



**REPUBLIC OF TURKEY
ADANA ALPARSLAN TÜRKES SCIENCE AND TECHNOLOGY
UNIVERSITY**

**INSTITUTE OF GRADUATE SCHOOL
DEPARTMENT OF COMPUTER ENGINEERING**

**IDENTICAL USER MATCHING ON CROSS ONLINE SOCIAL
NETWORKS: A TWO-STEP APPROACH BASED ON FACE
RECOGNITION**

**ÖMER AYANA
MASTER OF SCIENCE**



**REPUBLIC OF TURKEY
ADANA ALPARSLAN TÜRKER SCIENCE AND TECHNOLOGY
UNIVERSITY**

**INSTITUTE OF GRADUATE SCHOOL
DEPARTMENT OF COMPUTER ENGINEERING**

**IDENTICAL USER MATCHING ON CROSS ONLINE SOCIAL
NETWORKS: A TWO-STEP APPROACH BASED ON FACE
RECOGNITION**

**ÖMER AYANA
MASTER OF SCIENCE**

**SUPERVISOR
Assoc. Prof. Dr. ALİ İNAN**

ADANA 2022

I hereby declare that all information in this thesis has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all information that is not original to this work.

[Signature]

Ömer AYANA



ABSTRACT

IDENTICAL USER MATCHING ON CROSS ONLINE SOCIAL NETWORKS: A TWO-STEP APPROACH BASED ON FACE RECOGNITION

Ömer AYANA

Department of Computer Engineering

Supervisor: Assoc. Prof. Dr. Ali İNAN

August 2022, 112 pages

As the number of online social networks (OSNs) increases and the services distinct OSNs offer diversify, the number of individuals with accounts in more than one OSN increases. Consequently, the data of such users is spread over different OSNs. The problem of matching identical profiles on cross OSNs tries to unify OSN accounts of the same individual on different OSNs. Matching identical profiles not only allows having more information about a matched user, but also enables us to better understand users' behavior. Existing studies on matching identical profiles rely on profile attributes (such as age, gender, e-mail and education), content shared by users (such as tweets, photos, location check-ins) and links. However, relying on profile attributes is not reliable because any malicious user can easily spoof profile attributes to clone an OSN account. Furthermore, many users hide their profile attribute values due to privacy concerns, which bears the matching effort fruitless. Regarding user generated content, short and dirty texts within tweets are not easily processed, and location check-ins are much less frequently shared over many OSNs except Foursquare and alike.

In this thesis, we propose a solution that relies on shared photos and network structure. Since face biometrics are quite personal and more distinctive features than any other alternative OSN feature, our solution provides an accurate and reliable solution towards matching identical users. Our novel solution has two-phases: the search phase and the match phase. The search phase inspects all photos shared by a user, extracts faces, identifies the profile owner's faces automatically and generates a face vector signature from a sample of these faces. Using signatures reduces algorithmic complexity of the process and also makes it more reliable. In the

search phase, pairs of cross OSN accounts are compared over their signatures. Given an input source profile, the search phase acts as a filter to limit potential matches in the destination OSN to a small set of candidates. Subsequently, in the match phase, a ranked list of identical profiles are compiled using the rate of overlap between the links of the matched OSN profiles. Since there are many links of each account, we restrict the match phase to only profile photos of the link.

Experimental evaluation is performed over 3.000 real-world, labelled OSN accounts from Facebook and Instagram. The results indicate that in the search phase, 89% of the given input source profiles are located over the destination OSN. 58% of them are in top-1 rank, 76% of them are in top-2 rank, and 87% of them are in top-4 rank. Moreover, the match phase matches users with an f-score of $66\% \pm 4$. Finally, it is shown that the proposed search phase provides faster and more accurate matching than existing solutions.

Keywords: user profile matching, online social networks, face feature vectors, face recognition, user links, rule-based matching.

ÖZET

ÇEVİRİMİÇİ SOSYAL AĞLARDA ÖZDEŞ KULLANICI EŞLEŞTİRME: YÜZ TANIMAYA DAYALI İKİ ADIMLI YAKLAŞIM

Ömer AYANA

Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Ali İNAN

Ağustos 2022, 112 sayfa

Çevrimiçi sosyal ağların (OSN) sayısı arttıkça ve farklı OSN'lerin sunduğu hizmetler çeşitlendikçe, birden fazla OSN'de hesabı olan bireylerin sayısı artmakta. Sonuç olarak, bu tür kullanıcıların verileri farklı OSN'lere yayılır. Çapraz OSN'lerde aynı profilleri eşleştirme problemi, aynı kişinin OSN hesaplarını farklı OSN'lerde birleştirmeye çalışır. Aynı profilleri eşleştirmek, yalnızca eşleşen bir kullanıcı hakkında daha fazla bilgiye sahip olmamızı sağlamakla kalmaz, aynı zamanda kullanıcıların davranışlarını daha iyi anlamamızı sağlar. Aynı profilleri eşleştirmeye yönelik mevcut çalışmalar, profil özelliklerine (yaş, cinsiyet, e-posta ve eğitim gibi), kullanıcılar tarafından paylaşılan içeriğe (tweet'ler, fotoğraflar, konum kontrolleri gibi) ve bağlantılara dayanmaktadır. Ancak, herhangi bir kötü niyetli kullanıcı bir OSN hesabını klonlamak için profil özniteliklerini kolayca taklit edebileceğinden, profil özniteliklerine güvenmek güvenilir değildir. Ayrıca, birçok kullanıcı, eşleştirme çabasını sonuçsuz bırakan profil öznitelik değerlerini gizlilik endişeleri nedeniyle gizler. Kullanıcı tarafından oluşturulan içerikle ilgili olarak, tweet'lerdeki kısa ve kirli metinler kolayca işlenmez ve konum kontrolleri Foursquare ve benzerleri dışında birçok OSN'de çok daha az sıklıkla paylaşılır.

Bu tezde, paylaşılan fotoğraflara ve ağ yapısına dayanan bir çözüm öneriyoruz. Yüz biyometrisi, diğer tüm alternatif OSN özelliklerinden oldukça kişisel ve daha ayırt edici özellikler olduğundan, çözümümüz aynı kullanıcıları eşleştirmeye yönelik doğru ve güvenilir bir çözüm sunar. Yeni çözümümüzün iki aşaması vardır: arama aşaması ve eşleşme aşaması. Arama aşaması, bir kullanıcı tarafından paylaşılan tüm fotoğrafları inceler, yüzleri çıkarır,

profil sahibinin yüzlerini otomatik olarak belirler ve bu yüzlerin örneklerinden bir yüz vektör imzası oluşturur. İmza kullanmak, işlemin algoritmik karmaşıklığını azaltır ve daha güvenilir hale getirir. Arama aşamasında, çapraz OSN hesap çiftleri, imzaları üzerinden karşılaştırılır. Bir girdi kaynağı profili verildiğinde, arama aşaması, hedef OSN'deki potansiyel eşleşmeleri küçük bir aday grubuyla sınırlamak için bir filtre görevi görür. Daha sonra, eşleşme aşamasında, eşleşen OSN profillerinin bağlantıları arasındaki örtüşme oranı kullanılarak sıralı bir özdeş profil listesi derlenir. Her hesabın çok sayıda bağlantısı olduğundan, eşleşme aşamasını yalnızca bağlantının profil fotoğraflarıyla sınırlandırıyoruz.

Facebook ve Instagram'dan 3.000 gerçek, etiketli OSN hesabı üzerinden deneysel değerlendirme yapılmıştır. Sonuçlar, arama aşamasında, verilen giriş kaynağı profillerinin %89'unun hedef OSN üzerinde bulunduğunu göstermektedir. Bunların %58'i ilk 1'de, %76'sı ilk 2'de ve %87'si ilk 4'te yer almaktadır. Ayrıca, eşleşme aşaması, kullanıcıları ± 4 f-skoru ile eşleştirir. Son olarak, önerilen arama aşamasının mevcut çözümlere göre daha hızlı ve daha doğru eşleştirme sağladığı gösterilmiştir.

Anahtar Kelimeler: kullanıcı profili eşleştirme, çevrimiçi sosyal ağlar, yüz özellik vektörü, yüz tanıma, kullanıcı bağlantıları, kural tabanlı eşleştirme.

ACKNOWLEDGEMENTS

First of all, I would like to express my gratitude to my valuable supervisor Assoc. Prof. Dr. Ali İNAN, who helped me come up with this thesis and guided me at every step with all his experience and knowledge. I just want to let you know that it would have been very difficult for me to overcome the thesis without the generous information and feedback he always gave for the betterment of the thesis. I would like to thank my Supervisor for his supports.

Another thanks to my friends Deniz Furkan KANBAK, Ömer Can TOLUN, Dilara NİHADİOĞLU, Rahmi KAVAK, and Gülizar Gizem TOLUN for their intellectual and motivational support during the implementation and writing of my thesis.

My last thank you is to my mother Sevim AYANA and my father Zülfü AYANA, who have always been by my side, loving me unconditionally and supporting me in my success, for all their supports.

TABLE OF CONTENTS

ABSTRACT	iii
ÖZET	v
ACKNOWLEDGEMENTS.....	vii
TABLE OF CONTENTS.....	viii
LIST OF FIGURES	xi
LIST OF TABLES	xii
NOMENCLATURE	xiii
1. INTRODUCTION.....	14
1.1. Brief Survey Of Existing Studies On Profile Matching	16
1.1.1. Profile Attribute-Based Approaches	16
1.1.2. Content-Based Approaches	17
1.1.3. Network-Based Approaches.....	18
1.1.4. Hybrid Approach	18
1.2. Outline Of The Proposed Solution	19
1.3. Outline Of The Problem Definition	21
2. LITERATURE REVIEW	24
2.1. Profile Attribute-Based Approach	24
2.2. Content-Based Approach.....	25
2.3. Network-Based Approach.....	27
2.4. Hybrid Approach.....	28
2.5. Studies Made With Profile Photos And Faces In Posts.....	29
2.6. Studies Made With Profile Photos And Faces In Posts To Match Identical Profiles	30
3. MATERIALS AND METHODS	33
3.1. Face Detection Algorithms.....	33
3.1.1. You Only Look Once version 3 (YOLOv3)	33
3.1.2. Deep Neural Network (DNN).....	35
3.1.3. Histogram Of Oriented Gradients (HoG).....	35
3.1.4. Cascade Classifier	36

3.1.5.	MultiTask Convolutional Neural Network (MTCNN)	37
3.2.	Face Recognition Algorithms	38
3.2.1.	Facenet	40
3.2.2.	Vggface2	41
3.2.3.	Dlib	41
3.3.	Clustering Algorithms	42
3.3.1.	k-Means	42
3.3.2.	Agglomerative Nesting (AGNES)	43
3.4.	Datasets	44
3.4.1.	Dlib Dataset	44
3.4.2.	Vggface Dataset	45
3.4.3.	AboutMe Dataset	46
3.5.	Proposed Solution	46
3.5.1.	Outline Of The Search Phase	46
3.5.2.	Outline Of The Match Phase	49
3.5.3.	Determination Of The Threshold For Face Recognition	52
3.5.4.	Profile Owner Identification	54
3.5.5.	Determination Of The Face Recognition Algorithm And Matching	56
3.5.6.	Search Phase And Signature Generation	59
3.5.7.	Match Phase	61
4.	RESULTS AND DISCUSSIONS	64
4.1.	Dataset Collection	64
4.2.	Determination Of The Face Detection Algorithm	66
4.3.	Determination Of The FV Threshold For FR Models	69
4.4.	Determination Of The Profile Owner	72
4.5.	Testing Alternative Approaches For The Match Phase	73
4.5.1.	Under Constant $ N $, Varying t	73
4.5.2.	Under Constant t , Varying $ N $	75
4.6.	Testing Signature Generation Approaches For FR	77
4.7.	Face Detection On OSN Data And Determination Of Profile Owner's Vectors	81
4.8.	Overall Results On Real-World OSN Data Using Only The Search Phase	84
4.8.1.	Number Of Candidate Identical Profiles	85

4.8.2. Our Model Versus SVM.....	86
4.9. Determination Of <i>thrLink</i> And Overall Results On Real-World OSN Data After Match Phase	87
5. CONCLUSIONS	89
6. RECOMMENDATIONS.....	91
REFERENCES	92
CURRICULUM VITAE.....	112



LIST OF FIGURES

Figure 1.1. An example of identical user matching on cross OSNs.....	22
Figure 3.1. Structure detail of YOLOv3.It uses Darknet-53 as the backbone network and uses three scale predictions	35
Figure 3.2. Haar-like features.....	37
Figure 3.3. Overall Structure of MTCNN	38
Figure 3.4. General steps of face recognition.....	39
Figure 3.5. Triplet Loss function.....	40
Figure 3.6. K-Means clustering example	43
Figure 3.7. Agglomerative (AGNES) clustering example	44
Figure 3.8. Examples of images in different poses	45
Figure 3.9. Some example images from the Vggface dataset	45
Figure 3.10. Proposed searching model structure based on user-shared photo.....	47
Figure 3.11. Signature Generator	48
Figure 3.12. Proposed matching model structure based on profile photos of links	50
Figure 3.13. Fetch Links	51
Figure 4.1. Examples of different face detection scenarios	67
Figure 4.2. Facenet scenario with 100000 k and euclidean metric	69
Figure 4.3. Experiment results of 3 different approaches for face recognition by f-score and execution time ($ N $ constant).....	75
Figure 4.4. Experiment results of 3 different approaches for face recognition by f-score and execution time (t constant)	76
Figure 4.5. Distance distributions with Vggface2 ($t = 8$) (green dots show calculated distances between two identical signatures, red dots show the distances between two non-identical signature pairs)	77
Figure 4.6. Face recognition results with generated signatures	81
Figure 4.7. Account numbers by vector number of profile holders	83
Figure 4.8. Average number of matched identical instagram profiles for each facebook profile	85
Figure 4.9. Performance of finding identical profiles	86
Figure 4.10. Results of determining thrLink	88
Figure 4.11. Results of hybrid model.....	88

LIST OF TABLES

Table 1.1. Our notations	23
Table 3.1. Details of FR models	40
Table 4.1. Detailed information about the dataset created	65
Table 4.2. Face detection results	68
Table 4.3. Face verification threshold for Dlib with original image size	70
Table 4.4. Face verification threshold for Dlib with 80x80 image size	70
Table 4.5. Face verification threshold for Dlib with 120x120 image size	71
Table 4.6. Face verification threshold for Facenet model	71
Table 4.7. Face verification threshold for Vggface model	71
Table 4.8. Profile owner determination results.....	72
Table 4.9. Verification treshold and face recognition results for Dlib with original image size	78
Table 4.10. Verification treshold and face recognition results for Dlib with 80x80 image size	79
Table 4.11. Verification treshold and face recognition results for Dlib with 120x120 image size.....	79
Table 4.12. Verification treshold and face recognition results for Facenet.....	80
Table 4.13. Verification treshold and face recognition results for Vggface.....	80
Table 4.14. Vectorization results made with Dlib and Vgfface on 10 profiles	82
Table 4.15. Number of faces and vectors detected on OSN accounts	82
Table 4.16. Model and parameters used	84

NOMENCLATURE

OSN	: Online Social Network
FD	: Face Detection
FR	: Face Recognition
FV	: Face Verification
CA	: Clustering Algorithm



1. INTRODUCTION

Online social networks (OSNs) are virtual communication environments that allow real users to communicate with each other independent of location, such that users get informed about current events and share their ideas, and provide content about their interests (Rathore, Sharma, Loia, Jeong, & Park, 2017; Yassein, Aljawarneh, & Wahsheh, 2019). According to the analysis (The Global State of Digital in July 2022, 2022), as of July 2022, there are currently 4.70 billion social media users around the world, and each user spends 2 hours and 29 minutes on social media daily. The first reason for users to use social media is to stay in touch with their friends by 47.6%, to fill their spare time with 36.3%, and to read news stories by 34.8%. Based on the number of monthly active users, as of January 2022, the most popular OSNs are Facebook (2.9 billion users), Instagram (1.5 billion users) and Tiktok (1 billion users) (Statista, 2022).

With such large numbers of active users come along a plethora of OSN shares in the form text, photo, video and location posts. OSN platforms are rich environments for collecting data on human interaction in ready to consume, digital form. Therefore, OSNs provide rich and abundant ground of data for a wide variety of research subjects. For example, in the field of psychology: depressed OSNs users can be detected based on their posts (Islam et al., 2018; Tadesse, Lin, Xu, & Yang, 2019; Yoon, Kleinman, Mertz, & Brannick, 2019), in social analysis research: fake news can be identified accurately (Tschitschek, Singla, Gomez Rodriguez, Merchant, & Krause, 2018), credibility of users can be quantified (Abbasi & Liu, 2013), general sentiment analyses on popular subjects can be performed based on shared comments (Marcec & Likic, 2021), for marketing: patterns of consumer decision making processes can be examined (Boon-Long & Wongsurawat, 2015; Karpinska-Krakowiak, 2016; Toor, Husnain, & Hussain, 2017), or communities on social media can be detected automatically (Bedi & Sharma, 2016) for the network analysis.

On the other side, OSN data may also be utilized adversely. Notice that OSN profiles of individuals contain basic personal data like age, gender, birthday, education. Additionally, viewers of a profile page may learn about talents, interests, professional history or friends and family of the respective profile owner. In most cases, data posted on an OSN are public by default, which means the data is visible to anyone. Various studies have inspected potential

harm in careless use of OSNs including but not limited to the following: cyberbullying (Dinakar, Jones, Havasi, Lieberman, & Picard, 2012), cyberstalking (Frommholz et al., 2016), faking or cloning an account (Khaled, El-Tazi, & Mokhtar, 2019; Xiao, Freeman, & Hwa, 2015). These problems can have significant effects on both individuals and societies. According to the research (Aboujaoude, Savage, Starcevic, & Salame, 2015), 20-40 percent of young people are affected psychologically by being exposed to cyberbullying on social media. Fake accounts can spread negative information to the society with the news they shared and even affect the election results (Shu et al., 2017). Cloned accounts can sell fake products on behalf of the real company or make the person look bad to their friends by posting inappropriate content about them (Devmane & Rana, 2014). In addition, depending on the content that users share, their political opinions or the organizations they support may also emerge (Preotiuc-Pietro, Hopkins, Liu, & Ungar, 2017).

One means of protection against such attacks is to actively use privacy settings provided by the OSN platforms. Privacy settings let profile owners to determine who is allowed to observe the shared data. However, hiding data is not fail-safe since there exists accurate inference attacks based on machine learning algorithms. Many user profile attributes are in danger such as age (Dey, Tang, Ross, & Saxena, 2012), education (Mislove, Viswanath, Gummadi, & Druschel, 2010), location (M. Li et al., 2014), occupation (Gong & Liu, 2018). More complex attack scenarios predict the next destination of a user (Xue, Wu, Li, Hong, & Zhou, 2017), or predict users' age, gender, and education based on their previous locations (H. Li, Zhu, Du, Liang, & Shen, 2018).

Widespread popularity of OSNs have eventually brought together specialization in OSN services (Peng et al., 2018). Typically, users follow current news (political, economic, etc.) on Twitter, share pictures of special moments with their friends on Instagram, announce their professional updates on LinkedIn and tag places they visit on Foursquare (Y. Li, Zhang, Peng, Yin, & Xu, 2018). A single user now has distinct accounts on different OSNs (Global Social Media Account Ownership 2018 | Statista, 2018; The Global State of Digital in July 2022, 2022). As a result, a single OSN user's shares and connections are scattered over multiple OSNs. Combining these distinct profiles of an individual is critical to accurately analyzing the success of any OSN inference task. This problem is called *matching/recognizing identical profiles* (Y.

Li et al., 2018). It is important to note that a user may create multiple profiles on the same OSN in which case, the problem converts to identifying duplicate/clone profiles.

1.1. Brief Survey Of Existing Studies On Profile Matching

There is a prohibitively large number of profiles on prominent OSNs, the problem of matching identical profiles is a challenging task worth studying independently. The basic approach in the literature involves comparing profile data contained in the profiles to be mis/matched. However, studies differ in which component(s) of an input pair of profiles are considered. Existing studies can be divided into 4 categories according to the profile information and content used to find identical profiles: (i) profile attribute-based, (ii) content-based, (iii) network-based and (iv) hybrid (L. Wang, Hu, Zhang, & Cao, 2019; Zhou, Liang, Zhang, & Ma, 2016). Next, these four categories are reviewed briefly. An extended review of the literature is given in Chapter 2.

1.1.1. Profile Attribute-Based Approaches

In profile attribute-based approaches, users' personal information such as username, age, education, location, birthday and e-mail are taken into account. Perito et al. (Perito, Castelluccia, Kaafar, & Manils, 2011) propose a model that performs user matching only through the username. The presented model measures the probability of belonging to the same person by evaluating the uniqueness value of two given usernames, and calculates an entropy for given usernames.. The results show that high entropy usernames can be chosen by the same user and therefore increase the probability of matching, while low entropy usernames are chosen by more than one user and therefore are not suitable for finding identical profiles.

While username is an easy attribute to collect and process to find identical profiles, a user can use different usernames to stay anonymous on OSNs, or by coincidence, the same username can be used by distinct users. As a solution, to find identical profiles across OSNs, Iofciu et al. propose a model based on users' tags in addition to usernames. The tags here are the additional notes that users share to better express their shared pictures, videos, and location. In order to calculate the similarity of the tags of users, each user is modeled as a vector using BM25 and TF-IDF methods and these vectors are created from the BM25 and TF-IDF values of the tags (Iofciu, Fankhauser, Abel, & Bischoff, 2011).

Vosecky et al. develop another model based on profile attributes (Vosecky, Hong, & Shen, 2009). Here, each profile is represented with a feature vector of sensitive information. 6 profile

attributes are selected: nickname, birthday, gender, university, high school, home town. The difference of this study from the other studies is that a weight is determined according to the importance of each feature and it is decided how much it would affect the result. The matching rule of the two profiles is rule based – similarity beyond a threshold implies matched profiles.

While profile attributes are very powerful features for recognizing users, some users do not fill most attributes or share incorrect values on purpose to obscure their identity, and users may define different values in different OSNs. For example, a user can define a personal e-mail address on Facebook for the e-mail attribute, and a work e-mail address on LinkedIn. Besides these, large OSNs can have many users with the same attribute values, and public attributes can be unreliable because they can easily be spoofed by adversaries. In addition, since the tags are short texts and the same tags may be preferred by too many users, it may be insufficient to find identical profiles.

1.1.2. Content-Based Approaches

Attackers can also consider both photo/video and text contents shared by the users. Li et al propose a User Generated Content (UGC) model based on the assumption that a user would behave the same in different OSNs (Y. Li et al., 2018). In other words, they aim to find identical profiles by assuming that a user shared the location of the place they visit or a photo they take on a different OSN account. Based on this assumption, they create spatial, temporal and content-based features for each profile. They use different similarity measurement techniques for each feature.

Similar to (Y. Li et al., 2018), Goga et al. also propose a content-based model that includes geo-location of users' posts, the timestamps of users' posts, and the user's writing style. In order to compare the performance, they repeat the previously suggested username-based identical profile matching model. While the results show that the geo-location feature is a decisive feature for finding identical profiles, the proposed model cannot improve upon the username-based work (Goga, Lei, et al., 2013).

Although the location patterns of the users may be preferable to detect a user, it is difficult to obtain location information in each profile due to sparsity. Only a small number of users share their location information. Finally, the writing style solution makes it difficult to detect identical

profiles with the feature obtained from short sentences on OSNs such as Twitter due to length limits imposed.

1.1.3. Network-Based Approaches

Network-based approaches locate identical profiles based on similarity over connections (or friendship links) that a user creates with other users on the same OSN. Zhou et al. propose a semi-supervised Friend Relationship-Based User Identification (FRUI) model. FRUI first determines known identical users (seed users) and obtained features on their similarities. They then perform information propagation on the network using these seeds to find new identical users on the network. Here the similarities between unmapped users are calculated according to the mapped users. Finally, FRUI model calculates a match degree for all candidate user matched pairs (UMPs), and only UMPs with top ranks are considered as identical users (Zhou et al., 2016).

Li et al. use machine learning algorithms to obtain a solution that is more robust than FRUI (Y. Li, Su, Yang, & Gao, 2020). They analyze the similarities of k-hop neighbors. Here, feature analyzes are obtained in 3 different ways: (1) user friend-relationships (number of common friends), (2) user friend-clique (friend groups with the least and most interaction), and (3) clustering coefficient of the ego-network (the probability that two friends of a user are also friends). Feature extraction is performed on both labeled and unlabeled pairs, and separate features are obtained according to both positive and negative instances within the labeled pairs. Connections that users make in their social circles are personal, and a user establishes similar, if not exactly the same, connections with people in their social circle on different OSNs. This provides an opportunity to find identical profiles as these links cannot be easily imitated.

1.1.4. Hybrid Approach

Finally, in the hybrid structure, diverse variances of previous three models are used in combination. Bartunov et al. propose joint link-attribute algorithm (JLA) that uses profile information and neighborhood links to match identical users across OSNs (Bartunov, Korshunov, Park, Ryu, & Lee, 2012). They rank a list that keeps candidate matches and set the most similar profile as the exact match. JLA also applies propagation information to identify the best matching pairs in the network.

Liu et al. propose a framework called HYDRA, which calculates similarity among users by both examining users' behavior in long-term periods and creating the core social structure of

each user (S. Liu, Wang, Zhu, Zhang, & Krishnan, 2014). The motivation for examining heterogeneous data in different periods is as follows: there is a possibility that users may go to the same place, share the same tweets/images on the same day, but they will not show the same pattern in the long run. For the matching process, using labeled and unlabeled pairs, multi-objective optimization is applied to build a linkage function that maximizes matching success rate. Experimental results show that HYDRA is superior to existing models.

