary crossentropy loss for all the predictor outputs $s_i, i \in \{0, 1, \ldots, M+1\}$ described in 5.1, 5.2, 5.3, weighted by the inverse of priors. Then

Table 5.1: Detailed network architecture.

S_i, $i \in \{1, 2, \ldots, M\}$	S_0	S_{M+1}
conv3-32 $\rightarrow$ BN $\rightarrow$ ReLU	conv3-32 $\rightarrow$ BN $\rightarrow$ ReLU	conv3-512 $\rightarrow$ BN $\rightarrow$ ReLU
2D MaxPool 3×3, Stride 2	2D MaxPool 3×3, Stride 2	global average pooling
conv3-64 $\rightarrow$ BN $\rightarrow$ ReLU	conv3-64 $\rightarrow$ BN $\rightarrow$ ReLU	merge(F_0, F_s)
conv3-64 $\rightarrow$ BN $\rightarrow$ ReLU	2D MaxPool 3×3, Stride 2	dense-256D$\rightarrow$ ReLU
2D MaxPool 3×3, Stride 2	conv3-64 $\rightarrow$ BN $\rightarrow$ ReLU	dropout(0.2)
conv3-128 $\rightarrow$ BN $\rightarrow$ ReLU	2D MaxPool 3×3, Stride 2	
2D MaxPool 3×3, Stride 2	conv3-128 $\rightarrow$ BN $\rightarrow$ ReLU	
conv3-128 $\rightarrow$ BN $\rightarrow$ ReLU	2D MaxPool 3×3, Stride 2	
conv3-256 $\rightarrow$ BN $\rightarrow$ ReLU	conv3-128 $\rightarrow$ BN $\rightarrow$ ReLU	
	conv3-256 $\rightarrow$ BN $\rightarrow$ ReLU	
	2D MaxPool 3×3, Stride 2	
	conv3-256 $\rightarrow$ BN $\rightarrow$ ReLU	

the loss L incurred on image I is

$$L(I) = \sum_{i=0}^{M+1} \sum_{j=0}^{K-1} w_{I,j} \log(s_i(j)). \tag{5.4}$$

In 5.4, $w_{I,j}$ is a weight based on the ground truth I_j and the prior probability p_j of attribute j being present, which is precomputed on the training set. The weight $w_{I,j}$ defined in 5.5 helps to mitigate the challenges due to unbalanced class distributions which are prevalent in datasets like CelebA [36]. $w_{I,j}$ can be expressed as

$$w_{I,j} = \begin{cases} p_j, & \text{if } I_j = 0 \\ 1 - p_j & \text{if } I_j = 1. \end{cases} \tag{5.5}$$

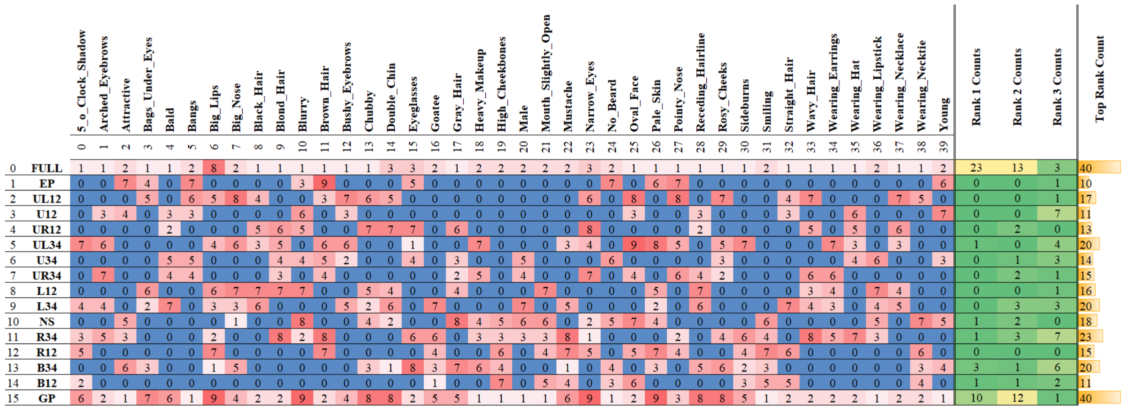
5.1.2 Optimal segment selection for output pruning

Intuitively, not all segments predict all attributes well. Therefore it is counterproductive to train the network to produce all K predictions from all M segment networks. Instead, we follow a data-driven approach to prune the segment networks.

Stage 1: Initially, we predict all $K = 40$ attributes from all M segment networks, the full face network and the global prediction network. Hence each attribute is predicted by $M + 2$ networks. After training for several epochs, we evaluate the detection accuracy of each of the M segment networks, S_i. For each attribute, we sort the $M + 2$ networks according to accuracy on validation set, and pick the top d networks for each attribute. The global predictor (GP) and the full face network can be expected to be among the best predictors most of the time, since they have a top view of the sum of parts. The rest of the $d - 2$ predictors of each attribute are segment networks and therefore the most associated N_i attributes for segment i where $i \in \{1, 2, \ldots, M\}$ and $N_i \leq K$ is determined this way.

In Fig.5.2 a table for $d = 7$ and $M = 14$ is shown where for the segment networks (row three and below) the non-zero numbers for different attribute columns denote the attributes assigned to the segment after pruning. The total number of attributes assigned to the segment networks after pruning are shown in the last column.

Stage 2: After the association of attributes with segments as described above, a second round of training is performed. The pruning process in stage 1 allows the segment networks to focus on attributes that they perform best on, without having



		5_o_Clock_Shadow	Arched_Eyebrows	Attractive	Bags_Under_Eyes	Bald	Bangs	Big_Lips	Big_Nose	Black_Hair	Blond_Hair	Blurry	Brown_Hair	Bushy_Eyebrows	Chubby	Double_Chin	Eyeglasses	Goatee	Gray_Hair	Heavy_Makeup	High_Cheekbones
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
0	FULL	1	1	2	1	1	2	8	2	1	1	1	1	1	1	3	3	2	1	2	2
1	EP	0	0	7	4	0	7	0	0	0	0	3	9	0	0	0	5	0	0	0	0
2	UL12	0	0	0	5	0	6	5	8	4	0	0	3	7	6	5	0	0	0	0	0
3	U12	0	3	4	0	3	3	0	0	0	0	6	0	3	0	0	0	0	0	0	0
4	UR12	0	0	0	0	2	0	0	0	5	6	5	0	0	7	7	7	0	6	0	0
5	UL34	7	6	0	0	0	0	4	6	3	5	0	6	6	0	0	1	0	0	7	0
6	U34	0	0	0	0	5	5	0	0	0	4	4	5	2	0	0	4	0	3	0	0
7	UR34	0	7	0	0	4	4	0	0	0	3	0	4	0	0	0	0	0	2	5	0
8	L12	0	0	0	6	0	0	6	7	7	7	7	0	0	5	4	0	0	4	0	0
9	L34	4	4	0	2	7	0	3	3	6	0	0	0	5	2	6	0	7	0	0	0
10	NS	0	0	5	0	0	0	0	1	0	0	8	0	0	4	2	0	0	8	4	5
11	R34	3	5	3	0	0	0	2	0	0	8	2	8	0	0	0	6	6	0	3	3
12	R12	5	0	0	0	0	0	7	0	0	0	0	7	0	0	0	0	4	0	0	6
13	B34	0	0	6	3	0	0	1	5	0	0	0	0	0	3	1	8	3	7	6	4
14	B12	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	7
15	GP	6	2	1	7	6	1	9	4	2	2	9	2	4	8	8	2	5	5	1	1

		Male	Mouth_Slightly_Open	Mustache	Narrow_Eyes	No_Beard	Oval_Face	Pale_Skin	Pointy_Nose	Receding_Hairline	Rosy_Cheeks	Sideburns	Smiling	Straight_Hair	Wavy_Hair	Wearing_Earrings	Wearing_Hat	Wearing_Lipstick	Wearing_Necklace	Wearing_Necktie	Young	Rank 1 Counts	Rank 2 Counts	Rank 3 Counts	Top Rank Count
		20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39				
0	FULL	2	2	2	3	2	1	1	1	1	1	1	2	1	1	1	1	2	1	1	2	23	13	3	40
1	EP	0	0	0	0	7	0	6	7	0	0	0	0	0	0	0	0	0	0	0	6	0	0	1	10
2	UL12	0	0	0	6	0	8	0	8	0	7	0	0	4	7	0	0	0	7	5	0	0	0	1	17
3	U12	0	0	0	0	0	3	0	0	3	0	0	0	3	0	0	6	0	0	0	7	0	0	7	11
4	UR12	0	0	0	8	0	0	0	0	2	0	0	0	0	5	0	5	0	6	0	0	0	2	0	13
5	UL34	0	0	3	4	0	9	8	5	0	5	7	0	0	0	7	3	0	3	0	0	1	0	4	20
6	U34	5	0	0	0	6	0	0	0	0	3	0	0	0	0	0	4	6	0	0	3	0	1	3	14
7	UR34	4	0	0	7	0	4	0	6	4	2	0	0	0	6	6	0	0	0	0	0	0	2	1	15
8	L12	0	7	0	0	0	0	5	0	7	0	0	0	0	3	4	0	7	4	0	0	0	0	1	16
9	L34	7	0	5	0	0	0	2	0	6	0	0	0	7	4	3	0	4	5	0	0	0	3	3	20
10	NS	6	6	0	2	5	7	4	0	0	0	0	6	0	0	0	0	5	0	7	5	1	2	0	18
11	R34	3	3	8	1	0	0	0	2	0	4	6	4	0	8	5	7	3	0	0	0	1	3	7	23
12	R12	0	4	7	5	0	5	7	4	0	0	4	7	6	0	0	0	0	0	6	0	0	0	0	15
13	B34	0	0	1	0	4	0	3	0	5	6	2	3	0	0	0	0	0	0	3	4	3	1	6	20
14	B12	0	5	4	0	3	6	0	0	0	0	3	5	5	0	0	0	0	0	4	0	1	1	2	11
15	GP	1	1	6	9	1	2	9	3	8	8	5	1	2	2	2	2	1	2	2	1	10	12	1	40

Figure 5.2: The top ranked segments (including GP and Full face, row-wise) for each attribute (in the columns). The blue cells indicate that that particular segment was not used to predict that attribute in the stage 2 of training. The segments predict attributes that are localized in that region. For example, the bottom half segment predicts attributes related to facial hair.

to worry about attributes they are just not capable of predicting. Hence, in Fig. 5.1 the output of the FC layers are denoted as $s_0^K, s_1^{N_1}, \dots, s_M^{N_M}, s_{M+1}^K$, where $N_i \leq K$ denotes the dimension of the outputs, i.e. the number of attributes being predicted by the branch. We have intentionally assigned all the attributes to GP and FULL networks (as shown in Fig. 5.2), since the receptive field for these two networks encompasses the entire face. So, we make the inherent assumption that these two networks are capable of successfully predicting all the attributes and set the output to s_0^K and s_{M+1}^K, respectively for FULL and GP, with dimensions K for these two networks.

5.1.3 Committee Machines for Score-Level Fusion

While the global predictor or the full face networks has good predictive power, using only those two predictors does not harness the full potential of the proposed network architecture. To utilize all the segment networks along with the global and full face predictors, we describe two committee machines here namely, the Highest Ranked Predictor (HRP) and the Normalized Score Aggregation (NSA) methods, that perform score-level fusion. For both methods, using the validation set V, we first compute the optimal thresholds, t_{a,i_a} for each attribute $a \in \{1, 2, \ldots, K\}$ and for each predictor $i_a \subset \{0, 1, \ldots, M+1\}$ which are responsible for predicting a. For CelebA, the table in Fig. 5.2 shows information about i_a, which is an ordered set or tuple, ordered in descending order of validation accuracy. For example, if we consider the attribute 'goatee', then $a = 16$ and $i_{a=16} = (14, 0, 13, 12, 15, 11, 9)$, which correspond to B12, FULL, B34, R12, GP, R34 and L34. Denoting I_a as the ground truth of attribute a for sample I and $\mathbb{1}$ as the indicator function, the optimal thresholds that maximize validation accuracy are computed as

$$t_{a,i} = \underset{t \in [0,1]}{\arg\min} \sum_{I \in V} \mathbb{1}_{I_a = \mathbb{1}_{S_i(I) < t}}. \tag{5.6}$$

We denote the visibility of a segment j for image I by $V_{I,j} \in \{0, 1\}$. Clearly $V_{I,0} = V_{I,15} = 1$, since both the Full Face Predictor and Global Predictor Network can predict attributes no matter what the occlusion is. Finally, we define an ordered set i_a^v which contains the top usable predictors for attribute a (ones that have visible segments) as

$$i_a^v = (x : x \in i_a, V_{I,x} = 1). \tag{5.7}$$

Hence, i_a^v for a given test image is computed by selecting only the visible segments from i_a. During test phase, the segment visibility is determined from the fiducial landmark detector results. Then, i_a^v simply selects the branches that are visible and are capable of predicting a particular attribute well, based on evaluation on training data i_a.

5.1.3.1 Highest Ranked Predictor (HRP) Committee Machine

After the completion of the two training stages, we evaluate the performance of each of the predictors (segment, full face and global networks) on the validation set to find a hierarchy of best performing predictors for each attribute. The results are shown in Fig. 5.2. For example, we can see that the best predictors for 'goatee' are B12, FULL, B34, R12, GP, R34 and L34, in descending order of validation accuracy.

When making a prediction for an attribute, we find the topmost usable predictor $j = i_a^v(1)$ that is usable/visible for that image. This score from predictor $S_j(I_j)$ is then thresholded with the optimal threshold of that segment for that attribute, which was precomputed from the validation set following 5.6. The prediction outcome based on the optimal threshold would be

$$P_a(I) = \mathbb{1}_{S_j(I_j) < t_{a,j}}. \tag{5.8}$$

In the example on 'goatee', we use the prediction of B12 ($= i_a(0)$), and if that segment is not visible, we use FULL ($= i_a(1)$).

5.1.3.2 Normalized Score Aggregation (NSA) Committee Machine

In general, different predictors trying to predict the same attribute might have different optimal thresholds. Once they are aggregated (say, by taking their mean or product or median), one needs to calculate the optimal threshold for the aggregate score. Instead, we could normalize the scores of the predictors so that after aggregation, the optimal threshold for the aggregate score is 0.5. [136] suggests a double sigmoid score normalization function for fusing scores from multiple predictors. However, it involves 2 hyper-parameters, which need to be found by cross-validation. Instead we propose a simpler normalization function below, which does not require any hyper-parameters.

Linear Threshold Normalization: Consider a binary classification problem, where we have to decide the class $C \in \{0, 1\}$, given a score $X \in [0, 1]$. We assume that the optimal threshold t that maximizes separation between the 2 classes is known, and therefore, $P(C = 1|X = t) = 0.5$. Considering a transformation $Y = T(X)$, we wish to identify the function T, such that, $P(C = 1|Y = 0.5) = 0.5$. Hence,

$$\begin{aligned} P(C = 1|Y = 0.5) &= P(C = 1|T(X) = 0.5) \\ &= P(C = 1|X = T^{-1}(0.5)). \end{aligned} \tag{5.9}$$

If we choose an invertible function T, such that $T(t) = 0.5$, then above equation yields $P(C = 1|Y = 0.5) = 0.5$. Thus, given multiple scores X_i, and their optimal thresholds t_i, we can transform the scores to $Y_i = T(X_i)$, so that after aggregating Y_i, say by averaging, the optimal threshold is 0.5.

For our algorithm, we use a piecewise linear transformation as follows that satisfies the $T(t) = 0.5$ criterion discussed above.

$$T_t(x) = \begin{cases} (0.5/t) \times x, & \text{if } 0 \leq x \leq t \\ \\ (0.5x + 0.5 - t)/(1 - t) & \text{if } t < x \leq 1. \end{cases} \tag{5.10}$$

We transform the scores of at least $p = 5$ top predictors out of the visible ones, i_a^v, using (5.10) to yield Z in (5.11).

$$Z = \{T_{t_{a,i_a^v(k)}}(S_{i_a^v(k)}(I_{i_a^v(k)})) : k \in \{1, 2, \ldots, min(|i_a^v|, p)\}\} \tag{5.11}$$

Finally, we use an aggregation function A on Z for the prediction. The decision rule using the aggregator function is

$$P_a(I) = \mathbb{1}_{A(Z) \geq A(\{1-z : z \in Z\})}. \tag{5.12}$$

Possible choices of aggregator functions are:

- *Bayes' Rule or Product Rule*: As discussed in the score fusion literature [137] we can use a product rule to combine the decisions of N binary classifiers according to the following decision rule

$$\prod^{N} P(C = 1|x) \lesseqgtr \prod^{N} (1 - P(C = 1|x)). \tag{5.13}$$

Thus the product aggregator function is $A(Z) = \prod_{z \in Z} z$ for all $z \in Z$.

- *Median Rule*: As proposed in [138], the median aggregator function is $A(Z) = med(Z)$.

5.1.4 Segment Dropout and Hierarchy of Best Predictors for Handling Occlusion

Segment Dropout: When training the network with image I, only a subset of the $M = 14$ segments might be present. The visible segments are randomly dropped with probability 30% when training. This is called Segment Dropout, which was introduced in [134] to augment the dataset for handling occlusion. When a certain segment is not present in a face the input to corresponding segment branch is zero. In order to make SPLITFACE robust against such cases and generalize better to detect attributes from available segments, random segment dropout is performed.

Hierarchy of Best Predictors: As described in the earlier section, we compute a hierarchy of predictors i_a^v that are visible. Thus even if a face is partially occluded and the best segment is not available, the other segments provide reasonable predictive power to the committee machine.

The unique architecture of SPLITFACE allows the use of predictor hierarchy and thereby improving the detection accuracy. In addition, it ensures that even if some part of the input face is not visible due to occlusion or failure of the face detector, the attribute detection network would rely on the visible segments to still make a good prediction. Note that the input to GP are the features from all the segment networks and the full face network, and our partial face augmentation approach during training enables it to handle missing segments while predicting attributes.

5.2 Experimental Setup, Evaluation and Discussion

5.2.1 Datasets

We use the CelebA [81] and LFWA [81] datasets for both training and evaluation. Also, to evaluate the SPLITFACE's capability for handling partially visible faces when estimating facial attributes, we created several variations of these two datasets and evaluate the performance of SPLITFACE on those variations. We follow the data augmentation scheme described in [83] for generating partially visible faces by cropping the images keeping only L12 or L34 or R12 or R34 or U12 or U34 portion. We replace the rest of the pixels with white pixels. Hence, we create six variations of both datasets and named them C$-P$ and L$-P$, respectively for CelebA and LFWA, where $P \in \{$L12, L34, R12, R34, U12, U34$\}$ [1]. Some sample images for C$-P$ dataset are shown in Fig. 5.3.

5.2.2 Implementation Details

The proposed network has $26,090,334$ trainable parameters, which are tuned using the adaptive moment estimation (ADAM) optimizer [139]. The initial learning rate was set to 0.001. For CelebA, we train the network for 10 epochs for both stages,

[1]Bounding boxes for the partial CelebA and partial LFWA datasets are available at `https://drive.google.com/open?id=16hL7g3d6dfvbdvwarYfT6zNcNNXcRLlr`

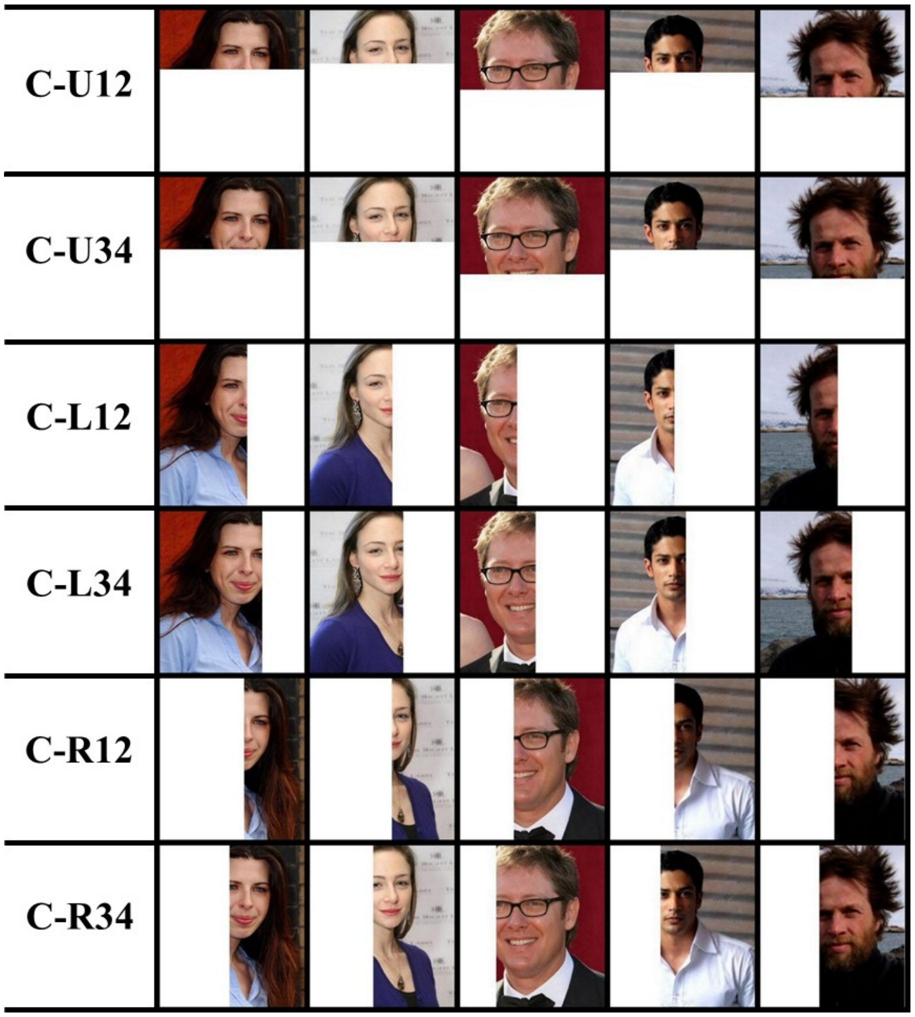


Figure 5.3: Modified CelebA dataset samples for partial faces.

while for LFWA, we train it for 180 epochs in the first stage and 270 epochs in the second stage. The full face region is resized to $196 \times 196 \times 3$ and given as input to the full face branch, which the inputs to the facial segment branches are all resized to $64 \times 64 \times 3$. The experiments were performed on NVIDIA Quadro P6000 GPUs, with training batch size of 200, and the code was written using the KERAS python library [140] with tensorflow [141] backend. Apart from segment dropout, horizontal flipping was applied for data augmentation. Among state-of-the-art methods, the authors of AFFACT [1] have provided the source code in their paper, which we

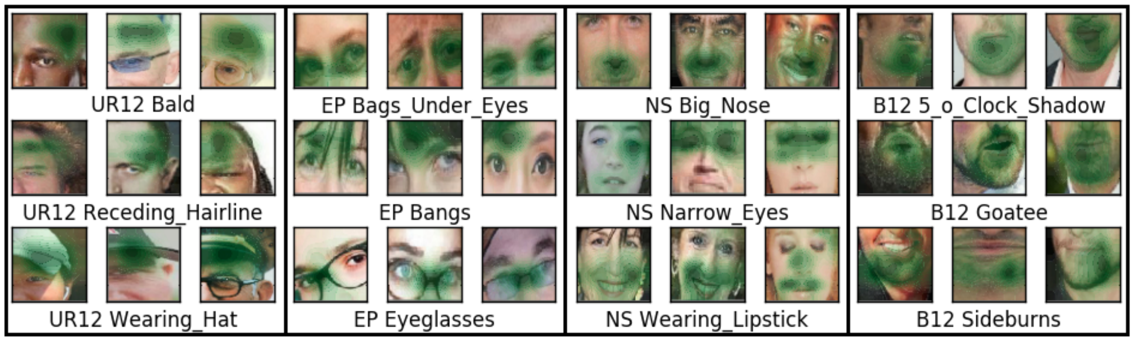


Figure 5.4: Visualization of Class Activation Maps for four different facial segments (UR12, EP, NS and B12 in the four quarters from left to right) and some attributes estimated by the corresponding block.

used for performance comparison on partial face datasets. However, the accuracy obtained from this implementation is slightly less than the accuracy reported in [1], perhaps because we have not applied test time data augmentation. For all the other methods, we directly report the results in corresponding publications.

5.2.3 Visualizing Network Response using Class Activation Maps

The class activation map was proposed in [133] to visualize the localization properties of the network. Given a network which terminates in a Global Average Pooling (GAP) layer followed by a dense layer, we can compute the CAM C_i of a particular class i as a weighted average of the activation maps of the layer just before the GAP layer as

$$C_i = \sum_{j}^{N} w_{i,j} M_j, \tag{5.14}$$

where M_j is the j^{th} feature map in a feature tensor of depth N just before the GAP layer and $w_{i,j}$ is the corresponding weight of the dense layer after the GAP layer. In Fig. 5.4, we show the CAM superimposed on some facial segments from CelebA.

Clearly, the activation maps are localized in interpretably meaningful regions. It can be seen from Fig. 5.4 that for all the three attributes shown for UR12 ('bald', 'receding hairline' and 'wearing hat') and B12 ('5 o clock shadow', 'goatee' and 'sideburns'), the network focuses on the same region: the top corner of the head for UR12 and the chin and cheeks for B12. On the other hand, for segments EP and NS, the attention shifted to different regions for different attributes. For example, for segment EP, the attribute 'bags under eyes' is predicted when the network has high response near the eyes, 'bangs' are predicted when the response is high near the forehead and 'eyeglasses' are predicted by looking at the bridge of the nose. Similarly the NS segment network shifted its attention to the nose, eyes or lips to predict 'big nose', 'narrow eyes' and 'wearing lipstick', respectively.

5.2.4 Performance Comparison on Original CelebA and LFWA datasets

The performance of the proposed method is compared with state-of-the-art methods in Tables 5.2 and 5.3 on the original CelebA and LFWA datasets, respectively. Among state-of-the-art methods, the result of AFFACT is directly reported from [1], while the column AFFACT Unaligned contains results that we found by evaluating the full faces we use in our experiment. Since the source codes were not available for any of the other state-of-the-art methods, we report the results directly from corresponding publications. The column titled 'Prior' shows the accuracies obtainable by only applying the knowledge from the prior probabilities of the presence or absence of an attribute in the datasets. It can be seen that a staggering mean accuracy of 80.57% in CelebA and 71.27% in LFWA is achievable by only using the prior probabilities in decision making. Even though state-of-the-art methods and the proposed method increases this number by more than 10%, for certain attributes such as Big Lips and Narrow Eyes in Table 5.2, the prior is higher than the trained methods for most of the approaches. We put the prior column in the table as a baseline for evaluation.

The last five columns in Tables 5.2 and 5.3 show the attribute-wise accuracy and the mean accuracy for Full, GP, HRP, NSA Product rule and NSA Median rule, respectively, in both tables. It can be seen from these tables that the committee machine approaches boost the results obtained from Full and GP for most of the attributes. The mean accuracy of 90.42% for CelebA and 85.85% for LFWA obtained from the NSA Product Rule closely matches state-of-the-art results presented in the table. Note that we adopted a very simple six layer convolutional network for the Full face branch that achieves 90.72% accuracy on CelebA and 84.02% accuracy on LFWA. The result for CelebA is boosted for HRP but degrades slightly for NSA methods. On the other hand, for LFWA, the committee machine approaches improve the overall performance. Since LFWA is a much smaller dataset and hence the trained network over-fitted greatly on the training set, this boost in result shows that the proposed committee machine approaches, especially NSA, generalizes well due to their ensemble aggregation mechanism. In later sections, we will present results for partially visible faces, where the committee machine approaches consistently improves over Full and GP branches and hence the adaptation of such methods is justified for practical purposes even with a slight loss in accuracy for the original dataset.

Table 5.2: Attribute detection performance comparison on the CelebA dataset in terms of individual and mean detection accuracy for the attributes.

									Proposed	Proposed	Proposed Committee Machine		
Attributes	**Prior**	**LENet+**	**MOON [36]**	**MCNN+**	**DMTL [2]**	**AFFACT [1]**	**AFFACT**	**PaW [84]**	**FULL**	**GP**	**HRP**	**NSA**	
		Anet [81]		**AUX [37]**			**Unaligned [1]**					**Prod. Rule**	**Med. Rule**
5_o_Clock_Shadow	88.83	91.00	94.03	94.51	95.00	94.21	94.09	94.64	93.96	90.00	93.96	93.01	93.13
Arched_Eyebrows	73.41	79.00	82.26	83.42	86.00	82.12	81.27	83.01	83.39	83.44	83.39	82.44	82.56
Attractive	51.36	81.00	81.67	83.06	85.00	82.83	80.36	82.86	82.71	82.86	82.86	83.13	82.76
Bags_Under_Eyes	79.55	79.00	84.92	84.92	85.00	83.75	84.89	84.58	85.12	79.72	85.12	84.63	84.86
Bald	97.72	98.00	98.77	98.90	99.00	99.06	97.82	98.93	98.46	97.88	98.46	97.98	98.03
Bangs	84.83	95.00	95.80	96.05	99.00	96.05	95.49	95.93	95.65	95.72	95.72	95.73	95.71
Big_Lips	75.91	68.00	71.48	71.47	96.00	70.88	71.42	71.46	67.29	67.29	67.29	69.78	69.28
Big_Nose	76.44	78.00	84.00	84.53	85.00	83.82	81.83	83.63	83.91	81.85	83.36	81.31	83.81
Black_Hair	76.10	88.00	89.40	89.78	91.00	90.32	85.88	89.84	88.88	72.85	88.88	88.82	89.03
Blond_Hair	85.09	95.00	95.86	96.01	96.00	96.07	95.17	95.85	95.70	95.68	95.70	95.04	95.76
Blurry	94.86	84.00	95.67	96.17	96.00	95.50	94.52	96.11	95.87	94.95	95.87	95.04	95.96
Brown_Hair	79.61	80.00	89.38	89.15	88.00	89.16	87.72	88.50	88.42	87.64	88.42	85.59	88.25
Bushy_Eyebrows	85.63	90.00	92.62	92.84	92.00	92.41	90.59	92.62	92.41	92.20	92.41	91.82	92.66
Chubby	94.23	91.00	95.44	95.67	96.00	94.98	95.10	95.46	94.69	94.69	94.69	93.90	94.94
Double_Chin	95.35	92.00	96.32	96.32	97.00	96.18	95.94	96.26	95.43	95.43	95.68	95.23	95.80
Eyeglasses	93.54	99.00	99.47	99.63	99.00	99.61	99.38	99.59	99.43	99.48	99.30	99.58	99.51
Goatee	93.65	95.00	97.04	97.24	99.00	97.31	97.21	97.38	96.51	95.41	96.70	95.88	96.68
Gray_Hair	95.76	97.00	98.10	98.20	98.00	98.28	97.89	98.21	97.57	95.99	97.57	95.80	97.45
Heavy_Makeup	61.57	90.00	90.99	91.55	92.00	91.10	90.82	91.53	91.18	91.51	91.51	91.55	91.59
High_Cheekbones	54.76	87.00	87.01	87.58	88.00	86.88	86.11	87.44	87.08	87.54	87.54	87.62	87.61
Male	58.06	98.00	98.10	98.17	98.00	98.26	97.29	98.39	97.58	98.14	98.14	98.09	97.95
Mouth_Slightly_Oper	51.78	92.00	93.54	93.74	94.00	92.60	92.82	94.05	93.62	93.91	93.91	93.90	93.78
Mustache	95.92	95.00	96.82	96.88	97.00	96.89	96.89	96.90	96.12	96.12	96.12	96.16	95.86
Narrow_Eyes	88.41	81.00	86.52	87.23	90.00	87.23	87.15	87.56	86.79	85.13	86.84	87.31	86.88
No_Beard	83.42	95.00	95.58	96.05	97.00	95.99	95.33	96.22	95.77	96.17	96.17	95.57	96.17
Oval_Face	71.68	66.00	75.73	75.84	78.00	75.79	74.87	75.03	75.40	70.45	75.40	75.75	74.93
Pale_Skin	95.70	91.00	97.00	97.05	97.00	97.04	96.97	97.08	96.90	95.80	96.90	96.72	97.00
Pointy_Nose	72.45	72.00	76.46	77.47	78.00	74.83	76.24	77.35	76.13	71.45	76.13	76.46	76.47
Receding_Hairline	91.99	89.00	93.56	93.81	94.00	93.29	91.74	93.44	92.55	91.52	92.55	92.40	92.25
Rosy_Cheeks	93.53	90.00	94.82	95.16	96.00	94.45	94.54	95.07	94.59	92.83	94.59	94.51	94.79
Sideburns	94.37	96.00	97.59	97.85	98.00	97.83	97.46	97.64	96.83	96.09	96.83	96.01	97.17
Smiling	52.03	92.00	92.60	92.73	94.00	91.77	90.45	92.73	92.42	92.74	92.74	92.89	92.70
Straight_Hair	79.14	73.00	82.26	83.58	85.00	84.10	82.17	83.52	83.11	79.04	83.11	82.36	80.41
Wavy_Hair	68.06	80.00	82.47	83.91	87.00	85.65	83.37	84.07	83.28	63.58	83.28	83.10	81.70
Wearing_Earrings	81.35	82.00	89.60	90.43	91.00	90.20	90.33	89.93	90.41	90.48	90.41	89.72	89.44
Wearing_Hat	95.06	99.00	98.95	99.05	99.00	99.02	98.66	99.02	98.71	95.79	98.71	98.42	98.74
Wearing_Lipstick	53.04	93.00	93.93	94.11	93.00	91.69	92.99	94.24	92.66	93.23	93.23	94.00	93.21
Wearing_Necklace	87.86	71.00	87.04	86.63	89.00	87.85	87.55	87.70	87.54	86.22	87.54	87.50	85.61
Wearing_Necktie	92.70	93.00	96.63	96.51	97.00	96.90	96.43	96.85	96.66	95.61	96.66	95.24	96.05
Young	77.89	87.00	88.08	88.48	90.00	88.66	86.21	88.59	87.95	88.45	88.45	86.93	88.01
Mean Accuracy	**80.57**	**87.30**	**90.94**	**91.29**	**92.60**	**91.01**	**90.32**	**91.23**	**90.72**	**88.87**	**90.80**	**90.42**	**90.61**

5.2.5 Cross-Dataset Testing Accuracies

In table 5.4, we present the cross-dataset testing performances of AFFACT, DMTL and SPLITFACE (NSA product rule). For AFFACT and the proposed method, we present two accuracies separated by /, the first one is for using the optimal threshold obtained from the validation set to find detection results (for AFFACT) or to normalize scores before applying product rule (for proposed). And, the

Table 5.3: Attribute detection performance comparison on the LFWA dataset in terms of individual and mean detection accuracy for the attributes.

					Proposed	**Proposed**	**Proposed Committee Machine**		
Attributes	**Prior**	**LENet+**	**MCNN+**	**DMTL** [2]	**FULL**	**GP**	**HRP**	**NSA**	
		Anet [81]	**AUX** [37]					**Prod. Rule**	**Med. Rule**
5_o_Clock_Shadow	59.76	84	77.06	80	74.72	74.72	74.72	77.47	77.59
Arched_Eyebrows	72.35	82	81.78	86	78.78	78.78	78.78	81.82	81.72
Attractive	62.09	83	80.31	82	77.44	77.44	77.44	80.25	80.16
Bags_Under_Eyes	59.52	83	83.48	84	79.11	79.11	79.11	82.98	82.62
Bald	88.94	88	91.94	92	91.69	91.51	91.51	90.97	91.88
Bangs	83.57	88	90.08	93	89.72	89.72	89.72	90.89	90.71
Big_Lips	64.07	75	79.24	77	75.47	77.54	77.54	79.10	78.97
Big_Nose	69.62	81	84.98	83	80.23	80.23	80.23	82.95	83.13
Black_Hair	85.53	90	92.63	92	91.63	92.22	92.22	92.34	92.49
Blond_Hair	95.75	97	97.41	97	97.31	97.31	97.31	97.47	97.47
Blurry	84.66	74	85.23	89	85.41	85.41	85.41	86.41	86.42
Brown_Hair	62.02	77	80.85	81	79.22	79.22	79.22	81.12	80.93
Bushy_Eyebrows	53.58	82	84.97	80	80.73	82.41	82.41	84.42	84.26
Chubby	64.31	73	76.86	75	74.13	75.19	75.19	76.13	76.06
Double_Chin	65.58	78	81.52	78	77.82	79.19	79.19	80.76	80.49
Eyeglasses	80.23	95	91.30	92	89.69	90.76	90.76	91.72	91.50
Goatee	77.41	78	82.97	86	81.72	81.72	81.72	83.30	83.01
Gray_Hair	83.94	84	88.93	88	87.94	87.94	87.94	88.37	88.46
Heavy_Makeup	87.21	95	95.85	95	94.80	94.80	94.80	95.38	95.39
High_Cheekbones	63.34	88	88.38	89	86.53	86.53	86.53	88.34	88.34
Male	76.02	94	94.02	93	92.17	92.17	92.17	92.81	92.60
Mouth_Slightly_Open	57.02	82	83.51	86	79.03	79.03	79.03	82.70	82.50
Mustache	89.03	92	93.43	95	91.92	91.92	91.92	93.27	92.97
Narrow_Eyes	63.45	81	82.86	82	78.94	80.07	80.07	82.86	82.75
No_Beard	73.08	79	82.15	81	79.27	79.27	79.27	80.65	80.77
Oval_Face	52.37	74	77.39	75	74.19	74.19	74.19	76.51	76.80
Pale_Skin	50.82	84	93.32	91	88.36	90.16	90.16	91.00	90.97
Pointy_Nose	68.4	80	84.14	84	81.50	82.92	82.92	83.63	84.20
Receding_Hairline	56.36	85	86.25	85	83.91	83.91	83.91	85.09	84.90
Rosy_Cheeks	81.46	78	87.92	86	85.55	85.55	85.55	87.19	87.08
Sideburns	69.38	77	83.13	80	79.42	79.42	79.42	81.89	81.76
Smiling	56.65	91	91.83	92	88.65	88.65	88.65	90.77	90.80
Straight_Hair	60.1	76	78.53	79	77.09	78.10	78.10	79.27	78.91
Wavy_Hair	57.94	76	81.61	80	77.02	77.02	77.02	78.55	78.28
Wearing_Earrings	85.1	94	94.95	94	94.20	94.20	94.20	94.59	94.75
Wearing_Hat	86.57	88	90.07	92	89.81	90.23	90.23	90.25	90.23
Wearing_Lipstick	83.22	95	95.04	93	93.71	93.71	93.71	94.07	94.07
Wearing_Necklace	78.54	88	89.94	91	88.71	88.71	88.71	89.45	89.59
Wearing_Necktie	63.13	79	80.66	81	79.55	79.55	79.55	81.70	81.40
Young	78.59	86	85.84	87	83.90	83.90	83.90	85.55	85.68
Mean Accuracy	**71.27**	**83.85**	**86.31**	**86.15**	**84.02**	**84.36**	**84.36**	**85.85**	**85.82**

Table 5.4: Cross dataset results. The three numbers for each Train-Test pair are for AFFACT [1], DMTL [2] and the Proposed method, respectively, from left to right.

Train/Test	**CelebA**			**LFWA**		
CelebA	89.07/90.32	92.6	90.39/87.14	79.5/73.84	73	79.32/74.56
LFWA	-/-	70.2	78.15/77.88	-/-	86	85.99/85.28

second accuracy is obtained by using the mid value of the score range (0 for AFFACT which gives scores between -1 and $+1$, and 0.5 for proposed method for which score is between 0 and 1) as the threshold. Higher accuracies are obtained by using optimal thresholds for the proposed method, while for AFFACT the accuracies dropped slightly. It can be seen from this table that, for cross-dataset testing (trained on CelebA and tested on LFWA or vice versa), the proposed method outperforms both AFFACT and DMTL with a relatively large margin. This again proves the generalization capability of SPLITFACE, which is achieved by the combination of its unique architecture with the committee machine.

Next, we evaluated the performances SPLITFACE on the modified CelebA and LFWA partial face datasets. The results for evaluation on same and cross-dataset are presented in tables 5.5 and 5.6. In table 5.5, results are presented form AFFACT and SPLITFACE network, both trained on the original CelebA training set and tested on the original and modified CelebA and LFWA datasets. The results for both using and not using optimal threshold (using 0 for AFFACT and 0.5 for SPLITFACE as threshold instead) are shown in the table. It can be seen that SPLITFACE, especially NSA with product rule and optimal threshold outperforms AFFACT in terms of accuracy for full face dataset, and both cross-domain and partial datasets. The differences are more prominent when using the optimal

Table 5.5: Networks trained on CelebA and tested on both full and partial CelebA and LFWA datasets.

Method		CelebA	C-U12	C-U34	C-L12	C-L34	C-R12	C-R34	LFWA	L-U12	L-U34	L-L12	L-L34	L-R12	L-R34
	AFFACT	**90.32**	77.98	81.86	80.56	84.93	80.18	85.07	73.84	**68.83**	71.12	69.01	73	69.21	**73.44**
Witout	**Full**	86.76	80.99	84.34	83.86	85.39	83.45	85.71	73.52	67.28	70.25	69.94	72.69	70.01	72.67
Optiamal	**HRP**	86.93	81.46	84.51	84.23	85.97	84.3	86.29	73.54	67.51	70.51	70.08	72.13	70.38	72.64
Threshold	**NSA Prod rule**	87.14	**83.24**	**85.53**	**84.6**	**86.79**	**84.84**	**86.5**	**74.56**	68.61	71.45	70.34	73.2	**70.35**	73.32
	NSA Med rule	87.07	83.22	85.47	84.51	86.75	84.76	86.44	74.3	68.49	**71.59**	**70.36**	**73.24**	70.27	73.1
	AFFACT	89.07	83.05	85.60	84.98	87.47	85.33	87.69	79.5	74.77	77.55	74.97	78.34	74.85	78.19
With	**Full**	90.72	83.99	87.14	87.19	89.33	87.46	89.63	72.32	66.96	69.37	68.22	71.1	68.72	71.25
Optimal	**HRP**	**90.8**	84.27	87.59	87.11	89.31	87.82	89.71	72.67	67.28	69.94	67.94	70.91	68.42	71.35
Threshold	**NSA Prod rule**	90.39	85.3	**88.08**	**88.12**	**89.87**	**88.42**	**90.02**	79.32	**75.76**	77.81	**76.7**	**78.77**	**76.51**	**78.57**
	NSA Med rule	90.61	**85.47**	88.16	87.59	89.76	88.1	90.01	**79.86**	75.28	**78.34**	76.1	78.73	75.74	78.49

Table 5.6: Networks trained on LFWA and tested on both full and partial CelebA and LFWA datasets.