1.2. Outline Of The Proposed Solution

Our observations from the brief survey of existing studies on profile matching can be summarized as follows:

- There exists no studies to accurately find identical profiles of all users. Namely, solutions that can find only a certain percentage of identical profiles are presented.
- Relying on profile attributes is not reliable since these can be spoofed by malicious OSN users or made private by users.
- Users can display the same behavior on multiple OSNs by sharing the same content (photo/video, text, location, etc.) in more than one OSN account. However, it is not easy to extract the writing style of users from short texts, and location information is shared less frequently on OSNs than other features.
- Users can establish similar social connections with other users on different OSNs.
- The use of different models as in hybrid approaches increases the success performance.

Based on these observations, in this thesis we propose a novel hybrid model that relies on both profile content and network structure. Our solution is a two-phase approach involving a *search phase* and then a subsequent *match phase*. Given an input source profile, the search phase acts as a filter to limit potential matches in the destination OSN to a small set of candidates. The match phase then picks a single specific profile from this candidate set.

The search phase inspects all photos shared in a profile, detects human faces within these photos and then generates a signature of the profile owner's face. Two profiles are considered a candidate match if their signatures are sufficiently similar. The match phase incorporates connections of the profiles to be matched such that matching profiles are recognized by similarity over profile photos of their connections. Notice that both phases rely on existing studies on face detection (FD) (Kumar, Kaur, & Kumar, 2019), face recognition (FR) (Guo & Zhang, 2019; Hassaballah & Aly, 2015) and face verification (FV) (Cao, Wipf, Wen, Duan, &

Sun, 2013; Crosswhite et al., 2018). While FD algorithms enable us to detect faces in posts shared by users, FR models generate feature vectors from each detected face, allowing us to measure similarity between vectors. In this way, FV decides whether two faces belong to the same person, according to the similarity of the two vectors.

Our contributions can be summarized as follows:

- Our solution is reliable in the search phase because face biometrics contained in detected faces are overly personal and cannot be imitated easily by adversaries, resulting in a more distinguishing feature than those utilized by existing work.
- Our solution is reliable in the match phase because connections cannot be imitated easily.
- We automatically identify the faces of the profile owner under the assumption that the owner's face appears the most frequently in all profile photos. To the best of our knowledge, this difficulty is considered for the first time in the literature.
- Our solution is efficient in the search phase because only face-signatures of two profiles are evaluated in their similarity. This way, an entire set of user photos is summarized into a single face signature vector, generated from a random subset of faces assumed to belong to the profile owner.
- Our solution is efficient in the match phase because only profile photos of friends (as opposed to all photos shared in the profile) are utilized.
- We test five different FD algorithms to determine the best performing FD algorithm for our problem setting. The DNN algorithm provide the best result among FD algorithms with 94% accuracy rate.
- We test three different FR algorithms on an external dataset and fine tune the FV threshold for each model empirically. The Vggface model is determined as the most suitable FR model with an f-score of 95%.
- Combining Vggface model and AGNES clustering, the faces of profile owners are found with 86% accuracy and 76% f-score.
- In the search phase, 89% of the profiles searched are found. 58% of them are in top-1 rank, 76% of them are in top-2 rank, and 87% of them are in top-4 rank.
- Finally, the match phase matches users with an f-score of $66\% \pm 4$.

We believe that the proposed search phase would be useful by itself in various application scenarios: detecting fake/cloned profiles created by copying a user's photos, locating duplicate profiles of the same user within a single OSN (i.e., de-duplication), finding criminal profiles on OSNs using photos obtained from criminals, and in its most general form, deciding whether two sets of photos belong to the same person.

1.3. Outline Of The Problem Definition

In this section, we formally define the problem of matching identical profiles based on profile photos obtained from distinct OSNs and introduce our notation. The notation is also summarized in Table 1.1 for reference purposes.

Let M and N represent two distinct OSNs. Let m be an account on M , and n be an account on N . When enumerated, we refer to individual accounts of M (resp. N) with m_i (resp. n_j). F_{m_i} is the set of feature vectors of faces contained in photos of posts by m_i . We assume that the connections of an account m can be accessed with $links(m_i)$ such that $m' \in links(m)$ implies m' and m are connected within M . Each vector in F_{m_i} is obtained by face recognition model and is an element of $\{R\}^d$.

This thesis searches for profiles identical to each $m_i \in M$ over N such that for any $n_j \in N$, matching the pair (m_i, n_j) implies the two accounts belong to the same individual. The decision is based on vector sets of the respective accounts and profile photos of their links/connections.

Every account posts multiple photos. The k^{th} photo of an account we denote with p_k . Account notation is not involved since in all cases, the relevant account is obvious from the context.

It is typical that a single photo contains several faces. Some of these are not even human faces as posting pets for example is very common. We use f and g to denote faces, and f_l to refer to the l^{th} face within a photo p . Feature representation of a face is denoted with $\vec{f}$ or $\vec{g}$ when necessary. Sets of faces are represented with F and G . When associated with a particular account, we quantify sets with the account notation as in F_m or F_{m_i} .

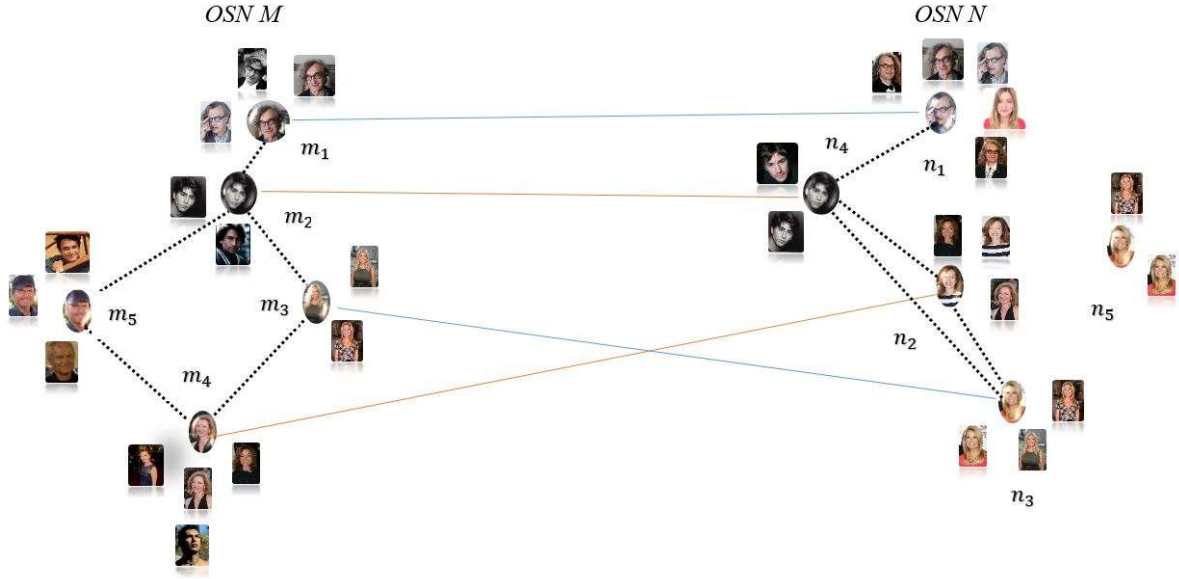


Figure 1.1. An example of identical user matching on cross OSNs

An example run of matching identical profiles is given in Figure 1.1. Here, circular images indicate profile photos and rectangular images are photos that appear on that profile. Dotted lines show (bidirectional) links between profiles, and straight lines indicate that the connected profiles on the cross OSNs belong to the same person. Orange straight lines are labeled profiles used in the training phase, and blue straight lines are determined to be identical profiles by the matching model. The matching decision is based on the similarities of the faces in the posts shared by the users and the links that the users disclose. For example, although n_3 and n_5 share similar photos, only m_3 and n_3 have similar links, so m_3 and n_3 are marked as identical profiles.

Consider account m . An arbitrary photo p_k of the account may not contain any faces of the profile owner. For example, the owner might be posting a relative's photo. During the search and match phases, we need to be able distinguish foreign faces (i.e., faces that do not belong with the owner). We refer to the owner with o , such that o_{m_i} is the owner of m_i and O_{m_i} is the set of faces that belong to the owner. Instead of using the all vectors in F_{m_i} , using only the vectors of the profile owner's faces O_{m_i} reduces complexity and improves accuracy.

Candidate identical profiles: In real life, if any m_i and n_j are identical profiles, then O_{m_i} and O_{n_j} should be more similar to each other than any other user's vector set. Based on this assumption, a similarity is calculated between each m_i and all $n \in N$. Thus, a list of the most similar candidate profiles that can be identical to m_i is obtained. This list is denoted with denoted N' .

Matching identical profiles: The search phase may yield an N' where $|N'| > 1$. In other words, there may be more than one profile belonging to m_i in N' after the search phase. Even worse, in the case that the search has failed, the only profile in N' may not be identical to m_i . The profiles that are not identical to m_i may be a profile that creates a fake or cloned account using photos of m_i or that have photos of a person who is very similar to the owner of the m_i . At this stage, their similarity is measured using connections $m' \in \text{links}(m_i)$ and $n' \in \text{links}(n_j)$ to eliminate non-identical profiles. If m_i and n_j are indeed identical, pairs of m' and n' are expected to be more similar than for other users.

Table 1.1. Our notations

Term	Description
$\mathcal{M}, \mathcal{N}$	Distinct online social networks (OSN)
M, N	Sample OSNs
m, n	An account of M, N
m_i, n_j	Accounts i and j on M, N
p, q	Photo of M, N
p_k	The k^{th} photo k of an account
f, g	Faces of a photo
f_l	The l^{th} face of photo f
$\vec{f}, \vec{g}$	Vector representation of photos f, g
F, G	Set of vectors
F_m	Vector set for account m
F_{m_i}	Vector set for account m_i
O	Vector set of an account owner
O_{m_i}	Vector set of owner for account m_i
$\vec{s}_{m_i}, \vec{s}_{n_j}$	Signature of account m_i, n_j
N'	Set of candidate accounts $n \in N' \subseteq N$ that may be identical to m
$m' \in \text{links}(m_i)$	Friendship links of m_i
m'_i, n'_j	Vector set of profile photos of m', n'_j
thr	Threshold for face verification
$thrSig$	Threshold for signature comparison
$thrLink$	Threshold for comparison of two friends where $m' \in \text{links}(m)$ and $n' \in \text{links}(n)$

The remainder of the thesis is structured as follows. A comprehensive review of existing work in the area is presented in Chapter 2. In Chapter 3, we describe the materials and methods used and the details of the proposed solution. Chapter 4 presents our experiment scenarios and discusses the results, and finally we conclude in Chapter 5.

2. LITERATURE REVIEW

In this chapter, we examine the existing matching models (previously reviewed briefly in Section 1.1). Since we propose a profile matching model based on users' faces in their photos and profile photos of their friends, we also review both studies on other topics and in finding identical profiles, made with profile photos and faces in posts shared by users.

2.1. Profile Attribute-Based Approach

Matching identical profiles using the attributes that users share to provide detailed information about themselves is a highly studied topic. While some studies are carried out only by considering the username (J. Liu et al., 2013; Perito et al., 2011; Zafarani & Liu, 2009, 2013), some studies use the content shared by users in addition to the username (Iofciu et al., 2011). Besides these, some researchers only work on the display name preferred by the users (Y. Li, Peng, Ji, Zhang, & Xu, 2017) and other studies use more than one profile attribute and matched users (Goga, Loiseau, Sommer, Teixeira, & Gummadi, 2015; Motoyama & Varghese, 2009; Raad, Chbeir, & Dipanda, 2010; Vosecky et al., 2009).

Zafarani and Liu (2009) suggest using the username to determine the username that a user uses on a different community, that is, to recognize identical users in different communities. They conduct experiments on a total of 12 communities such as Flickr, YouTube, Twitter and achieve an average accuracy rate of 66%. Y. Li et al. (2017) propose a supervised model for linking identical profiles, taking into account the display name preferred by users. The assumption here is that the selected display name is not random because it represents the people, and it has a high variety. Another reason for choosing display name is that it is very low cost to collect and process. In that study, many features are obtained by using many string techniques such as longest common substring similarity and Jensen-Shannon distance for each display name and given as input to machine learning algorithms. Experiments are performed with 7 different classifiers on three different OSNs. While Logistic Regression(LR) provides the best results, they obtain f-scores of 96.24%, 92.49%, and 90.68%, respectively.

Motoyama and Varghese (2009) propose a searching and matching model on two different OSNs (Facebook and MySpace). The searching and matching methods are carried out with the

multiple profile attributes of the users. While testing different combinations of attributes for the searching, a feature vector consisting of personal information such as age, birthday, hometown, location, education is produced for each profile during the training and testing phase of a boosting classifier. In addition, friend lists are also collected for each profile to measure the similarity of their friend lists on both Facebook and MySpace. The results show that the name, country, location, school attributes contribute more to the search method. Like Vosecky et al (2009), Raad et al (2010) develop a framework that uses multiple profile attributes to match identical profiles by weighting the attributes either manually or automatically.

Although working with profile attributes provides good results in some cases, profile attributes can be imitated by other users or are features that some users make private for privacy reasons. Also, in large-scale OSNs, there may be quite a lot of candidate profiles with the same name or attribute value. It therefore raises some concerns to establish a robust model.

2.2. Content-Based Approach

Content-based matches are based on the assumption that user-generated activities are unique and personal. As a result, while some researchers propose a model based on the similarities in users' posts/tweets/comments (Ahmad & Ali, 2019; Almishari & Tsudik, 2012; Roedler, Kergl, & Rodosek, 2017), some studies are carried out based on the similarity of the location information of the places users visit and the temporal information of the posts users share (Goga, Lei, et al., 2013; Kong, Zhang, & Yu, 2013; Y. Li et al., 2018; Lim, Lu, Chen, & Kan, 2015).

Almishari and Tsudik (2012) present a simple method by which they could find out which author wrote the reviews, based on the writing style of the authors, and then match the anonymous reviews written on different sites with the real authors. The study is carried out on approximately 1 million reviews and 2,000 authors collected on the Yelp site. Each collected review is divided into 4 different token parts and tested separately: unigrams, digrams, rating, and category. Naive Bayes (NB) and Kullback-Leibler Divergence (KLD) Metric are used as models. The results show that in experiments with digrams tokens, both KLD and NB are able to correctly match authors to reviews by almost 99.5% accuracy rate, and even achieve 83% linking success rate with the only unigrams tokens. Although it is somewhat easy to detect the writing style of users in forums and blogs where long posts are allowed (even this depends on

the variety of words users use), it is not appropriate on platforms such as Twitter that offer users the chance to write limited content.

Lim et al. (2015) analyze the behavioral patterns displayed by users on 6 different OSNs (Flickr, Google+, Tumblr, Twitter, YouTube and Instagram), both in terms of temporal and topical signatures. The study shows that a user expressed themselves with different profile descriptions on each OSN and profile description is not suitable for finding identical profiles. In addition, it is seen that users mostly produce their content on YouTube and Instagram and share most of these contents on Twitter and Tumblr by cross-network sharing. This showed that it is possible to match users based on content. Kong et al. (2013) propose a novel Multi-Network Anchoring (MNA) model to predict one-to-one anchor links (the type of link that indicates that two different accounts belong to the same person) on cross OSNs. To find anchor links on OSNs, spatial (location of home, restaurant etc.), temporal (frequency of time slots of shared posts), social links (common friends), and text information (most used words) features are extracted by using generated data by each profile. Later, the obtained features are given as input to a machine learning algorithm (here, SVM). The study is conducted on two different OSNs: Twitter and Foursquare. To test the performance of MNA, they run existing techniques on the dataset and show that MNA is superior to existing techniques.

Riederer et al. propose a model in which identical profiles can be found if there are enough timestamped location information shared by users (Riederer, Kim, Chaintreau, Korula, & Lattanzi, 2016). A footprint is created for each user using location information, and it suggests finding identical accounts based on the similarity of footprints. The proposed model is tested on many OSNs and the results show that although the model gives good results, it needs a lot of timestamped location information, if available, to get good results. The fact that users share their location information only on some platforms (almost most of the studies were done on the same platform, for example, Foursquare), as well as the fact that only a certain users share this information, can reduce the probability of finding many identical profiles. Also, since everyone can go to the same restaurant or gym, users may only have a different footprint if they draw quite different location patterns. In addition, users need to share a lot of location information so that footprints can be formed. Finally, extracting temporal, location and text information from each user is both difficult and costly. When these are considered, timestamped location information is limited to being applicable on most OSNs.

2.3. Network-Based Approach

Studies in this area provide a solution to the problem of finding identical profiles by examining the interactions/relationships between users on OSNs (Korula & Lattanzi, 2014; Y. Li et al., 2020; Tan et al., 2014; Zhou et al., 2016). In addition to these, some studies also offer protection methods to the users to protect their privacy and be aware of the risks (Shen & Jin, 2014).

Tan et al. (2014) propose a semi-supervised framework that models each user's social relationships on the hypergraph by creating a hypergraph for each OSN. Here, each user is converted to a low-dimensional embedding on the social hypergraph, and then similarities are calculated on the created embeddings. Manifold Alignment on Hypergraph (MAH) is applied with labeled data so that the embeddings of identical users are close to each other. To mapping users, candidate users are ranked by similarity value and marked top one as identical user. The results showed good performance for user mapping on hypergraphs. Contrary to study (Zhou et al., 2015), Zhou et al. (2016) develop an unsupervised model that simply infers multiple feature vectors (embedding) from friendship links for each user without the need for prior knowledge (seed users), and they call this model as Friend Relationship-based User Identification algorithm without Prior knowledge (FRUI-P). Then, they calculate one-to-one similarity between the embeddings and matched candidate pairs. Although FRUI-P seems to lag behind existing models in terms of recall value, it is shown to be superior to existing techniques in terms of precision.

Shen and Jin (2014) present two supervised methods using users' profile attributes (such as name, location and profile photo) and friendship information: the first is the User Accounts Linkage Inference (UALI) model to find identical profiles on OSNs, and the second is Information Control Mechanism (ICM) model, which increases the privacy level of users while a certain amount of information is still visible, reducing the likelihood of users finding identical profiles. After extracting the features for each profile, they are trained with 4 different classifiers. Both proposed models are tested on 3 different OSNs: Google+, Twitter and Foursquare. The UALI model is shown to perform better than existing models in most cases. Finally, the ICM model show that 80% of the users are protected against the existing UALI models. Network-based studies show that the relationships established by users are much better than profile information to distinguish users and provide an opportunity to find identical profiles.

2.4. Hybrid Approach

With the hybrid model, researchers use profile attributes, social relations and posts shared by users in a model with many different variations to make profiles more distinctive and more distinguishable from each other (Bartunov et al., 2012; Bennacer, Nana Jipmo, Penta, & Quercini, 2014; Johansson, Kaati, & Shrestha, 2013; Malhotra, Totti, Meira, Kumaraguru, & Almeida, 2012; Soltani & Abhari, 2013; Wang et al., 2019, Jain et al. 2013).

Jain et al (2013) propose two entity searching models, content and network-based (friendship links), and combine them with the traditional profile-attribute based searching method. Here, it is preferred to use username, name, location, profile picture and URL (link indicating a user's other OSN account - Self-Identification), which are public in both profiles, as profile attributes. For the content search method, a search is performed on tweets and posts shared by users, and the cosine metric is applied to calculate similarity between tweets/posts. For network search, the exact match is applied on the friendship lists obtained from both different profiles. Since each searching method returns a list of candidate profiles, they determine the real identical profile in the returned list by applying the entity matching method. Here they apply two different matching methods: syntactic and image matching. The overall whole architecture (searching and matching methods) is named Finding Nemo. The results show that the proposed search model improved over the traditional search method. Johansson et al (2013) suggest stylometric matching and time-based matching methods to detect users who create multiple accounts and comment on the forums. In stylometric matching, a feature vector "writeprints" is created for each user by extracting the most used words according to the content of the comments made by the users. In time-based matching, a feature vector is created for each user, taking into account the times of the comments/posts shared by each user. Here, similarity between vectors is obtained using the Euclidean distance metric. During the matching, they rank the users in the candidate list returned by the proposed model according to their similarities. When matching is made considering both top-1 and top-3, it is seen that the time-based matching method is superior to stylometric matching in both cases, while it is seen that the combination of both models gives much better results.

Soltani and Abhari (2013) also present a hybrid model to find identical profiles across multiple social media. The model basically uses 3 different data types: personal identities (PI) (such as gender, location, education, profession, email), social identities (SI) (such as tweets/posts), and relational identities (RI) (such as friendships). They give a weight according to the importance

of each attribute. The proposed model is tested on Facebook, LinkedIn, and Twitter. The results show that it is highly dependent on the number of tweets and posts, and when tested by obtaining 5,10,20 tweets for each user, it shows that the recall value of the model remains below 20% in all cases, while the precision increases above 60% when the number of tweets is 20. Finally, Malhotra et al. (2012) create a digital footprint for each user by using features such as location, description, profile image and links to find identical profiles in social networks, while Bennacer et al. (2014) develop a model that iteratively finds new identical pairs at the end of each step using a certain number of candidate pairs initially.

2.5. Studies Made With Profile Photos And Faces In Posts

In social media, the number of photos shared by users creates big data and it becomes worth for researching, understanding and using. Zheng et al. (Zheng, Yuan, Chang, & Wu, 2016) examine the profile photos preferred by users in social media according to their gender and region of residence and present differences. Guntuku et al. (Guntuku, Preotiuc-Pietro, Eichstaedt, & Ungar, 2019) also examine the relationship between depression and anxiety with pictures shared by users in their study on Twitter. While they say that users with depression and anxiety shared pictures with similar features (grayscale and low aesthetic cohesion), the results show that the mental states of the users could be determined when the shared pictures are taken into account. In another study, Bakhshi et al. (Bakhshi, Shamma, & Gilbert, 2014) observe how photos with human faces affect the relationship between users on social media. The study is done on 1 million Instagram images, and the results show that posts with human faces receive over 32% more likes and comments. This shows that the human face affects the interaction between people more. In addition, other studies made with profile photos are as follows (Jung, An, Kwak, Salminen, & Jansen, 2017; Mei et al., 2020; Wei & Stillwell, 2017).

Apart from these, Choi et al. suggest using a multiple face recognition (FR) model, in contrast to traditional methods, for the problem of face annotation (determining who the faces in the captured picture belong to) (Choi, De Neve, Plataniotis, & Ro, 2011). To determine who the other faces in a user's photo gallery belong to, they propose a model that would judge the faces of users' contacts. In the result section, they obtain output by combining the results from more than one FR engine. Finally, the proposed model shows much better results than the models using a single FR engine. Dantone et al. also propose an algorithm that recognizes and tracks users in real time on mobile devices. During the training phase of the proposed algorithm, 300 thousand images collected on Facebook are used (Dantone et al. 2011). Lastly, Ilia et al. (Ilia, Polakis,

Athanasopoulos, Maggi, & Ioannidis, 2015) develop a mechanism that determines which users will not be shown the faces in the shared photo according to user authority and blurs the relevant faces, in order to prevent a photo shared on OSN from getting into the hands of other users and causing bad results.

2.6. Studies Made With Profile Photos And Faces In Posts To Match Identical Profiles

Since facial biometrics of people is unique to each person, cannot be changed, and is a more distinguishing feature than user-generated content, it is also preferred in the problem of finding identical profiles. In this section, we examine studies that use profile photos and faces from users' posts to match identical profiles on cross OSN. Some studies add to their models the similarities of users' profile photos/avatars, in addition to profile attributes, friendship links, and posts shared by users (Goga, Perito, Lei, Teixeira, & Sommer, 2013, Jain et al., 2013; Goga et al., 2015; S. Liu et al., 2014; Oliseenko & Abramov, 2021). Here traditional methods such as RGB-histogram image matching and Scale-Invariant Feature Transform (SIFT) similarity are used when calculating the similarity between two profile photos.

Contrary to the existing techniques, we propose a searching strategy based on the similarity of the faces in the users' posts, and a matching model based on the similarity of the profile photos of their friends. We do not use any extra information (like profile attribute) in both methods. Users can take a photo with their friends or a family member and share it on their profiles. For this reason, when photos in a profile are collected, people's faces may be in these photos, apart from the profile owner. Therefore, in the searching method, we first identify the faces of the profile owners automatically, then we search for identical profiles according to the similarity of the profile owners' faces. In a similar study, Sokhin et al. propose a model for matching identical profiles by calculating similarity between faces in posts shared by users (Sokhin, Butakov, & Nasonov, 2019). Instead of using all the faces collected from a profile, they define a defining vector (DV) for each profile. For this, the faces in the photos of each profile are detected with the MTCNN algorithm and converted into a feature vector with the Facenet face recognition model. The DV is then created by averaging the vectors of the faces that appear most frequently in a profile. In detail, a hierarchical clustering is performed to find the most frequently seen faces, and a threshold value of 0.8 (fixed) is used in this. At the end of clustering, clusters with frequently appearing faces are obtained. The assumption here is that the most dense cluster (the cluster with the most faces) will belong to the account owner. Finally, for matching identical profiles, the distance between the DVs created for each profile

is calculated using the distance metric and the profiles closest to each other are matched as identical profiles. Here, a threshold is used as an additional filtering for matching: if the closest distance value does not exceed the threshold (here, they set to 0.65), the two profiles are matched, otherwise it is decided that the two profiles belong to different people. The results obtain an f-score value of 0.872 when the DV is created using the first 2 densest clusters for each profile. Finally, they show by experiment that it is not appropriate to match using only profile avatars. However, this study is sensitive to fake and cloned accounts. They find only one identical profile for each profile. However, the profile marked as a identical may be a fake or cloned profile or a profile consisting of photos of someone similar to the profile owner. Therefore, additional approval is needed.