Method		CelebA	C-U12	C-U34	C-L12	C-L34	C-R12	C-R34	LFWA	L-U12	L-U34	L-L12	L-L34	L-R12	L-R34
Without	**Full**	75.87	64.84	71.67	70.93	73.67	70.77	74	83.52	67.11	74.76	75.08	79.74	75.8	80.24
Optimal	**HRP**	75.94	65.03	71.94	70.78	73.64	70.58	74.02	83.84	67.59	75.36	74.93	79.81	75.7	80.45
Threshold	**NSA Prod rule**	77.88	**67.85**	**75.63**	**71.39**	**75.95**	**71.23**	**75.82**	**85.28**	**70.47**	**79.78**	**75.74**	**81.62**	**76.35**	**82.52**
	NSA Med rule	**78.22**	67.72	75.55	71.3	75.84	71.16	75.72	85.18	70.3	79.64	75.62	81.51	76.25	82.43
With	**Full**	76.37	65.46	71.74	70.97	74	70.83	74.35	84.02	69.38	76.05	76.09	80.51	76.74	80.86
Optimal	**HRP**	76.58	66	72.18	71.22	74.21	71.16	74.61	84.36	70.13	76.66	76.42	80.85	77.06	81.28
Threshold	**NSA Prod rule**	**78.15**	**69.54**	**76.14**	**72.3**	**76.68**	**72.22**	**76.63**	**85.99**	**73.26**	**81.4**	**77.4**	**82.84**	**78.32**	**83.46**
	NSA Med rule	78.13	68.13	75.61	71.75	75.92	71.79	76.03	85.82	72.12	80.79	76.75	82.22	77.41	83.06

thresholds, which show that threshold normalization step with a piece-wise linear function can boost the overall performance. Similar scenario is found in table 5.6, where SPLITFACE is trained on the original LFWA training set and tested on both original and partial CelebA and LFWA datasets. Since no pre-trained version of AFFACT on LFWA is publicly available, the results for AFFACT could not be provided in this table. Note that in both tables 5.5 and 5.6, the committee machine approaches improves over the full face branches, especially for partial face datasets. This improvement can be attributed to the unique architecture of SPLITFACE that harnesses local information from unoccluded facial segments and to the ensemble aggregation approach by using the committee machine.

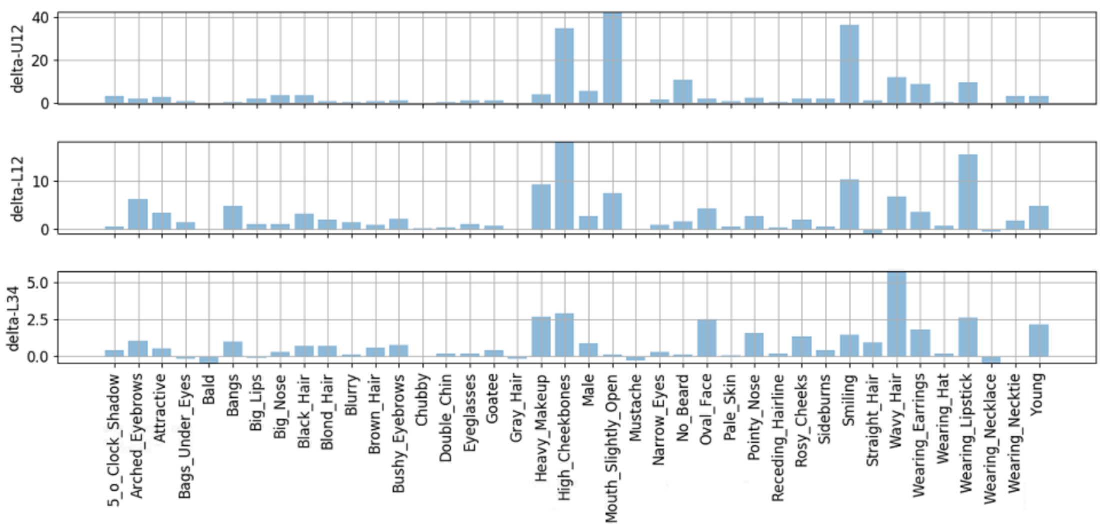


Figure 5.5: Attribute-wise comparison of performance changes (w.r.to the performance on the unoccluded faces in CelebA) on the C-U12, C-L12 and C-L34 modified datasets. The vector of differences are denoted as delta-U12, delta-L12 and delta-L34 respectively.

5.2.6 Analysis of Performance Degradation with Occlusion

One obvious observation from tables 5.5 and 5.6 is that the attribute detection accuracies decrease with increasing occlusion. For example, all the methods achieve higher accuracies for upper three-fourth faces present in C-U34 and L-U34 in comparison to C-U12 and L-U12, respectively. In this section, we explore the effect of occlusion on the accuracy of each attribute using Fig 5.5 which plots the decrease in accuracy of SPLITFACE (after stage 1 before output pruning) for the partial CelebA datasets, C$-P, P \in \{$U12, L12, L34$\}$ described in section 5.2.1 with respect to full face accuracy. The differences are denoted as delta-U12, delta-L12 and delta-L34, respectively. We observe that SPLITFACE fails in C-L12 and C-L34 for the same attributes such as wavy hair, high cheekbones and wearing lipstick, since part of the right side of the faces are occluded in both cases. On the other hand, C-U12 has reduced performance for attributes like ‘mouth slightly open’, ‘no beard’ and ‘smiling’, which are attributes localized in the lower part of the face, which is not visible in C-U12. So, SPLITFACE avoids catastrophic failures during occlusion, since the prediction accuracy of other attributes remain near constant and only invisible localized attributes’ performance degrade. The output pruning step of SPLITFACE removes the attributes for which a segment performs badly in the first stage. When trained, SPLITFACE utilizes information from different segments to bolster its decision about an attribute as well as fill up the gaps in attributes in one segment by using information from other segments which predict those missing attributes.

5.2.7 Performance for Partial Face Augmentation

We also trained SPLITFACE with training samples from the modified partial face datasets. When training, samples from the modified datasets were picked with a 0.3 probability in each batch while the rest of the samples came from the original datasets. The performances of the networks trained in this manner are presented in tables 5.7 and 5.8. In comparison to tables 5.5 and 5.6, we can see that for both partially modified CelebA and LFWA datasets, the performances improved greatly when the partial faces are augment the training samples in addition to segment dropout.

In this chapter, we introduced SPLITFACE, an algorithm for facial attribute extraction utilizing multiple facial segments, a unique deep convolutional network,

Table 5.7: Performance of SPLITFACE trained on original and modified CelebA (70-30 ratio).

Methods	**CelebA**	**C-U12**	**C-U34**	**C-L12**	**C-L34**	**C-R12**	**C-R34**
Full	90.42	88.01	89.7	89.56	90.02	89.54	90.07
HRP	90.18	87.94	89.55	89.23	89.77	89.21	89.71
NSA Prod rule	90.39	88.43	89.77	89.76	90.02	**89.86**	90.11
NSA Med rule	**90.52**	**88.46**	**90.05**	**89.85**	**90.16**	89.85	**90.21**

Table 5.8: Performance of SPLITFACE trained on original and modified LFWA (70-30 ratio).

Methods	**LFWA**	**L-U12**	**L-U34**	**L-L12**	**L-L34**	**L-R12**	**L-R34**
Full	83.93	80.31	83.01	81.6	83	81.93	83.37
HRP	85.46	**82.39**	84.45	**83.36**	84.59	**83.56**	84.79
NSA Prod rule	**86.04**	82.06	84.87	83.07	**84.97**	83.43	85.14
NSA Med rule	85.97	82.17	**84.88**	83.06	84.87	83.4	**85.23**

and a committee machine approach for ensemble aggregation. Through extensive experimentation, we have shown that the proposed method outperforms state-of-the-art facial attribute extraction methods for partially visible faces. Also, utilizing a committee machine approach, SPLITFACE achieved better generalization and superior performance across domains. Moreover, when trained with both segment dropout and partial face data, the network achieved even higher attribute detection accuracy for partially visible faces. As for future research, since the segments are heavily overlapping and therefore can assist each other greatly, similar performance might be achievable with smaller input images. Finally, it would be interesting to see if a cross-stitch network [142] can improve performance when connected to the segment network branches at certain intervals, by allowing the segment networks to share data.

Chapter 6: User Authentication Using Location Trace History

6.1 Person Authentication using Trace Histories (PATH)

The location service of smartphones returns geographical location of the user based on GPS and WiFi network. In addition to the latitude and longitude information, the exact day, time and duration of being in a proximity can be extracted from the location service data (Fig. 6.1) which are very useful for modeling the pattern of a user's location trace. We define the problem of location-based authentication as Person Authentication using Trace Histories (PATH). In general, the PATH problem has three challenges:

1. Clustering of Geo-location points to form observation states taking into account the temporal information.

2. Handling unforeseen observation states and learning a model for each user using sequential patterns inferred from the observation states.

3. Generating verification score from a test sequence using the trained model.

In Fig. 6.2, a schematic of the proposed verification system is shown. Basically, there are three steps. First, from the training geo-location data of user x, location clusters are formed and the cluster centers and radius are extracted. Then, from

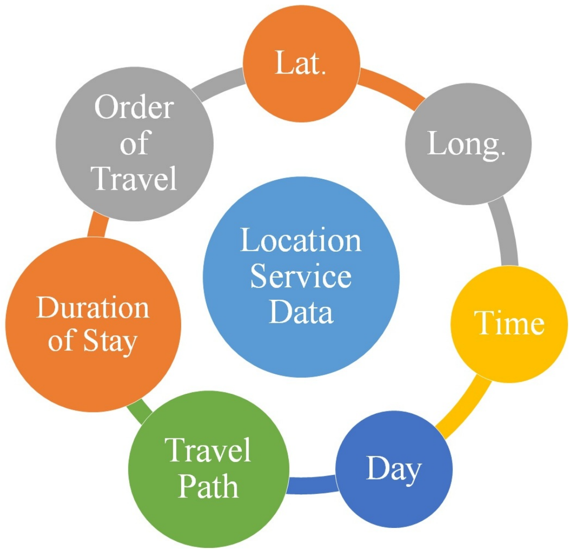


Figure 6.1: Useful information obtained from the location service of the smartphone.

the sequence of geo-location data with time-stamps and session information, the sequence of training observations is obtained which is used to train the verification model for user x. Finally, from every n observations in the test data, scores are generated to verify user x using the previously trained model.

6.1.1 Geo-location Points to Observation States

Since, this is a verification problem as opposed to recognition, only the information about the legitimate user is available during training. For a user, the data collected by the location service is a sequence of geo-location points $P = \{p_1, p_2, \ldots, p_n\}$, where, each point $p_i \in P$ contains the longitude (p_i^{Long}), latitude (p_i^{Lat}) and time stamp (p_i^T). These points are sampled at variable rates based on the speed of movement and therefore the points might not be equally spaced in time. A location trace can be formed by connecting the Geo-location points according to their time series as shown with the red points connected with arrows in Fig. 6.3.

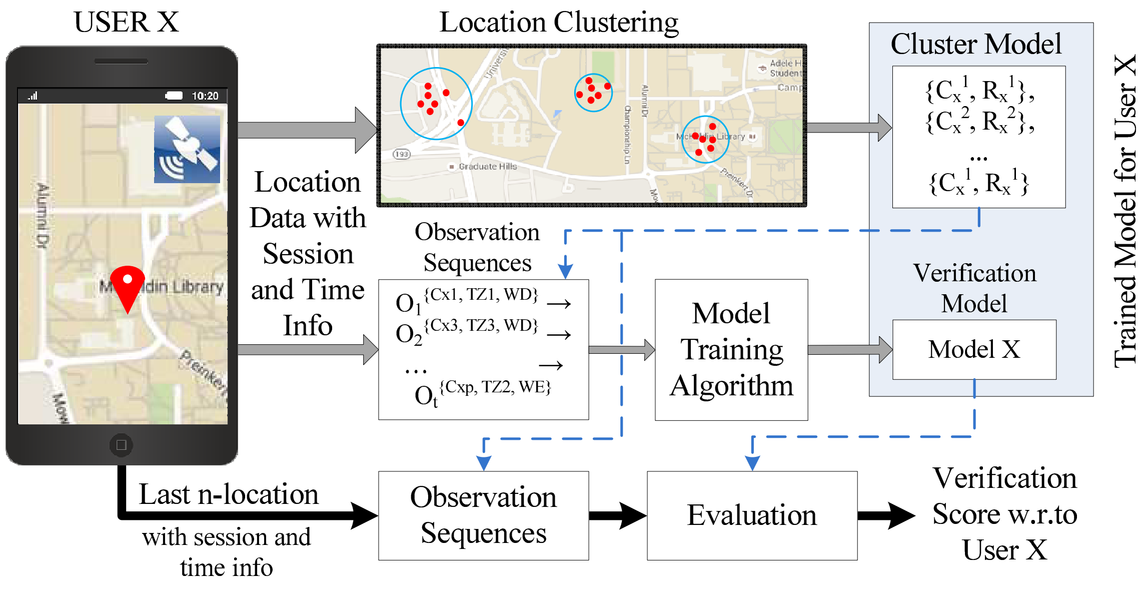


Figure 6.2: System overview for handling the PATH problem.

In order to take into account the duration of stay in a certain locality, the geo-location traces are sampled once every three minutes. Thus, if a user is in the same location for a while, the location logger will log the same location at three minute interval. The historical Geo-location points of the user obtained this way are clustered into $\eta = 1, \ldots, N$ clusters, namely $C^1 \ldots C^N$, using the DBSCAN algorithm [125] based on geographical distances (GeoDist) between data points. The maximum distance between a point from the center of the cluster in which that point belongs is set to be below a certain value R_{max} meters. A cluster C^j is completely defined by $\{C^j, R^j\}$, where $C^j : \{c^j_{Lat}, c^j_{Long}\}$, $j \in \eta$, consists of the latitude and longitude of the cluster center, respectively, and R^j is the radius of the cluster where $R^j \leq R_{max}$. An example of such clustering is shown in Fig. 6.3, where the cluster C^1_i represents presence in a residential area and C^2_i in an office building in an university suggesting plausible regions like home, university etc. that the user i would visit.

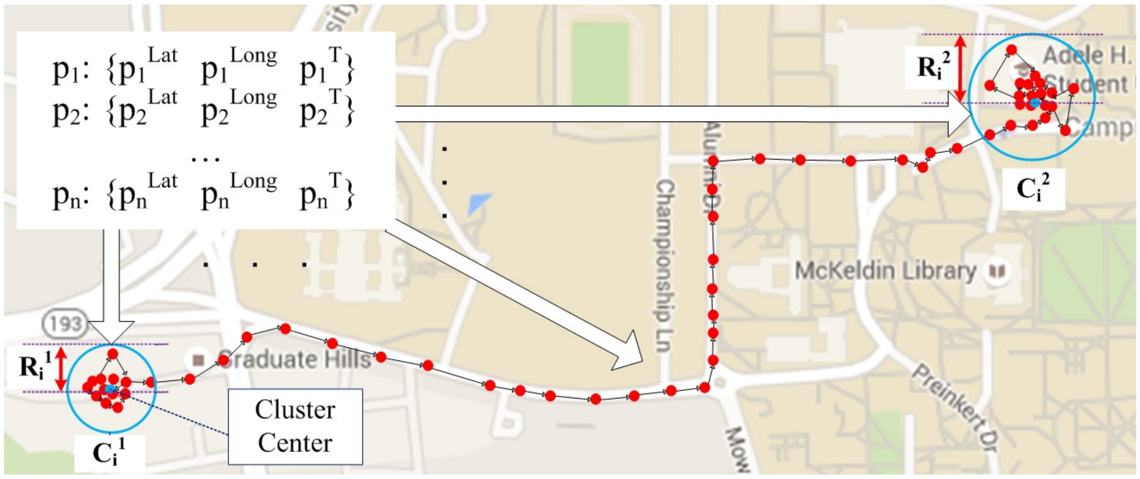


Figure 6.3: Geo-location Points and Clusters.

Two types of additional clusters, Transit (Tr) and Unknown (Unk), are assigned for each user. If the user is traveling, causing location information to change rapidly ($\geq 2ms^{-1}$), then those geo-locations points are assigned to Tr. For each of the other data points p_m that are not inside any location cluster and also not in the Transit cluster, the nearest known location cluster to each of those points within a radius of $M = 10000$ meters is determined by calculating $GeoDist(j,m)$, the geological distance between p_m and all C^js. Then those data points are assigned to the cluster Unk_ℓ, where,

$$\ell = \begin{cases} \arg\min_j GeoDist(j,m), & \text{if } GeoDist(j,m) \leq M \\ \\ \infty, & \text{otherwise} \end{cases} \tag{6.1}$$

for $m = 1, \ldots, n$, $p_m \notin C^j$, $j \in \eta$, $p_m \notin Tr$.

So, to summarize, there would be N location clusters C^j and N corresponding nearby unknown clusters Unk_j for $j \in \eta$, one more cluster denoted as Unk_{inf} and the transit cluster Tr totaling the number of cluster to be $2N + 2$. Data points at each cluster are assigned to six different observations based on day and time information. Weekdays and weekend data points are flagged with WD and WE. Also, the whole day is divided into three time zones (TZs) - TZ1 (12:01 am to 8:00 am), TZ2 (8:01 am to 4:00 pm) and TZ3 (4:01 pm to 12:00 pm). Thus, there are $(2N + 2) \times 2 \times 3$ possible observation states for the locations, transition and unknowns. One additional observation state, namely, $Null$, is considered in order to take the sparsity of the data into account. The $Null$ is inserted at the end of each day. It signifies the unavailability of location, time zone, number of observation samples and observation states in between consecutive sessions occurring in two different days.

Now, it is possible that many of the observation states are not present in the training data and yet, they may appear in the test data because of the following reasons

1. The location service of the phone might only collect data when the phone is turned on and in use. Also, some user prefer to turn off the location service when the battery is low.

2. Some unknown states near known locations might not occur during the training phase.

3. Data for all time zones and days might not be present for all locations.

6.1.2 Handling Unforeseen Observations to Learn User Models

In order to verify the user, it is imperative to take the unforeseen observations into account rather them assigning zero emission probability to them. Fortunately, the probability distribution of the occurrence of all the states can be smoothed out using the estimated values from the training data. Laplace smoothing is one easy choice to make sure that the probability does not go to zero at all, however, it does not take into account the prior information about available states. For example, assume that the observation $C^1 - TZ1 - WE$ is not present in the training set, i.e. the historical location log of the user does not contain any information of the user being at C^1 during timezone 1 on weekends. The prior probability of its occurrence $P(C^1 - TZ1 - WE)$ can be still be approximated from the probability of the user being in C^1 during $TZ1$ and the probability of the user being in C^1 during weekends by assuming that the two events are independent. In the next section, the integration of this assumption into the proposed HMM training model is elaborated.

From the temporally sorted training observation sequence, three different approaches are presented for user verification. The approaches are:

1. Simple time-sequence matching
2. Markov Chain Models
3. MSHMM models - a Hidden Markov Model with the proposed marginal smoothing

These approaches are discussed in details in the next section.

6.1.3 User Verification Methods

Three different user verification methods are discussed here. After the pre-processing step, the observations are available as a time series for each user and user-wise models are generated using this data.

6.1.3.1 Sequence Matching (SM) Method

The sequence matching algorithm is presented in 1. The algorithm computes match ratio between the training sequence vector of user i with the last n sequence values of a user j. The bigger the match ratio, the more likely it is that the users are same. While computing the sequence matches, the temporal order of the sequences are taken into account.

Algorithm 1 Sequence Matching Algorithm

```
procedure SeqMatching($S^i_{tr}, S^j_{te}$)  ▷ Training Sequence Vector of user $i$ ($S^i_{tr}$), n-last Test Sequence Vector of user $j$ ($S^j_{te}$)
    $S_c \leftarrow 0$                                        ▷ Sequence Counter
    $S_t \leftarrow 1$                                        ▷ Sequence Track Variable
    for $v_{tr} \in S^i_{tr}$ do
        if $v_{tr} == S^j_{te}[S_t]$ then
            $S_t \leftarrow S_c + 1$                          ▷ Element Matched
            if $St == |S^j_{te}|$ then
                $S_t \leftarrow 0$
                $S_c \leftarrow S_c + 1$
            end if
        end if
    end for
    $r \leftarrow (S_c + 1.0)|S^j_{te}| + \frac{S_t}{|S^i_{tr}|+|S^j_{te}|)}$
    return $r$                                                ▷ The match ratio is $r$
end procedure
```

6.1.3.2 Markov Chain (MC)-Based Verification

For Markov Chain-based verification, the probability of moving to a state depends only on the last visited state and the transition matrix for all probable states. The model X_n is a Markov chain for observation sequences of length n which is composed of a set of k-observation states $S = s_1, s_2, \ldots, s_k$, prior probability $\rho_i = Prob\{X_0 = i\}$ of entering state i, and a set of transitions $t_{i,j}$ where

$$t_{i,j} = Prob(X_n = s_j | X_{n-1} = s_i). \tag{6.2}$$

Given the prior and transition probabilities of the training data, the total probability of traversing any sequence of n consecutive observations $i_0, \ldots, i_n \in S$ is calculated as

$$Prob(X_0 = i_0, \ldots, X_n = i_n) = \rho_{i_0} t_{i_0,i_1} \ldots t_{i_{n-1},i_n} \tag{6.3}$$

For unforeseen states, Laplace smoothing is enforced by setting the prior probabilities of and transition probabilities from those states to a tiny value δ.

6.1.3.3 MSHMM Model for PATH

The proposed Marginally Smoothed Hidden Markov Models (MSHMMs) are specifically trained to handle unforeseen observations. HMM models are assumed to be generated by a Markov process with unobserved hidden states and are very effective for analyzing sequential data. The HMM model can be expressed as $\lambda = (\pi, A, B)$ where the parameters can be learned from the observed locations of the training set O given the vocabulary of possible observations V. Here, π is the initial hidden-state distribution, A is a time-dependent stochastic transition matrix between the hidden states, and B is a stochastic matrix with the probability of emitting a particular observation at a given state.

Algorithm 2 Modified Forward-Backward HMM Algorithm for training MSHMM model for PATH

```
procedure BaumWelchForPATH(training observations $O = \{o_1, o_2, \dots, o_T\}$ of length $T$, vocabulary of observations $V = \{v_1, v_2, \dots, v_M\}$ where $M \geq T$)
    initialization
    random initialization of $\pi$, $A$ and $B$. $\delta$.
    while until convergence do
        $\alpha_i(1) = \pi_i b_i(o_1)$
        $\alpha_i(t+1) = b_i(o_{t+1}) \sum_{j=1}^{N} \alpha_j(t) a_{ji} \forall t, i$
        $\beta_i(T) = 1$
        $\beta_i(t) = \sum_{j=1}^{N} \beta_j(t+1) a_{ij} b_j(o_{t+1}) \forall t, i$
        E-step
        $\gamma_i(t) = \frac{\alpha_i(t)\beta_i(t)}{\sum_{j=1}^{N} \alpha_j(t)\beta_j(t)} \forall t, i$
        $\xi_{i,j}(t) = \frac{\alpha_i(t) a_{ij} b_j(o_{t+1}) \beta_j(t+1)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i(t) a_{ij} b_j(o_{t+1}) \beta_j(t+1)} \forall t, i, j$
        Modified M-step
        $\widehat{\pi_i} = \gamma_i(1) \forall i$
        $\widehat{a_{ij}} = \frac{\sum_{t=1}^{T-1} \xi_{i,j}(t)}{\sum_{t=1}^{T-1} \gamma_i(t)} \forall t, i, j$
        if $\sum_{t=1}^{T} \mathbf{1}(O_t = v_k) \neq 0$ then
            $\widehat{b_i}(v_k) = \frac{\sum_{t=1}^{T} \mathbf{1}(O_t = v_k)\gamma_i(t) + \delta}{\sum_{t=1}^{T} \gamma_i(t) + T\delta} \forall t, i, j$
        else
            $\widehat{b_i}(v_k) = \frac{\sum_{t=1}^{T} \mathbf{1}(O_t^{L,TZ} = v_k^{L,TZ})\gamma_t(j) + \delta}{\sum_{t=1}^{T} \gamma_t(j) + T\delta} \times \frac{\sum_{t=1}^{T} \mathbf{1}(O_t^{L,W} = v_k^{L,W})\gamma_t(j) + \delta}{\sum_{t=1}^{T} \gamma_t(j) + T\delta} \forall t, i, j$
        end if
        Normalize and Update
        update $\pi = normalize(\widehat{\pi})$
        update $A = normalize(\widehat{A})$
        update $B = normalize(\widehat{B})$
    end while
    return $\pi$,$A$, $B$
end procedure
```

To learn the model, the HMMs are trained using the three-step Baum-Welch algorithm [143] shown in Algorithm 2. The initialization step sets $\lambda = (\pi, A, B)$ with random initial conditions. The parameters are then updated iteratively until convergence. In Algorithm 2, $\alpha_i(t) = Prob(y(1) = o_1, \ldots, y(t) = o_t, X(t) = i|\lambda)$ is the probability of seeing the partial observable sequence $o_1, \ldots, o_t$ and ending up in state i at time t and it is calculated recursively. N is the number of hidden states, a_{ij} refers to the j-th element of the i-row of the A and $b_j(o)$ refers to the emission probabilities of the j-th state in B for observation o.

The update equation of $\widehat{b_i}$ in the M-step is modified from the original Baum-Welch algorithm to assign non-zero emission probabilities to those observations of the vocabulary V that are not present in the training set O. As mentioned in the formulation of the PATH problem, an observation consists of the location, timezone and weekday/weekend information, i.e. $o_i = \{o_i^{L,TZ,WE}\} \forall i \in \{1, \ldots, T\}$. Originally the summation in the nominator of $\widehat{b_i}$ is only made over observed symbols equal to o_k, i.e. the indicator function $\mathbf{1}(O_t = v_k) = 1$ if $o_t = o_k$, and zero otherwise. But, this equation assigns zero emission probabilities for unforeseen observations, and will eventually pull the overall probability of observing a sequence to zero even if only one such unforeseen yet probable observation is present in the test sequence. One simple work-around is to use a smoothing technique such as Laplace Smoothing where $\widehat{b_i}$ is assigned a very small constant probability. However, the performance of Laplace smoothing is found to be very poor experimentally because of its empirical nature. A marginal smoothing method is proposed here which utilizes the location, timezone and weekend/weekday information of the observations to assign the emission probabilities. The method is based on the assumption that the probability of a user being in a location cluster at a certain timezone $P(v_k^L, v_k^{TZ})$ is independent of the probability of the individual being in a location on weekdays/weekends $P(v_k^L, v_k^W)$. Then, $\widehat{b_i}(v_k)$ can be expressed as

$$\begin{aligned}
\widehat{b_i}(v_k) &= P(O_t = v_k \mid X_t = i) \\
&= P(O_t^L = v_k^L, O_t^{TZ} = v_k^{TZ}, O_t^W = v_k^W \mid X_t = i) \\
&\approx P(O_t^L = v_k^L, O_t^{TZ} = v_k^{TZ} \mid X_t = i) \\
&\quad \times P(O_t^L = v_k^L, O_t^W = v_k^W \mid X_t = i) \qquad (6.4) \\
&\approx \frac{\sum_{t=1}^T \mathbf{1}(O_t^{L,TZ} = v_k^{L,TZ})\gamma_t(j)}{\sum_{t=1}^T \gamma_t(j)} \times \\
&\quad \frac{\sum_{t=1}^T \mathbf{1}(O_t^{L,W} = v_k^{L,W})\gamma_t(j)}{\sum_{t=1}^T \gamma_t(j)} \qquad (6.5)
\end{aligned}$$

where, the indicator function $\mathbf{1}(O_t^{L,TZ} = v_k^{L,TZ}) = 1$ if the location and timezone of o_t and v_k are the same irrespective of the day and zero otherwise, and $\mathbf{1}(O_t^{L,W} = v_k^{L,W}) = 1$ only if the location and day are the same irrespective of the timezone and zero otherwise.

If Laplace-smoothing is used instead of the marginal smoothing proposed here, then, when $\sum_{t=1}^T \mathbf{1}(O_t = v_k) == 0$, the update equation of $\widehat{b_i}(v_k)$ would be

$$\widehat{b_i}(v_k) = \frac{\delta}{\sum_{t=1}^T \gamma_i(t) + T\delta} \qquad (6.6)$$

for all t, i. Here, δ is a very small number, and therefore, a even smaller emission probability is being assigned to an unforeseen observation instead of 0.

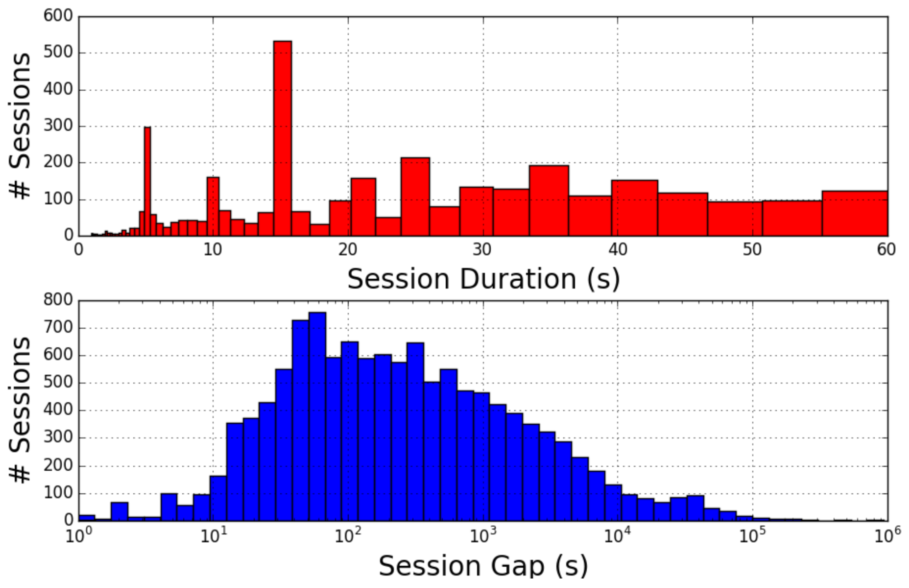


Figure 6.4: For UMDAA02 dataset: (Top) Histogram of duration of sessions and (Bottom) Histogram of time gap between consecutive sessions.

6.1.4 Experimental Setup and Evaluation

Experiments are performed on two datasets: (1) the UMDAA02 geo-location dataset [31], and (2) the Geolife GPS tarjectory dataset [114]. The histogram of duration of geo-data collection sessions along with the histogram of time gap between consecutive sessions for the two datasets are shown in Figs. 6.4 and 6.5. It can be seen from these figures that for the UMDAA02 geo-location dataset, geo-location data is collected for at most 60 seconds after the user logs into the phone. On the other hand, for the Geolife dataset the data is seamlessly collected for long period of times. In fact, the session with the maximum duration is almost 11 days long. The reason behind this difference is that the UMDAA02 dataset is collected for authentication research using smartphones when the phone is being used, whereas, the GeoLife dataset is collected using GPS-phones and GPS loggers for individual and social behavioral research. Since the event of logging into a phone varies widely,

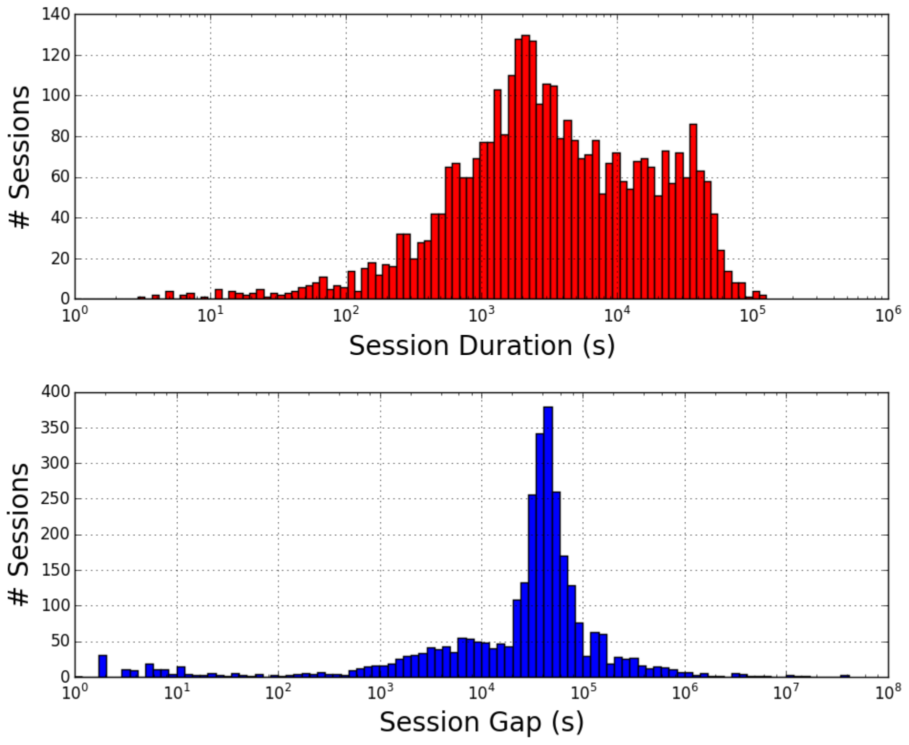
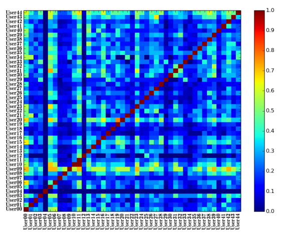


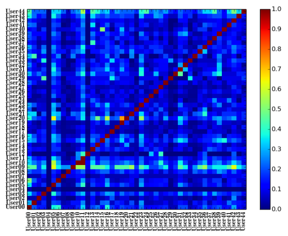
Figure 6.5: For Geo-Life data set: (Top) Histogram of duration of sessions and (Bottom) Histogram of time gap between consecutive sessions.

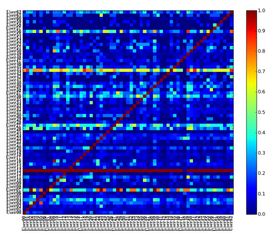
the session gap for the UMDAA02 dataset is spread more widely then the GeoLife dataset.

Considering the nature of the data collection process, experiments for the two methods are designed differently. Since the UMDAA02 dataset is small, sparse and contains user data for a little over a week for each user, the first 70% of chronologically sorted data of each user is used for training the model for that user and the rest are used for evaluation. On the other hand, for the GeoLife dataset, experiments are done on a total of 63 users who has geo-location data for 6 weeks or more. Location data for the 6-th week is used for evaluation, while those from the previous weeks are used for training user-wise models.

In Fig. 6.6(a) the distinguishability of location information of the users is


(a)


(b)


(c)

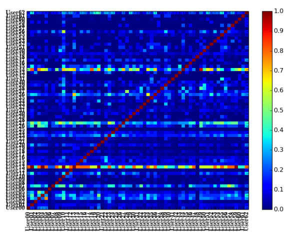

(d)

Figure 6.6: Similarity matrix depicting (a) location overlap for UMDAA02 dataset, (b) observations overlap for UMDAA02 dataset, (c) location overlap for the GeoLife dataset, and, (d) observations overlap for the GeoLife dataset.

depicted as similarity matrices for the UMDAA02 dataset. For a user, after determining the location clusters (considering $R_{max} = 20$ meters), the location traces are obtained for the training period of that user and all the other users considering those clusters. In figure 6.6(a), the percentage of common location clusters that any two users share is shown as a similarity matrix. It can be seen that in many cases two different user can have significant amount of overlaps. However, when time and day information are incorporated with the location data to generate the observations, the number of overlaps gets reduced, as can be seen from Fig. 6.6(b). Intuitively, considering the sequence information would minimize the similarity between two different users even more. Similar depictions for the GeoLife trajectory dataset are shown in Fig. 6.6(c) and 6.6(d), respectively.

The number of past observations n that is considered when evaluating the verification score has a direct impact on the EER. It can be seen from Fig. 6.7, that the EER decreases with increasing n, which is understandable since having greater number of past observations improves the predictability of the next observation. On the other hand, the EER gets larger if the value of R is too small or too big. The

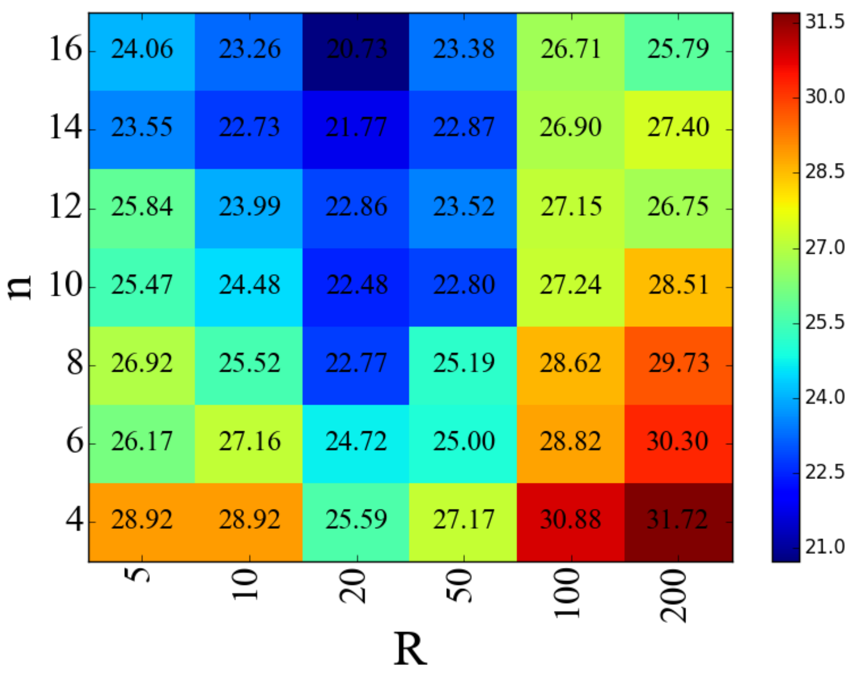


Figure 6.7: EER(%) heatmap for length of sequence (n) vs. the cluster maximum radius (R).

optimum value of R for the UMDAA02 dataset is found to be around 20 meters, as can be seen in the figure. This is a reasonable value since clusters of this diameter are neither too small to be rooms nor too big to be communities. Rather, they are most likely to be representative of buildings and for the PATH problem and other location-based prediction tasks, generally buildings like home, shopping mall, work-place, gym etc. are considered to translate the geo-location observation.

In Fig. 6.8, the performance of the four methods - SM, MSHMM, MC and HMM-lap are shown in terms of EER for varying n on the UMDAA02 dataset for $R_{max} = 20$. Given the unconstrained nature of the dataset, verification is a daunting task even with more robust modalities such as face [31]. Yet, the proposed MSHMM method achieved an EER of 20.73% for $n = 16$ outperforming all the other methods. In fact, for any value of n, the MSHMM models trained with 10 hidden

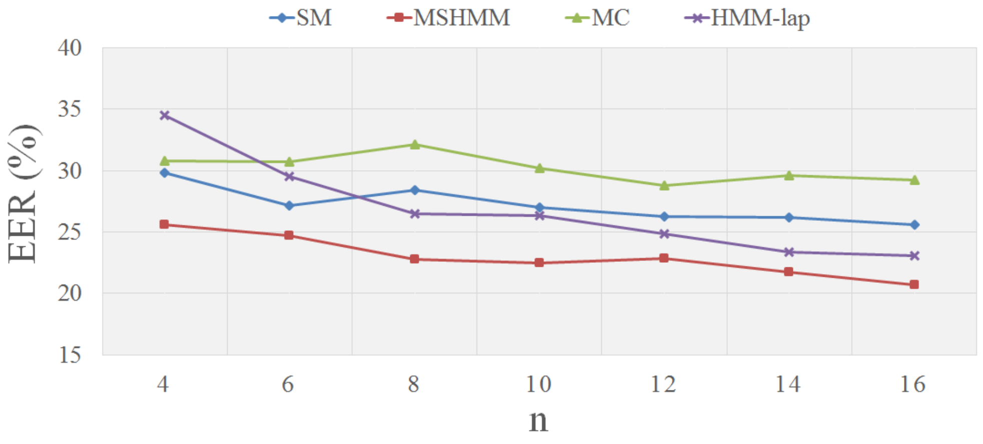


Figure 6.8: Comparison of SM, MC, MSHMM and HMM-lap methods in terms of EER (%) for the UMDAA02 dataset. For HMM-based methods the number of hidden states is 10.

states performs better then other methods.

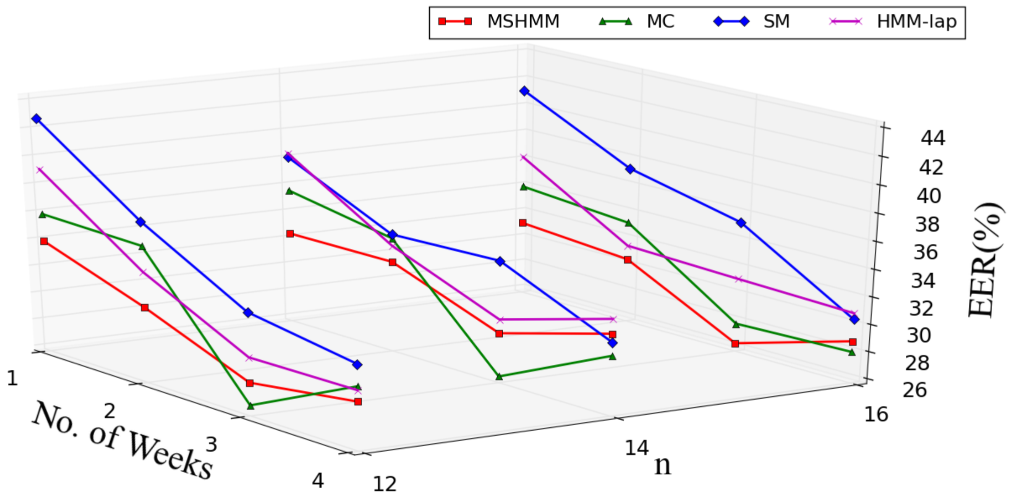


Figure 6.9: Comparison of SM, MC, MSHMM and HMM-lap mehthods in terms of EER (%) for varying n and varying number of weeks of training data. For HMM-based methods the number of hidden states is 10.

Finally, in Fig. 6.9, the performances of the four methods are compared for different n. while the number of past weeks for training is increased from 1 to 4. Understandably, the EER is showing a decreasing trend in general for increasing training data. The overall performance of the proposed MSHMM method (trained with 10 hidden states) is better than the other three methods across different training size and sequence length. The MC method also found to achieve better accuracy on this dataset. This is probably due to the fact that this dataset is less sparse and therefore there are fewer unforeseen observations which led to better estimation of the prior and transition probability matrix for MC.

Chapter 7: User Authentication Using Application-usage Information

In this chapter, the problem of application usage-based user authentication is presented in detail along with associated challenges and possible solutions. The impact of unknown application and unforeseen events on the authentication task is investigated and several methods for handling the active authentication problem effectively are described. Finally, a detailed analysis on the application usage data, experimental results and discussions are presented.