After reviewing the existing studies, we can summarize the literature as follows: the problem of matching identical profiles is a studied topic. For this, solutions are presented by using many features such as profile attributes, social relations, user content. Similarities of profile photos are also included in these. However, the number of studies based on the similarity of the faces in the posts shared by users is very few, and therefore it is a subject worth researching. We present our differences as follows: (1) Since the performance of face detection algorithms can vary according to different conditions, instead of using MTCNN directly on the OSN dataset, we test 5 different face detection algorithms (including MTCNN) on the dataset that contains faces are taken from different angles and composed of various color contrast to determine the most ideal face detection algorithm. (2) Likewise, instead of using the Facenet model directly as the face recognition model, we test 3 different face recognition (including Facenet) models on an external dataset and determine the most ideal face verification threshold for each model. Unlike Sokhin et al. (2019), we generate signatures for each profile by averaging only t vectors of the profile owners' faces. We also perform a hierarchical clustering to find the vectors of the profile owners like Sokhin et al. (3) However, instead of the fixed value for the threshold, we use the experimentally determined FV threshold value for each FR model and also evaluate how accurately the 3 different FR models find the vectors of the profile owners' faces. We then present 2 different models to search and match identical profiles. (4) In the searching method, we calculate the distance between the signature of each profile whose identical profile is sought and the signatures of other profiles, and we obtain a candidate list of profiles that may be identical to the searched profile. In this way, if a profile has more than one identical profile, it is obtained with a single query. After that, we follow two different strategies in the matching

model. (5) First, we test 3 different approaches to determine the optimal matching method: supervised, unsupervised, and rule-based by using profile owners' vectors. (6) Second, we calculate the matching count between the profile photos of the friends of the profiles in the candidate list and the profile photos of the friends of the profile whose identical profile is sought. Finally, we mark the profile with the most matching count as an identical profile



3. MATERIALS AND METHODS

In this chapter, we present the existing datasets and methods that we rely on building our proposed solution. Section 3.1 covers face detection algorithms. In Section 3.2 we discuss face recognition algorithms. Then in Section 3.3, we briefly discuss clustering algorithms. The datasets are presented in Section 3.4. Finally in Section 3.5, we present our solution towards matching identical profiles on cross OSNs based on face identification

3.1. Face Detection Algorithms

Face detection is a well-studied computer vision technique that first checks for any faces on a given image and returns the location and size of each face, if any (Kumar, Kaur, & Kumar, 2019). It is the primary step in many automatic systems such as face recognition (AL-Allaf, 2014), face tracking (Vadakkepat, Lim, De Silva, Jing, & Ling, 2008) and facial expression recognition (Hernandez-Matamoros, Bonarini, Escamilla-Hernandez, Nakano-Miyatake, & Perez-Meana, 2016).

An important sub-task in this thesis is to accurately detect and cut faces in photos, particularly for the case of photos collected from OSN profiles. Detecting the face incorrectly (detecting the face in the non-face area or not detecting the existing face on the photo) or detecting and cutting the face smaller/larger than it is could prevent the collection of correct faces from the profiles and adversely affect the accuracy of the results. As a result, determining the best performing face detection algorithm is vital. The performance of a face detection algorithms varies with parameters like the size, brightness of a photo and shooting angles of the faces (Kumar et al., 2019). In this section, we briefly present 5 face detection algorithms: You Only Look Once versin 3 (YOLOv3), Deep Neural Network (DNN), Histogram of oriented Gradients (HoG), Cascade Classifier and MultiTask Convolutional Neural Network (MTCNN). These are prominent algorithms from the literature that have achieved good results.

3.1.1. You Only Look Once version 3 (YOLOv3)

A face/object detection algorithm should quickly return the bounding box coordinates of all existing faces/objects on an image, even with varying poses and scales. To determine the model that provides the most accurate and fastest performance, with the developments in both deep

learning techniques and convolutional neural networks (CNNs), alongside traditional methods (Malisiewicz, Gupta, & Efros, 2011; Peng, Tian, Wang, Li, & Huang, 2015), many CNN-based object/face detection models have been developed and presented (Cai, Fan, Feris, & Vasconcelos, 2016; W. Liu et al., 2016; Ren, He, Girshick, & Sun, 2017; Yang, Luo, Loy, & Tang, 2018; Zhang et al., 2017, Jiang et al, 2017). Besides these, YOLO is also another latest technology developed by Redmon et al. (Redmon, Divvala, Girshick, & Farhadi, 2016) for the object detection problem and it continues to be developed: the second version YOLOv2 was presented in 2017 (Redmon & Farhadi, 2017) and the third version YOLOv3 in 2018 (Redmon & Farhadi, 2018).

YOLO is the first one-stage detector based on CNNs. In the other words, YOLO first splits a given image into distinct grid cells and directly predicts the coordinates and classes of possible objects/faces by searching on each cell with a single network (Wang & Zheng, 2018). YOLO applies forward propagation only once over the network to detect objects, and uses only the given input image. While this makes YOLO run fast, it can also cause low detection accuracy, especially for small targets.

Some object detection studies are developed based on YOLO (Asha & Narasimhadhan, 2018; Hendry & Chen, 2019; Lan, Dang, Wang, & Wang, 2018; Lu et al., 2018). YOLOv3 consists of a total of 106 layers, including convolutional, residual and up-sampling layers, and uses the backbone of Darknet-53 (Mao, Sun, Liu, & Jia, 2019) that consists of 53 convolutional layers for feature extraction (Vinh & Anh, 2020). The object/face detection task is performed by the remaining 53 layers. Here, to increase detection accuracy over small targets, YOLOv3 performs detection over 3 different resolutions of the input image by down-sampling resolution to 32, 16 and 8. The general structure of YOLOv3 for object detection is given in Figure 3.1 (Mao et al., 2019). YOLOv3 is trained and tested on the COCO (Lin et al., 2014) dataset containing 80 different categories and produces good results. It is also recommended for face detection in both real-time environments and complex environments since it can detect small objects well and is fast (Weijun Chen, Huang, Peng, Zhou, & Zhang, 2021; Chong Li, Wang, Li, & Fei, 2020; Vinh & Anh, 2020; Wang & Zheng, 2018). In this thesis, YOLOv3 (Nguyen, 2022) trained on the WIDER FACE (Yang, Luo, Loy, & Tang, 2016) dataset is used for face detection.

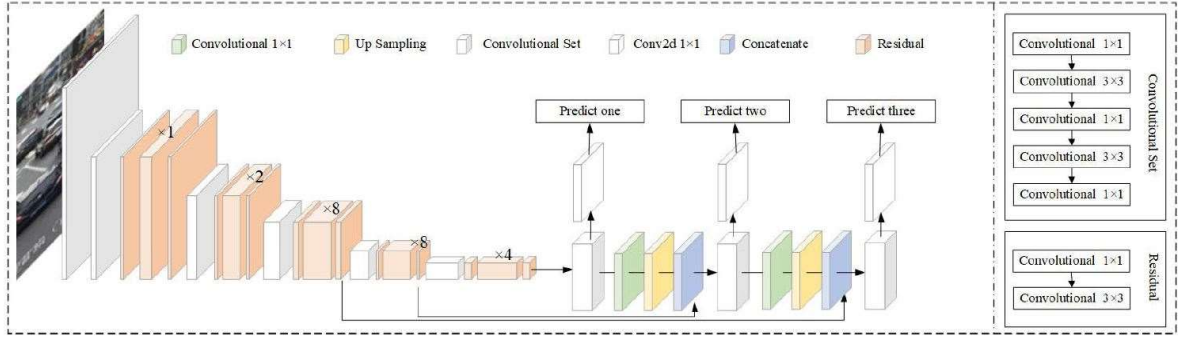


Figure 3.1. Structure detail of YOLOv3. It uses Darknet-53 as the backbone network and uses three scale predictions (Mao et al., 2019)

3.1.2. Deep Neural Network (DNN)

The deep neural network (DNN) is inspired by the Single Shot-multibox Detector (SSD) (Wei Liu et al., 2016) which is proposed for object detection and uses the backbone of the ResNet-10 (He, Zhang, Ren, & Sun, 2016) model. DNN is also a one stage detector like YOLO, it can capture multiple objects using multibox over a single shot (Nagrath et al., 2021). It is fast enough to be used in real-time application scenarios and shows high-accuracy detection rate. In this thesis, we use OpenCV DNN (2018) face detector.

3.1.3. Histogram Of Oriented Gradients (HoG)

Histogram of Oriented Gradients (HoG) is a traditional and simple feature identification method. Since it is a simple and fast calculation method, it is a highly preferred for object/face detection (Dalal & Triggs, 2005; H. Ren & Li, 2014; Surasak, Takahiro, Cheng, Wang, & Sheng, 2018; Žemgulys, Raudonis, Maskeliunas, & Damaševičius, 2018). HoG divides a given image into small grid cells and calculates the histogram of the gradient directions with pixels in each grid. The resulting feature vector shows the magnitude of gradients of each region area. The gradient of the points where the corners and edges exist are usually large and this allows detection of objects/faces (Surasak et al., 2018). The gradient feature vector of a pixel (x,y) is obtained with Equation 1 and 2, the magnitude and direction of gradient for pixel (x,y) is calculated by Equation 3 and 4 as follows:

$$G_x(x,y) = H(x+1,y) - H(x-1,y) \quad (1)$$

$$G_y(x,y) = H(x,y+1) - H(x,y-1) \quad (2)$$

$$G(x,y) = \sqrt{G_x(x,y)^2 + G_y(x,y)^2} \quad (3)$$

$$a(x,y) = \tan^{-1}\left(\frac{G_x(x,y)}{G_y(x,y)}\right) \quad (4)$$

In the equations, G_x represents the horizontal gradient and G_y represents the vertical gradient.

Once all gradient feature vectors are obtained, these vectors are concatenated to create a single and long feature vector. Lastly, these vectors are used as input for the machine learning algorithm (mostly, SVM) to learn patterns on feature vectors of objects/faces. When a new feature vector is given, the trained model decides whether an object/face exists or not. HoG is suggested in face detection apart from object detection (Adouani, Ben Henia, & Lachiri, 2019; Cerna, Cámara-Chávez, & Menotti, 2013). In this thesis, we use the HoG method (King, 2022) offered by Dlib (King, 2009). This Dlib version of HoG was trained with SVM on the LFW dataset (Huang, Mattar, Berg, & Learned-Miller, 2008).

3.1.4. Cascade Classifier

Viola and Jones propose the Cascade Classifier for detection of objects based on supervised algorithms (Viola & Jones, 2001). The algorithm consists of the following 4 steps:

a) Haar Feature Selection consists of some pre-defined filtering features. These are line features, edge features, and four-rectangle features which are made of black and white pixels. Examples of these filters are presented in Figure 3.2 (Jadhav, Lone, Matey, Madamwar, & Jakhete, 2021). Instead of using the pixel values of the given image directly, they generate a feature by shifting each filter in all possible places on the image. Each feature is obtained by subtracting the sum of the pixel values from the white rectangle to the sum of the pixel values from the black rectangle. In this way, a feature pool is created by visiting all possible places with all filters.

b) Integral image is produced to make summation of pixel values quickly with a low computational cost. This is done by creating a temporary image with the same size as the original image.

c) Adaboost, an effective boosting algorithm, is used to determine the best features from the feature pool obtained from the training dataset to detect faces with the least error rate.

d) Cascading classifiers are applied to ignore non-face windows as quickly as possible. Here, instead of applying all features on a given sub-window, grouped features are tested one-by-one

in stages. If a given sub-window is rejected in any stage, it is discarded. Otherwise, it is sent to next stage. The sub-window that passes all stages will be a face region. In the cascading step, different classifiers are used together to perform the stages. Multiple face detection studies are made with Cascade Classifier (Cuimei Li, Qi, Jia, & Wu, 2017; Priadana & Habibi, 2019; Sharifara, Mohd Rahim, & Anisi, 2015). In this thesis, we use the Cascade Classifier offered by OpenCV (OpenCV, 2015).

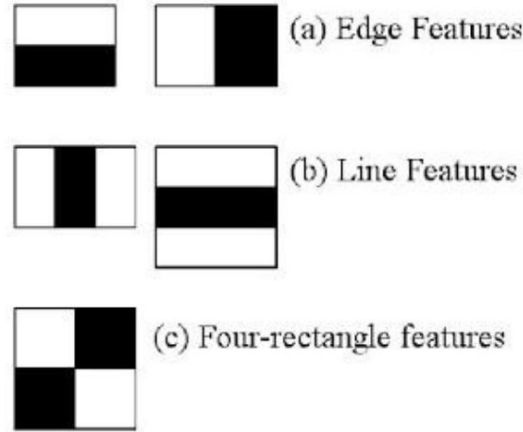


Figure 3.2. Haar-like features (Jadhav et al., 2021)

3.1.5. MultiTask Convolutional Neural Network (MTCNN)

MultiTask Convolutional Neural Network (MTCNN) is a CNN-based model proposed for solving face detection and alignment problems in real time (K. Zhang, Zhang, Li, & Qiao, 2016). MTCNN basically consists of three different and consecutive networks: Proposal Network (P-Net), Refine Network (R-Net), and Output Network (O-Net). In order to detect targets of different scales and create a robust model, MTCNN produces an image pyramid by resizing a given image to different sizes before it enters the network, and these scaled images are sent to the network as input. In the first subnet stage (P-Net), candidate windows that may contain a target are generated quickly with light CNN and vectors of bounding box regression are obtained for each candidate. At end of the P-Net, they apply non-maximum suppression (NMS) to eliminate false-positive candidates. All candidates returned by P-Net are processed by R-Net that has a more complex CNN network. R-Net performs additional conditions and applies border regression with NMS to reject the candidates that has low confidence score. Finally, O-Net returns faces regions with the bounding boxes coordinates by selecting the best candidates and produces positions of the five facial landmarks (N. Zhang, Luo, & Gao, 2020).

The general model structure (K. Zhang et al., 2016) of MTCNN is given in Figure 3.3 (K. Zhang et al., 2016). MTCNN is tested on WIDER FACE, Fddb, and AFLW datasets along with traditional methods to test its performance and it is shown to have better detection performance and faster speed. MTCNN is used for face detection in many studies (Ghofrani, Toroghi, & Ghanbari, 2019; Jose, Greeshma, Mithun Haridas, & Supriya, 2019; Ma & Wang, 2019). In this thesis, we use the Python implementation of MTCNN (Centeno, 2021).

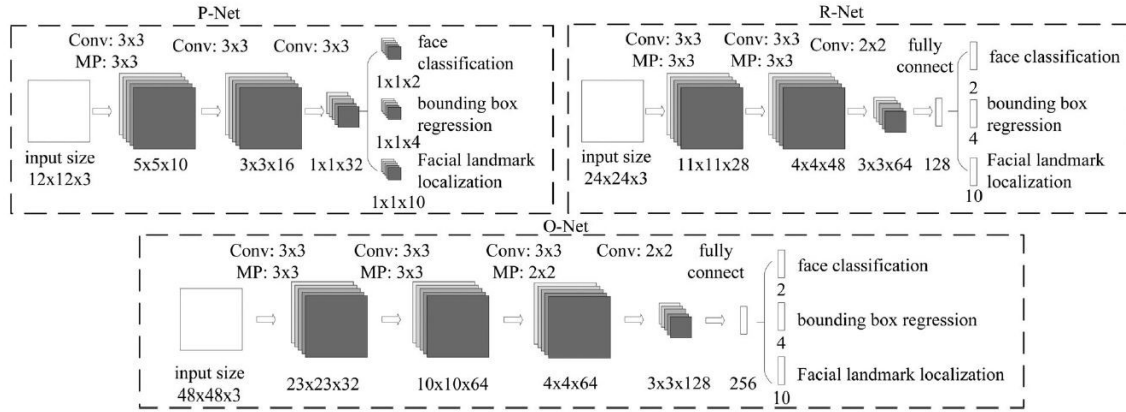


Figure 3.3. Overall Structure of MTCNN (K. Zhang et al., 2016)

3.2. Face Recognition Algorithms

In this section, models previously proposed for the face recognition problem and used in this thesis are presented in detail. This research area is still very popular for computer vision.

Face recognition (FR) is the process of matching similar faces using human biometrics (Guo & Zhang, 2019). Since the biological characters of people such as eyes, fingerprints and facial features cannot be stolen and are unique to individuals, FR is highly preferred in systems that require security. There are two specific instances of FR: (i) face verification (FV) and (ii) face identification (FI). FV is the problem of deciding whether two given face images belong to the same person by comparing them one-to-one and then returning either true or false (X. Cao, Wipf, Wen, Duan, & Sun, 2013; Crosswhite et al., 2018; Nguyen & Bai, 2011; Sankaranarayanan, Alavi, Castillo, & Chellappa, 2016). Unlike FV, FI performs one-to-many matching. In other words, the face given as belonging to a person is searched in a database of faces of many people and the problem is to determine who it belongs to. The result of the FI process a list of people with a similar faces to the face given (Guo & Zhang, 2019). Algorithms developed for FR create a feature vector (i.e., an embedding) for each face and calculate a

distance (alternatively, similarity) between these feature vectors and decide compare the distance with the threshold. If the measured distance is above the threshold (Schroff, Kalenichenko, & Philbin, 2015), faces should belong to distinct people. The basic steps respectively applied for FR are presented in Figure 3.4 (Hassaballah & Aly, 2015).

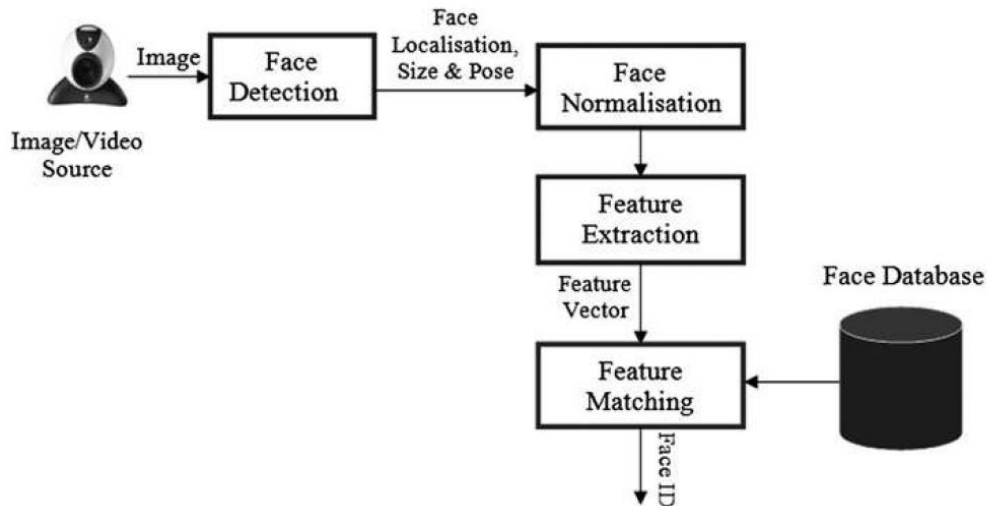


Figure 3.4. General steps of face recognition (Hassaballah & Aly, 2015)

Face recognition models based on traditional methods such as local binary pattern (LBP), principal components analysis (PCA), and gabor wavelet (GW) have been proposed before (Gumus, Kilic, Sertbas, & Ucan, 2010; Mohammed, Minhas, Jonathan Wu, & Sid-Ahmed, 2011; Vinay, Shekhar, Murthy, & Natarajan, 2015; Zhao & Wei, 2017). In addition, many CNN-based FR models have been developed, especially with the developments in deep learning (Q. Cao, Shen, Xie, Parkhi, & Zisserman, 2018; Weiyang Liu et al., 2017; Schroff et al., 2015; Taigman, Yang, Ranzato, & Wolf, 2014). When the different approaches are examined, it is seen that while traditional methods use one or two layers to examine an approach based on the change in textures, histogram-based or pixel distribution; deep learning methods extract features with more layers and different loss function approaches (Wang & Deng, 2021). Therefore, in this thesis, we use Facenet (Schroff et al., 2015), Vggface2 (Q. Cao et al., 2018) and Dlib's FR model (King, 2017a), which are both open source and are instances of deep learning models. The input sizes accepted by the models, the sizes of the embedding they create, the distance measurement metric they use and the face recognition thresholds determined for each model are summarized in Table 3.1.

Table 3.1. Details of FR models

Model	Input size	Embedding size	Distance metric	Threshold
Facenet	160x160	128	Euclidean	1.242
Vggface	224x224	2048	Cosine	0.58
Dlib	150x150 and different sizes	128	Euclidean	0.6

3.2.1. Facenet

Facenet is a model proposed by Google researchers for solving the FR problem by generating a 128-byte feature vector (embedding) $f(x)$ for each face x into feature space R^d (Schroff et al., 2015). Facenet, which learns mappings over the Euclidean space based on the measured distances of the faces, uses triplet loss (TL) (Weihua Chen, Chen, Zhang, & Huang, 2017) to make closer the embeddings of the same person, and farther the embeddings of distinct persons. The squared L2 distance is used to measure distance. Once each of the embedding is created for each face, the goal is to calculate similarity of the two embeddings and decide whether they belong to the same or different people. The working logic of TL is presented in Figure 3.5 (Schroff et al., 2015).

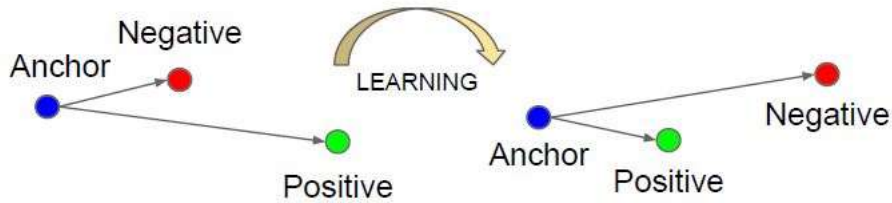


Figure 3.5. Triplet Loss function (Schroff et al., 2015)

TL minimizes the distance between an anchor $f(x_i^a)$ and a positive sample $f(x_i^p)$ of the person i and maximizes the distance between the $f(x_i^a)$ and a negative sample of any person $f(x_i^n)$. This is done for each triplet pair in the data set as follows (Schroff et al., 2015):

$$(f(x_i^a) - f(x_i^p))^2 + a < (f(x_i^a) - f(x_i^n))^2, \forall (f(x_i^a), f(x_i^p), f(x_i^n)) \in T \quad (5)$$

where a is the margin (distinction) between positive and negative embedding pairs and T is the set of the all triplets in the training set. Then loss is minimized as follows:

$$L = \sum_i^X [(f(x_i^a) - f(x_i^p))^2 - (f(x_i^a) - f(x_i^n))^2 + a]_+ \quad (6)$$

where X is the number of the total triplets.

Two different networks, Zeiler and Fergus (Zeiler & Fergus, 2014) and Inception (Szegedy et al., 2015) are used to test the performance on the Labeled Faces in the Wild (LFW) dataset. Different sizes for the input size of images are tested: 220x220, 224x224, 160x160, 96x96, 165x165, and 140x116. In addition, different embedding sizes are also tested: 64, 128, 256, 512. As a result, an overall accuracy of 99.63 is achieved. The best result is obtained with 160x160 input size and 128 embedding. The threshold value determined for classification in this is 1.242.

3.2.2. Vggface2

Contrary to traditional methods, deep learning models need a training set when extracting feature vectors for a face. FR performance depends on the quality of the created training set as well as the model created. In other words, the dataset should represent all groups of the society. It should consist of a rich dataset for age, pose and ethnicity.

Vggface2 (Q. Cao et al., 2018) offers a dataset that is both quite large and comprehensive in terms of age, ethnicity, and pose. The created dataset consists of 3.31 million images for 9131 people. Cao et al. (2018) use the architectures of the CNN-based ResNet-50 (He et al., 2016) and SE-ResNet (Hu, Shen, & Sun, 2018) models to perform a training on Vggface2 and MS-Celeb-1M datasets for FR. Then they test the trained model on 3 different IJB datasets. Here, they use 224x224 as input size and produce a feature vector of 2048 size as output for each image in both models. For measuring distance, Cosine similarity is preferred. The results show that the feature vectors obtained with both ResNet and SE-ResNet trained on the VGGface2 dataset are suitable for FR and are superior to other studies. It is also reported here that SE-ResNet outperforms ResNet. Therefore, we use the Vggface2+SE-ResNet model in this thesis. We call the Vggface2+SE-ResNet model as Vggface for short and present it as such in the results.

3.2.3. Dlib

In 2017, the developer of the Dlib framework, King (King, 2009), developed a powerful FR model based on the ResNet-34 (He et al., 2016) and presented it in Dlib. This model is built by removing a few layers from ResNet-34 and reducing the number of filters per layer by half

(King, 2017b). To train the created model, around 3 million face images for 7485 people are used and tested on the LFW dataset. Input size of the model is decided as 150x150. Here, the biggest advantage of the model is that faces that are different from 150x150 size are resized to 150x150 and given as input to the network. In this way, unlike other models, it accepts image inputs with different sizes. The model outputs a feature vector (embedding) of 128 bytes for each face. Thus, it is decided whether the given two embeddings belong to the same person by calculating the similarity (distance) of the embeddings. The Euclidean distance is used as the distance metric. The results show that the model provides 0.993 accuracy for FR on LFW. In addition, according to the results, the best classification threshold is 0.6. In this thesis, we call this model as Dlib and present the results as such.