7.0.1 Active/Continuous Authentication of Smartphones

Active, continuous or implicit authentication are different terminologies for the same authentication approach in which the rightful user of mobile devices is authenticated throughout the entire session of usage [144] [11] [31]. In recent years, active authentication research has gained a lot of attention because of the increased security risks and complexity of password, token-based, multi-factor and other explicit authentication systems [11]. In active authentication, the wide range of sensor data available on the mobile devices are utilized to learn one or more templates for the legitimate user during a training session. The templates are used in the background to continuously authenticate the user during regular usage and based on the amount of deviation from the templates the device itself starts restricting access to phone applications and utilities starting from the most sensitive ones [31]. Most popular modalities for active authentication are front camera face images [145] [146] [13], touch screen gesture data [19] [14] [147], accelerometer and gyroscope data [103] [61] [148], location data [3] etc. Suitability of different behavioral biometric such as touch and keystroke dynamics, phone pick-up patterns, gait dynamic, and patterns from location trace history have been explored for active authentication [149] [150] [3]. Combinations of multiple biometric have been demonstrated to produce robust authentication on real-life data[1].

7.1 Problem Formulation

The application usage data from smartphones coupled with the timing information can be used to determine the exact day time and duration of using any application. It is assumed here that there might be certain pattern in the usage of different applications at different time of the day or during weekdays and weekends. Hence, a state-space model can be intuitively considered for modeling the pattern of application usage for a particular user. Models for different users are assumed to be different because of the differences in lifestyle of each individual. Therefore, the state-space model of a user can effectively be considered as a model for the pattern of life of that user and can be used to differentiate the user from others. There

[1]`http://www.biometricupdate.com/201506/atap-division-head-previews-behavioral-biometrics-system-at-google-io`

are however several challenges to this approach when applied to the authentication problem using application usage:

- Forming observation states from the application data and corresponding timing information.
- Training a state-space model in a way that it can handle unforeseen observations during testing.
- Generating verification scores from sequential observation data.

Each of these challenges and the proposed solutions are discussed here.

7.1.1 Application Names to Observation States

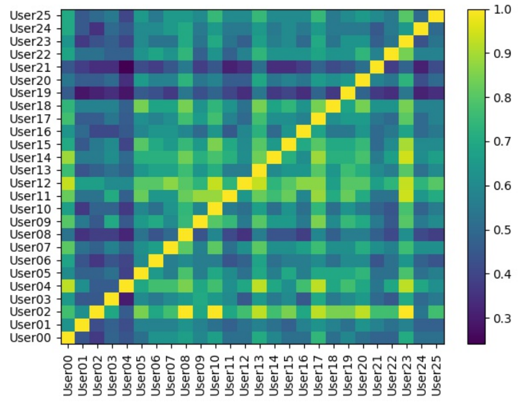
Incorporating the temporal information with the application name is a challenge because the user can use an application at any time, and therefore the power set of all applications and all probable time is intractable even if we sample at a relatively high frequency. For example, if there are N number of applications and if we sample every 5 minutes, then there would be 480 unique time stamps in a day and 3360 timestamps in a week. This would mean a total of $3360 \times N$ observation states for the applications in a week. However, for a single application, most of these observation states will either not occur or occur very infrequently in the training set. Hence, training a reliable state-space models with this sparsely occurring observation states will be difficult.

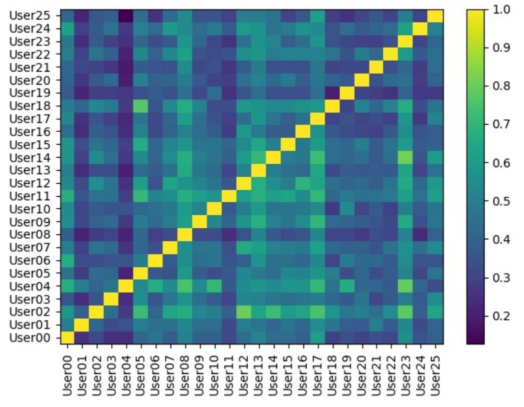
In this regard, the time-zone and weekday/weekend flag idea are adopted from [3]. By dividing the day into three distinct time zones (TZs), namely, TZ_1

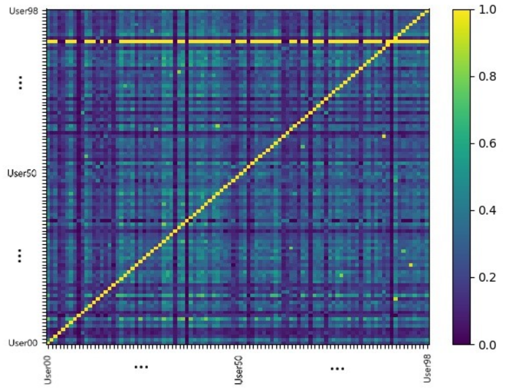
(12:01 am to 8:00 am), TZ_2 (8:01 am to 4:00 pm) and TZ_3 (4:01 pm to 12:00 pm), and denoting weekday/weekend with a flag $W(t) \in W_D, W_E \forall t$, respectively, the total number of possible observation states is kept limited to $6N$. The functions $TZ(t)$ and $W(t)$ maps any time t into one of the corresponding timezone and weekday/weekend, respectively. The impact of converting application tags into observations on verifying the users of the UMDAA-02 app-usage data and the Securacy datasets can be visualized from Figs. 7.1(a)-(b) and 7.1(c)-(d), respectively. The similarity matrix in Figs. 7.1(a) depicts the percentage of common applications between two users in UMDAA-02 training dataset, whereas, the similarity matrix in Fig. 7.1(b) depicts the percentage of common observations between any two users on the same dataset. It is clear that the similarity of observations between two different users is less than the similarity of applications. The effect is less visible on the Securacy dataset (Figs. 7.1(c)-(d))because the subjects came from a diverse population than the subjects of the UMDAA-02 dataset. Hence, the similarity of applications is less pronounced, yet, the differences between application similarity and observation similarity are still present.

7.1.2 Taking Unknown Applications into Account

Now, in order to handle unknown applications that might be present in the test set, an additional application name U is considered. The U application adds 6 observation states when combined with TZs and W. Note that in the training set the probability of having any U application is very low or zero, and all the observations


(a)


(b)


(c)

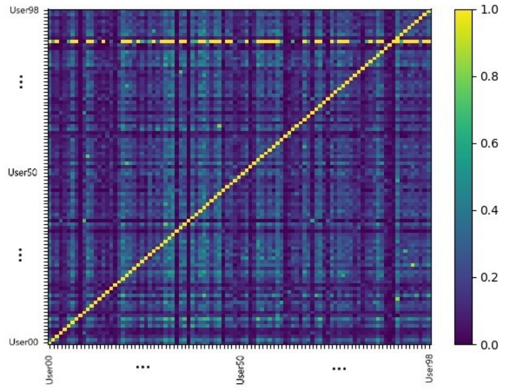

(d)

Figure 7.1: Similarity matrix depicting (a) application name overlap, and (b) observations overlap for the training set of the UMDAA-02 dataset. Similarly, (c) and (d) depicts the application overlap and observations overlap for the training set of the Securacy dataset

with U are assigned a very small prior probability $(10e - 20)$ when state-space models are trained. Also, it is ensured for state-space models that the emission probability for the states with U application does not go to zero, in order to prevent zero probability score during testing when unknown applications are encountered. If the total number of unique applications used by user X in the training set is A_x, then any application α_y of the test user Y in the test set $\bar{A}_y$ will be denoted as U if $\alpha_y \notin A_x$. In [3], the authors addressed similar issues for geo-location data by considering even more additional states such as nearby unknowns. However, proximity is a vague concept for application data and therefore only U is considered here. Note that, any observation with an unknown application is unforeseen by default, but an unforeseen observation with some other application name is not unknown.

Note that, apart from U, unforeseen observations might be present in the test set. For example, in the training set an application α_x might only occur in weekdays at timezones TZ_1 and TZ_2 while the same application might be used in the test set at time zone TZ_3 on a weekday. In that case, the test observation (α_x, TZ_3, W_D) would be unforeseen in the training set. For state space models, this problem is handled by generating all possible combinations applications, time zone and day flag and use them to construct the model. If one such observation is not present in the training set, it is assigned non-zero prior and emission probabilities to ensure that they do not bring down the probability of a test sequence to zero.

7.1.3 Handling Uncertainty

Now that unknown applications and unforeseen observation states are addressed, we tackle the creation of observation states via binning of time-stamped data. In most cases, the data collection is done in sessions, where a session starts with unlocking the phone and stops when the phone is locked again. Even if this is not the case, there can be very long idle times between consecutive usage of a phone, during which, authentication is a redundant operation and no application is running in the foreground [151]. Hence, there can be a big gap between the start-time for an application and the stop time of the previous application in the data log. This time gap might be as small as several seconds to as big as several days even for a user who owns a smartphone for regular use [152]. The sparsity introduced by this time gap is handled in two ways. At the beginning of each session (unlocking of the phone) a dummy observation state Ψ is introduced. The state-space model is expected to learn that Ψ is a time gap which might or might not cause a change in the time zone. For example, the last used application might be in TZ_1 before the closing of a session. Then the next session may occur in either TZ_1 or TZ_2 or TZ_3 of the same day. If the next session is in the next day or if the day changes within a running session, then an additional flag Δ is introduced which denotes the transition into next day. The time zone and weekday/weekend flags are ignored for observations Ψ and Δ.

So, taking the six probable observations for U and Ψ and Δ observations into consideration, the total number of possible observation states for user X would be $6N + 6U + \Psi + \Delta$.

7.1.4 System Overview

A diagram depicting an application-usage-based user verification system is shown in Fig. 7.2. Once the observation sequence is extracted, a verification model can be trained based on the patterns in the sequence. The verification model can be a state space model, a string matching approach or even a recurrent neural

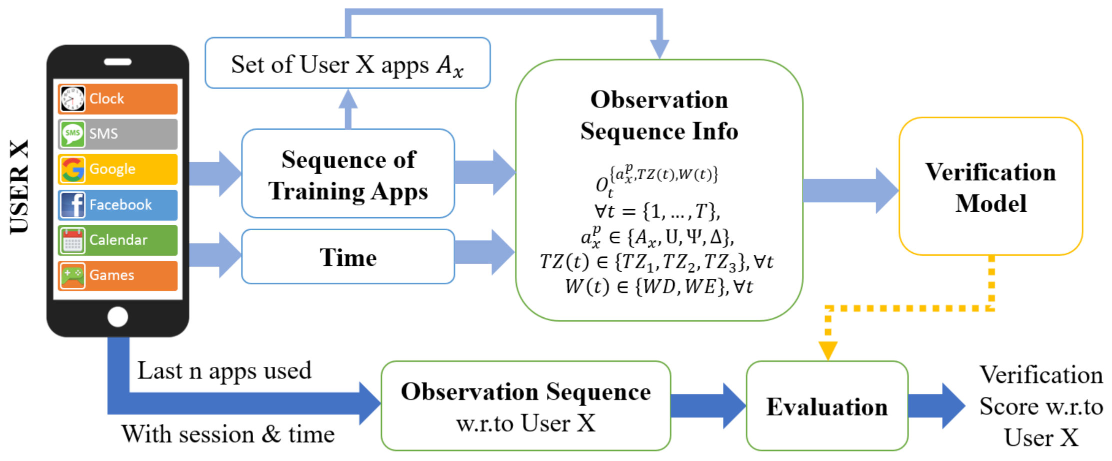

USER X
Clock
SMS
Google
Facebook
Calendar
Games
Set of User X apps A_x
Sequence of Training Apps
Time
Observation Sequence Info
$O_t^{\{a_x^p, TZ(t), W(t)\}}$
$\forall t = \{1, \ldots, T\}$,
$a_x^p \in \{A_x, \text{U}, \Psi, \Delta\}$,
$TZ(t) \in \{TZ_1, TZ_2, TZ_3\}, \forall t$
$W(t) \in \{WD, WE\}, \forall t$
Verification Model
Last n apps used
With session & time
Observation Sequence w.r.to User X
Evaluation
Verification Score w.r.to User X

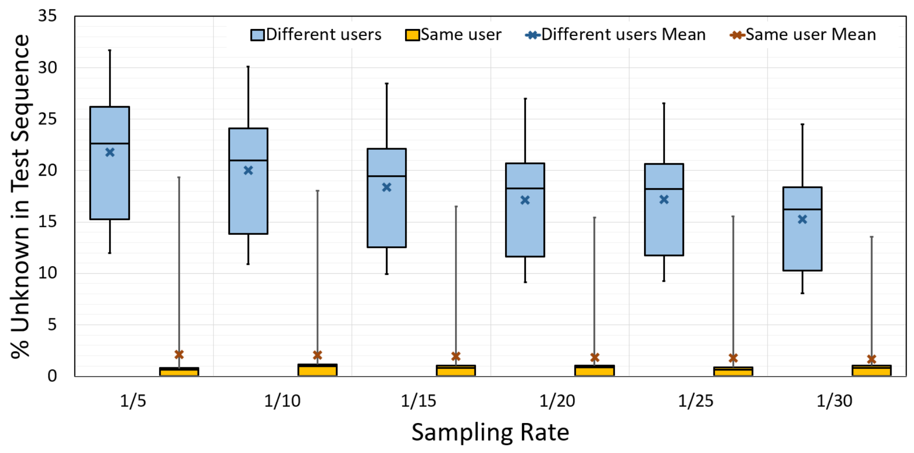
Figure 7.2: Overview of an application-usage-based user verification system for mobile devices.

network, depending on data availability and need. For state-space models, once training for a user is done, the model can be used to generate scores for the last n test observation sequences created using the same protocol that was used during the training phase. The score can be thresholded to obtain the verification decision. For more simpler methods such as sequence matching, unknown applications and unforeseen observations are difficult to handle. For the authentication problem, the unknown and unforeseen play key roles, described in the next section.

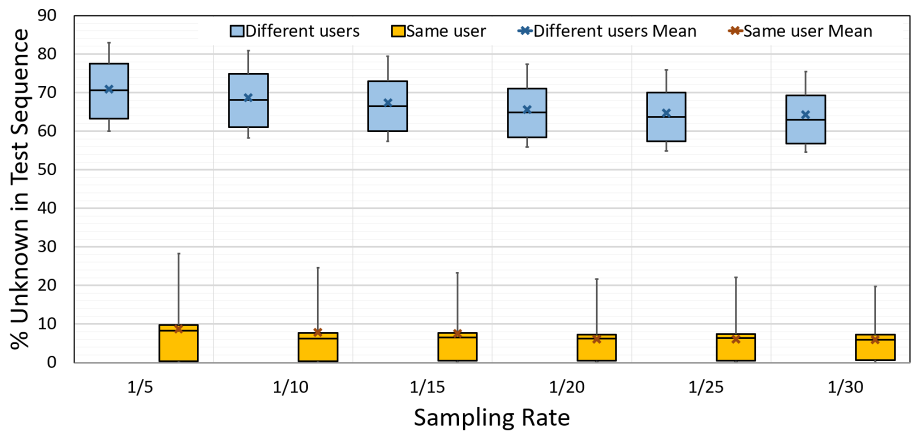
7.2 The Role of Unknown application and Unforeseen Observations in User Verification

7.2.1 Statistics of unknown applications in the test data

If an application is present in the test set but not encountered in the training set, the application is denoted with U as unknown application in the proposed



(a)



(b)

Figure 7.3: Boxplots depicting the percentage of unknown application in test data for (a) UMDAA-02 dataset, and (b) Securacy dataset, for different sampling rates. Note that the average percentage of unknown applications used by the the different user is much bigger than that for same user on both datasets.

formulation. Intuitively, the prevalence of U will be much higher if the test set comes from a different user or from an intruder of the phone, while for the legitimate user the test set will have fewer unknown applications. This intuition is verified on the application usage data from both UMDAA-02 and Securacy datasets, as can be seen from the box plots in Fig. 7.3.

Note that the gap between the whisker plots for same user and different users is larger for the Securacy dataset in comparison to UMDAA-02 dataset. Securacy is a larger dataset with more users, more data per user and more variation in user demographies compared to UMDAA-02 in which the subjects were from a narrow age range and were all affiliated with the same institution. Hence, it shows that among the general population, even the selection of applications varies widely between users.

7.2.2 Impacts of binary decision based on unforeseen events

Two simple experiments with unknown applications U and unforeseen observations are performed on the UMDAA-02 and Securacy datasets to evaluate their role in user verification. The observations for each user are chronologically sorted and the earliest 70% observations are considered for training and the rest for testing. Now, for any user i in the training set, a sequence of training observations S_i^{tr} is obtained along with the set of unique applications A_i. Now, each test sequence of a user is compared with the training sequence and application lists of the training subjects and different binary hard decision rules are applied in two experiments. In the first experiment, the binary decision rule is based on occurrence of an application in the test set that is not present in the training set. In the second experiment, the decision is taken based on the occurrence of an unforeseen observation in the test set. In both cases, if there is even a single occurrence of an unknown application or an unforeseen observation, then the match score is set to 0.0, otherwise it is set to 1.0. The matching algorithms for the two experiments are shown in (3) and (4), respectively. The data sampling rates for both these experiments were set to 1/30 per second, which resulted in $\sim$ 16863 training-test sequence pairs for the UMDAA-02 application-usage dataset and $\sim$ 846331 training-test sequence pairs for the Securacy dataset. The number of users with adequate training and test data is 26 in UMDAA-02 and 99 in Securacy, leading to an average of $\sim$ 647 and $\sim$ 8549 pairs per user, respectively.

Algorithm 3 Binary Decision Rule based on Unknown Applications

```
procedure BinUnk($A_i$, $S_j^{te}$)  ▷ List of unique applications of user $i$ ($A_i$), n-last Test Sequence Vector of user $j$ ($S_j^{te}$)
    for $v^{te} \in S_j^{te}$ do                ▷ Loop through all test observations
        $a^{te} \leftarrow v^{te}[0]$           ▷ Get the application name from the test observation
        if $a^{te} \notin A_i$ then
            return 0.0        ▷ Return score 0.0 if any unknown application is encountered
        end if
    end for
    return 1.0      ▷ Return score 1.0 if no unknown application in test sequence
end procedure
```

Algorithm 4 Binary Decision Rule based on Unforeseen Observations

```
procedure BinUnfore($S_i^{tr}$, $S_j^{te}$)  ▷ Sequence of training observations for user $i$ ($S_i^{tr}$), n-last Test Sequence Vector of user $j$ ($S_j^{te}$)
    for $v^{te} \in S_j^{te}$ do                ▷ Loop through all test observations
        if $v^{te} \notin S_i^{tr}$ then
            return 0.0        ▷ Return score 0.0 if any unforeseen observation is encountered
        end if
    end for
    return 1.0      ▷ Return score 1.0 if no unforeseen observation in test sequence
end procedure
```

Results for several evaluation metrics namely, sensitivity, specificity, F1-score and accuracy - all in percentage, obtained through the two experiments on the two

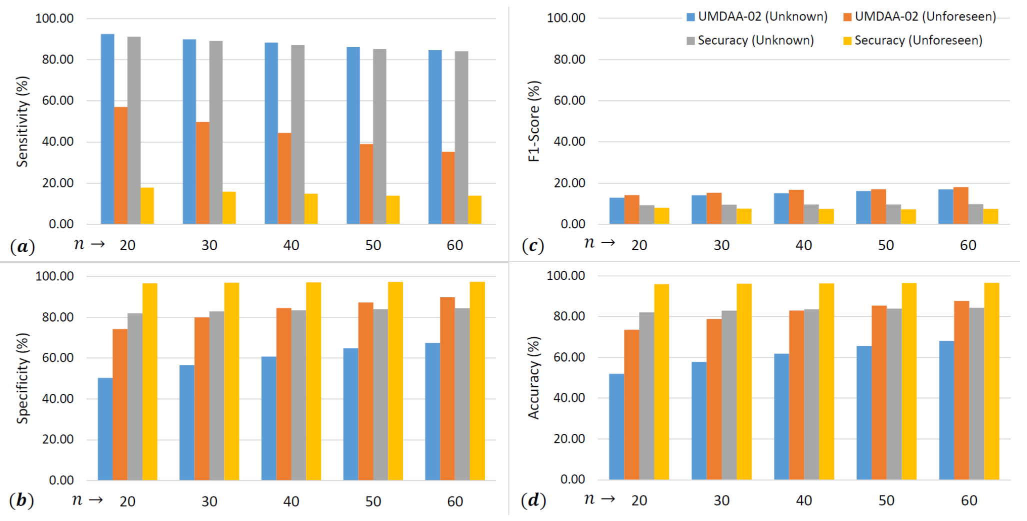


Figure 7.4: (a) Sensitivity, (b) Specificity, (c) F1-Score, and (d) Accuracy (in %) obtained by varying sequence length n for Securacy and UMDAA-02 application-usage data for using the Binary Hard Decision rule based on unknown applications and unforeseen observations.

datasets are shown in Fig. 7.4(a)-(d). The definition of these metrics are as follows:

$$Sensitivity = \frac{TP}{TP + FN} \times 100\% \tag{7.1}$$

$$Specificity = \frac{TN}{TN + FP} \times 100\% \tag{7.2}$$

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \times 100\% \tag{7.3}$$

$$F1 - Score = \frac{2TP}{2TP + FP + FN} \times 100\% \tag{7.4}$$

where, TP, FP and FN are the numbers of true positive, false positive and false negative detections, respectively. High Sensitivity implies smaller number of false-negatives, while high Specificity implies less false-positives. Accuracy over 50% denotes that the true values outweighs the false predictions. Finally, F1-Score implies better overall precision and recall.

Fig. 7.4 gives the following interesting insights about the impact of the unknown applications and unforeseen observations on the performance metrics for the two datasets.