3.3. Clustering Algorithms

Datasets consisting of large data are both difficult to process by machine learning algorithms and to analyze (Sajana, Sheela Rani, & Narayana, 2016). Clustering is a method of grouping heterogeneous data in such a way that similar records according to their characteristics are placed in the same cluster (Verma, Srivastava, Chack, Diswar, & Gupta, 2012). Clustering is an unsupervised method that divides large-volume datasets into subgroups, making it easy to process and facilitates the emergence of existing patterns. Clustering algorithms are basically divided into 5 groups according to the clustering criteria and approaches they use while clustering: partitioning based, hierarchical, density based, grid based, and model based algorithms (Sajana et al., 2016; Xu & Tian, 2015). Among various alternatives, we use k-Means, which is a partitioning algorithm to match identical profiles on cross OSNs. We also experiment with AGNES, a hierarchical clustering algorithm to find the profile owner vectors from all vectors of a given profile

3.3.1. k-Means

Due its simplicity and speed, k-Means is one of the partition-based clustering algorithms that has been tested and used in many studies (Alhawarat & Hegazi, 2018; Dehariya, Shrivastava, & Jain, 2010; Meng, Liang, Cao, & He, 2018; Park, Kim, Jeong, & Youn, 2013). k-Means splits the given dataset into k distinct clusters according to the distance closeness of data. For this, k-Means initially selects k samples from the dataset $X = (x_1, x_2, \dots, x_r)$ where r is the total number of instances. Each selected sample is called a centroid c and represents a cluster C . Then, the algorithm calculates a distance between each x and each c using a distance metric

(Euclidean, cosine, etc.) and assigns each x to the cluster with the closest centroid. After that, the average is taken using all x within each cluster and the new centroid is updated. These steps are repeated until there is no change in the clusters (Verma et al., 2012). In this way, k clusters containing only data that are close to each other are obtained. An example of k-Means (at $k=3$) is given in Figure 3.6 (Pal, 2021). In this thesis, we use the version of k-Means implemented in the Sklearn library (Sklearn.cluster.KMeans — Scikit-Learn 0.21.3 Documentation, 2019).

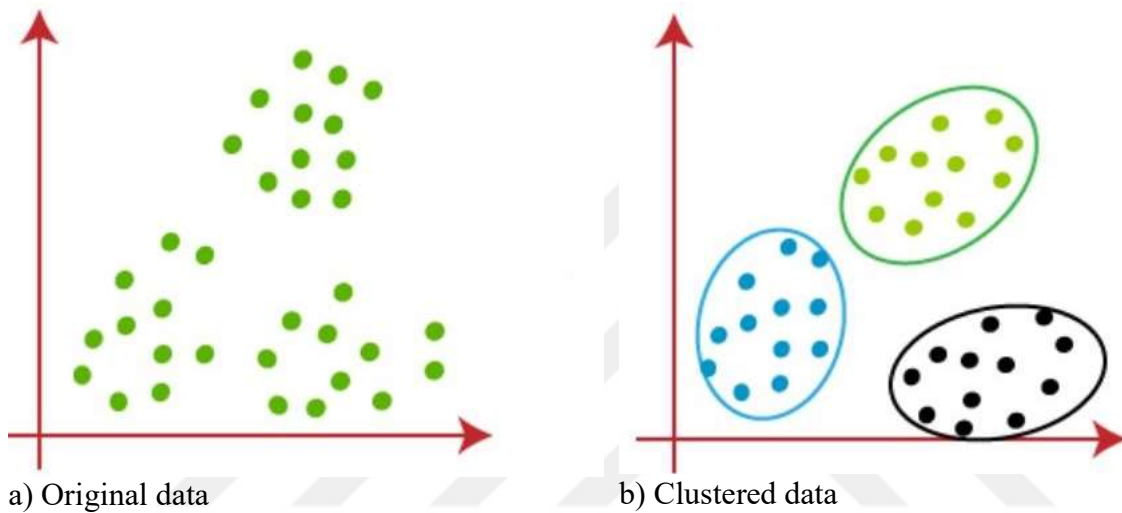


Figure 3.6. K-Means clustering example (Pal, 2021)

3.3.2. Agglomerative Nesting (AGNES)

Agglomerative Nesting (AGNES) is a bottom-up hierarchical clustering method that creates a tree or hierarchy on the data for clustering called a dendrogram (Sajana et al., 2016). Initially, a distinct cluster is created for each record of the dataset X . Then, using an appropriate distance metric (e.g., Euclidean, cosine, etc.) two of the closest clusters are merged at each iteration. AGNES uses three different linkages when finding closest cluster pairs: single, complete and average. Single linkage is calculated based on the similarity between two members in two different clusters, while complete linkage is calculated based on the distance of the two most dissimilar members in the two different clusters. Average linkage is obtained with the average of the distances of each members of the two clusters (Garima, Gulati, & Singh, 2015).

Merged, new cluster is inserted into the dendrogram as the parent node of the clusters combined together. Eventually, only one cluster that contains all records is obtained (Verma et al., 2012).

An example of AGNES is shown in Figure 3.7 (Hierarchical Clustering (Unsupervised Learning Algorithm), 2021). Some studies using AGNES are as follows (Diraco, Leone, & Siciliano, 2015; Ip, Fong, & Liu, 2010; Lung & Zhou, 2010). In this thesis, we use the version of AGNES implemented in the Sklearn library (Sklearn.cluster.AgglomerativeClustering, 2019).

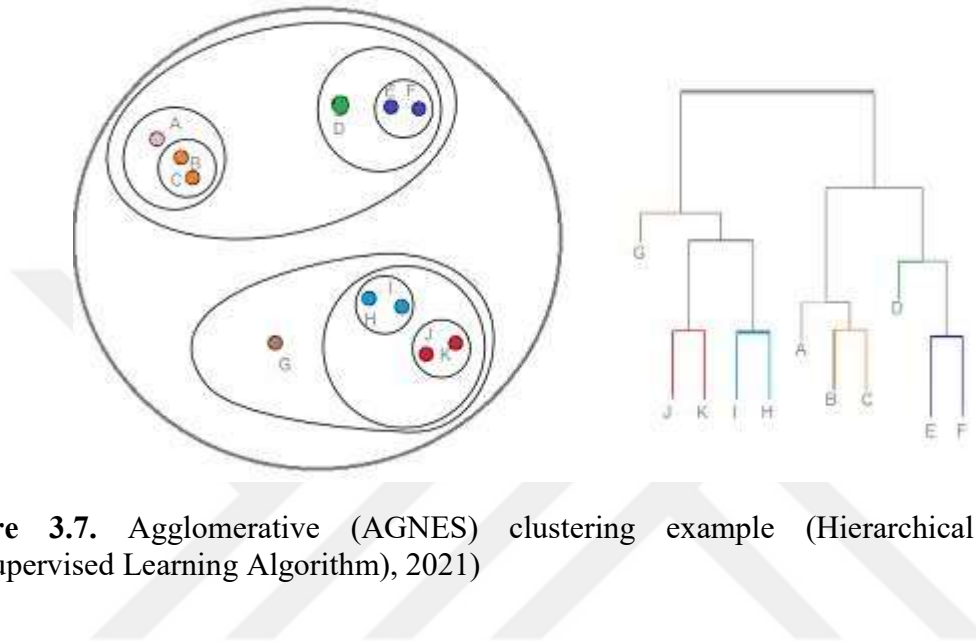


Figure 3.7. Agglomerative (AGNES) clustering example (Hierarchical Clustering (Unsupervised Learning Algorithm), 2021)

3.4. Datasets

The data collected from social media is quite complex, requires pre-processing and is large-volume. OSN data may not be suitable for every algorithm to work both quickly and successfully. Therefore, in this thesis, we use a total of 4 datasets, including 3 datasets that have been studied before and made available to the public, and one that is created in this thesis. These 3 datasets are used to determine the best algorithms/models for the proposed models as well as their ideal parameters

3.4.1. Dlib Dataset

The Dlib dataset (King, 2016a) was created by King, the founder of the Dlib library (King, 2009) and published publicly on Dlib's blog (King, 2016b). This dataset consists of 6975 face images in a wide variety of poses, as well as front-facing shots collected from image datasets such as ImageNet, AFLW, Pascal VOC, VGG, WIDER, which are very popular in image processing studies in the literature. In addition, the dataset contains information about how many faces there are in each picture and in which areas. In this thesis, 2922 images selected

from this dataset are used and some examples are given in Figure 3.8 (King, 2016a).



Figure 3.8. Examples of images in different poses (King, 2016a)

3.4.2. Vggface Dataset

The Vggface (Visual Geometry Group - University of Oxford, 2015) dataset was made publicly available by Parkhi et al. (Parkhi, Vedaldi, & Zisserman, 2015). The dataset contains 1000 images of 2622 celebrities (actor or politician etc.). The celebrities are from different ethnicities and pictures of a celebrity are taken at different times and in different poses. This is the primary reason of our interest in the dataset. In this thesis, 300 randomly selected pictures are used for each celebrity. As an example, pictures of celebrity names Amanda Seyfried and Aamir Khan are given in Figure 3.9.



a) Amanda Seyfried



b) Aamir Khan

Figure 3.9. Some example images from the Vggface dataset (Visual Geometry Group - University of Oxford, 2015)

3.4.3. AboutMe Dataset

One of the challenging problems of this thesis is to identify and verify known identical accounts of a user on cross OSNs, such that the posts and friends' profile photos of the respective user will be collected before working on the OSN. Therefore, it is preferred to use a previously tested and publicly available dataset. In this thesis, the AboutMe (Lim, Lu, Chen, & Min-Yen, 2017) dataset created by Lim et al. (Lim, Lu, Chen, & Kan, 2015) is used. This dataset consists of 15,595 user identities and post IDs for six OSNs (Flickr, Google+, Instagram, Tumblr, Twitter and YouTube). Information is complete for at least 4 of these 6 OSNs for each user.

3.5. Proposed Solution

In this section we present our two-phase proposed solution. The first phase performs a search over the destination OSN accounts based on faces in the photos shared by the source user. The outcome of the search phase is a list of candidate destination profiles that could be identical to the source profile. The second phase is the match phase where the source user's links in the source and destination OSNs are used to further filter and rank candidates obtained in the search phase. For matching links over the respective OSNs, we restrict the match phase to profile photos of the links. The real identical profile of the profile sought is matched over a ranking of the candidates based on number of matched links.

3.5.1. Outline Of The Search Phase

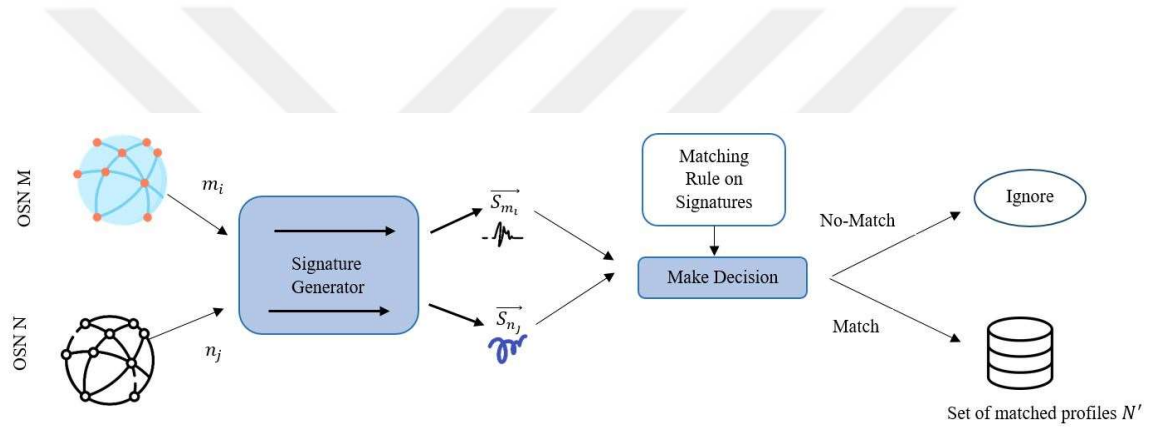
In this section, a demonstration of the search phase is presented. Our search phase is based on the similarity of users' faces, as biometrics of human faces are unique to individuals and are more distinctive than user-generated features. However, extracting faces of the profile owner is not so easily achievable since photos may contain no faces at all, multiple faces where only one or even none belongs to the owner. Our solution for the search phase is based on similarity of the faces detected in photos posted on a profile. Other than the rather innocent assumption that the owner's face appears more often in these photos than any other individual, our solution requires no extra information on the profile owner. It is important to note that studies with so little background information requirement is very limited.

We propose a threshold-based approach in addition to machine learning and deep learning models. We use face detection algorithms (reviewed in Section 3.1) to detect faces in given photo and face recognition models (reviewed in Section 3.2) to create a feature vector from

each face. On two OSNs referred to as M and N, Figure 3.10 outlines the search phase. There are 6 steps to be followed:

- (1) collecting all photos p of an OSN account $m \in M$,
- (2) detecting each face f in each photo p ,
- (3) converting f to fixed length vectors $\vec{f}$,
- (4) identifying profile owners O_{m_i} ,
- (5) generating a signature $\vec{S}_{m_i}$ for each account m_i and,
- (6) searching OSN using generated signatures.

The outcome is a list of profiles that may be identical to each searched account m_i .



In Figure 3.10, the first 5 steps are represented by the “Signature Generator” block to create a unique signature $\vec{S}_{m_i}$ for each m_i . The last step (6) is carried out by the “Make Decision” block, which decides whether two signatures $\vec{S}_{m_i}$ and $\vec{S}_{n_j}$ belong to identical profiles. The workflow for the “Signature Generator” block is given in Figure 3.11.

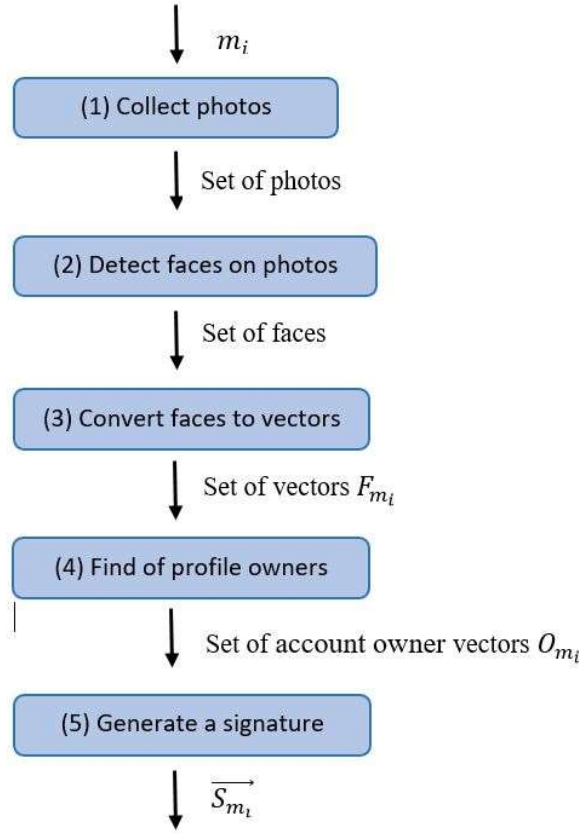


Figure 3.11. Signature Generator

Next, we briefly elaborate on the individual steps.

1. Collection of photos: In this step, each account $m_i \in M$ is visited and each photo p_k shared by m_i is stored. In order to avoid privacy concerns, in this thesis we concentrated only on publicly shared photos.

2. Detection of faces in the collected pictures: Each photo p_k collected from m_i is input to a face detection algorithm and only the areas with the face are cut separately. The result is a set of images cut appropriately.

3. Conversion of detected faces to vectors: Each cut face f_l are transformed into fixed length vectors $\vec{f}_l$ by face recognition model and added to vector set F_{m_i} . The vector length d changes according to the face recognition model. In other words, each face recognition algorithm transforms a p into a different size vector $\vec{p} = \{R\}^d$ based on the model structure it has. The goal here is to measure the similarity of faces converted to vectors using distance calculation metrics

such as Euclidean and Cosine. The distance obtained by measuring the similarity of the similar faces will be closer than the other faces.

4. Identification of profile owners: The problem of having photos of people (friends, spouses, children, etc.) other than the profile owner in OSN profiles is common. Therefore, before deciding whether m_i and n_j belong to the same person, one should determine the account owner of each account using F_{m_i} to obtain and use the only the vectors of the account owner's faces O_{m_i} .

5. Generating a signature for each profile owner: Comparing two OSN profiles m_i and n_j is not easy because (a) $|O_{m_i}|$ and $|O_{n_j}|$ varies, (b) matching every possible pair of face vectors is time consuming, and finally (c) image backgrounds, orientations and scale may be vastly different. To overcome these difficulties and build an efficient and accurate matching rule of profiles, we devise a signature-based solution where each profile's signature depends on a random sample of t face vectors. Even though efficient, this approach requires setting the parameter t - number of face vectors involved in signature generation - appropriately.

Our signature generation scheme is rather straightforward. We propose computing the column-wise average of t vectors of O_{m_i} . The result is the signature $\overrightarrow{S_{m_i}}$ and comparing two signatures, rather than using all vectors, is sufficient.

6. Make decision based on similarity of signatures: Signatures are robust hashes of the owners. But we still require a binary decision mechanism that will decide whether two signatures match. Here, the condition of being identical is decided according to the similarity of signatures $\overrightarrow{S_{m_i}}$ and $\overrightarrow{S_{n_j}}$. Notice that multiple user pairs may be matched by the decision rule. Consequently, we refer to matches of the search phase "candidate matches". These user-wise lists are further refined by the match phase.

3.5.2. Outline Of The Match Phase

In this scenario, a matching approach is followed based on the profile photos of a user's friends. The search phase generates a set of profiles that may be identical with $m_i \in N'$ for each profile m_i . However, there may be more than one profile n within N' and n may not be identical to m_i . As stated in the problem definition, the profiles not identical to m_i may be a profile that creates a fake or cloned account using photos of m_i or that have photos of a person who is very similar

to the owner of the m_i . Therefore, to eliminate and remove non-identical profiles from N' , an extra filtering method is needed.

At this stage, we propose applying a secondary match phase that compares profile photos of connections of m_i denoted $links(m_i)$ with the profile photos of friends of each $n_j \in N'$. Individuals tend to establish links with the same people (in their social circle, work environment or they meet in OSN) on different OSNs. This is a difficult situation for others to imitate and, makes it easier for us to distinguish between profiles.

The match phase is comprised of 5 steps:

- (1) collecting the links $n' \in links(n_j)$ from each n_j ,
- (2) downloading the profile photo p of each profile in $links(n')$,
- (3) detecting the the face f of the downloaded p ,
- (4) converting each detected f to a fixed length vector $\vec{f}$ and adding $\vec{f}$ to vector set of profile photos n'_j , and finally,
- (5) deciding whether the vector sets of two profile photos m'_i and n'_j belong to identical profiles.

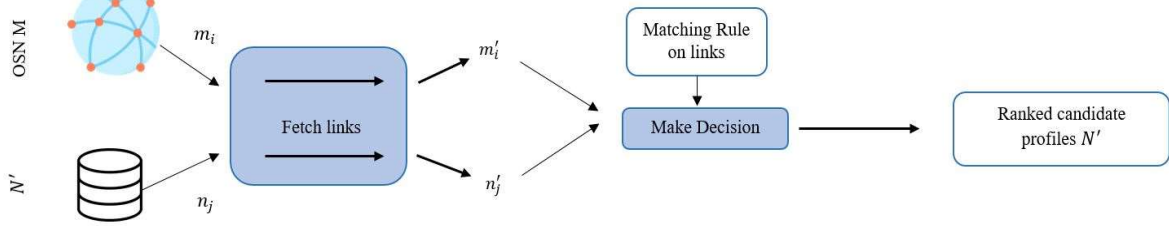


Figure 3.12. Proposed matching model structure based on profile photos of links

An outline of the match phase is shown in Figure 3.12. In this scenario, the first 4 steps are performed in the “Fetch links” block to create a vector set of profile photos of $links(n'_i)$ for each n_i . The last step (5) is applied in the “Make Decision” block, which decides whether m'_i and n'_j are identical profiles. Workflow for fetching links is given in Figure 3.13.

Next, we briefly elaborate on the individual steps.

1. Fetching links: In this step, each account $n_j \in N'$ is visited and its links $links(n_j)$ with the other accounts in N are collected.

2. Downloading profile photo of each link: Once links are collected, current profile photo p of each link in n' is downloaded.

3. Detection of faces in the collected profile photo: In this step, face f on each collected p are detected with the face detection algorithm and only the areas with the face are cut separately.

4. Conversion of detected faces to vectors: Each cut face f is transformed into a fixed length vector $\vec{f}$ as in step 3 of Section 3.5.1. Then each obtained $\vec{f}$ is added to vector set of profile photos of friends n'_j of n_j .

5. Make decision based on vectors set of profile photos: The purpose of this step is to decide whether the vector sets of profile photos of $links(m'_i)$ and $links(n'_j)$ obtained from two different profiles $m_i \in M$ and $n_j \in N'$ belong to identical profiles. For this, the similarity (distance) is calculated between each vector in m'_i and all vectors in n'_j , and this similarity is compared to a matching threshold. At this stage, we rank the all $n_j \in N'$ according to number of matches between m'_i and n'_j , then mark n_j at the top-1 rank as the identical profile of the m_i . If all numbers of matches are equal, we return “no identical profiles found” since no tie-breaker is available.

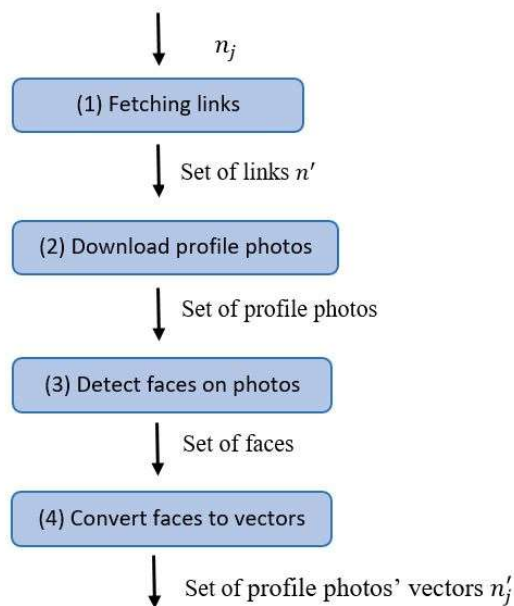


Figure 3.13. Fetch Links

3.5.3. Determination Of The Threshold For Face Recognition

The search and match phases of our proposed solution could be formulated as instances of the face verification (FV) problem – a special case of face recognition. Consider, for example, determining only the vectors of the profile owner O_{m_i} from the face vector set F_{m_i} of each profile m_i in the search phase method. Alternatively, comparing two friends' profile photos to decide whether they are the same person in our match phase is an FV problem. An important step of applying existing FV algorithms is to determine the threshold in the decision rule. Each FR model extracts feature vectors of different sizes for an image, and therefore the FV threshold value of each model will be different. In this section, we describe how we determine the thresholds for the distinct face recognition models we utilize.

In this thesis, we use 3 different face recognition models. Therefore, for each FR, we determine a threshold thr that best distinguishes the distance between a pair of vectors $\vec{f}, \vec{g}$. The distance here is calculated as follows:

$$dis = metric(\vec{f}, \vec{g}) \quad (7)$$

where dis is the distance between the two vectors and $metric$ is a distance calculation method for vector spaces such as Euclidean or cosine.

If dis is less than or equal to thr , $\vec{f}$ and $\vec{g}$ belongs to the same person, and otherwise the vectors may belong to two different people. If thr is set too large, the rate of recognizing two distinct individual's faces as a match (false-positive) would increase. On the other hand, if thr is set too tight, the rate of mismatching the same individual's faces (false-negative) would increase. The aim is to determine a thr that reduces both errors and allows us to make an accurate estimate. This problem is an optimization problem that can be solved with unsupervised or supervised approaches. In this thesis, we prefer to use a supervised approach to determine the most appropriate thr value. For this, a training dataset, which contains faces that are previously labeled for different people are required.

We assume that each person represents an OSN account m . Then, each face f of m is converted to a feature vector $\vec{f}$ by an FR model and added to the vector set F_m of m . Since this is a learning and optimization problem, the generated data set should consist of an equal number of samples for both positive and negative classes (i.e., balanced). Algorithm 1 explains the steps of creating

this dataset and learning the appropriate threshold. The experimental results of the scenario are presented in Section 4.3.

Algorithm 1:

t is the number of vectors selected for each m_i , *Sample* is a list that keeps randomly selected vectors from each vector set F_{m_i} , *Lbl* is used to give each m_i a unique label (initial value 0), *Labels* the list that holds *Lbls*, *Klist* is list that keeps different sub sample size, *Metrics* = { *euclidean*, *cosine* } is the list of distance metrics and x is the total number of accounts used.

```

1: For each  $m_i$  ( $i$  from 1 to  $x$ ) do:
2:    $F_{m_i} \leftarrow$  all face vectors of  $m_i$ 
3:    $F'_{m_i} \leftarrow$  randomly select  $t$  vectors from  $F_{m_i}$ 
4:   Add  $F'_{m_i}$  to Sample
5:   Add Lbl to Labels for  $t$  times and increase it by 1
6: For each  $k$  in Klist do:
7:   Sub_Sample = []
8:   Sub_Labels = []
9:   For  $j=1$  to  $k$ :
10:     $rand \leftarrow$  generate random value  $\in \{0,1\}$ 
11:    if  $rand = 0$ :
12:       $(\vec{s1}, \vec{s2}) \leftarrow$  two random vectors from Sample where  $Labels[\vec{s1}] \neq Labels[\vec{s2}]$ 
13:    else:
14:       $(\vec{s1}, \vec{s2}) \leftarrow$  two random vectors from Sample where  $Labels[\vec{s1}] == Labels[\vec{s2}]$ 
15:    Sub_Labels  $\leftarrow rand$ 
16:    Sub_Sample  $\leftarrow [\vec{s1}, \vec{s2}]$ 
17:   For each mts in Metrics do:
18:     Distances = []
19:     For each ss in Sub_Sample:
20:        $dis \leftarrow mts(ss[0], ss[1])$  and add dis to Distances
21:   Determine threshold thr that separates two classes well using Distances and Sub_Labels

```

In lines 1-5, t vectors are randomly selected F'_{m_i} from each account's vector set F_{m_i} and added to the *Sample* list. At the same time, each m_i is given a unique label *Lbl*, and this *Lbl* is added

to the *Labels* list t times. This way, a vector pool *Sample* is created randomly. Next, a *sub-sample* vector and *sub-labels* lists with different sizes k are generated from the *Sample* and *Labels* lists in lines 6-16. The reason for using different k here is to observe the changes above the threshold when the sample size changes (increases or decreases). Another important point here is that the created *sub-sample* vector list should contain an equal number of samples for each class to be balanced. Therefore, at each step, a random value $rand \in \{0,1\}$ is generated. Thus, a close number of 0's and 1's would be produced. Then, two vectors $(\vec{s1}, \vec{s2})$ are randomly selected from *Sample* based on $rand$ and added to the *sub-sample* as a single row in lines 11-16. If the $rand$ is 0, two vectors belonging to the different people are selected, otherwise, 2 vectors belonging to the same person are selected. After that, $rand \in \{0,1\}$ is added to *sub-labels*. This way, vector pairs belonging to the same person and different people, randomly selected from the *Sample* pool, are obtained. Finally, in lines 17-20, the distance dis is calculated for the pair $(\vec{s1}, \vec{s2})$ in the *sub-samples* for each metric mts and added to the *Distances* list. Then, using *Distances* and *sub-labels*, the thr that best separates the classes is determined in line 21. That is, following a supervised approach, we observe how accurately the classification is made according to the distance values as the threshold changes, and we determine a thr that classifies at the highest level of success according to the given distance.

3.5.4. Profile Owner Identification

OSN users can share a special moment in daily life, both alone and with other people. For this reason, people other than the account owner (his/her spouse, child or friend) can also be in a photo collected from an OSN profile. This makes it difficult for an outsider to identify which person in the photos is the real owner of this profile. For this reason, in order to decide whether two profiles belong to the same person on cross OSNs, first of all, it is necessary to determine the profile owner and make an evaluation using only the profile owner's faces. Thus, we perform a clustering method using thr over vector set F_m of account m to obtain vectors of profile owner O_m . Detailed explanation about clustering methods is given in Section 3.3. Clustering method returns cluster sets $C = \{C_i, i \in [1, r]\}$ where r is the total number of clusters returned by clustering operation. We assume that the profile owner will have more vectors and at the end of clustering, O_m would be $\max(|C_i|)$ where $1 \leq i \leq r$. To measure the performance of the clustering approach applied to find vectors of profile owners O_m , we perform a scenario on faces labelled with their owners.

We assume that each person represents an OSN account m . Each face f of m is converted to a vector $\vec{f}$ by an FR model and added to the vector set F_m of m . We create a simulated profile for each m_i and add the vectors of other profiles with a different number besides the profile owner to the vector set of m_i . Finally, we evaluate the accuracy of the clustering method. The proposed scenario is given in Algorithm 2. The experimental results are presented in Section 4.4.

Algorithm 2:

t is the number of vectors selected for each m_i , y is the number of accounts to be simulated and *Indexes* is the list of indexes of y randomly selected individuals, x is the total number of accounts used and *CA* is the clustering algorithm tested.

1. For $i=0$ to y do:
2. $index \leftarrow$ randomly select an index $r \in [1 \dots x]$ such that r not in *Indexes*
3. Add $index$ to *Indexes*
4. For each i in *Indexes* do:
5. *Vectors* = []
6. *Lables* = []
7. *Lbl* = 0
- 8: $F_{m_i} \leftarrow$ all face vectors of m_i
9. $F'_{m_i} \leftarrow$ randomly select t vectors from F_{m_i}
10. Add F'_{m_i} to *Vectors*
11. Add *Lbl* to *Labels* for t times and increase it by 1
12. $other_people_size \leftarrow$ random $\in [1, 10]$
13. For $j=0$ to $other_people_size$ do:
14. $s \leftarrow$ randomly select a people index $r \in [1 \dots x]$ and $r \neq i$
15. $other_people_sample_size \leftarrow$ random $\in [1, T*2]$
16. $F'_{m_s} \leftarrow$ randomly select $other_people_sample_size$ vectors F_{m_s}
17. Add F'_{m_s} to *Vectors*
18. Add *Lbl* to *Labels* for $other_people_sample_size$ times and increase it by 1
19. *Predictedx* $\leftarrow$ cluster *Vectors* with *CA* using threshold *thr* and take vectors in cluster whose has maximum vectors
20. *Actualx* $\leftarrow$ Find mode of *Labels* and take its vectors from *Vectors*

Random and unique y indexes $index \in [1, x]$ are selected to simulate OSN profiles and added to the *Indexes* list in lines 1-3. Then, t vectors are randomly selected to build F'_{m_i} from each vector set F_{m_i} and added to the *Vectors* list. At the same time, each m_i is given a unique label Lbl , and is added to the *Labels* list t times. This way, a vector pool *Vectors* is created randomly for m_i with the lines 4-11. Line 12 randomly picks how many different people $other_people_size \in [1, 10]$ should be in the profile of m_i apart from the profile owner. In lines 13-18, random index ($s \in [1, x]$ and $s \neq i$) is chosen up to $other_people_size$. Then, we randomly determine the of vectors $other_people_sample_size \in [1, t*2]$ of each selected person m_s will be added to the *Vectors*. The fact that $other_people_sample_size$ is between 1 and $t*2$ indicates that the number of vectors for someone other than the profile owner can be more than the number of vectors of the profile owner. This is a simulation that shows that in a normal OSN profiles, a profile has more pictures of someone other than the profile owner. Then, vectors are randomly selected into F'_{m_s} from a person's vector list F_{m_s} are added to the *Vectors*. After that, m_s is given a unique label Lbl , and is added to the *Labels* $other_people_sample$ vectors times. This way, different numbers of vectors of more than one person are added in the *Vector* and it becomes suitable for representing an OSN profile. In line 19, the vectors in *Vectors* are clustered with the clustering algorithm *CA* using the face verification threshold thr learned in Section 4.3 for each scenario. It is expected that vectors whose distance is less than or equal to thr are included in the same cluster. After the clustering operation is performed, the vectors in the cluster with the highest number of vectors *Predictedx* are retrieved. By taking the mode of the *Labels* in line 20, it is determined which person's vector is the most in the *Vectors* and these vectors are taken to be *Actualx*. The assumption here is that the person with the most vectors owns that profile.

3.5.5. Determination Of The Face Recognition Algorithm And Matching

To find most suitable matching approach and face recognition model, 3 different approaches with 3 face recognition models are tested in order to determine the right model structure that can be run on OSNs in terms of both accuracy and time. The 3 different approaches tested here are (i) a supervised algorithm, (ii) an unsupervised algorithm, and finally (iii) a rule-based face recognition are performed.

In the proposed search phase, we mention the main problems with comparing two profiles on cross OSNs, m_i and n_j . One of these is the $|O_{m_i}|$ and $|O_{n_j}|$ could be different in size, and the

other the determination of the number $t \in [1, |O_{m_i}|]$ of faces to be used for comparison purposes. To answer these questions, two different cases are tested on labelled face sets. With this scenario, face set of each person is divided into two disjoint subsets and each subset is used to represent OSN accounts $m \in M$ and $n \in N$. Thus, same index for i and j shows that m_i and n_j are identical profile. For example, when $i=0$ and $j=0$, m_i and n_j are identical profiles. Then, each face f of m is converted to a vector $\vec{f}$ by an FR model and added to the vector set F_m of m . Here, a searched person $m \in M$ and a search space $n \in N$ are created for each case. Each m_i is searched in N and whether there is an identical person is tested. The goal here is to simulate two different OSNs crosswise. In the first case, while $|N|$ is the same at each step, the number of vectors used t for each person m_i increases. The aim is to see how the performance of matching identical people changes as the t increases. In the second case, while the t is the same at each step, $|N|$ increases. The aim here is to observe the success of the models in matching identical people as the $|N|$ increases. Since the number of vectors $|O_{m_i}|$ and $|O_{n_j}|$ could be different, $t/4$ vectors of each m_i are used, while t vectors are used for each n_j for both cases. This represents holdout method by creating 80% train set and 20% test set. The execution steps of supervised algorithm, unsupervised algorithm and rule-based approach for the first case are given in Algorithm 3. The experimental scenario is carried out in Section 4.5.

Algorithm 3:

y is the number of accounts to be simulated and *Indexes* is the list of indexes of y randomly selected individuals, x is the total number of accounts used, and *Tlist* is the list of t values tested.

1: For $k=0$ to y do:

2: $index \leftarrow$ randomly select an index $r \in [1,x]$ such that r not in *Indexes*

3: $i \leftarrow index$

4: $j \leftarrow index$

5: Add (i,j) to *Indexes* // m_i and n_j are identical profiles

6: For t in *Tlist* do:

7: *VectorsM* = []

8: *LabelsM* = []

9: *VectorsN* = []

10: *LabelsN* = []

```

11:  Lbl = 0
12:  For each (i,j) in Indexes do:
13:       $F'_{n_j} \leftarrow$  randomly select t vectors from  $F_{n_j}$ 
14:      Add  $F'_{n_j}$  to VectorsN
15:      Add Lbl to LabelsN for t times
16:       $F'_{m_i} \leftarrow$  randomly select t/4 vectors  $F_{m_i}$  where  $|F'_{n_j} \cap F'_{m_i}| = 0$ 
17:      Add  $F'_{m_i}$  to VectorsM
18:      Add Lbl to LabelsM for t/4 times and increase it by 1
19:      -----perform supervised algorithm-----
20:      Train supervised algorithm with VectorsN and LabelsN
21:      -----perform unsupervised algorithm-----
22:      ClusterLabels  $\leftarrow$  Cluster VectorsN with unsupervised algorithm and return cluster
labels of VectorsN
23:      -----perform rule-based face recognition-----
24:      PredictionThreshold = []
25:      For z=0 to length of VectorsM do:
26:          MatchedLabels = []
27:          For c=0 to length of VectorsN do:
28:              dis  $\leftarrow$  distance_metric(VectorsMz, VectorsNc)
29:              if dis  $\leq$  thr:
30:                  Add LabelsNc to MatchedLabels
31:          Add mode of MatchedLabels to PredictionThreshold

```

In lines 1-5, *y* random and unique indices $index \in [1,x]$ are selected to simulate *M* and *N* and are added to *Indexes* as a pair (*i, j*). Here, agreeing indices of *i* and *j* imply that m_i and n_j are identical profiles. So, an identical pair of profiles is selected at each step. Then, *t* vectors are randomly selected F'_{n_j} from each person's vector list F_{n_j} and added to the *VectorsN* list. At the same time, each n_j is given a unique label *Lbl*, and is added to the *LabelsN* list *t* times. In addition, *t/4* vectors are also randomly selected F'_{m_i} from F_{m_i} and added to the *VectorsM* list. Also, *Lbl* is added to the *LabelsM* list *t/4* times. This way, two distinct vector pools are created randomly in lines 6-18. After that, the supervised algorithm is trained with *VectorsN* and

LabelsN in line 20. In line 22, *VectorsN* is clustered with the unsupervised algorithm (here, $K = y$) and a list that contains the cluster label of each vector is obtained as *ClusterLabels*. Finally, threshold-based search is being tested in lines 24-31. Each vector *VectorsM_z* is searched in *VectorsN*. The search is performed by measuring a distance *dis* between each *VectorsM_z* and *VectorsN_c* vectors. If *dis* is less than or equal to the threshold *thr*, label of *VectorsN_c* (*LabelsN_c*) is added to the list of *MatchedLabels_j* maintained for each *VectorsM_j*. Here *distance_metric* and *thr* shows the appropriate metric for each FR model in Section 4.3. In line 29, the most matched label is determined by taking the mode of the *MatchedLabels_j* list and added to the *PredictionThreshold* list for the *VectorsM_j*.

3.5.6. Search Phase And Signature Generation

Different matching approaches to match identical profiles using vectors of profile owners from each profile are described in Section 3.5.5. In these approaches, it is shown that a certain number of vectors belonging to the profile owners should be used for each profile and that stages such as training, clustering or threshold based search should be applied using these vectors. However, there are some barriers to the suitability of running these approaches on OSNs.

With supervised (resp. unsupervised) methods, adding a new profile to the OSN requires repeating the training (resp. clustering) steps. Additionally, these methods cannot return null answers indicating the search is unsuccessful.

Threshold-based matching has its own barriers. Similarity between every pair of vectors of every pair of profiles should be evaluated. The brute-force complexity is prohibitively high.

In order to eliminate these obstacles, in this thesis, we recommend to use a certain number t of vectors of the profile owners O_{m_i} for each profile m_i , and to average them column-wise, thus creating a single signature $\overrightarrow{S_{m_i}}$. Quite naturally, the search phase operates over these signatures and similarity in signature pairs should indicate identical profiles. Since signatures have the same dimensionality as a single face vector, complexity is reduced dramatically. Additionally, there will be no need for retraining or re-clustering since the similarities are calculated only over signatures, and if a profile has more than one identical profile, it can be found.

A signature is generated according to Equation 8 below.

$$\overrightarrow{S_{m_{ij}}} = \frac{1}{t} \sum_{j=1}^t O_{m_{ij}}[x], \quad \forall x \in [1, d] \quad (8)$$

where $O_{m_{ij}}$ shows the j^{th} vector of O_{m_i} and d is the length of each face vector $\vec{f}$.

We next focus on the comparison two signatures $\vec{S_{m_i}}$ and $\vec{S_{n_j}}$. We propose a threshold based face verification approach and determine the threshold $thrSig$ that best separates the distance between two signatures $\vec{S_{m_i}}$ and $\vec{S_{n_j}}$. The distance here is calculated as

$$dis = metric(\vec{S_{m_i}}, \vec{S_{n_j}}) \quad (9)$$

where dis is the distance between the two signatures, and $metric$ is one of Euclidean, cosine.

If dis is less than or equal to $thrSig$, we can decide that the signatures $\vec{S_{m_i}}$ and $\vec{S_{n_j}}$ belong to the same profile, otherwise, these can be belong two different profiles. As in Section 3.5.3, it is very important to determine the $thrSig$ that provides the best success rate. To determine the optimal $thrSig$ value, Algorithm 1 is used. The only difference is in how the initial *Sample* list is created. Instead of using t vectors for each m_i , y distinct signatures are created using Equation 8 for each m_i and added to the *Sample* list. Here, each $\vec{S_{m_i}}$ is generated from randomly selected t vectors from F_{m_i} . All remaining steps are performed in the same way for $thrSig$.

Algorithm 4:

LabelsM is the list that keeps label of each m_i , *LabelsN* is the list that keeps label of each n_j , *Prediction* indicates the list that keeps labels of identical users predicted of each m_i on N , *Null identifier* means that m_i does not match with any n_j and $metric \in \{ euclidean, cosine \}$ shows the distance measurement metric

- 1: For each $\vec{S_{m_i}}$ (i from 1 to $|M|$) do:
- 2: For each $\vec{S_{n_j}}$ (j from 1 to $|N|$) do:
- 3: $dis \leftarrow metric(\vec{S_{m_i}}, \vec{S_{n_j}})$
- 4: if $dis \leq thrSig$:
- 5: Add label of n_j in *LabelsN* to *Prediction*
- 6: if m_i does not match with any n_j , add *Null identifier* to *Prediction*

Once the appropriate *thrSig* and distance measurement metric has been determined, Algorithm 4 is applied to search for identical profiles on M and N . For each m_i in M , and each n_j in N we compare the distance between the signatures $\overrightarrow{S_{m_i}}$ and $\overrightarrow{S_{n_j}}$ in line 3 and check if the distance exceeds the threshold in line 4. If *dis* is less than or equal to *thrSig*, m_i is matched with n_j as an identical user by adding label of n_j to *Prediction*. If there is no match after $\overrightarrow{S_{m_i}}$ has been compared with all $\overrightarrow{S_{n_j}}$, *Null identifier* is added to *Prediction* in line 6.

3.5.7. Match Phase

One of the greatest opportunities provided by OSNs is that interpersonal communication always remains active regardless of the medium. It is created by establishing a link with the accounts of people in the social circle of each user on this OSN. As a result of this situation, it is definitely conceivable that the links created by a user in two different OSN platforms may be similar to each other. Thus, in this section, we decide whether two profiles m_i and $n_j \in N'$ are identical profiles by matching the link sets of the two profiles $m' \in \text{links}(m_i)$ and $n' \in \text{links}(n_j)$. For this, using the proposed model in Section 3.5.2, the profile photo p of each link in n' is downloaded and a set of vectors of profile photos n'_j is created for each account n_j by an FR model. Then, we calculate how much m'_i and n'_j match, and decide whether m'_i and n'_j belong to identical profiles based on the number of matches. Identical profiles are expected to have more matches than non-identical profiles. The matching process is given Algorithm 5. Here, a similarity (distance) *dis* is calculated between each vector in m'_i and all vectors in n'_j , and then it is decided whether a vector in m'_i matches any vector in n'_j using the face verification threshold *thr* mentioned in Section 3.5.3. Then, the total number of matches between m'_i and n'_j is determined.

Algorithm 5:

numberOfcommonlinks is the total number of matches between m'_i and n'_j , and *metric* $\in \{euclidean, cosine\}$ shows the distance measurement metric

1: *numberOfcommonlinks* $\leftarrow 0$

2: For $w=0$ to $|m'_i|$) do:

3: For $z=0$ to $|n'_j|$) do:

```

4:       $dis \leftarrow metric(m'_i[w], n'_j[z])$ 
5:      if  $dis \leq thr$ :
6:           $numberOfcommonlinks ++$ 
7:          break
8: return  $numberOfcommonlinks$ 

```

We consider multiple different strategies to determine the identical profile.

In our first strategy, also stated in the Section 3.5.2, we rank all the $n_j \in N'$ according to the number of matches and assign n_j at the top-1 rank as identical with m_i .

In our second strategy, we use the link-matched threshold $thrLink$. If the number of matches is greater or equal $thrLink$, we return 1, otherwise 0. Returning 1 indicates that m'_i and n'_j are identical profiles. Algorithm 6 outlines a supervised algorithm to determine the optimal $thrLink$ value. The match phase is performed on the vector sets of the profile photos of each m_i and its identical profile $n \in N$, and the number of matching vectors $numberOfcommonlinks_identical$ is determined. The same step is performed on another profile n that is not identical to m_i and the number of matching vectors $numberOfcommonlinks_not_identical$ is calculated. Each match count is added to the *Sample* list. This way, the match set is obtained by comparing the profile photo vector sets obtained from both identical and non-identical profiles. At the end, the ideal $thrLink$ value is determined to distinguish identical profiles according to the number of matches.

Algorithm 6:

Sample is the list that keeps the number of matches between each m'_i and both identical and non-identical profiles of m_i , *labels* represents the list that keeps label of selected pairs and T is the randomly selected vectors from each m'_i

1: For each m_i (i from 1 to x) do:

2: $SetM \leftarrow$ randomly select T vectors from m'_i

3: $SetN_identical \leftarrow$ randomly select T vectors from vector set of profile photos of $n \in N$ which is the identical profile of m_i

4: $numberOfcommonlinks_identical \leftarrow$ perform Algorithm 5 with $SetM$ and $SetN_identical$

5: *Add numberOfcommonlinks_identical to Sample*
 6: *Add 1 to Labels*
 7: *SetN_not_identical $\leftarrow$ randomly select T vectors from vectors set of profile photos of $n \in N$ which is not identical profile of m_i*
 8: *numberOfcommonlinks_not_identical $\leftarrow$ perform Algortihm 5 with SetM and SetN_not_identical*
 9: *Add numberOfcommonlinks_not_identical to Sample*
 10: *Add 0 to Labels*
 11: *Determine threshold thrLink that is separate two class well using Sample and Labels*



4. RESULTS AND DISCUSSIONS

This chapter presents the basic statistics of the dataset we created for empirical analysis and the results of the experimental evaluation we conducted.

4.1. Dataset Collection

In this section, we explain the detailed steps of our data collection efforts and present the statistics of the resulting OSN samples. Since we aim to match identical users in cross OSNs using face recognition methods, we prefer to use OSNs where users tend to share more photos and user connections can be accessed easily. Our natural choice is to use Facebook as OSN M and Instagram as OSN N . To create a common base of users from these two OSNs, we rely on the AboutMe dataset of Section 3.4.3, which contains the identity information of 15,595 users' Flickr, Google+, Instagram, Tumblr, Twitter, and YouTube accounts.

Our dataset contains a comprehensive image pool consisting of poses taken from various angles of people from different ethnicities and different age ranges. The AboutMe dataset contains 11,922 users with Instagram user ID information. To identify users who are still active and have public profiles, we implement a crawler in the Python programming language and visit each profile n_j of these 11,922 users. Since the links $n' \in \text{links}(n)$ on N - Instagram is one-way, each profile n_j has two different friendship lists: $n' \in \text{followers}(n_j)$ shows the list of people who follow n_j , $n'' \in \text{followings}(n_j)$ shows the list of people that n_j follows. We set $n^* = n' \cap n''$ worked with those profiles supporting $n^* > 200$. Since n^* contains followers that are followed back, the links contained in n^* are much stronger and therefore much more suitable for the match phase. This filter has left us with 5,614 profiles.

As the source OSN M , we work with Facebook. People share a lot of pictures other than text in Facebook and accessing friend lists is easy if public visibility is allowed. The AboutMe dataset does not readily list Facebook accounts of participating individual. We implement a crawler in the Python programming language to browse the AboutMe sites of the 5,614 Instagram users qualified above, and collect their Facebook profile. 3,420 of these accounts shared their Facebook information on their AboutMe sites, out of which the Facebook accounts of only 1,096 are still active. We visit each active profile $m_i \in M$, and detect 512 accounts with public

friends list and public profile. In this thesis, we use 2500 of the 5614 qualified Instagram accounts and 500 of 512 qualified Facebook accounts.

The friendship connections on Facebook are bi-directional, which implies there is only one friendship list m' for each m_i . Once again, we develop two separate crawlers in Python for M and N to visit each $m = \{m_i, i \in [1, 500]\}$ and $n = \{n_j, j \in [1, 2500]\}$ and download the photos shared by each m_i and the profile photos of m' . The first crawler collects the URLs of photos for each m_i . If m_i shared more than 1500 photos, we restrict ourselves to 1500 randomly chosen posts, otherwise we download all photos available. The second crawler downloads profile photos of each connection m' . Detailed information about the dataset created is given in Table 4.1.

Table 4.1. Detailed information about the dataset created

Term	Amount
# of accounts in M	500
# of accounts in N	2500
# of identical users ($M \cap N$)	500
Total # of photos from M	201,216
Total # of photos from N	1,901,848
Average # of photos for each m	402
Average # of photos for each n	760
Average # of photos for each $m \in (M \cap N)$	402
Average # of photos for each $n \in (M \cap N)$	762
Average m' for each m	430
Average n' for each n	895
Average n'' for each n	852
Average n^* for each n	264
Average m' for each $m \in (M \cap N)$	430
Average n^* for each $n \in (M \cap N)$	415

An average Instagram user n_j shares more photos than an average Facebook user m_i (760 vs. 402). Additionally, although an average n_j has more followers and followees, the number of

mutual friends on both lists n^* is less than the average m' . This shows that the followers of Instagram users differ significantly from those who follow them.

4.2. Determination Of The Face Detection Algorithm

In this section, we test 5 popular face detection algorithms: YOLOv3, DNN, HoG, Cascade, and MTCNN. For the experiment scenario, 2922 images are selected and used from the Dlib dataset consisting of 6975 photos of Section 3.4.1. Each photo p in the dataset contains a different number of faces f and the pixel $\{x, y, width, height\}$ of each f is labelled where x and y are the upper left corner coordinates of f , $width$ is the width of f , $height$ is the height of f . We run 5 face detection algorithms on each p .

The performance of face detection algorithms can vary depending on the size of the photo, the angular degrees of the faces, and the brightness of the photo. In order to measure the performance of the algorithms in different scenarios, each randomly selected p from 2922 photos is tested in different sizes, by rotating at different degrees and at different brightness. Each randomly selected p is resized separately to 120x120, 220x220, 330x330 and 480x480 dimensions, rotated clockwise at 45, 90, 135, 180 degrees, and finally the brightness of the picture is changed to $\pm 50, 100, 150$ and saved. In addition to these, we also keep the original versions of the pictures, which have not been modified, for testing. As a result, these 5 algorithms are tested under 15 different conditions in 4 different scenarios. Some examples are given in Figure 4.1. The results are presented in Table 4.2.

The evaluation metrics used are as follows:

$$\text{Accuracy (acc)} = \frac{TP}{n} \quad (10)$$

$$\text{Error per face (epf)} = \frac{FP+F}{n} \quad (11)$$

$$\text{Error per image (epi)} = \frac{FP+F}{\# \text{ of sample images}} \quad (12)$$

where true-positive (TP) is the number of correctly captured faces, false-positive (FP) indicates the number of faces detected even though there is no face, false-negative (FN) is the number of faces that are not detected, and n represents the total number of faces on 2922 photos.

Equation 10 presents the total rate of correctly captured faces, Equation 11 is the rate of incorrectly detected faces, and Equation 12 is the percentage of incorrectly detected faces.