- With increasing sequence length n, the specificity is increasing gradually for all the cases, while sensitivity is decreasing. The decrease in sensitivity is probably due to the fact that the probability of having an unknown application in the sequence increases with increasing sequence size, thereby increasing the chances for false negatives. On the other hand, with increasing n more sequences are denoted as negatives, which in effect reduces the number of false positives and therefore increases specificity.

- The sensitivity drops drastically when unforeseen observations are used instead of unknown applications as decision criteria. This is understandable, since the number of false negatives increases rapidly when all sequences with at least on unforeseen observations are marked as data from a different user.

- The number of false positives decreases when unforeseen observations are considered for decision instead of unknowns. This leads to a jump in specificity for a fixed n. In general, the specificity is much higher for Securacy dataset in comparison to UMDAA-02. This proves that there are more unknown applications and unforeseen applications in Securacy when comparing a user with others. Securacy being a more diverse and larger dataset has wider variation of information, which leads to this phenomenon. Here, the training data for each user is longer, meaning that they are much closer representation of real life and therefore, an unknown application or unforeseen observation is actually a different user's data in most cases.

- Higher sensitivity, however, does not mean that for real life data a simple binary classifier based on unforeseen observations is reasonably good. The F1-Score is very low for both datasets, which means either precision or recall or both of therm are very low. Since in the active authentication using application usage, the number of positive pairs is largely outweighed by the number of negative pairs, it can be assumed that $FP >> FN$ and $TN >> TP$. Since Precision=$\frac{TP}{TP+FP}$ and Recall=$\frac{TP}{TP+FN}$, that means, Recall>Precision. With increasing n, FN increases, while FP decreases, leading to reduction in recall and increase in precision. However, given the fact that the F1-Score does not improve much with increasing n, it can be assumed that Recall reduces steeply while Precision does not improve much.

- Irrespective of deciding with unknown or unforeseen, the accuracy is always lower for the UMDAA-02 dataset in comparison to Securacy dataset. Even though the application-usage information in Securacy is much larger than UMDAA-02, probably due to the high demographical similarity among the subjects of UMDAA-02, the binary hard measure performs poorly in comparison to Securacy. In practice, there could not be any assumption made about the demographic similarity or dissimilarity of an user and an intruder - hence, using neither unknown applications nor unforeseen observations as a hard decision metric cannot be a practical solution to the active authentication problem.

- The experiment once again proves that 'accuracy' is not a good performance metric when the number of samples between classes is severely biased. In this example, the average percentage of positive pairs in the dataset is $\sim 3.85\%$ on UMDAA-02 dataset and $\sim 1.01\%$ in the Securacy dataset. Being an open set problem, the task is to deal with heavily biased data towards negative samples and better performance measures in this regard would be receiver operating characteristic (ROC) curves and equal error rates (EER) instead of accuracy.

7.2.3 Impacts of ignoring unforeseen events

Now that the impact of unforeseen events on the authentication problem are established, a slightly more advanced sequence matching approach based on Levenshtein Distance [153] also known as the Edit-Distance (ED) [154] is performed to study the impact of ignoring the unknown observations and unforeseen events. When matching a sequence s_1 to another sequence s_2 of the same length, the original ED calculates the number of deletions, insertions, or substitutions required to transform s_1 to s_2. For the active authentication problem, let's assume that a test observation sequence S^{te} of length n is to be matched with any training observation sequence S^{tr} of length N, where, intuitively $N > n$. Since each observation consists of an application name, timezone and day flag, when a mismatch occurs, the distance can be assumed to be different depending on the amount of match. For example, if only the application name matches, then the timezone and day flag needs to be substituted, leading to two operations. Based on this fact, the a modified algorithm for

Algorithm 5 Pseudocode for the modified edit-Distance algorithm.

```
procedure M-ED($S^{tr}$, $S^{te}$)    ▷ Training observation sequence of a user ($S^{tr}$) of length $N$, $n$-last Test observation Sequence of any user ($S^{te}$), where length($S^{tr}$)$> n$.
    $D \leftarrow [1, 2, \ldots, n]$
    for $j = 0$ to $n - 1$ do
        $d \leftarrow zeros[1 : n]$
        $d[0] \leftarrow [j + 1]$
        for $i = 0$ to length($S^{tr} - 1$) do
            if $S^{te}[j] == S^{tr}[i]$ then
                $d[i + 1] \leftarrow D[i]$    ▷ Exact match, no operation needed.
            else
                $A_1, T_1, W_1 \leftarrow S^{tr}[i]$    ▷ Extract application name, timezone and day flag from the observations.
                $A_2, T_2, W_2 \leftarrow S^{te}[j]$    ▷ Extract application name, timezone and day flag from the observations.
                NOp $\leftarrow 0$
                if $A_1 == A_2$ and ($T_1 == T_2$ or $W_1 == W_2$) then
                    NOp $\leftarrow 1$    ▷ One substitution needed if only timezone or day does not match.
                else if $A_1 == A_2$ then
                    NOp $\leftarrow 2$    ▷ Two substitution needed if neither timezone nor day are matching.
                else
                    NOp $\leftarrow 3$    ▷ Three substitution operation for no match.
                end if
                $d[i + 1] \leftarrow$ NOp+ $\min(D[i], D[i + 1], d[i - 1])$
            end if
            $D \leftarrow d$
        end for
    end for
    return $D[n - 1]$
end procedure
```

Table 7.1: Performance of the M-ED algorithm in terms of EER (%) for three types of test sequences - all observations, all except the ones with unknown applications and all without unforeseen observations. Experiment performed on the UMDAA-02 dataset with fixed sampling rate at $1/30s^{-1}$.

n	**%EER**		
	All Obs.	**No Unknown Apps.**	**No Unforeseen Obs.**
20	43.20	49.22	48.96
30	39.03	44.72	46.70
40	36.97	43.64	45.01
50	35.53	42.19	44.16
60	34.31	42.47	43.29

edit distance (M-ED) is presented in (5). Using this algorithm, three different tests are performed on the UMDAA-02 dataset, the results for which are given in Table 7.1. In the first test, all test observations are included, while in the next two tests, the observations with unknown applications, and the unforeseen observations are ignored. In order to ignore the unforeseen observations, for any training sequence, each test sequence is compared to find the unforeseen observations and removed from the test sequence. For unknown applications, the corresponding observation is removed. This operation reduced the number of samples per user from 891 to 458 and 245, respectively, and the number of unique application in the test data went from 61 to 60 and 45. As can be seen from Table 7.1, the lowest EERs for any value of n are obtained when all observations are considered. Ignoring both unknown applications and unforeseen observations make the verification task difficult. Also, for practical purposes, ignoring samples will cause latency in decision making, which can greatly reduce the recall of an active authentication system.

In the next section, some suitable modeling approaches for the application usage-based active authentication problem are discussed.

7.3 Suitable Modeling Techniques

In light of the outcomes of the experiments presented in the previous section, it can be asserted that the application-usage-based verification models must be capable of taking into account unknown applications and unforeseen observations. A popular approach to model temporal data sequences is to use state-space models such as Mobility Markov Chains or HMMs which can model time variation of the data. However, these methods are not capable of handling unforeseen events by default. For example, any unforeseen event will be given a zero emission probability in these models, and therefore, the models will be somewhat like the binary decision model that was discussed earlier. However, simple modifications to these models can improve the usability of these methods when unforeseen events are present as discussed in chapter 6 for geo-location data. In this research work, the three state-space models namely, the Markov Chain (MC)-based Verification, HMM with Laplacian Smoothing (HMM-lap) and Marginally Smoothed HMM (MSHMM), described in section 6.1.3, are employed for application-usage-based verification task and the performances are compared.

For the MC method, the prior probability for unknown and unforeseen events are set to a very small nonzero probability of $\delta = e^{-20}$ (Laplace-smoothing) when training a model X_T for observation sequences of length T. For MC, the probability of transitioning to an observation state o_j depends only on the probability of the last observation state o_i, i.e.

$$\tau_{i,j} = Prob(X_T = o_j | X_{T-1} = o_i). \tag{7.5}$$

If the prior probability of entering any state i is $p_i = Prob\{X_0 = i\}$ with respect to the set of observations for user-z O_T^z , then the total probability of traversing any sequence of n consecutive observations $i_0, \ldots, i_n \in O_T^z$ is calculated as

$$Prob(X_0 = i_0, \ldots, X_n = i_n) = p_{i_0} \tau_{i_0,i_1} \ldots \tau_{i_{n-1},i_n} \tag{7.6}$$

Similar to the MC method, in HMM-lap method Laplacian Smoothing of the emission probabilities is considered alogn with HMM to incorporate unforeseen observations as discussed in [3]. The number of hidden states is fixed to 20 for all the experiments and the maximum number of iteration is set to 50.

The most suitable approach for handling unforeseen observations is the Marginally Smoothed Hidden Markov Model (MSHMM) introduced in [3]. To adopt the approach for the active authentication problem, the marginal probabilities of the presence of an application in the training sequence of a user for each time-zone and day flags are precomputed. Assuming that the probability of user-x using application a_x^i at time-zone $TZ(t)$ at time t, $P(a_x^i, T_j)$ is independent of the probability of user-x using the application at location $W(t)$, $P(a_x^i, W(t))$ at time t, the emission probability from state s to observation o_t, $\widehat{e}_s(o_t)$ is $P(O_t^{\{a_x, TZ(t), W(t)\}} = o_t^{\{a_x, TZ(t), W(t)\}} | X_t = s)$ if

$o_t \in O_t^{\{a_x^p, TZ(t), W(t)\}}$. Otherwise,

$$\begin{aligned}\widehat{e}_s(o_t) &= P(O_t^{\{a_x, TZ(t)\}} = o^{\{a_x, TZ(t)\}} | X_t = s) \times \\ &\quad P(O_t^{\{a_x^p, W(t)\}} = o_t^{\{a_x, W(t)\}} | X_t = s), \end{aligned} \tag{7.7}$$

where $P(o_t^{\{a_x, TZ(t)\}} = max(\delta, P(a_x, TZ(t)))$ and, $P(o_t^{\{a_x, W(t)\}} = max(\delta, P(a_x, W(t)))$. By definition, the MSHMM approach is capable of differentiating between unknown applications and unforeseen observations with known applications, as well as, the more frequent vs. less frequent applications occurring at different time zones and days.

In the next section, experimental results for these three verification methods are discussed in detail for performance comparison.

7.4 Experimental Results

The performances of M-ED, MC, HMM-lap and MSHMM algorithms for the full test sequences of the UMDAA-02 application usage dataset are shown in Table 7.2, where, the sampling rate has been varied from one sample every 5 seconds to one sample every 30 seconds with intervals of 5 seconds, while the number of previous observations n is varied from 20 to 60 with intervals of 10. It can be seen from the table that with smaller sampling rate and bigger n, the EER drops for all the methods. The MSHMM outperforms every other method in every case, which can be attributed to the improved modeling capability of the method due to marginal

Table 7.2: Application-usage-based verification performance comparison for UMDAA-02 dataset across different methods based on EER (%) for varying sequence length (n) and sampling rate. The number of hidden states is fixed at 20 and maximum number of iteration is 50 for HMM-based methods.

n	**Method**	**Sampling Rate**					
		1/5	**1/10**	**1/15**	**1/20**	**1/25**	**1/30**
20	**M-ED**	42.96	42.92	44.12	43.64	43.09	43.2
	MC	40.86	40.53	40.27	39.48	40.39	36.78
	HMM-lap	38.49	38.35	37.82	37.39	38.83	36.77
	MSHMM	37.3	37.3	36.67	35.93	35.63	34.82
30	**M-ED**	42.7	41.71	40.18	38.17	37.58	39.03
	MC	40.29	39.18	38.21	40	39.04	36.82
	HMM-lap	37.28	37.2	36.68	37.73	37.89	37.45
	MSHMM	36.23	36.87	35.74	36.87	35.99	35.79
40	**M-ED**	41.7	38.64	38.41	38.13	37.45	36.97
	MC	39.29	40.57	38.13	39.62	41.97	35.89
	HMM-lap	37.37	37.88	36.75	36.07	39.11	34.62
	MSHMM	35.4	35.65	34.026	34.4	36.58	32.54
50	**M-ED**	40.69	37.98	36.19	35.55	35.58	35.53
	MC	40.34	37.92	38.67	36.96	39.57	33.56
	HMM-lap	36.97	36.01	36.48	34.72	36.7	33.95
	MSHMM	35.95	34.41	34.67	32.41	35.27	30
60	**M-ED**	38.69	35.93	35.32	35.72	34.97	34.31
	MC	38.33	37.5	37.5	38.01	35.91	34.35
	HMM-lap	35.31	35.48	34.18	33.15	36.05	34.35
	MSHMM	34.036	34.92	32.78	33.33	34.3	31.93

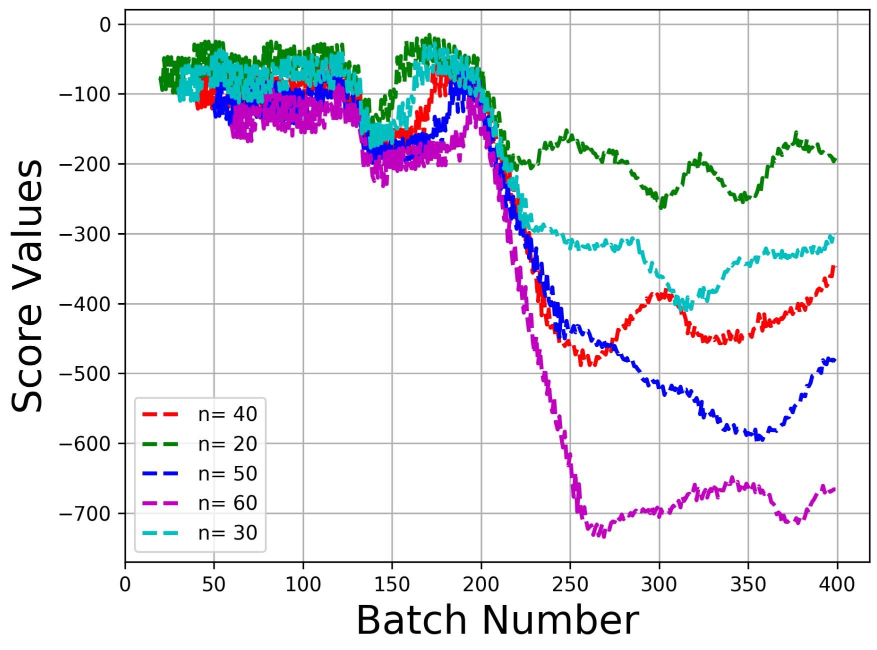


Figure 7.5: Average change in MSHMM scores in response to intrusion on the UMDAA-02 application-usage data.

smoothing. For a practical verification system, the sampling rate and value of n would determine the latency of decision making. In many cases, a sample every 30 second might be too late and therefore the system designer should choose these parameters carefully.

As for n, intuitively with more historical data the performance should improve all the time. In order to determine the impact of n and also to get an idea about the latency of MSHMM when intrusion occurs, a different experiment was performed where a different user's data is appended with the legitimate user's data to simulate intrusion. To be more precise, for each user of the UMDAA-02 dataset, 200 consecutive observations from the test sequence starting from a random index are appended with 200 consecutive observations from the test sequences of all the other users (start index picked randomly) and the whole sequence is evaluated using

MSHMM for different n values. The average score values across all users are plotted in Fig. 7.5 for different n values. When the observations from a different user starts to enter a batch (at 200-th batch), the average scores returned by MSHMM for each batch drops vividly, as can be seen from the figure. Also, the figure clearly shows the drop is larger for large n values - justifying the intuition that considering more historical data is advantageous in this regard. As for latency, if the score of -200 is considered as a threshold for decision making, then for all $n = 60$, the intrusion will be detected within ~ 5 batches, i.e. withing 2.5 minutes from the inception of intrusion.

Finally, for the Securacy dataset, the performances of MSHMM, HMM-lap, MC and M-ED are presented in Table. 7.3. Similar to the UMDAA-02 dataset results, MSHMM outperforms the other methods by a good margin. Note that the EER values are much lower for this dataset for the state-space models, which is understandable since it has already been demonstrated in Fig. 7.1(c) that the users are quite separable in this dataset even if only application names are considered. However, M-ED faces difficulty in exploiting the separability of the observations since is not capable of modeling temporal variations as effectively as state-space models.

Based on results of the experiments presented in this work, it can be asserted that application-usage data might be useful as a soft biometric for user verification for bolstering the decision in a multi-modal authentication scenario. Given the fact that the application-usage data is readily available and easy to track without using much battery or computational power, real-time score generation is possible. The

Table 7.3: App-based verification EER(%) comparison for Securacy dataset across different methods [3] for different n values. Number of Hidden States is set to 20 and sampling rate is $1/30s^{-1}$.

n	**MSHMM**	**MC**	**HMM-lap**	**M-ED**
20	17.23	19.286	19.66	35.09
30	16.75	18.9967	19.59	32.88
40	16.38	18.7074	19.19	31.4
50	16.26	17.9475	19.22	30.53
60	16.16	17.6443	18.38	30.58

experiments also depict that the verification scores show rapid change for intrusion within several minutes. Hence, the latency is not too high for a soft biometric measure. However, even though state-space models can be made to work well with some modifications, the equal error rate for a diverse dataset is still around $\sim$ 16%, which needs further improvement. In this regard, bigger training datasets and keeping longer usage history might be helpful. In addition, if computational constraints can be loosened, then more sophisticated high-performance methods such as deep neural networks can be employed to minimize the EER.

To summarize, in this chapter, the challenging problem of active authentication using application usage data has been formulated and systematically tackled to obtain viable solutions. Through several experiments, the impact of unknown applications and unforeseen observations on the authentication problem has been investigated and it is established that for this problem inclusion of the uncertain events are necessary to obtain better performances. In this regard, a modified edit distance algorithm has been introduced, the performance of which is compared with three state-space models namely, Markov Chain, HMM with Laplacian Smoothing and Marginally-Smoothed HMM, in terms of EER. Experiments were performed on the UMDAA-02 and the Securacy application-usage datasets. The experiments revealed some very interesting insights about the differences between the two datasets. Also, we addressed different aspects of important practical considerations such as intrusion detection, latency, observation history and sampling rate. As for future work, the M-ED method might be further improved by varying the distances for the three different cases based on the marginal probabilities. Also, recurrent neural network (RNN)-based models might be able to learn more discriminative properties of application-usage patterns. However, RNNs require huge amount of data for useful training, which the two datasets presented here lacks. Another interesting research direction would be the joint training of application sequence and some other sequential data such as the location data to improve the authentication performance. Finally, since application information are also suitable context for other modalities, application data sequences can have duel utilization (as a separate modality and also as context) in more advanced active authentication schemes.

Chapter 8: Future Research Directions

In this chapter, we present directions for future research based on the works presented in this dissertation and/or based on ideas that are closely related to authentication research.

8.1 Facial Segment-based Face Verification

The concept of facial segments can have a significant impact on reliably verifying partially visible and occluded faces. At present, most face verification networks are trained in a data-driven manner where the verification of occluded faces is not explicitly handled, rather it is expected that if the training dataset has sufficient examples of partially visible and occluded faces then the model will learn to identify/verify them. However, in generic face verification datasets, partial visibility is an infrequent phenomenon, whereas, in the mobile domain, it is highly prevalent. Therefore, similar to traditional face detector networks, verification networks that are trained in the traditional way are not expected to achieve satisfactory precision and recall for AA tasks. Our proposal for using facial segments to extract local information and merging them for global decision making can make a difference in this regard. A network with architecture similar to DRUID can be trained where the top branches will be replaced with attention networks [155] [156] [157], one for each of the facial segments. The approach would have similarity with the multi-attention approach presented in [158] but instead of allowing the network to choose the attention region, it will be enforced by a loss utilizing the ground truth mask for each particular segment. Each branch will also have a classification task during training in order to ensure that the network learns to identify the subject separately based on each estimated facial segments that are visible. The features that are used for segment-wise classification will be concatenated and passed through higher layers and finally reduced to a more compact, global feature representation tasked to identify the person. Once trained, the network can be used to extract local and global deep features from any face image. For any two faces, the deep features can be compared to measure similarity for verification task. Several key aspects of this tentative network and the training mechanism are as follows:

- Based on our approach, there can be a total of 14 facial segments present for a single face. Hence, it will require huge memory to fit the fully connected layers if the number of training classes is very large. In this regard, we would suggest using a dataset like UMDFaces that has ≈ 8000 training classes, but a huge number of samples per class to keep the memory usage in check.

- Even if we keep the number of training classes relatively low, the architecture of network would permit data augmentation by cropping the input images to ensure the presence of a large number of partially visible faces (following the augmentation mechanism presented in 4.2.1.2). The augmentation will ensure that the network learns to depend on the visible segments to identify a person instead of relying only on a global representation.

- The proposed formulation will also help investigate the impact of different facial segments on the verification task and enable to experiment on different feature level fusion mechanisms for extracting a global face representation from local features.

With the multi-task architecture, extensive data augmentation and local to global feature representation mechanism, it can be expected that the proposed network will perform well on cross-domain face verification tasks (as demonstrated for DRUID and SPLITFACE) and definitely on verifying partially visible or occluded faces.

8.2 Enforcing Rank Constraint on Deep Feature Extraction Networks for Verification

There has been on-going research on low-dimensional feature representation of faces and objects for verification tasks. Discriminative low-dimensional representation speeds up search operations for large datasets. In traditional verification approaches, a deep neural network is trained on a multi-class classification task using a categorical cross-entropy or a similar loss in order to get a good feature representation in an intermediate (usually second to last) layer. The features do not have any order or rank and are considered equally important for verification. One can enforce a rank condition on the feature layer during training using two different approaches. In the first approach, we can introduce a modified dropout layer in between the feature layer and the final fully connected classification layer. The dropout layer will have gradually increasing dropout probability for the neurons of the feature layer following a linear or exponential function. During training, the model will learn to rely on the neurons will minimum or no dropout, while the dependency will gradually decrease with increasing dropout. Hence, the feature vector will be ranked according to the importance of the feature. The rapidness of the decay can be controlled by changing the curvature of the decay function. Another approach could be to apply a gradually decaying cap on the magnitude of the neurons in the feature layer following a linear or exponentially decaying function. That way, neurons with larger weights will be able to accommodate a wider variety of values and will also have a large impact on the distance/similarity measure during verification, while neurons with smaller weights will contribute less in the decision making for verification. The network will learn about this variation in weights when training for classification and will try to accommodate more discriminative information in the neurons with larger weights. Hence, the feature vector will be ranked according to importance.

We did some preliminary experiments on this concept and trained several models with and without the ranking-constraint for a car-model verification task on the Comprehensive Cars (CompCars) dataset [159]. Our results show that with the ranking constraint, the networks achieve noticeable improvement in ROC curves at lower dimensions in comparison to randomly picked entries from the feature vectors for a network trained without the constraint.

8.3 Reinforcement Learning for Conditional Detection of Facial Attributes

There are several facial attribute pairs that are highly correlated. For example, in [37], the authors reported that there are strong positive correlations between heavy makeup and wearing lipstick, chubby and double chin, mouth slightly open and smiling etc., while there are strong negative correlations among male and wearing lipstick, male and heavy makeup, goatee and no beard etc. It is also intuitive that there are several attributes that can directly impact other attributes. For example, if we have a prior information for a face to have a goatee, it becomes highly likely that the person is an adult male and also asserts right away that no-beard or 5 o'clock shadow are not true and having heavy makeup is less likely. It would be interesting to train a network that would detect the attributes one by one in a sequence starting from the attribute that it can detect most reliably. The probability estimate of all future attributes will be conditioned on all currently estimated attribute probabilities. The system can be realized with a Deep-Q learning architecture for reinforcement learning [160] [161] [162] in which the Deep-Q network will take the image and all the previous estimates as inputs to dictate an attribute detector on which attribute to estimate next. The attribute detector will take the full image, the prior estimates and the information on which attribute to detect next as input and produce an estimate for that attribute conditioned on the priors. The detector might be pre-trained with ground truth values for randomly picked attributes. We can expect the Deep-Q agent to be able to produce an optimum sequence for detecting the attributes in a fast and reliable manner. The agent can also be expected to learn a stopping criterion because the detector network might converge to the best solutions earlier. It would be interesting to analyze the sequences of attributes to be detected that the Deep-Q agent produces. For example, it is not intuitive whether the agent will dictate the detector to detect local attributes first and gradually move towards the global or the other way around. Also, the network will learn to handle conditional probabilities for the attributes and therefore hopefully will learn and exploit the correlation among different attributes and improve the results.

8.4 Development of Multi-modal User Verification Model

So far we have evaluated the discriminative power of location traces and application usage data for user verification. Fusing these modalities with other non-face modalities such as the accelerometer, gyroscope, touch dynamics data battery and memory usage, while using the foreground application as a context, can be an effective approach to obtain a robust model for user authentication. Since, different apps require different types of handling and activities from the user's part, comparing across different applications is intuitively not practical. The sensor data are temporal and hence needs to be divided into fixed length time windows to train recurrent neural networks. Also, features from multiple modalities need to be fused together to strengthen the discriminative power of the feature vectors.

Chapter 9: Conclusion

In this dissertation, we have presented our work on active user authentication using different sensor data from smartphones. Our benchmark evaluation of state-of-the-art algorithms on the mobile domain modalities showed that those algorithms are not tailored to perform well in the mobile domain. We addressed the very basic problem of shortage of realistic, unsupervised, multi-modal active authentication dataset and collected one for this research the technical help from Google. We developed customized techniques to address the needs of active authentication, such as facial segment-based face detection techniques like DRUID and SPLITFACE which can reliably detect face and facial attributes, respectively, from partially visible and MSHMM, the location-based user verification model that takes unforeseen data and data sparsity into account while training models for the authorized users.

More specifically, we developed the facial segment-based face detector (FSFD) and its more advanced variant SegFace for partially visible face detection. Then, we re-defined the FSFD method in terms of neural network architecture to develop the DeepSegFace algorithm. Furthermore, considering the requirements of AA, we developed an end-to-end trainable deep regression-based single face detector, namely, DRUID, that does not require the proposal generator and outperforms all the previous methods and the state-of-the-art methods with a big margin. Then we extended the idea of facial segments to the task of facial attribute detection from partially visible and occluded faces. In this regard, we developed the SPLITFACE network that takes the facial segments into account, estimates the local and global attributes separately and combines them using a committee machine approach to obtain more robust estimates of all attributes.

In another track, we modified the traditional Hidden Markov Model to handle unforeseen states and sparse temporal data by using marginal probabilities of location traces. The method, namely, Marginally Smoothed HMM (MSHMM), produces user verification score based on several last visited locations and corresponding timing information. We observed MSHMM to perform effectively for application-usage-based user authentication task as well. A modified edit-distance was proposed as a simpler approach to show the impact of unforeseen observations and unknown applications on the authentication task. We also performed experiments on intrusion detection latency, observation history and sampling rate for the application-usage-based active authentication problem.

Based on the concepts and formulations presented in this dissertation, we presented several future research direction on active authentication. The idea of facial segment can be utilized to develop a face verification model for partially visible and occluded faces. The facial attribute detection problem can be formulated as a reinforcement learning problem where the reinforcement agent will guide the network to select which attribute to estimate next based on already estimated attributes. We also presented an approach for enforcing a rank constraint on deep neural networks to obtain ranked deep features that would help in searching on large datasets and in training compressed networks. Finally, we presented an approach for fusing several non-face modalities such as accelerometer, gyroscope, swiping pattern etc to develop a multi-modal authentication scheme using a Siamese recurrent network. We hope that this dissertation will be helpful for computer vision and machine learning researchers in the future to determine a realistic pathway to achieve satisfactory level of authentication from unconstrained smartphone usage data.

Bibliography

[1] M. Günther, A. Rozsa, and T. E. Boult, "AFFACT - alignment free facial attribute classification technique," *CoRR*, vol. abs/1611.06158, 2016. [Online]. Available: http://arxiv.org/abs/1611.06158

[2] H. Han, A. K. Jain, S. Shan, and X. Chen, "Heterogeneous face attribute estimation: A deep multi-task learning approach," *CoRR*, vol. abs/1706.00906, 2017. [Online]. Available: http://arxiv.org/abs/1706.00906

[3] U. Mahbub and R. Chellappa, "Path: Person authentication using trace histories," in *2016 IEEE 7th Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON)*, Oct 2016, pp. 1–8.

[4] U. Mahbub, V. M. Patel, D. Chandra, B. Barbello, and R. Chellappa, "Partial face detection for continuous authentication," in *2016 IEEE International Conference on Image Processing (ICIP)*, Sept 2016, pp. 2991–2995.

[5] U. Mahbub, S. Sarkar, and R. Chellappa, "Pooling facial segments to face: The shallow and deep ends," in *12th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2017)*, May 2017.

[6] E. Chin, A. P. Felt, V. Sekar, and D. Wagner, "Measuring user confidence in smartphone security and privacy," in *Proc. 8th Symp. Usable Privacy and Security*, ser. SOUPS, 2012, pp. 1:1–1:16.

[7] I. T. Fischer, C. Kuo, L. Huang, and M. Frank, "Smartphones: Not smart enough?" in *Proc. of the 2nd ACM Workshop Security and Privacy in Smartphones and Mobile Devices*, ser. SPSM, 2012, pp. 27–32.

[8] (2014) Smart phone thefts rose to 3.1 million in 2013. urlhttp://www.consumerreports.org/cro/news/2014/04/smart-phone-thefts-rose-to-3-1-million-last-year/index.htm.

[9] Verizon RISK Study. (2013) Data breach investigations report. urlhttp://www.verizonenterprise.com/resources/reports/rp_data-breach-investigations-report-2013_en_xg.pdf.

[10] M. Meeker and L. Wu. (2013, May) Kleiner perkins caufield and byers (kpcb): Internet trends. [Online]. Available: http://www.kpcb.com/blog/2013-internet-trends

[11] V. M. Patel, R. Chellappa, D. Chandra, and B. Barbello, "Continuous user authentication on mobile devices: Recent progress and remaining challenges," *IEEE Signal Processing Magazine*, vol. 33, no. 4, pp. 49–61, July 2016.

[12] P. Samangouei, V. M. Patel, and R. Chellappa, "Attribute-based continuous user authentication on mobile devices," in *International Conference on Biometrics Theory, Applications and Systems (BTAS), Arlington, VA*, Sept 2015.

[13] M. E. Fathy, V. M. Patel, and R. Chellappa, "Face-based active authentication on mobile devices," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[14] T. Feng, Z. Liu, K.-A. Kwon, W. Shi, B. Carbunar, Y. Jiang, and N. Nguyen, "Continuous mobile authentication using touchscreen gestures," in *Homeland Security (HST), 2012 IEEE Conf. on Technologies for*, Nov. 2012, pp. 451–456.

[15] H. Zhang, V. Patel, M. Fathy, and R. Chellappa, "Touch gesture-based active user authentication using dictionaries," in *Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on*, Jan 2015, pp. 207–214.

[16] L. Araujo, J. Sucupira, L.H.R., M. Lizarraga, L. Ling, and J. Yabu-Uti, "User authentication through typing biometrics features," *Sig. Process., IEEE Trans.*, vol. 53, no. 2, pp. 851–855, 2005.

[17] R. P. Guidorizzi, "Security: Active authentication," *IT Professional*, vol. 15, no. 4, pp. 4–7, July 2013.

[18] D. Crouse, H. Han, D. Chandra, B. Barbello, and A. K. Jain, "Continuous authentication of mobile user: Fusion of face image and inertial measurement unit data," in *International Conference on Biometrics (ICB)*, May 2015.

[19] M. Frank, R. Biedert, E. Ma, I. Martinovic, and D. Song, "Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 1, pp. 136–148, Jan. 2013.

[20] M. Jakobsson, E. Shi, P. Golle, and R. Chow, "Implicit authentication for mobile devices," in *Proceedings of the 4th USENIX Conference on Hot Topics in Security*, ser. HotSec'09. Berkeley, CA, USA: USENIX Association, 2009, pp. 9–9. [Online]. Available: http://dl.acm.org/citation.cfm?id=1855628.1855637

[21] E. Shi, Y. Niu, M. Jakobsson, and R. Chow, "Implicit authentication through learning user behavior," in *Proceedings of the 13th International Conference on Information Security*, ser. ISC'10. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 99–113. [Online]. Available: http://dl.acm.org/citation.cfm?id=1949317.1949329

[22] N. Clarke, *Transparent User Authentication: Biometrics, RFID and Behavioural Profiling*, 1st ed. Springer Publishing Company, Incorporated, 2011.

[23] R. Ranjan, V. M. Patel, and R. Chellappa, "Hyperface: A deep multi-task learning framework for face detection, landmark localization, pose estimation, and gender recognition," *CoRR*, vol. abs/1603.01249, 2016. [Online]. Available: http://arxiv.org/abs/1603.01249

[24] Y. Sun, X. Wang, and X. Tang, "Deep convolutional network cascade for facial point detection," in *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '13. Washington, DC, USA: IEEE Computer Society, 2013, pp. 3476–3483. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2013.446

[25] S. S. Farfade, M. J. Saberian, and L.-J. Li, "Multi-view face detection using deep convolutional neural networks," in *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, ser. ICMR '15. New York, NY, USA: ACM, 2015, pp. 643–650. [Online]. Available: http://doi.acm.org/10.1145/2671188.2749408

[26] D. Ramanan, "Face detection, pose estimation, and landmark localization in the wild," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, ser. CVPR '12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 2879–2886. [Online]. Available: http://dl.acm.org/citation.cfm?id=2354409.2355119

[27] V. Jain and E. Learned-Miller, "Fddb: A benchmark for face detection in unconstrained settings," University of Massachusetts, Amherst, Tech. Rep. UM-CS-2010-009, 2010.

[28] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," University of Massachusetts, Amherst, Tech. Rep. 07-49, October 2007.

[29] M. Kostinger, P. Wohlhart, P. Roth, and H. Bischof, "Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, Nov 2011, pp. 2144–2151.

[30] S. Sarkar, V. M. Patel, and R. Chellappa, “Deep feature-based face detection on mobile devices,” in *2016 IEEE International Conference on Identity, Security and Behavior Analysis (ISBA)*, Feb 2016, pp. 1–8.

[31] U. Mahbub, S. Sarkar, V. M. Patel, and R. Chellappa, “Active user authentication for smartphones: A challenge data set and benchmark results,” in *Biometrics Theory, Applications and Systems (BTAS), 2016 IEEE 7th Int. Conf.*, Sep. 2016.

[32] C. McCool, S. Marcel, A. Hadid, M. Pietikainen, P. Matejka, J. Cernocky, N. Poh, J. Kittler, A. Larcher, C. Levy, D. Matrouf, J.-F. Bonastre, P. Tresadern, and T. Cootes, “Bi-modal person recognition on a mobile phone: using mobile phone data,” in *IEEE ICME Workshop on Hot Topics in Mobile Multimedia*, Jul. 2012.

[33] P. Viola and M. 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*, vol. 1, 2001, pp. I–511–I–518 vol.1.

[34] P. Samangouei and R. Chellappa, “Convolutional neural networks for facial attribute-based active authentication on mobile devices,” in *International Conference on Biometrics Theory, Applications and Systems (BTAS), Arlington, VA*, Sept 2016.

[35] K. He, Y. Fu, and X. Xue, “A jointly learned deep architecture for facial attribute analysis and face detection in the wild,” *CoRR*, vol. abs/1707.08705, 2017. [Online]. Available: http://arxiv.org/abs/1707.08705

[36] E. M. Rudd, M. Günther, and T. E. Boult, *MOON: A Mixed Objective Optimization Network for the Recognition of Facial Attributes.* Cham: Springer International Publishing, 2016, pp. 19–35. [Online]. Available: https://doi.org/10.1007/978-3-319-46454-1_2

[37] E. Hand and R. Chellappa, “Attributes for improved attributes: A multi-task network utilizing implicit and explicit relationships for facial attribute classification,” 2017. [Online]. Available: https://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14749

[38] N. Kumar, A. Berg, P. Belhumeur, and S. Nayar, *Attribute and simile classifiers for face verification*, 2009, pp. 365–372.

[39] N. Kumar, A. Berg, P. N. Belhumeur, and S. Nayar, “Describable visual attributes for face verification and image search,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 10, pp. 1962–1977, Oct 2011.

[40] D. A. Vaquero, R. S. Feris, D. Tran, L. Brown, A. Hampapur, and M. Turk, “Attribute-based people search in surveillance environments,” in *2009 Workshop on Applications of Computer Vision (WACV)*, Dec 2009, pp. 1–8.

[41] Q. Xu, J. Erman, A. Gerber, Z. Mao, J. Pang, and S. Venkataraman, "Identifying diverse usage behaviors of smartphone apps," in *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, ser. IMC '11. New York, NY, USA: ACM, 2011, pp. 329–344. [Online]. Available: http://doi.acm.org/10.1145/2068816.2068847

[42] "Spotlight on consumer app usage part-1," App Annie, Tech. Rep., 2017. [Online]. Available: http://files.appannie.com.s3.amazonaws.com/reports/1705_Report_Consumer_App_Usage_EN.pdf

[43] Y. Xu, M. Lin, H. Lu, G. Cardone, N. Lane, Z. Chen, A. Campbell, and T. Choudhury, "Preference, context and communities: A multi-faceted approach to predicting smartphone app usage patterns," in *Proceedings of the 2013 International Symposium on Wearable Computers*, ser. ISWC '13. New York, NY, USA: ACM, 2013, pp. 69–76. [Online]. Available: http://doi.acm.org/10.1145/2493988.2494333

[44] V. Srinivasan, S. Moghaddam, A. Mukherji, K. K. Rachuri, C. Xu, and E. M. Tapia, "Mobileminer: Mining your frequent patterns on your phone," in *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ser. UbiComp '14. New York, NY, USA: ACM, 2014, pp. 389–400. [Online]. Available: http://doi.acm.org/10.1145/2632048.2632052

[45] P. Welke, I. Andone, K. Blaszkiewicz, and A. Markowetz, "Differentiating smartphone users by app usage," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ser. UbiComp '16. New York, NY, USA: ACM, 2016, pp. 519–523. [Online]. Available: http://doi.acm.org/10.1145/2971648.2971707

[46] H. Li, X. Liu, and Q. Mei, "Predicting smartphone battery life based on comprehensive and real-time usage data," *CoRR*, vol. abs/1801.04069, 2018. [Online]. Available: http://arxiv.org/abs/1801.04069

[47] S. L. Jones, D. Ferreira, S. Hosio, J. Goncalves, and V. Kostakos, "Revisitation analysis of smartphone app use," in *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ser. UbiComp '15. New York, NY, USA: ACM, 2015, pp. 1197–1208. [Online]. Available: http://doi.acm.org/10.1145/2750858.2807542

[48] V. Kostakos, D. Ferreira, J. Goncalves, and S. Hosio, "Modelling smartphone usage: A markov state transition model," in *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ser. UbiComp '16. New York, NY, USA: ACM, 2016, pp. 486–497. [Online]. Available: http://doi.acm.org/10.1145/2971648.2971669

[49] D. Ferreira, E. Ferreira, J. Goncalves, V. Kostakos, and A. K. Dey, "Revisiting human-battery interaction with an interactive battery

interface," in *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ser. UbiComp '13. New York, NY, USA: ACM, 2013, pp. 563–572. [Online]. Available: http://doi.acm.org/10.1145/2493432.2493465

[50] J. Kaasila, D. Ferreira, V. Kostakos, and T. Ojala, "Testdroid: Automated remote ui testing on android," in *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia*, ser. MUM '12. New York, NY, USA: ACM, 2012, pp. 28:1–28:4. [Online]. Available: http://doi.acm.org/10.1145/2406367.2406402

[51] H. Falaki, R. Mahajan, S. Kandula, D. Lymberopoulos, R. Govindan, and D. Estrin, "Diversity in smartphone usage," in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '10. New York, NY, USA: ACM, 2010, pp. 179–194. [Online]. Available: http://doi.acm.org/10.1145/1814433.1814453

[52] T. Feng, J. Yang, Z. Yan, E. M. Tapia, and W. Shi, "Tips: Context-aware implicit user identification using touch screen in uncontrolled environments," in *Proceedings of the 15th Workshop on Mobile Computing Systems and Applications*, ser. HotMobile '14. New York, NY, USA: ACM, 2014, pp. 9:1–9:6. [Online]. Available: http://doi.acm.org/10.1145/2565585.2565592

[53] H. Khan and U. Hengartner, "Towards application-centric implicit authentication on smartphones," in *Proceedings of the 15th Workshop on Mobile Computing Systems and Applications*, ser. HotMobile '14. New York, NY, USA: ACM, 2014, pp. 10:1–10:6. [Online]. Available: http://doi.acm.org/10.1145/2565585.2565590

[54] S. Mondal and P. Bours, "Does context matter for the performance of continuous authentication biometric systems? An empirical study on mobile device," in *International Conference of the Biometrics Special Interest Group (BIOSIG)*, 2015, pp. 1–5.

[55] R. Murmuria, A. Stavrou, D. Barbará, and D. Fleck, "Continuous authentication on mobile devices using power consumption, touch gestures and physical movement of users," in *International Workshop on Recent Advances in Intrusion Detection.* Springer, 2015, pp. 405–424.

[56] O. Riva, C. Qin, K. Strauss, and D. Lymberopoulos, "Progressive authentication: Deciding when to authenticate on mobile phones," in *Proc. of the 21st USENIX Conf. on Security Symp.*, ser. Security'12. Berkeley, CA, USA: USENIX Association, 2012, pp. 15–15. [Online]. Available: http://dl.acm.org/citation.cfm?id=2362793.2362808

[57] Y. Liu and A. Simpson, "Privacy-preserving targeted mobile advertising: requirements, design and a prototype implementation," *Software: Practice*

and Experience, vol. 46, no. 12, pp. 1657–1684. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/spe.2403

[58] L. Fridman, S. Weber, R. Greenstadt, and M. Kam, "Active authentication on mobile devices via stylometry, application usage, web browsing, and GPS location," *IEEE Systems Journal*, vol. 11, no. 2, pp. 513–521, 2017.

[59] A. Serwadda, V. V. Phoha, and Z. Wang, "Which verifiers work?: A benchmark evaluation of touch-based authentication algorithms." in *BTAS*. IEEE, 2013, pp. 1–8. [Online]. Available: http://dblp.uni-trier.de/db/conf/btas/btas2013.html#SerwaddaPW13

[60] M. Derawi, C. Nickel, P. Bours, and C. Busch, "Unobtrusive user-authentication on mobile phones using biometric gait recognition," in *Intelligent Inform. Hiding and Multimedia Signal Process. (IIH-MSP), 2010 6th Int. Conf.*, Oct. 2010, pp. 306–311.

[61] A. Primo, V. Phoha, R. Kumar, and A. Serwadda, "Context-aware active authentication using smartphone accelerometer measurements," in *Comput. Vision and Pattern Recognition Workshops, IEEE Conf.*, June 2014, pp. 98–105.