As the fourth evaluation metric, execution time (ex) shows the total run time for 2922 photos, and lastly, average execution time (aex.) shows the execution time for a single photo p assuming uniform distribution.



a) original



b) Images rotated 180 degrees



c) Pictures with brightness reduced by 50

Figure 4.1. Examples of different face detection scenarios

Analyzing the results of images of different sizes, we observe that as the image size decreases, there is a decrease in accuracy. For the execution time, especially with HoG, Cas and MTCNN, the execution time increased as the size increased. Here, while the original size and 480x480 size images provide almost the same accuracy, it is seen that the execution time on the original size images is less.

Table 4.2. Face detection results

	Size				Rotating				Brightness						Org.
Algorithm	120	220	330	480	45	90	135	180	50	100	150	-50	-100	-150	
YOLOv3	0.87	0.90	0.91	0.91	0.81	0.67	0.73	0.76	0.89	0.86	0.80	0.90	0.84	0.70	0.91
	0.16	0.13	0.13	0.12	0.22	0.36	0.29	0.27	0.14	0.17	0.23	0.14	0.20	0.32	0.13
	0.17	0.14	0.14	0.13	0.23	0.38	0.32	0.29	0.15	0.18	0.25	0.15	0.21	0.34	0.14
	1604	1613	1417	1425	1317	1345	1301	1296	1283	1295	1337	1297	1368	1321	1247
	0.54	0.55	0.48	0.48	0.45	0.46	0.44	0.44	0.43	0.44	0.45	0.44	0.46	0.45	0.42
DNN	0.93	0.94	0.94	0.94	0.85	0.71	0.75	0.75	0.93	0.90	0.84	0.92	0.87	0.75	0.94
	0.12	0.11	0.11	0.11	0.20	0.33	0.30	0.29	0.12	0.14	0.19	0.13	0.17	0.28	0.11
	0.13	0.12	0.12	0.12	0.21	0.35	0.32	0.31	0.12	0.15	0.20	0.14	0.18	0.30	0.12
	151	193	159	159	156	156	159	150	175	155	165	160	177	207	117
	0.05	0.06	0.05	0.05	0.05	0.05	0.05	0.05	0.06	0.05	0.05	0.05	0.06	0.07	0.04
HoG	0.45	0.48	0.48	0.49	0.35	0.24	0.25	0.28	0.48	0.47	0.44	0.48	0.44	0.37	0.48
	0.54	0.52	0.52	0.51	0.64	0.75	0.74	0.72	0.52	0.53	0.56	0.52	0.56	0.62	0.51
	0.58	0.56	0.56	0.55	0.69	0.81	0.80	0.77	0.56	0.57	0.60	0.56	0.60	0.67	0.55
	32	91	200	407	118	117	117	117	118	118	117	117	120	117	118
	0.01	0.03	0.06	0.14	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04	0.04
Cascade	0.12	0.40	0.42	0.429	0.22	0.21	0.21	0.22	0.40	0.38	0.31	0.41	0.38	0.34	0.41
	0.87	0.59	0.58	0.587	0.78	0.78	0.79	0.78	0.60	0.62	0.68	0.59	0.62	0.66	0.59
	0.94	0.64	0.62	0.631	0.84	0.84	0.85	0.84	0.65	0.67	0.73	0.64	0.66	0.71	0.63
	50	61	76	116	64	64	63	64	65	64	64	64	65	61	65
	0.01	0.02	0.02	0.040	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02
MTCNN	0.84	0.85	0.86	0.86	0.75	0.54	0.58	0.56	0.84	0.77	0.65	0.84	0.79	0.65	0.86
	0.18	0.19	0.18	0.19	0.29	0.47	0.45	0.46	0.20	0.25	0.37	0.20	0.25	0.36	0.18
	0.20	0.20	0.20	0.21	0.31	0.51	0.48	0.50	0.21	0.27	0.39	0.22	0.27	0.39	0.20
	2902	2885	3246	3464	2795	2802	2782	2781	2785	2758	2697	2819	2799	2730	2800
	0.99	0.98	1.11	1.18	0.95	0.95	0.95	0.95	0.95	0.94	0.92	0.96	0.95	0.93	0.95

As the angle of the faces in the images is rotated clockwise from 0 to 180 degrees, there is a remarkable decrease in the performance of the algorithms, especially the lowest performance is seen at 90-degree angle. Apart from that, rotating pictures give almost the same execution times as the original sized pictures.

We also observe that increasing or decreasing the brightness in the images adversely affects the performance of the algorithms.

While DNN achieves the best performance in almost all cases, YOLOv3 shows the closest performance. However, considering execution times, it is seen that DNN outperforms YOLOv3 with a much lower execution time. Cascade provides the worst result.

Based on these results, we prefer employing DNN for face detection and use the original images without preprocessing – resizing in particular.

4.3. Determination Of The FV Threshold For FR Models

In this section, the FV threshold thr that decides whether the pair of vectors $(\vec{f}, \vec{g})$ of faces (f, g) collected from OSN profiles belong to the same person according to the distance between $\vec{f}$ and $\vec{g}$, is determined.

We test 5 scenarios created with the Vggface dataset and 3 FR models separately using the Algorithm 1 of Section 3.5.3. In this scenario, the number of vectors t selected for each m_i is 30. Each experiment is run for three different pool sizes $klist$: 20.000, 50.000, 100.000. The purpose of this is to see the change in thr as the number of samples increases and to determine an overall thr for each model. We use the distance metrics alternatives Euclidean, cosine.

The results for the 5 scenarios are given in Tables 4.3, 4.4, 4.5, 4.6 and 4.7, respectively. Recall, precision, accuracy, f-score and execution time are given for each result. Figure 4.2 shows the effects of the threshold with respect to accuracy, precision, recall and f-score on a Facenet scenario. According to the results, as threshold gets tighter, precision increases, while recall, accuracy and f-score decrease. On the other hand, if threshold is too high, recall increases, but precision, accuracy and f-score decrease. It seems appropriate to choose the threshold that will maximize the f-score for a balanced result and evaluation. Thus, the threshold value given here is selected to maximize f-score. Finally, the distance metric and thr , which provides the best results with the specified model, are determined.

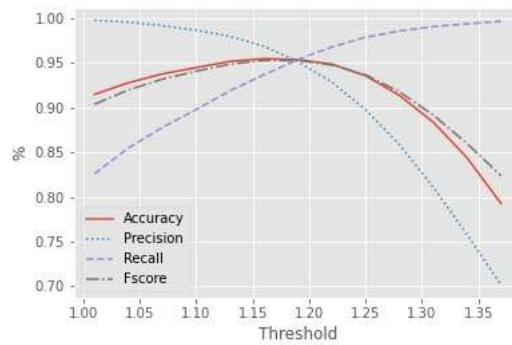


Figure 4.2. Facenet scenario with 100000 k and euclidean metric

The results show that in all scenarios, the Euclidean distance metric takes less time to run than Cosine distance metric. When evaluated on the basis of f-score, it is seen that Euclidean performs better in studies with Dlib, and Cosine performs better in studies with Vggface model. In the Facenet scenario both metrics provide almost the same results. In comparison on a model basis, it is seen that Dlib is superior to the other two models. In addition, 3 different studies with Dlib show that resizing face images to a different size than the original causes a slight decrease in f-score. Finally, increasing the sample size does not have much effect on the threshold and f-score.

Table 4.3. Face verification threshold for Dlib with original image size

Sample Size	20k		50k		100k	
Distance Metric	Euclidean	Cosine	Euclidean	Cosine	Euclidean	Cosine
Threshold	0.655	0.097	0.660	0.099	0.658	0.098
Recall	0.989	0.984	0.985	0.978	0.985	0.977
Precision	0.987	0.982	0.985	0.979	0.987	0.985
Accuracy	0.989	0.984	0.985	0.979	0.987	0.981
F-score	0.988	0.983	0.985	0.978	0.986	0.981
Time	0.1576	0.7171	0.4089	1.7932	0.7999	3.4399
Determined parameters	Metric	Euclidean	threshold	0.65		

Table 4.4. Face verification threshold for Dlib with 80x80 image size

Sample Size	20k		50k		100k	
Distance Metric	Euclidean	Cosine	Euclidean	Cosine	Euclidean	Cosine
Threshold	0.617	0.080	0.610	0.080	0.600	0.080
Recall	0.979	0.953	0.973	0.947	0.965	0.948
Precision	0.969	0.976	0.971	0.977	0.979	0.975
Accuracy	0.974	0.966	0.973	0.964	0.973	0.963
F-score	0.974	0.965	0.972	0.962	0.972	0.961
Time	0.1885	0.7181	0.4268	1.7453	0.7967	3.3273
Determined parameters	Metric	Euclidean	threshold	0.610		

Table 4.5. Face verification threshold for Dlib with 120x120 image size

Sample Size	20k		50k		100k	
Distance Metric	Euclidean	Cosine	Euclidean	Cosine	Euclidean	Cosine
Threshold	0.610	0.080	0.610	0.085	0.620	0.080
Recall	0.963	0.942	0.963	0.959	0.973	0.942
Precision	0.968	0.966	0.972	0.959	0.963	0.971
Accuracy	0.967	0.956	0.968	0.960	0.969	0.959
F-score	0.965	0.954	0.968	0.959	0.968	0.956
Time	0.1666	0.6832	0.4119	1.7134	0.8089	3.3910
Determined parameters	Metric	Euclidean	threshold	0.61		

Table 4.6. Face verification threshold for Facenet model

Sample Size	20k		50k		100k	
Distance Metric	Euclidean	Cosine	Euclidean	Cosine	Euclidean	Cosine
Threshold	1.180	0.710	1.180	0.687	1.160	0.670
Recall	0.942	0.950	0.950	0.945	0.937	0.936
Precision	0.965	0.958	0.956	0.962	0.969	0.970
Accuracy	0.955	0.956	0.955	0.955	0.955	0.955
F-score	0.953	0.954	0.953	0.953	0.953	0.953
Time	0.2187	0.9605	0.4530	2.3745	0.8592	4.7513
Determined parameters	Metric	Euclidean	threshold	1.17		

Table 4.7. Face verification threshold for Vggface model

Sample Size	20k		50k		100k	
Distance Metric	Euclidean	Cosine	Euclidean	Cosine	Euclidean	Cosine
Threshold	122.770	0.580	123.160	0.590	123.340	0.590
Recall	0.889	0.933	0.895	0.939	0.890	0.937
Precision	0.950	0.969	0.943	0.963	0.941	0.961
Accuracy	0.923	0.953	0.924	0.954	0.919	0.951
F-score	0.919	0.951	0.919	0.951	0.915	0.949
Time	0.2187	1.0310	0.5311	2.5776	1.0645	5.1098
Determined parameters	Metric	Cosine	threshold	0.59		

4.4. Determination Of The Profile Owner

In this section, we evaluate Algorithm 2 of Section 3.5.4 for finding the profile owner vectors $O_{m_i} \in F_{m_i}$ of each m_i . 5 different scenarios are tested under the assumption that the account owner's face appears the most often. To simulate OSN profiles, number of accounts to be simulated y is set to 1000, total number of accounts used x is set to 2622. The number of vectors t selected for each m_i is 30, clustering algorithm (CA) is AGNES and linkage criteria is single. The reason why we prefer AGNES here is that it does a hierarchical clustering process only according to the distance threshold, without the need for a fixed cluster number as an input at the beginning. Lastly, the distance measurement metric and thr values used in the clustering process for each scenario are selected according Section 4.3.

The results of the 5 different scenarios are given in Tables 4.8. Our evaluation criteria includes recall, precision, accuracy and f-score, as well as total execution time and the average time taken to find each O_{m_i} (avg. time). The evaluation metrics are as follows:

$$True\text{-}positive (tp): |Actual \cap Predicted| \quad (13)$$

$$False\text{-}positive (fp): |Predicted - Actual| \quad (14)$$

$$False\text{-}negative (fn): |Actual - Predicted| \quad (15)$$

$$True\text{-}negative (tn): |Vectors| - tp - fp - fn \quad (16)$$

where *Actual* is O_{m_i} , *Predicted* is the vector set return by CA and *Vectors* represents the vector pool created to simulate profile of each m_i .

Tablo 4.8. Profile owner determination results

Algorithm	Dlib with original size	Dlib with size 80	Dlib with size 120	Facenet with size 160	Vggface with size 224
Total time	21.1074	24.0961	23.6286	33.0601	173.2848
Avg. time	0.0211	0.0241	0.0236	0.0331	0.1733
Recall	0.889	0.772	0.786	0.767	0.756
Precision	0.901	0.787	0.801	0.787	0.774
Accuracy	0.940	0.871	0.881	0.867	0.863
Fscore	0.894	0.777	0.792	0.775	0.762

The results show that Dlib outperforms the other models in terms of both time and accuracy for finding the profile owner. The study of Dlib on 2 different resized images shows that resizing adversely affects success. Although Facenet and Vggface produce almost the same results in terms of accuracy, it is seen that the Vggface model takes longer time.

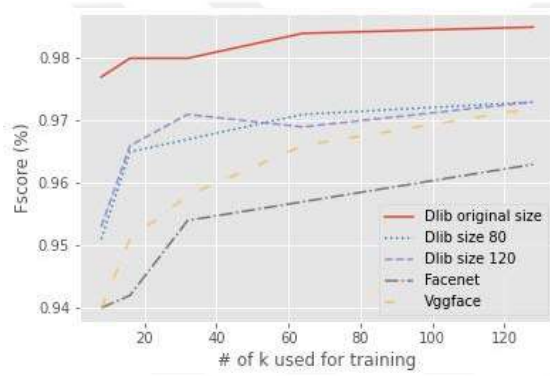
4.5. Testing Alternative Approaches For The Match Phase

In Section 3.5.5, we presented 3 different approaches for face recognition. Each approach is run separately for 5 different scenarios using Algorithm 3 and two different OSN profiles are created in each scenario. As the supervise algorithm, we use Support Vector Machines (SVM). As the unsupervised algorithm, we employ k-Means. Finally, for the rule-based approach, *thr* and distance measurement metrics specified in Section 4.3 are used. Our reason for choosing SVM here is that it has been studied extensively in the literature for the face recognition problem and it provides remarkable results (Guo, Chen, & Li, 2017; Dadi & Mohan Pillutla, 2016; Zhang, Qi, & Chen, 2015; Sharif, Naz, Yasmin, Shahid, & Rehman, A. 2017). In the experimental phase, the version of SVM offered by the Sklearn library (Sklearn.svm.SVC — Scikit-Learn 0.22 Documentation, 2019) is used. The values we use for the parameters of the SVM are as follows, respectively, kernel is RBF, C is 1, gamma is scaled and degree is 3. We do not perform any parameter tuning because our primary goal here is to observe the changes of supervised algorithms as $|N|$ and t change in the face recognition problem. We use SVM as the representative of supervised models, as SVM is highly preferred for face recognition among supervised models before. However, we are aware that the performance of SVM could be improved significantly if parameters fine-tuned.

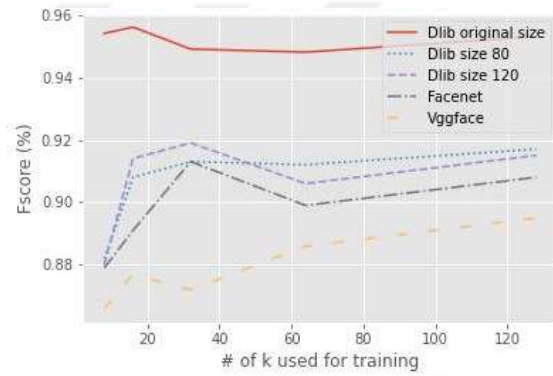
4.5.1. Under Constant $|N|$, Varying t

Here, the case where $|N| = 1000$ is constant is executed and the results are presented. 5 different values are tested for t : 8, 16, 32, 64, and 128. The results of 5 different scenarios are given in Figure 4.3 over f-score and execution time as evaluation metrics. Subfigures 4.3.a and 4.3.d are for SVM, 4.3.b and 4.3.e are for k-Means and finally 4.3.c and 4.3.f are for the rule-based FR approach.

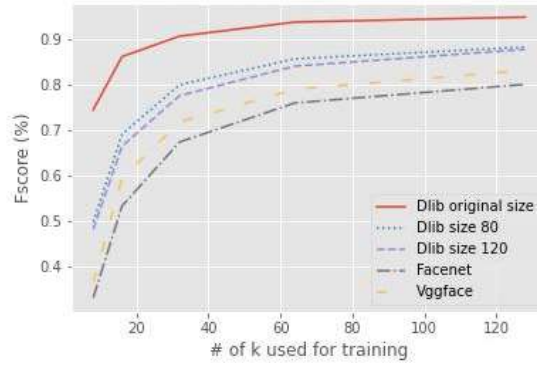
The results show that the accuracy in the FR process based on SVM and rule-based approaches vary according to t . As t increases, the chance of learning increases in the SVM model, while the chance of matching vectors being obtained from the same person in rule-based FR increases and an increase in success is observed. Increasing t causes a decrease in k-Means in some cases, indicating that k-Means cannot efficiently cluster larger clusters as the t increases. Model-based evaluation shows that Dlib is superior to the other two models in terms of both f-score and execution time. Apart from that, Vggface provides better f-score than Facenet in 2 of the 3 scenarios (i.e., SVM and rule-based), but it lags behind in terms of execution time. Finally, the evaluation of the approaches shows that SVM, is more suitable for FR both in terms of f-score and execution time compared to k-Means and rule-based approaches.



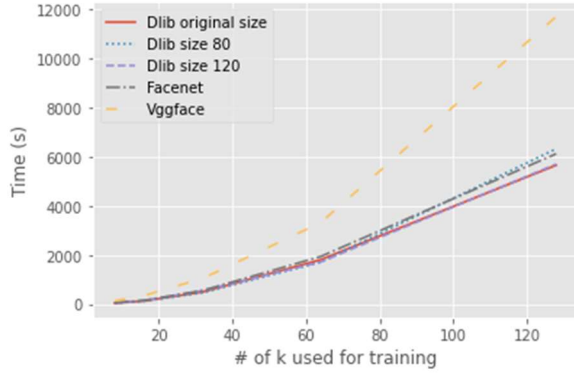
a) F-score with SVM



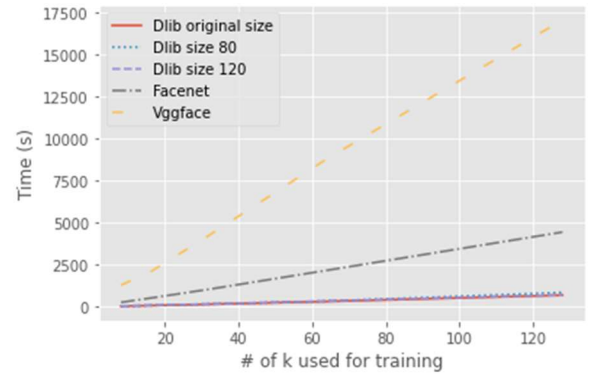
b) F-score with K-Means



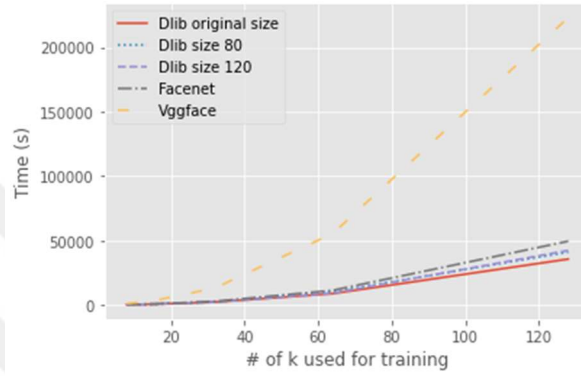
c) F-score with Threshold



d) Execution time with SVM



e) Execution time with K-Means



f) Execution time with Threshold

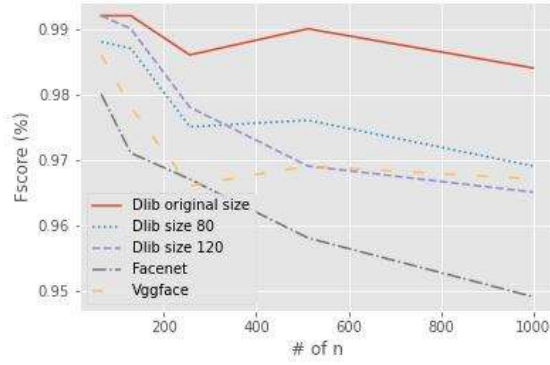
Figure 4.3. Experiment results of 3 different approaches for face recognition by f-score and execution time ($|N|$ constant)

4.5.2. Under Constant t , Varying $|N|$

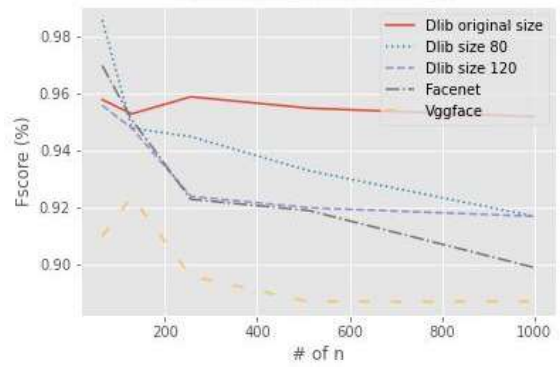
In this scenario t is constant at 32 and $|N|$ varies. We test 5 different values for $|N|$: 64, 128, 256, 512, and 1000. The results of the 5 different models are given in Figure 4.4 along f-score and execution time as evaluation metrics. Subfigures 4.4.a and 4.4.d are for SVM, 4.4.b and 4.4.e are for k-Means and finally 4.4.c and 4.4.f are for the rule-based FR approach.

The results show that as $|N|$ increases, in all 3 approaches, FR performance decreases. However, among the 3 different approaches, the threshold-based approach experiences the biggest decrease. As in Section 4.5.1, Dlib outperforms the other two models, producing better f-score and runtime. Likewise, it turns out that Vggface produces better f-score than Facenet in two scenarios (i.e., SVM and threshold-based approaches) even though execution time is longer.

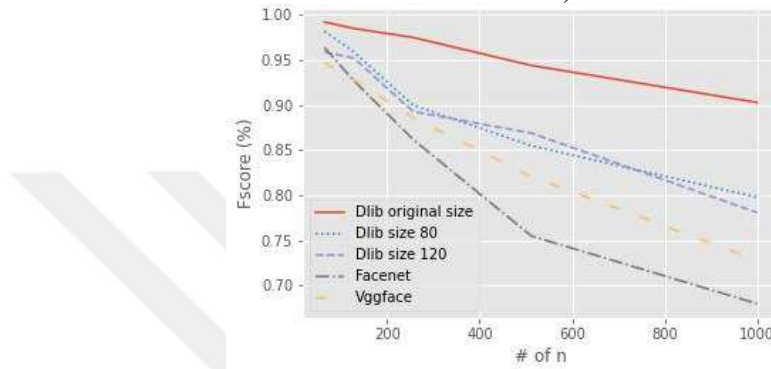
Making an overall assessment, the supervised approach SVM is superior in both evaluation metrics and for FR, vectors produced with Dlib are much more suitable.



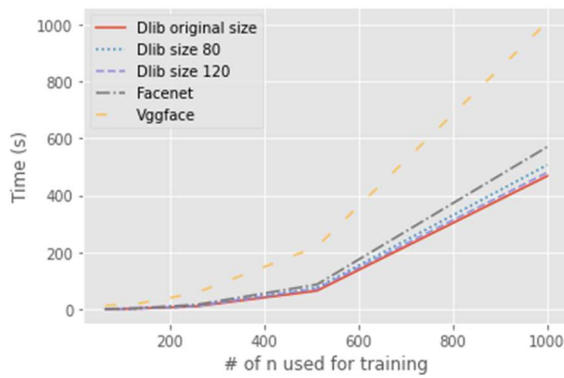
a) F-score with SVM



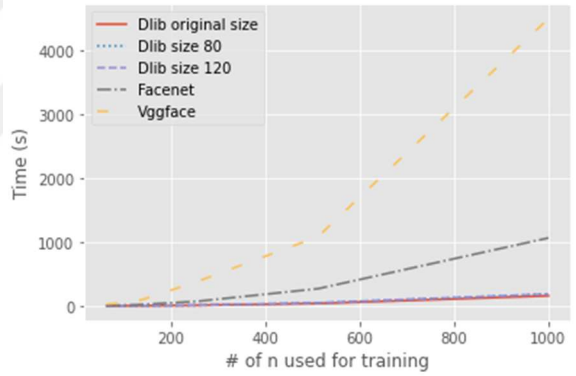
b) F-score with K-Means



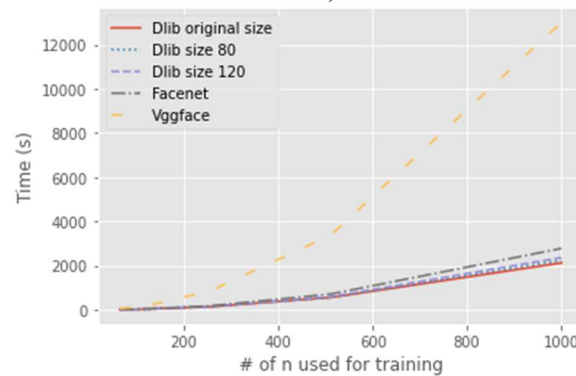
c) F-score with Threshold



d) Execution time with SVM



e) Execution time with K-Means



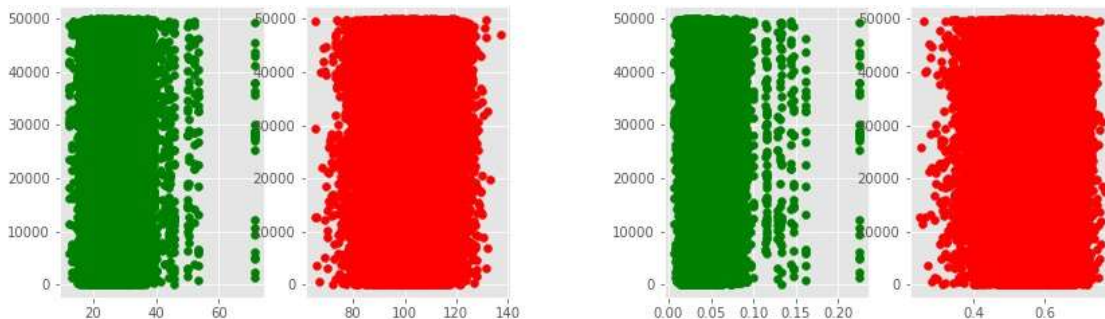
f) Execution time with Threshold

Figure 4.4. Experiment results of 3 different approaches for face recognition by f-score and execution time (t constant)

4.6. Testing Signature Generation Approaches For FR

In this section, we test the search model that generates a signature $\overrightarrow{S_{m_i}}$ for each profile m_i using only t vectors of the profile owner O_{m_i} . We run this model for the FR problem to test the accuracy of its performance before testing it on OSNs. The most appropriate threshold $thrSig$ that decides whether signatures $\overrightarrow{S_{m_i}}$ and $\overrightarrow{S_{n_j}}$ are identical or not according to the distance between the two given signatures, and distance measurement method are determined for 5 different scenarios using the Algorithm 1. Here, the total signature number y generated for each m_i is 3 and the sub sample size k is 50.000. Distance measurement methods tested are Euclidean and Cosine, and 4 different values of t (that are 8, 16, 32, 64) to observe the performance of the proposed model as t increases.