[62] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 580–587. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2014.81

[63] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[64] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[65] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European Conference on Computer Vision*. Springer, 2016, pp. 21–37.

[66] H. Chan and W. Bledsoe, "A man-machine facial recognition system: Some preliminary results," in *TR*, 1965.

[67] T. Sakai, T. Nagao, and T. Kanade, "Computer analysis and classification of photographs of human faces," *Seminar*, pp. 55–62, October 1973.

[68] S. Zafeiriou, C. Zhang, and Z. Zhang, "A survey on face detection in the wild," *Comput. Vis. Image Underst.*, vol. 138, no. C, pp. 1–24, Sep. 2015. [Online]. Available: http://dx.doi.org/10.1016/j.cviu.2015.03.015

[69] C. Zhang and Z. Zhang, "A survey of recent advances in face detection," Microsoft Research, Tech. Rep. MSR-TR-2010-66, June 2010.

[70] C. Huang, H. Ai, Y. Li, and S. Lao, "High-performance rotation invariant multiview face detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 4, pp. 671–686, April 2007.

[71] S. Li, L. Zhu, Z. Zhang, A. Blake, H. Zhang, and H. Shum, "Statistical learning of multi-view face detection," in *Computer Vision ECCV 2002*, ser. Lecture Notes in Computer Science, A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, Eds. Springer Berlin Heidelberg, 2002, vol. 2353, pp. 67–81. [Online]. Available: http://dx.doi.org/10.1007/3-540-47979-1_5

[72] S. Bileschi and B. Heisele, "Advances in component based face detection," in *Analysis and Modeling of Faces and Gestures, 2003. AMFG 2003. IEEE International Workshop on*, Oct 2003, pp. 149–156.

[73] M. Marin-Jimenez, A. Zisserman, M. Eichner, and V. Ferrari, "Detecting people looking at each other in videos," *International Journal of Computer Vision*, vol. 106, no. 3, pp. 282–296, 2014. [Online]. Available: http://dx.doi.org/10.1007/s11263-013-0655-7

[74] M. Mathias, R. Benenson, M. Pedersoli, and L. Van Gool, "Face detection without bells and whistles," in *ECCV*, 2014.

[75] S. Liao, A. K. Jain, and S. Z. Li, "A fast and accurate unconstrained face detector," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 211–223, Feb. 2016. [Online]. Available: http://dx.doi.org/10.1109/TPAMI.2015.2448075

[76] R. Ranjan, S. Sankaranarayanan, C. D. Castillo, and R. Chellappa, "An all-in-one convolutional neural network for face analysis," *CoRR*, vol. abs/1611.00851, 2016. [Online]. Available: http://arxiv.org/abs/1611.00851

[77] S. Yang, P. Luo, C. C. Loy, and X. Tang, "From facial parts responses to face detection: A deep learning approach," in *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[78] Z.-W. Huo, X. Yang, C. Xing, Y. Zhou, P. Hou, J. Lv, and X. Geng, "Deep age distribution learning for apparent age estimation," *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 722–729, 2016.

[79] E. Eidinger, R. Enbar, and T. Hassner, "Age and gender estimation of unfiltered faces," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 12, pp. 2170–2179, Dec 2014.

[80] G. Guo and G. Mu, "Joint estimation of age, gender and ethnicity: Cca vs. pls," in *2013 10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, April 2013, pp. 1–6.

[81] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, Dec 2015.

[82] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 91–99. [Online]. Available: http://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf

[83] U. Mahbub, S. Sarkar, and R. Chellappa, "Partial face detection in the mobile domain," *CoRR*, vol. abs/1704.02117, 2017. [Online]. Available: http://arxiv.org/abs/1704.02117

[84] H. Ding, H. Zhou, S. K. Zhou, and R. Chellappa, "A deep cascade network for unaligned face attribute classification," *CoRR*, vol. abs/1709.03851, 2017. [Online]. Available: http://arxiv.org/abs/1709.03851

[85] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, "Face recognition: A literature survey," *ACM Comput. Surv.*, vol. 35, no. 4, pp. 399–458, Dec. 2003. [Online]. Available: http://doi.acm.org/10.1145/954339.954342

[86] J. C. Chen, V. M. Patel, and R. Chellappa, "Unconstrained face verification using deep cnn features," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, March 2016, pp. 1–9.

[87] T. Ahonen, S. Member, A. Hadid, M. Pietikinen, and S. Member, "Face description with local binary patterns: Application to face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 2037–2041, 2006.

[88] S. Xie, S. Shan, X. Chen, and J. Chen, "Fusing local patterns of gabor magnitude and phase for face recognition," *IEEE Transactions on Image Processing*, vol. 19, no. 5, pp. 1349–1361, May 2010.

[89] Y.-C. Chen, V. M. Patel, P. J. Phillips, and R. Chellappa, "Dictionary-based face recognition from video," in *Proceedings of the 12th European Conference on Computer Vision - Volume Part VI*, ser. ECCV'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 766–779. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-33783-3_55

[90] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 1701–1708.

[91] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Proceedings of the 27th International Conference on Neural Information Processing Systems*, ser. NIPS'14. Cambridge, MA, USA: MIT Press, 2014, pp. 1988–1996. [Online]. Available: http://dl.acm.org/citation.cfm?id=2969033.2969049

[92] J. C. Chen, R. Ranjan, A. Kumar, C. H. Chen, V. M. Patel, and R. Chellappa, "An end-to-end system for unconstrained face verification with deep convolutional neural networks," in *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, Dec 2015, pp. 360–368.

[93] L. Lin, G. Wang, W. Zuo, F. Xiangchu, and L. Zhang, "Cross-domain visual matching via generalized similarity measure and feature learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PP, no. 99, pp. 1–1, 2016.

[94] Y. Meng, D. S. Wong, R. Schlegel, and L.-f. Kwok, *Touch Gestures Based Biometric Authentication Scheme for Touchscreen Mobile Phones.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 331–350. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-38519-3_21

[95] C. Bo, L. Zhang, X.-Y. Li, Q. Huang, and Y. Wang, "Silentsense: Silent user identification via touch and movement behavioral biometrics," in *Proceedings of the 19th Annual International Conference on Mobile Computing &#38; Networking*, ser. MobiCom '13. New York, NY, USA: ACM, 2013, pp. 187–190. [Online]. Available: http://doi.acm.org/10.1145/2500423.2504572

[96] H. Lu, A. Brush, B. Priyantha, A. Karlson, and J. Liu, "Speakersense: Energy efficient unobtrusive speaker identification on mobile phones," in *The Ninth International Conference on Pervasive Computing (Pervasive 2011)*, June 2011. [Online]. Available: https://www.microsoft.com/en-us/research/publication/speakersense-energy-efficient-unobtrusive-speaker-identification-on-mobile-phones/

[97] X. X. Yu Zheng, "Mining individual life pattern based on location history." IEEE, May 2009. [Online]. Available: https://www.microsoft.com/en-us/research/publication/mining-individual-life-pattern-based-on-location-history/

[98] S. Gambs, M.-O. Killijian, and M. N. n. del Prado Cortez, "Next place prediction using mobility markov chains," in *Proc. the First Workshop on Measurement, Privacy, and Mobility*, ser. MPM '12, 2012, pp. 3:1–3:6.

[99] Y. Zheng, Y. Chen, Q. Li, X. Xie, and W.-Y. Ma, "Understanding transportation modes based on gps data for web applications," *ACM Trans. Web*, vol. 4, no. 1, pp. 1:1–1:36, Jan. 2010. [Online]. Available: http://doi.acm.org/10.1145/1658373.1658374

[100] D. J. Patterson, L. Liao, D. Fox, and H. Kautz, *Inferring High-Level Behavior from Low-Level Sensors.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 73–89. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-39653-6_6

[101] W.-Y. M. Yu Zheng, Xing Xie, "Understanding mobility based on gps data." Association for Computing Machinery, Inc., Sep. 2008. [Online]. Available: https://www.microsoft.com/en-us/research/publication/understanding-mobility-based-on-gps-data/

[102] J. H. Manish Gupta, *Applications for Pattern Discovery Using Sequential Data Mining.* IGI Global, Jan. 2012. [Online]. Available: https://www.microsoft.com/en-us/research/publication/applications-for-pattern-discovery-using-sequential-data-mining/

[103] S. Gambs, M.-O. Killijian, and M. N. n. del Prado Cortez, "Show me how you move and i will tell you who you are," in *Proc. 3rd ACM SIGSPATIAL Int. Workshop on Security and Privacy in GIS and LBS*, ser. SPRINGL '10, 2010, pp. 34–41.

[104] W. Mathew, R. Raposo, and B. Martins, "Predicting future locations with hidden markov models," in *Proc. 2012 ACM Conf. Ubiquitous Computing*, ser. UbiComp '12, 2012, pp. 911–918.

[105] H. Cao and M. Lin, "Mining smartphone data for app usage prediction and recommendations: A survey," *Pervasive and Mobile Computing*, vol. 37, pp. 1 – 22, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1574119217300421

[106] A. Tongaonkar, S. Dai, A. Nucci, and D. Song, "Understanding mobile app usage patterns using in-app advertisements," in *Passive and Active Measurement*, M. Roughan and R. Chang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 63–72.

[107] A. Shema and D. E. Acuna, "Show me your app usage and i will tell who your close friends are: Predicting user's context from simple cellphone activity," in *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, ser. CHI EA '17. New York, NY, USA: ACM, 2017, pp. 2929–2935. [Online]. Available: http://doi.acm.org/10.1145/3027063.3053275

[108] K. Halunen and A. Evesti, "Context-aware systems and adaptive user authentication," in *Evolving Ambient Intelligence*, ser. Communications in Computer and Information Science, M. OGrady, H. Vahdat-Nejad, K.-H. Wolf, M. Dragone, J. Ye, C. Rcker, and G. OHare, Eds. Springer International Publishing, 2013, vol. 413, pp. 240–251. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-04406-4\protect\

unhbox\voidb@x\bgroup\def,{_}\let\futurelet\@let@token\let\protect\relax\protect\edefcmr{cmtt}\protect\xdef\OT1/cmr/m/it/12{\OT1/cmr/m/n/12}\OT1/cmr/m/it/12\size@update\enc@update_\egroup24

[109] A. Ross and A. Jain, "Information fusion in biometrics," *Pattern Recogn. Lett.*, vol. 24, no. 13, pp. 2115–2125, Sept. 2003. [Online]. Available: http://dx.doi.org/10.1016/S0167-8655(03)00079-5

[110] I. G. Damousis and S. Argyropoulos, "Four machine learning algorithms for biometrics fusion: A comparative study," *Applied Computational Intelligence and Soft Computing*, vol. 2012, no. 242401, 2012.

[111] N. Neverova, C. Wolf, G. Lacey, L. Fridman, D. Chandra, B. Barbello, and G. W. Taylor, "Learning human identity from motion patterns," *CoRR*, vol. abs/1511.03908, 2015. [Online]. Available: http://arxiv.org/abs/1511.03908

[112] N. Eagle and A. (Sandy) Pentland, "Reality mining: Sensing complex social systems," *Personal Ubiquitous Comput.*, vol. 10, no. 4, pp. 255–268, Mar. 2006. [Online]. Available: http://dx.doi.org/10.1007/s00779-005-0046-3

[113] C. Shepard, A. Rahmati, C. Tossell, L. Zhong, and P. Kortum, "Livelab: Measuring wireless networks and smartphone users in the field," *ACM SIGMETRICS Performance Evaluation Review*, vol. 38, no. 3, pp. 15–20, jan 2011.

[114] Y. Zheng, X. Xie, and W.-Y. Ma, "Geolife: A collaborative social networking service among user, location and trajectory." *IEEE Data Eng. Bull.*, vol. 33, no. 2, pp. 32–39, 2010. [Online]. Available: http://dblp.uni-trier.de/db/journals/debu/debu33.html#ZhengXM10

[115] H. F. Yu Zheng, *Geolife GPS trajectory dataset - User Guide*, Jul. 2011. [Online]. Available: https://www.microsoft.com/en-us/research/publication/geolife-gps-trajectory-dataset-user-guide/

[116] D. Ferreira, V. Kostakos, A. R. Beresford, J. Lindqvist, and A. K. Dey, "Securacy: An empirical investigation of android applications' network usage, privacy and security," in *ACM Conference on Security & Privacy in Wireless and Mobile Networks*, ser. WiSec '15. ACM, 2015, pp. 11:1–11:11.

[117] R. Ranjan, V. M. Patel, and R. Chellappa, "A deep pyramid deformable part model for face detection," in *Biometrics Theory, Applications and Systems (BTAS), 2015 IEEE 7th Int. Conf.*, 2015, pp. 1–8.

[118] A. Kumar, R. Ranjan, V. M. Patel, and R. Chellappa, "Face alignment by local deep descriptor regression," *CoRR*, vol. abs/1601.07950, 2016. [Online]. Available: http://arxiv.org/abs/1601.07950

[119] T. Ojala, M. Pietikainen, and T. Maenpaa, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” *IEEE Trans. Pattern Anal. and Mach. Intell.*, vol. 24, no. 7, pp. 971–987, July 2002.

[120] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances Neural Inform. Process. Syst. 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[121] J. Chen, V. M. Patel, and R. Chellappa, “Unconstrained face verification using deep CNN features,” *CoRR*, vol. abs/1508.01722, 2015. [Online]. Available: http://arxiv.org/abs/1508.01722

[122] D. Yi, Z. Lei, S. Liao, and S. Z. Li, “Learning face representation from scratch,” *CoRR*, vol. abs/1411.7923, 2014. [Online]. Available: http://arxiv.org/abs/1411.7923

[123] L. Breiman, “Random forests,” *Mach. Learning*, vol. 45, no. 1, pp. 5–32. [Online]. Available: http://dx.doi.org/10.1023/A:1010933404324

[124] J. H. Friedman, “Greedy function approximation: A gradient boosting machine,” *Annals of Statistics*, vol. 29, pp. 1189–1232, 2000.

[125] J. Sander, M. Ester, H.-P. Kriegel, and X. Xu, “Density-based clustering in spatial databases: The algorithm gdbscan and its applications,” *Data Min. Knowl. Discov.*, vol. 2, no. 2, pp. 169–194, June 1998. [Online]. Available: http://dx.doi.org/10.1023/A:1009745219419

[126] R. Ranjan, S. Sankaranarayanan, C. D. Castillo, and R. Chellappa, “An all-in-one convolutional neural network for face analysis,” in *2017 12th IEEE International Conference on Automatic Face Gesture Recognition (FG 2017)*, May 2017, pp. 17–24.

[127] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05) - Volume 1 - Volume 01*, ser. CVPR ’05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 886–893. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2005.177

[128] C. J. Burges, “A tutorial on support vector machines for pattern recognition,” vol. 2, January 1998, pp. 121–167. [Online]. Available: https://www.microsoft.com/en-us/research/publication/a-tutorial-on-support-vector-machines-for-pattern-recognition/

[129] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *CoRR*, vol. abs/1409.1556, 2014.

[130] K. E. A. van de Sande, J. Uijlings, T. Gevers, and A. Smeulders, "Segmentation as Selective Search for Object Recognition," in *ICCV*, 2011. [Online]. Available: http://www.huppelen.nl/publications/ICCV2011SelectiveSearch.pdf

[131] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 770–778.

[132] yeephycho, "Tensorflow face detector," https://github.com/yeephycho/tensorflow-face-detection, 2017.

[133] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[134] U. Mahbub, S. Sarkar, and R. Chellappa, "Pooling facial segments to face: The shallow and deep ends," in *2017 12th IEEE International Conference on Automatic Face Gesture Recognition (FG 2017)*, May 2017, pp. 634–641.

[135] M. Lin, Q. Chen, and S. Yan, "Network in network," *CoRR*, vol. abs/1312.4400, 2013. [Online]. Available: http://arxiv.org/abs/1312.4400

[136] R. Cappelli, D. Maio, and D. Maltoni, "Combining fingerprint classifiers," in *Proceedings of the First International Workshop on Multiple Classifier Systems*, ser. MCS '00. London, UK, UK: Springer-Verlag, 2000, pp. 351–361. [Online]. Available: http://dl.acm.org/citation.cfm?id=648054.746359

[137] A. Jain, K. Nandakumar, and A. Ross, "Score normalization in multimodal biometric systems," *Pattern Recogn.*, vol. 38, no. 12, pp. 2270–2285, Dec. 2005. [Online]. Available: http://dx.doi.org/10.1016/j.patcog.2005.01.012

[138] K. Tumer and J. Ghosh, "Linear and order statistics combiners for pattern classification," *CoRR*, vol. cs.NE/9905012, 1999. [Online]. Available: http://arxiv.org/abs/cs.NE/9905012

[139] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[140] F. Chollet *et al.*, "Keras," urlhttps://github.com/keras-team/keras, 2015.

[141] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. J. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Józefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. G. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. A. Tucker, V. Vanhoucke, V. Vasudevan, F. B. Viégas, O. Vinyals, P. Warden,

M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *CoRR*, vol. abs/1603.04467, 2016. [Online]. Available: http://arxiv.org/abs/1603.04467

[142] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," *CoRR*, vol. abs/1604.03539, 2016. [Online]. Available: http://arxiv.org/abs/1604.03539

[143] L. R. Rabiner, "Readings in speech recognition," A. Waibel and K.-F. Lee, Eds. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1990, ch. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, pp. 267–296. [Online]. Available: http://dl.acm.org/citation.cfm?id=108235.108253

[144] S. Gupta, A. Buriro, and B. Crispo, "Demystifying authentication concepts in smartphones: Ways and types to secure access," *Mobile Information Systems*, vol. 2018, p. 16 pages, 2018. [Online]. Available: https://www.hindawi.com/journals/misy/2018/2649598/

[145] P. Samangouei, V. M. Patel, and R. Chellappa, "Facial attributes for active authentication on mobile devices," *Image and Vision Computing*, vol. 58, pp. 181 – 192, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0262885616300828

[146] A. Hadid, J. Heikkila, O. Silven, and M. Pietikainen, "Face and eye detection for person authentication in mobile phones," in *Distributed Smart Cameras, 2007. ICDSC '07. First ACM/IEEE International Conference on*, Sept 2007, pp. 101–108.

[147] H. Zhang, V. M. Patel, and R. Chellappa, "Robust multimodal recognition via multitask multivariate low-rank representations," in *IEEE Int. Conf. Automat. Face and Gesture Recogn.* IEEE, 2015.

[148] N. Neverova, C. Wolf, G. Lacey, L. Fridman, D. Chandra, B. Barbello, and G. Taylor, "Learning human identity from motion patterns," *IEEE Access*, vol. 4, pp. 1810–1820, 2016.

[149] A. Mahfouz, T. M. Mahmoud, and A. S. Eldin, "A survey on behavioral biometric authentication on smartphones," *Journal of Information Security and Applications*, vol. 37, pp. 28 – 37, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2214212617302417

[150] W.-H. Lee, X. Liu, Y. Shen, H. Jin, and R. B. Lee, "Secure pick up: Implicit authentication when you start using the smartphone," in *Proceedings of the 22Nd ACM on Symposium on Access Control Models and Technologies*, ser. SACMAT '17 Abstracts. New York, NY, USA: ACM, 2017, pp. 67–78. [Online]. Available: http://doi.acm.org/10.1145/3078861.3078870

[151] N. van Berkel, C. Luo, T. Anagnostopoulos, D. Ferreira, J. Goncalves, S. Hosio, and V. Kostakos, "A systematic assessment of smartphone usage gaps," in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ser. CHI '16. New York, NY, USA: ACM, 2016, pp. 4711–4721. [Online]. Available: http://doi.acm.org/10.1145/2858036.2858348

[152] K. Church, D. Ferreira, N. Banovic, and K. Lyons, "Understanding the challenges of mobile phone usage data," in *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*, ser. MobileHCI '15. New York, NY, USA: ACM, 2015, pp. 504–514. [Online]. Available: http://doi.acm.org/10.1145/2785830.2785891

[153] V. I. Levenshtein, "Binary Codes Capable of Correcting Deletions, Insertions and Reversals," *Soviet Physics Doklady*, vol. 10, p. 707, Feb. 1966.

[154] G. Navarro, "A guided tour to approximate string matching," *ACM Comput. Surv.*, vol. 33, no. 1, pp. 31–88, Mar. 2001. [Online]. Available: http://doi.acm.org/10.1145/375360.375365

[155] P. Rodrguez, G. Cucurull, J. M. Gonfaus, F. X. Roca, and J. Gonzlez, "Age and gender recognition in the wild with deep attention," *Pattern Recognition*, vol. 72, pp. 563 – 571, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320317302546

[156] B. Zhao, X. Wu, J. Feng, Q. Peng, and S. Yan, "Diversified visual attention networks for fine-grained object classification," *IEEE Transactions on Multimedia*, vol. 19, no. 6, pp. 1245–1256, June 2017.

[157] T. Sun, L. Sun, and D.-Y. Yeung, "Fine-grained categorization via cnn-based automatic extraction and integration of object-level and part-level features," *Image Vision Comput.*, vol. 64, no. C, pp. 47–66, Aug. 2017. [Online]. Available: https://doi.org/10.1016/j.imavis.2017.06.003

[158] H. Zheng, J. Fu, T. Mei, and J. Luo, "Learning multi-attention convolutional neural network for fine-grained image recognition," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, pp. 5219–5227.

[159] L. Yang, P. Luo, C. C. Loy, and X. Tang, "A large-scale car dataset for fine-grained categorization and verification," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 3973–3981.

[160] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, Nov 2017.

[161] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra,

S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015. [Online]. Available: http://dx.doi.org/10.1038/nature14236

[162] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," *IEEE Transactions on Neural Networks*, vol. 16, pp. 285–286, 1998.