The results are given respectively in Tables 4.9, 4.10, 4.11, 4.12, and 4.13 for each scenario. Here, $thrSig$ is determined by considering the value that will maximize the f-score. The specified $thrSig$ is shown in column Threshold. Verification f-score obtained with determined $thrSig$ is also given. The results show that $thrSig$ obtained with both Euclidean and Cosine for all different values of t achieve 1.0 f-score. As t increases, $thrSig$ decreases. This implies averaging t vectors of an account make the profile owner's signature more distinctive. And the signatures obtained from identical profiles become more similar to each other. The same is also observed for Vggface in Figure 4.5, where the distribution of distances between signatures obtained from both identical and non-identical profiles is depicted. We observe that both distributions differ visibly from each other. Using 8 vectors belonging to only the profile owner (i.e., setting $t = 8$) gives very good results.



a) Euclidean

b) Cosine

Figure 4.5. Distance distributions with Vggface2 ($t = 8$) (green dots show calculated distances between two identical signatures, red dots show the distances between two non-identical signature pairs)

Next, the model proposed in this thesis is tested on 5 different scenarios for the problem of matching an identical profiles using Algorithm 4 proposed in Section 3.5.6. For each scenario, $thrSig$ determined in this section are used. 2.622 person vectors in each scenario are split into two distinct vector sets to simulate two OSNs with profiles m and n . The results are given respectively in Tables 4.9, 4.10, 4.11, 4.12, and 4.13 for each scenario. We use recall, precision, accuracy and f-score as evaluation metrics. Execution time is given as “Total time” for each scenario. Comparative graphics of the 5 models are also presented in Figure 4.6. The results show that it is better to calculate the distance between two signatures with Cosine to match identical profiles in all scenarios. Here, as t increases, it is seen that f-score improves in most cases. Using 8 vectors belonging to only the profile owner provides considerably good results, especially in Dlib-original with 0.996 f-score and Vggface scenarios with 0.988 f-score. Finally, since only signatures are compared to find identical profiles, unlike other approaches, even if t increases, there is no increase in execution time.

Table 4.9. Verification treshold and face recognition results for Dlib with original image size

# of k for training		8	16	32	64
Euclidean	Total time (s)	8.3804	6.8307	6.7850	6.9774
	Threshold	0.39	0.34	0.32	0.26
	V. Fscore	1.000	1.000	1.000	1.000
	Recall	0.941	0.991	0.998	1.000
	Precision	0.946	0.993	0.998	1.000
	Accuracy	0.941	0.991	0.998	1.000
	Fscore	0.934	0.990	0.997	1.000
Cosine*	Total time (s)	15.9384	16.4261	15.9523	16.1419
	Threshold	0.04	0.03	0.03	0.02
	V. Fscore	1.000	1.000	1.000	1.000
	Recall	0.996	0.999	1.000	1.000
	Precision	0.998	0.999	1.000	1.000
	Accuracy	0.996	0.999	1.000	1.000
	Fscore	0.996	0.999	1.000	1.000

* indicates best result and selected metric for the face recognition

Table 4.10. Verification treshold and face recognition results for Dlib with 80x80 image size

# of k for training		8	16	32	64
Euclidean	Total time (s)	4.3532	3.4159	3.4388	3.3510
	Threshold	0.33	0.26	0.30	0.24
	V. Fscore	1.000	1.000	1.000	1.000
	Recall	0.878	0.984	0.991	0.999
	Precision	0.887	0.987	0.991	0.999
	Accuracy	0.878	0.984	0.991	0.999
	Fscore	0.864	0.983	0.989	0.999
Cosine*	Total time (s)	7.7005	7.6755	7.6316	7.6506
	Threshold	0.03	0.02	0.02	0.01
	V. Fscore	1.000	1.000	1.000	1.000
	Recall	0.981	0.997	1.000	1.000
	Precision	0.987	0.997	1.000	1.000
	Accuracy	0.981	0.997	1.000	1.000
	Fscore	0.980	0.996	1.000	1.000

Table 4.11. Verification treshold and face recognition results for Dlib with 120x120 image size

# of k for training		8	16	32	64
Euclidean	Total time (s)	4.9394	3.3970	4.4002	3.2065
	Threshold	0.33	0.27	0.32	0.25
	V. Fscore	1.000	1.000	1.000	1.000
	recall	0.866	0.972	0.957	0.999
	precision	0.871	0.978	0.959	0.999
	accuracy	0.866	0.972	0.957	0.999
	fscore	0.849	0.969	0.951	0.998
Cosine*	Total time (s)	9.9155	8.1303	7.5889	7.5588
	Threshold	0.03	0.02	0.02	0.01
	V. Fscore	1.000	1.000	1.000	1.000
	Recall	0.968	0.992	1.000	0.999
	Precision	0.977	0.995	1.000	0.999
	Accuracy	0.968	0.992	1.000	0.999
	Fscore	0.966	0.992	1.000	0.999

Table 4.12. Verification threshold and face recognition results for Facenet

# of k for training		8	16	32	64
Euclidean	Total time (s)	9.9108	13.7642	8.7935	9.0269
	Threshold	0.61	0.69	0.48	0.45
	V. Fscore	1.000	1.000	1.000	1.000
	recall	0.917	0.927	0.996	0.999
	precision	0.928	0.928	0.996	0.999
	accuracy	0.917	0.927	0.996	0.999
	fscore	0.908	0.916	0.996	0.999
Cosine*	Total time (s)	20.6488	19.2157	18.4916	19.3673
	Threshold	0.28	0.36	0.19	0.16
	V. Fscore	1.000	1.000	1.000	1.000
	Recall	0.962	0.999	0.999	1.000
	Precision	0.971	0.999	0.999	1.000
	Accuracy	0.962	0.999	0.999	1.000
	Fscore	0.959	0.999	0.998	1.000

Table 4.13. Verification threshold and face recognition results for Vggface

# of k for training		8	16	32	64
Euclidean	Total time (s)	20.8851	15.7295	19.3879	16.2060
	Threshold	68.00	59.13	60.26	50.55
	V. Fscore	1.000	1.000	1.000	1.000
	recall	0.900	0.980	0.988	0.999
	precision	0.914	0.985	0.989	0.999
	accuracy	0.900	0.980	0.988	0.999
	fscore	0.889	0.978	0.986	0.999
Cosine*	Total time (s)	48.8353	43.9959	42.4526	47.6101
	Threshold	0.31	0.24	0.21	0.17
	V. Fscore	1.000	1.000	1.000	1.000
	Recall	0.989	0.998	1.000	1.000
	Precision	0.993	0.999	1.000	1.000
	Accuracy	0.989	0.998	1.000	1.000
	Fscore	0.988	0.998	1.000	1.000

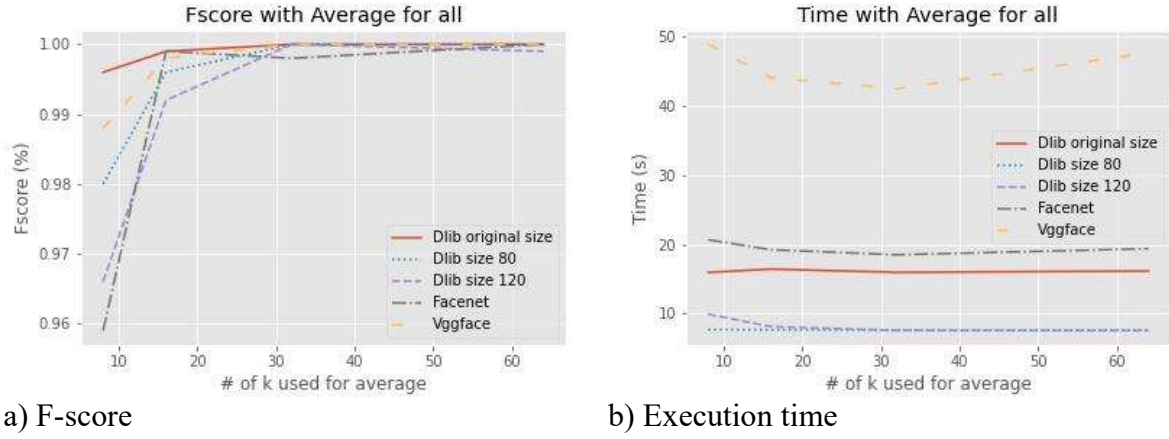


Figure 4.6. Face recognition results with generated signatures

4.7. Face Detection On OSN Data And Determination Of Profile Owner's Vectors

This section relies on real-word data collected from OSNs. We use Facebook as M and Instagram as N . Each face f_l on each photo p_k collected from each profile m_i is detected by the DNN algorithm tested in Section 4.2, and each detected f_l is cut from p_k . Then, each cut f_l is converted to vector $\vec{f}_l$ with our FR algorithm and added to vector set F_{m_i} of m_i . Here, we initially used Dlib as our FR algorithm as Dlib outperforms Vggface and Facenet models (refer to Sections 4.3, 4.4, 4.5 and 4.6). However, we later found out that Dlib could not convert each face to vector while performing vectorization. When the non-vectorizable faces are examined, the common features of the photos are determined as follows: Dlib does not work well in cases where the brightness of the photos is too low or too high, the faces are not fully visible, children's faces are present, and if there is any accessory (glasses, etc.) on the face. This shows that Dlib does not work well on OSN photos, which typically are dirtier than images of the Vggface dataset – faces are not completely visible for various reasons. For this reason, we switch to Vggface as our FR algorithms, since Vggface provides the second best performance for the vectorization process. The Vggface model outperforms Facenet model in both 2 of the 3 matching approaches tested and the proposed search phase. It is better in terms of FR success, although it requires more execution time than Facenet. In Table 4.14, vectorization results made with Dlib and Vggface on 10 profiles are given. The results show that while Vggface model converts each face to a vector, Dlib is unable to convert a significant number of faces to vectors. As a result, we convert each f_l to $\vec{f}_l$ with Vggface model.

Table 4.14. Vectorization results made with Dlib and Vggface on 10 profiles

Profile ID	# of total detected faces	# of encoded faces by Dlib	# of encoded faces by Vggface
1	396	170	396
2	482	201	482
3	31	13	31
4	393	199	393
5	249	130	249
6	394	162	394
7	328	179	328
8	202	70	202
9	374	147	374
10	36	12	36

After obtaining the each F_{m_i} , this step finds the vectors of the profile owner O_{m_i} for each m_i . For this, AGNES clustering tested in Section 4.4 is run on the each F_{m_i} with the distance measurement method (here, cosine) and threshold thr (0.59) determined for the Vggface model to obtain each O_{m_i} . The results are presented in Table 4.15. Figure 4.7 shows how many accounts have more vector numbers than the specified t value, according to $|O_{m_i}|$ and $|O_{n_j}|$ for both M and N . Here, values of 2, 8, 16, and 32 are used for t .

Table 4.15. Number of faces and vectors detected on OSN accounts

Term	Amount
# of total faces detected for M	120,422
# of total faces detected for N	637,767
Average faces for each m	241
Average faces for each n	255
Average profile owner's vectors for each m	69

Average profile owner's vectors for each n	89
Average profile owner's vectors for each $m \in (M \cap N)$	69
Average profile owner's vectors for each $n \in (M \cap N)$	97

Table 4.15 shows that Instagram users post photos with more human faces. Additionally, Instagram profile owners share their own photos more frequently than Facebook profile owners on the average.

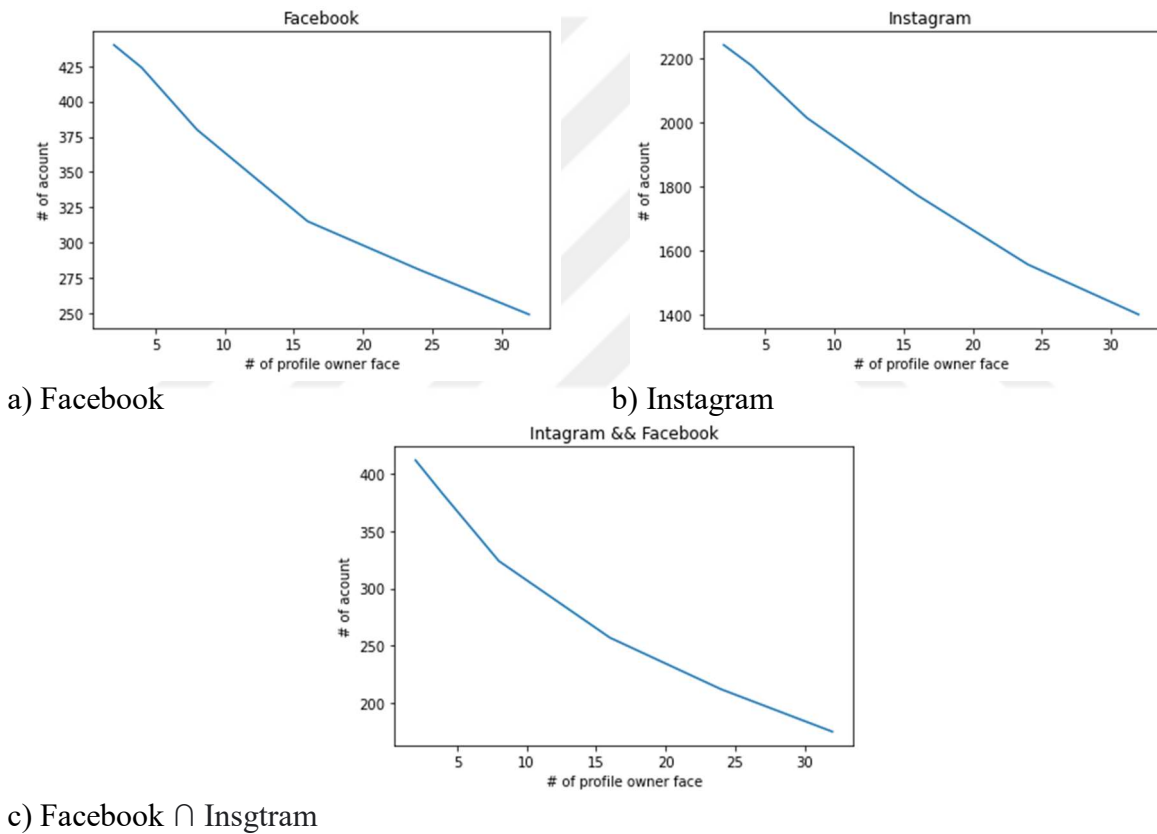


Figure 4.7. Account numbers by vector number of profile holders

Figure 4.7 shows that as t increases, the number of accounts on it decreases. It seems appropriate to keep t small in order to experiment with more profiles. When t is set to 8, 2015 account remain in the Instagram dataset and 380 accounts in the Facebook dataset. At the same time, 324 of the 500 identical accounts used in this thesis have more than 8 vectors. The proposed

search phase tested in Section 4.6 appears to work well for FR with a 0.988 f-score by the Vggface model when t is 8. For this reason, we prefer to set $t = 8$.

A summary of the models and parameters we used to find identical profiles on the OSN is given in Table 4.16.

Table 4.16. Model and parameters used

Term	Value
Face detection algorithm	DNN
Face recognition algorithm	Vggface
Distance measurement metric	Cosine
Input size	224x224
Embedding size	2048
Threshold for face verification (thr)	0.59
Threshold for signature comparison (thrSig)	0.31
Number of used vector for each profile (t)	8

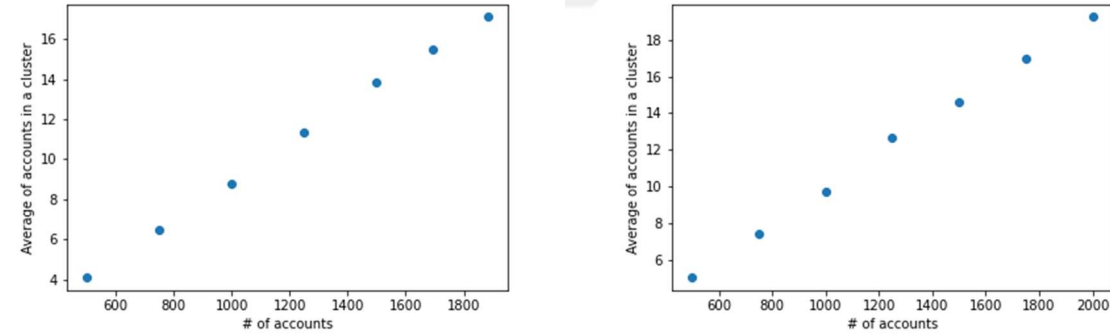
4.8. Overall Results On Real-World OSN Data Using Only The Search Phase

In this section, we test our search phase along – without applying the match phase. For the two OSNs M and N each O_{m_i} and O_{n_j} are labelled. Section 4.5 indicates that SVM is the best of the 3 matching approaches in terms of both accuracy and execution time. Here, we observe its performance on real-world OSN data. The results are finalized on 324 Facebook and 2015 Instagram accounts. The Instagram accounts of 324 Facebook accounts are also found among the 2015 Instagram accounts. Facebook accounts of the remaining Instagram accounts are not included in the Facebook set. In order to match identical profiles on M and N , we assume that each $m_i \in M$ is searched for among the accounts set of N . First, $\overrightarrow{S_{m_i}}$ is created for each m_i using $t = 8$ randomly selected vectors of O_{m_i} . The distance between $\overrightarrow{S_{m_i}}$ and $\overrightarrow{S_{n_j}}$ are measured with Cosine and $thrSig = 0.31$. The model returns a list of candidate accounts $N' \subset N$ that may be identical to m_i . To test the performance of our model, distinct sets of N with different sizes are created for fixed M . This way, performance is measured for varying $|N|$. 7 different values are tested for $|N|$: 500, 750, 1000, 1250, 1500, 1750, and 2000.

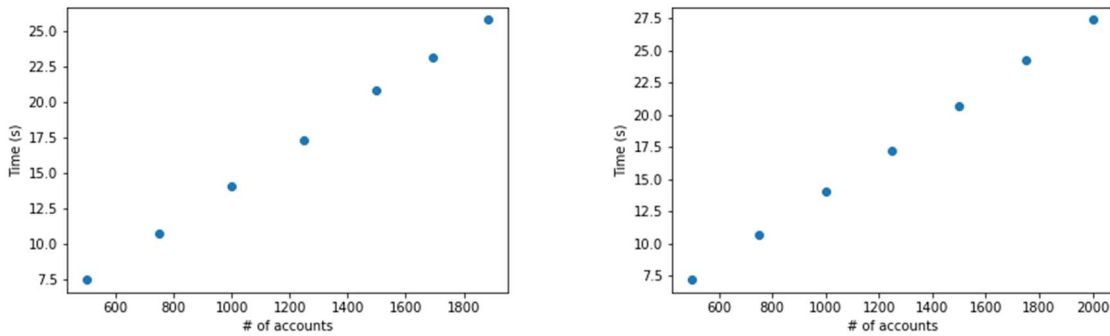
4.8.1. Number Of Candidate Identical Profiles

This experiment tests how many Instagram accounts in total are in N' while searching for m_i in N . The best result here is that only one Instagram profile would be in N' and that profile would be identical to m_i given absolutely identical profile n of m_i is in N . Otherwise, the worst result is that identical profile of m_i would not be in N' . If it is certainly known that the identical profile of m_i is not N , the best outcome is when $|N'|$ is 0. Other possible bad outcomes would having $|N'| \geq 1$. Large $|N'|$ indicates worse results for the model.

We test two different scenarios here. In scenario 1, identical profile n of none of the accounts m is in N . Scenario 2 is the case that identical profile n of each m_i is in N . Figure 4.8 shows the average of $|N'|$ for each m_i and total execution time for each $|N|$. The results show that when $|N|$ is 500, the average of $|N'|$ is at most 4 for each m_i in both scenarios and there is a linear increase in $|N'|$ as N increases. We conclude that each m_i could also be matched with non-identical accounts using the proposed model. But it seems clear that for each m_i , our model returns N' with 1/100th of N in a relatively short time compared to the other 3 approaches. The real question here to ask is: is the identical profile of m_i in N' ?



a) Average of $|N'|$ for each m_i in scenario 1 b) Average of $|N'|$ for each m_i in scenario 2

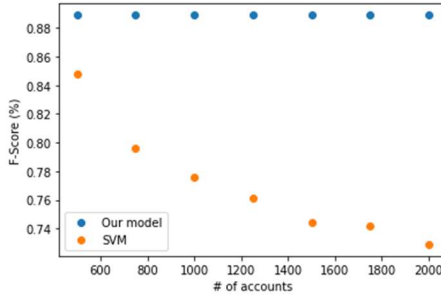


c) Execution time for scenario 1 d) Execution time for scenario 2

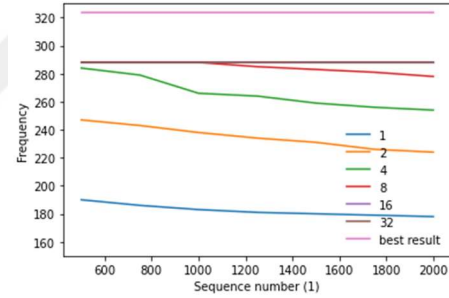
Figure 4.8. Average number of matched identical instagram profiles for each facebook profile

4.8.2. Our Model Versus SVM

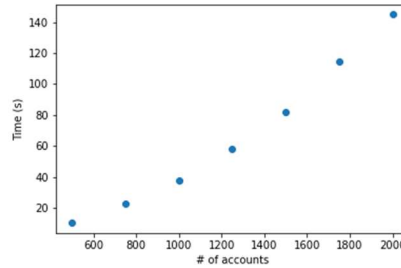
In this section, we compare the proposed search phase against SVM to find identical profiles of each m_i in N . In the scenario, identical Instagram profiles of 324 Facebook accounts are in the Instagram's account set. Here, each $n_j \in N'$ is sorted in ascending rank according to the distance between $\overrightarrow{S_{m_i}}$ and $\overrightarrow{S_{n_j}}$. According to this rank, the order s of n_j , which is the identical profile of m_i , is checked. Here, 6 different values are tested for s : 1, 2, 4, 8, 16, and 32. For example, value 4 checks whether n_j , the identical profile of m_i , is in top-4 rank of N' . In order to compare the performance of our model, we also apply SVM on N to find identical profiles of each m_i . Here, SVM returns only one candidate identical profile for each m_i . Figure 4.9.a presents the f-score results obtained with our model against the SVM model. Figure 4.9.b shows how many identical profiles correctly matched are in the first top s rank of N' , while Figure 4.9.c shows the execution time for SVM.



a) Fscore of our model and SVM



b) The order of the searched profile in the returned possible set



c) Exectuion of SVM

Figure 4.9. Performance of finding identical profiles

For each $|N|$, identical profiles of 288 of 324 facebook accounts are within N' returned by our model. When $|N|$ is 500, the identical profiles of 190 (58% of 324 profiles) Facebook profiles are in the top-1 rank in N' , the identical profiles of 247 Facebook accounts (76% of 324

profiles) are in top-2 rank, and the identical profiles of 284 Facebook accounts (87% of 324 profiles) are in top-4 rank in N' . Figure 4.9.b shows that while searching for identical profile of m on N , a set of possible identical profiles of m is obtained in a short time, and examining each $n \in N$ in top 4 or 8 accounts are sufficient to find the identical profile. SVM results show that as N increases, performance decreases, while execution time increases and SVM requires more runtime than the our search model. The biggest advantage of the proposed model over SVM is that when a new account is included in the OSN dataset, possible identical profiles for the new account can be searched in a short time without the need for training.

4.9. Determination Of *thrLink* And Overall Results On Real-World OSN Data After Match Phase

In this section, the proposed match phase that identifies identical profiles based on profile photos of links is tested. Each face f on each profile photo p collected from each link $m' \in \text{links}(m_i)$ is detected by the DNN algorithm, and each detected f is cut from p . Then, each cut f is converted to vector $\vec{f}$ with the Vggface model and added to vectors set of profile photos m'_i of m_i . Before testing the proposed approach, we determine threshold *thrLink* that decides whether two profiles are identical according to the number of matches between m'_i and n'_j . We run Algorithm 6 proposed in Section 3.5.7 on 324 Facebook and 1000 Instagram profiles. The Cosine metric determined for the Vggface model in Section 4.3 is used to measure the distance between two vectors, $\text{thr} = 0.59$. Instead of using all vectors in m'_i , T (here, 200) vectors randomly selected from m'_i are used for each profile m_i . The results are given in Figure 4.10 along evaluation metrics precision, recall, accuracy and f-score. We primarily focus on f-score. Results show that as *thrLink* increases, precision increases, while recall and f-score decrease. In addition, model accuracy is observed to be in the range of $57 \pm 5\%$. The best result for f-score is obtained with *thrLink* 3 with 63% percent. In other words, if the total number of matches of the profile photos of the users' friends is 3 or more, then the two profiles will be identical, otherwise they will not be identical with 63% f-score. In general, there is no notable difference between the number of matches of profile photos obtained from identical profiles and the number of matches of profile photos obtained from non-identical profiles. It turns out that binary classification using *thrLink* would not be suitable for matching identical profiles.

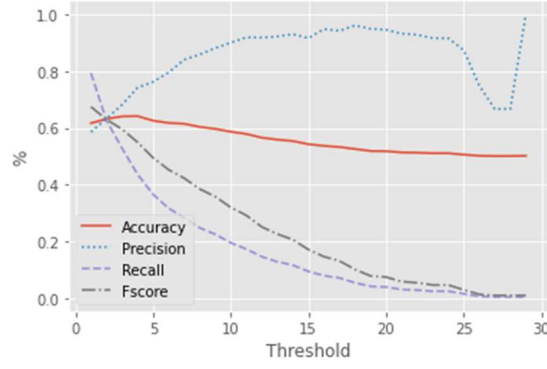


Figure 4.10. Results of determining thrLink

In this last experiment scenario, we perform our match phase over the candidates compiled according to Section 4.8.2. The search phase return an N' for each m_i , and N' may have profiles that are not identical to m_i . Therefore, to eliminate non-identical profiles in N' and identify the real identical profile of m_i , we use proposed matching model to compare vectors of profile photos of friends of each $n_j \in N'$ and m_i . This way we avoid binary classification, and match m_i and n_j , whose profile photos of friends m'_i and n'_j match the most.

The results are given in Figure 4.11 as f-score. The results show that the assumption of matching identical profiles by comparing the profile photos of the friends of the profiles on two OSNs gives remarkable results. Instead of deciding whether the profiles are identical by binary classification with the number of matches and *thrLink*, we observe that it is more appropriate to match the profiles m_i and n_j as identical profiles where m'_i and n'_j matched most.

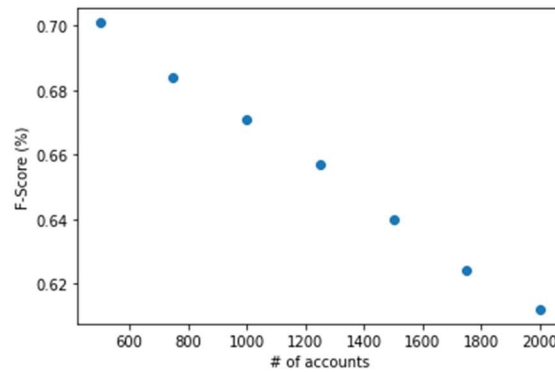


Figure 4.11. Results of hybrid model

5. CONCLUSIONS

The volume, veracity and velocity of data posted on online social networks increases on a daily basis. Almost every user has an account in more than one online social network and the type of content shared differs by the type of online social network. Public or private, many researchers try to match profiles of the same user on cross online social networks to learn more about the users they target. Creating an integrated pool of online social network data for a user, allows better capturing the user's pattern of behavior.

Many solutions have been presented to address the problem of matching identical profiles of a user. A considerable portion of the proposed solutions are based on similarities of profile attributes that can be easily spoofed by malicious profiles or left blank by users due to privacy concerns. Additionally, majority of studies perform matching on content produced by the users or connections established by users. However, most user-generated content (such as tweets, photos and location check-in) is limited in distinguishing users.

In this thesis, we propose a novel two-phase hybrid model to match identical profiles. Contrary to the existing studies, we present a search phase based on the similarities of the faces in the posts of the users, which has not been well-studied before, and a matching phase based on the social links users established on the same OSN. We aim our hybrid model to be more reliable, both because facial biometrics are more distinctive and unchangeable than previously studied features, and also because social links established by users are difficult to imitate by malicious accounts. In the search phase, we create a signature for each profile with the average of a small number of faces belonging to the profile owner and perform a quick search according to the similarities of these signatures. As a result, for each profile whose identical profile is searched, we obtain a list of candidate identical profiles with a single query. Then, matching phase picks a profile with the highest potential, which can be a real identical profile with the searched profile from the list of candidate profiles, based on the similarities of the links of the respective users.

We perform experimental results on data collected from two real OSNs Facebook and Instagram over 3.000 users. We test five different face detection (FD) algorithms to determine the best performing FD algorithm for our problem setting. We test three different face recognition (FR) models to determine the FR model that best performs feature vectorization for

each face. We also run AGNES clustering to determine the vectors of the profile owners. The results show that the DNN algorithm provide the best result among FD algorithms with 94% accuracy, the Vggface model is determined as the most suitable FR model with 95% f-score, and AGNES clustering can find faces of profile owners with 86% accuracy and 76% f-score. Our search phase captures identical profiles more successfully than traditional matching methods in all cases, and the matching phase reaches up to 70% f-score.

We believe that the proposed search phase would be useful by itself in various application scenarios: detecting fake/cloned profiles created by copying a user's photos, locating duplicate profiles of the same user within a single OSN (i.e., de-duplication), finding criminal profiles on OSNs using photos obtained from criminals, and in its most general form, deciding whether two sets of photos belong to the same person.

6. RECOMMENDATIONS

This thesis presents a two-phase hybrid model with a face recognition approach to match identical profiles in cross OSNs of a user. Here, we offer some suggestions for both improving the performance of the model and some future studies.

- (1) Commonly shared and distinguishing profile attributes may be utilized in combination with the proposed solution.
- (2) To strengthen the match phase, all photos of links can be processed instead of using just profile photos - but one has to consider the complexity of this approach.
- (3) The performance change can be observed by adjusting the parameter settings of the SVM, which we compare with our search method.
- (4) Link enrichment can be done for the profiles that hide their links - for example, a user's Facebook friends are hidden, but when their profile is viewed, people they comment on or like their photos are listed.
- (5) Finally, the photos posted by users can be examined and suggestions can be made regarding the risks of sharing. For example, if you share this photo, your OSN account may be linked with these accounts.

REFERENCES

- Abbasi, M. A., & Liu, H. (2013). Measuring user credibility in social media. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7812 LNCS. https://doi.org/10.1007/978-3-642-37210-0_48
- Aboujaoude, E., Savage, M. W., Starcevic, V., & Salame, W. O. (2015). Cyberbullying: Review of an old problem gone viral. *Journal of Adolescent Health*, Vol. 57. <https://doi.org/10.1016/j.jadohealth.2015.04.011>
- Adouani, A., Ben Henia, W. M., & Lachiri, Z. (2019). Comparison of Haar-like, HOG and LBP approaches for face detection in video sequences. *16th International Multi-Conference on Systems, Signals and Devices, SSD 2019*. <https://doi.org/10.1109/SSD.2019.8893214>
- Ahmad, W., & Ali, R. (2019). Social Account Matching in Online Social Media using Cross-linked Posts. *Procedia Computer Science*, 152, 222–229. <https://doi.org/10.1016/J.PROCS.2019.05.046>
- AL-Allaf, O. N. A. (2014). Review of Face Detection Systems Based Artificial Neural Networks Algorithms. *The International Journal of Multimedia & Its Applications*, 6(1). <https://doi.org/10.5121/ijma.2013.6101>
- Alhawarat, M., & Hegazi, M. (2018). Revisiting K-Means and Topic Modeling, a Comparison Study to Cluster Arabic Documents. *IEEE Access*, 6. <https://doi.org/10.1109/ACCESS.2018.2852648>
- Almishari, M., & Tsudik, G. (2012). Exploring linkability of user reviews. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7459 LNCS. https://doi.org/10.1007/978-3-642-33167-1_18
- Asha, C. S., & Narasimhadhan, A. V. (2018). Vehicle Counting for Traffic Management System using YOLO and Correlation Filter. *2018 IEEE International Conference on*

Electronics, Computing and Communication Technologies, CONECCT 2018.

<https://doi.org/10.1109/CONECCT.2018.8482380>

Bakhshi, S., Shamma, D. A., & Gilbert, E. (2014). Faces engage us: Photos with faces attract more likes and comments on instagram. *Conference on Human Factors in Computing Systems - Proceedings*. <https://doi.org/10.1145/2556288.2557403>

Bartunov, S., Korshunov, A., Park, S.-T., Ryu, W., & Lee, H. (2012). Joint link-attribute user identity resolution in online social networks. *Proceedings of the Workshop on Social Network Mining and Analysis (SNA-KDD)*.

Bedi, P., & Sharma, C. (2016). Community detection in social networks. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Vol. 6. <https://doi.org/10.1002/widm.1178>

Bennacer, N., Nana Jipmo, C., Penta, A., & Quercini, G. (2014). Matching user profiles across social networks. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8484 LNCS. https://doi.org/10.1007/978-3-319-07881-6_29

Boon-Long, S., & Wongsurawat, W. (2015). Social media marketing evaluation using social network comments as an indicator for identifying consumer purchasing decision effectiveness. *Journal of Direct, Data and Digital Marketing Practice*, 17(2). <https://doi.org/10.1057/dddmp.2015.51>

Cai, Z., Fan, Q., Feris, R. S., & Vasconcelos, N. (2016). A unified multi-scale deep convolutional neural network for fast object detection. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9908 LNCS. https://doi.org/10.1007/978-3-319-46493-0_22

Cao, Q., Shen, L., Xie, W., Parkhi, O. M., & Zisserman, A. (2018). VGGFace2: A dataset for recognising faces across pose and age. *Proceedings - 13th IEEE International Conference on Automatic Face and Gesture Recognition, FG 2018*. <https://doi.org/10.1109/FG.2018.00020>

Cao, X., Wipf, D., Wen, F., Duan, G., & Sun, J. (2013). A practical transfer learning algorithm for face verification. *Proceedings of the IEEE International Conference on*

Computer Vision. <https://doi.org/10.1109/ICCV.2013.398>

Centeno, I. de P. (2021, April 15). *ipazc/mtcnn*. GitHub. <https://github.com/ipazc/mtcnn>

Cerna, L. R., Cámara-Chávez, G., & Menotti, D. (2013). Face detection: Histogram of oriented gradients and bag of feature method. *Proceedings of the 2013 International Conference on Image Processing, Computer Vision, and Pattern Recognition, IPCV 2013*, 2.

Chen, Weihua, Chen, X., Zhang, J., & Huang, K. (2017). Beyond triplet loss: A deep quadruplet network for person re-identification. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017-January*.
<https://doi.org/10.1109/CVPR.2017.145>

Chen, Weijun, Huang, H., Peng, S., Zhou, C., & Zhang, C. (2021). YOLO-face: a real-time face detector. *Visual Computer*, 37(4). <https://doi.org/10.1007/s00371-020-01831-7>

Choi, J. Y., De Neve, W., Plataniotis, K. N., & Ro, Y. M. (2011). Collaborative face recognition for improved face annotation in personal photo collections shared on online social networks. *IEEE Transactions on Multimedia*, 13(1).
<https://doi.org/10.1109/TMM.2010.2087320>

Crosswhite, N., Byrne, J., Stauffer, C., Parkhi, O., Cao, Q., & Zisserman, A. (2018). Template adaptation for face verification and identification. *Image and Vision Computing*, 79.
<https://doi.org/10.1016/j.imavis.2018.09.002>

Dadi, H. S., & Mohan Pillutla, G. K. (2016). Improved Face Recognition Rate Using HOG Features and SVM Classifier. *IOSR Journal of Electronics and Communication Engineering*, 11(04). <https://doi.org/10.9790/2834-1104013444>

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005, I*. <https://doi.org/10.1109/CVPR.2005.177>

Dantone, M., Bossard, L., Quack, T., & Van Gool, L. (2011, November). Augmented faces. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)* (pp. 24 31). IEEE.

- Dehariya, V. K., Shrivastava, S. K., & Jain, R. C. (2010). Clustering of image data set using k-means and fuzzy k-means algorithms. *Proceedings - 2010 International Conference on Computational Intelligence and Communication Networks, CICN 2010*.
<https://doi.org/10.1109/CICN.2010.80>
- Devmane, M. A., & Rana, N. K. (2014). Detection and prevention of profile cloning in online social networks. *International Conference on Recent Advances and Innovations in Engineering, ICRAIE 2014*. <https://doi.org/10.1109/ICRAIE.2014.6909237>
- Dey, R., Tang, C., Ross, K., & Saxena, N. (2012). Estimating age privacy leakage in online social networks. *Proceedings - IEEE INFOCOM*.
<https://doi.org/10.1109/INFCOM.2012.6195711>
- Dinakar, K., Jones, B., Havasi, C., Lieberman, H., & Picard, R. (2012). Common sense reasoning for detection, prevention, and mitigation of cyberbullying. *ACM Transactions on Interactive Intelligent Systems, 2*(3). <https://doi.org/10.1145/2362394.2362400>
- Diraco, G., Leone, A., & Siciliano, P. (2015). People occupancy detection and profiling with 3D depth sensors for building energy management. *Energy and Buildings, 92*.
<https://doi.org/10.1016/j.enbuild.2015.01.043>
- Frommholz, I., al-Khateeb, H. M., Potthast, M., Ghasem, Z., Shukla, M., & Short, E. (2016). On Textual Analysis and Machine Learning for Cyberstalking Detection. *Datenbank-Spektrum, 16*(2). <https://doi.org/10.1007/s13222-016-0221-x>
- Garima, Gulati, H., & Singh, P. K. (2015). Clustering techniques in data mining: A comparison. *2015 International Conference on Computing for Sustainable Global Development, INDIACom 2015*.
- Ghofrani, A., Toroghi, R. M., & Ghanbari, S. (2019). Realtime Face-Detection and Emotion Recognition Using MTCNN and miniShuffleNet V2. *2019 IEEE 5th Conference on Knowledge Based Engineering and Innovation, KBEI 2019*.
<https://doi.org/10.1109/KBEI.2019.8734924>
- Global social media account ownership 2018 | Statista. (2018). Retrieved from Statista website: <https://www.statista.com/statistics/788084/number-of-social-media-accounts/>

- Goga, O., Lei, H., Parthasarathi, S. H. K., Friedland, G., Sommer, R., & Teixeira, R. (2013). Exploiting innocuous activity for correlating users across sites. *WWW 2013 - Proceedings of the 22nd International Conference on World Wide Web*. <https://doi.org/10.1145/2488388.2488428>
- Goga, O., Loiseau, P., Sommer, R., Teixeira, R., & Gummadi, K. P. (2015). On the reliability of profile matching across large online social networks. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2015-August*. <https://doi.org/10.1145/2783258.2788601>
- Goga, O., Perito, D., Lei, H., Teixeira, R., & Sommer, R. (2013). Large-scale Correlation of Accounts Across Social Networks. *University of California at Berkeley, Berkeley, California, Tech. Rep. TR-13-002*.
- Gong, N. Z., & Liu, B. (2018). Attribute inference attacks in online social networks. *ACM Transactions on Privacy and Security, 21*(1). <https://doi.org/10.1145/3154793>
- Gumus, E., Kilic, N., Sertbas, A., & Ucan, O. N. (2010). Evaluation of face recognition techniques using PCA, wavelets and SVM. *Expert Systems with Applications, 37*(9). <https://doi.org/10.1016/j.eswa.2010.02.079>
- Guntuku, S. C., Preotiuc-Pietro, D., Eichstaedt, J. C., & Ungar, L. H. (2019). What twitter profile and posted images reveal about depression and anxiety. *Proceedings of the 13th International Conference on Web and Social Media, ICWSM 2019*.
- Guo, G., & Zhang, N. (2019). A survey on deep learning based face recognition. *Computer Vision and Image Understanding, 189*. <https://doi.org/10.1016/j.cviu.2019.102805>
- Guo, S., Chen, S., & Li, Y. (2017). Face recognition based on convolutional neural network & support vector machine. 2016 IEEE International Conference on Information and Automation, IEEE ICIA 2016. <https://doi.org/10.1109/ICInfA.2016.7832107>
- Hassaballah, M., & Aly, S. (2015). Face recognition: Challenges, achievements and future directions. *IET Computer Vision, Vol. 9*. <https://doi.org/10.1049/iet-cvi.2014.0084>
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-December*. <https://doi.org/10.1109/CVPR.2016.90>

- Hendry, & Chen, R. C. (2019). Automatic License Plate Recognition via sliding-window darknet-YOLO deep learning. *Image and Vision Computing*, 87.
<https://doi.org/10.1016/j.imavis.2019.04.007>
- Hernandez-Matamoros, A., Bonarini, A., Escamilla-Hernandez, E., Nakano-Miyatake, M., & Perez-Meana, H. (2016). Facial expression recognition with automatic segmentation of face regions using a fuzzy based classification approach. *Knowledge-Based Systems*, 110. <https://doi.org/10.1016/j.knosys.2016.07.011>
- Hierarchical Clustering (Unsupervised Learning algorithm). (2021, April 24). Retrieved 2022, from smriti-mishra website:
<https://smritimishrain.wordpress.com/2021/04/25/hierarchical-clustering-unsupervised-learning-algorithm/>
- Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-Excitation Networks. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.
<https://doi.org/10.1109/CVPR.2018.00745>
- Huang, G. B., Mattar, M., Berg, T., & Learned-Miller, E. (2008, October). Labeled faces in the wild: A database for studying face recognition in unconstrained environments. In Workshop on faces in 'Real Life' Images: detection, alignment, and recognition.
- Ilija, P., Polakis, I., Athanasopoulos, E., Maggi, F., & Ioannidis, S. (2015). Face/off: Preventing privacy leakage from photos in social networks. *Proceedings of the ACM Conference on Computer and Communications Security, 2015-October*.
<https://doi.org/10.1145/2810103.2813603>
- Iofciu, T., Fankhauser, P., Abel, F., & Bischoff, K. (2011). Identifying Users Across Social Tagging Systems. *Icwsn*.
- Ip, A., Fong, S., & Liu, E. (2010). Optimization for allocating BEV recharging stations in urban areas by using hierarchical clustering. *Proc. - 6th Intl. Conference on Advanced Information Management and Service, IMS2010, with ICMLA2010 - 2nd International Conference on Data Mining and Intelligent Information Technology Applications*.
- Islam, M. R., Kabir, M. A., Ahmed, A., Kamal, A. R. M., Wang, H., & Ulhaq, A. (2018). Depression detection from social network data using machine learning techniques.

Health Information Science and Systems, 6(1). <https://doi.org/10.1007/s13755-018-0046-0>

Jadhav, A., Lone, S., Matey, S., Madamwar, T., & Jakhete, S. (2021). Survey on Face Detection Algorithms. *International Journal of Innovative Science and Research Technology*, 6(2).

Jain, P., Kumaraguru, P., & Joshi, A. (2013, May). '@ i seek'fb. me' identifying users across multiple online social networks. In *Proceedings of the 22nd international conference on WorldWide Web* (pp.1259-1268).

Jiang, H., & Learned-Miller, E. (2017, May). Face detection with the faster R-CNN. In *2017 12th IEEE international conference on automatic face & gesture recognition (FG 2017)* (pp. 650 657). IEEE.

Johansson, F., Kaati, L., & Shrestha, A. (2013). Detecting multiple aliases in social media. *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2013*.
<https://doi.org/10.1145/2492517.2500261>

Jose, E., Greeshma, M., Mithun Haridas, T. P., & Supriya, M. H. (2019). Face Recognition based Surveillance System Using FaceNet and MTCNN on Jetson TX2. *2019 5th International Conference on Advanced Computing and Communication Systems, ICACCS 2019*. <https://doi.org/10.1109/ICACCS.2019.8728466>

Jung, S. G., An, J., Kwak, H., Salminen, J., & Jansen, B. J. (2017). Inferring social media users' demographics from profile pictures: A Face++ analysis on twitter users. *Proceedings of the International Conference on Electronic Business (ICEB), 2017-December*.

Karpinska-Krakiwicz, M. (2016). The effects of social networking sites on consumer-brand relationships. *Journal of Computer Information Systems*, 56(3).
<https://doi.org/10.1080/08874417.2016.1153894>

Khaled, S., El-Tazi, N., & Mokhtar, H. M. O. (2019). Detecting Fake Accounts on Social Media. *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018*.
<https://doi.org/10.1109/BigData.2018.8621913>

- King, D. E. (2009). Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research, 10*.
- King, D. E. (2016a). Dlib.net. Retrieved 2022, from http://dlib.net/files/data/dlib_face_detection_dataset-2016-09-30.tar.gz
- King, D. E. (2016b). Easily Create High Quality Object Detectors with Deep Learning. <http://blog.dlib.net/2016/10/easily-create-high-quality-object.html>
- King, D. E. (2017a). dlib C++ library. GitHub. https://github.com/davisking/dlib/blob/master/examples/dnn_face_recognition_ex.cpp
- King, D. E. (2017b). High Quality Face Recognition with Deep Metric Learning. Retrieved 2022, from Dlib.net website: <http://blog.dlib.net/2017/02/high-quality-face-recognition-with-deep.html>
- King, D. E. (2022, August 2). *dlib C++ library*. GitHub. https://github.com/davisking/dlib/blob/master/dlib/image_processing/frontal_face_detector.h
- Kong, X., Zhang, J., & Yu, P. S. (2013). Inferring anchor links across multiple heterogeneous social networks. *International Conference on Information and Knowledge Management, Proceedings*. <https://doi.org/10.1145/2505515.2505531>
- Korula, N., & Lattanzi, S. (2014). An efficient reconciliation algorithm for social networks. *Proceedings of the VLDB Endowment, 7*(5). <https://doi.org/10.14778/2732269.2732274>
- Kumar, A., Kaur, A., & Kumar, M. (2019). Face detection techniques: a review. *Artificial Intelligence Review, 52*(2). <https://doi.org/10.1007/s10462-018-9650-2>
- Lan, W., Dang, J., Wang, Y., & Wang, S. (2018). Pedestrian detection based on yolo network model. *Proceedings of 2018 IEEE International Conference on Mechatronics and Automation, ICMA 2018*. <https://doi.org/10.1109/ICMA.2018.8484698>
- Li, Chong, Wang, R., Li, J., & Fei, L. (2020). Face detection based on YOLOv3. *Advances in Intelligent Systems and Computing, 1031 AISC*. https://doi.org/10.1007/978-981-13-9406-5_34

- Li, Cuimei, Qi, Z., Jia, N., & Wu, J. (2017). Human face detection algorithm via Haar cascade classifier combined with three additional classifiers. *ICEMI 2017 - Proceedings of IEEE 13th International Conference on Electronic Measurement and Instruments, 2018-January*. <https://doi.org/10.1109/ICEMI.2017.8265863>
- Li, H., Zhu, H., Du, S., Liang, X., & Shen, X. S. (2018). Privacy leakage of location sharing in mobile social networks: Attacks and defense. *IEEE Transactions on Dependable and Secure Computing, 15*(4). <https://doi.org/10.1109/TDSC.2016.2604383>
- Li, M., Zhu, H., Gao, Z., Chen, S., Yu, L., Hu, S., & Ren, K. (2014). All your Location are Belong to Us: Breaking mobile social networks for automated user location tracking. *Proceedings of the International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), 11-14-August-2014*. <https://doi.org/10.1145/2632951.2632953>
- Li, Y., Peng, Y., Ji, W., Zhang, Z., & Xu, Q. (2017). User Identification Based on Display Names Across Online Social Networks. *IEEE Access, 5*. <https://doi.org/10.1109/ACCESS.2017.2744646>
- Li, Y., Su, Z., Yang, J., & Gao, C. (2020). Exploiting similarities of user friendship networks across social networks for user identification. *Information Sciences, 506*. <https://doi.org/10.1016/j.ins.2019.08.022>
- Li, Y., Zhang, Z., Peng, Y., Yin, H., & Xu, Q. (2018). Matching user accounts based on user generated content across social networks. *Future Generation Computer Systems, 83*. <https://doi.org/10.1016/j.future.2018.01.041>
- Lim, B. H., Lu, D., Chen, T., & Kan, M. Y. (2015). #mytweet via Instagram: Exploring user behaviour across multiple social networks. *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2015*. <https://doi.org/10.1145/2808797.2808820>
- Lim, B. H., Lu, D., Chen, T., & Min-Yen, K. (2017). About.Me Dataset for User-centric OSN Analysis. Scholarbank.nus.edu.sg. <https://doi.org/10.25540/941F-64FU>
- Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Doll'ar, P., Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes

- in Bioinformatics), 8693 LNCS(PART 5). https://doi.org/10.1007/978-3-319-10602-1_48
- Liu, J., Zhang, F., Song, X., Song, Y. I., Lin, C. Y., & Hon, H. W. (2013). What's in a name? An unsupervised approach to link users across communities. *WSDM 2013 - Proceedings of the 6th ACM International Conference on Web Search and Data Mining*. <https://doi.org/10.1145/2433396.2433457>
- Liu, S., Wang, S., Zhu, F., Zhang, J., & Krishnan, R. (2014). HYDRA: Large-scale social identity linkage via heterogeneous behavior modeling. *Proceedings of the ACM SIGMOD International Conference on Management of Data*. <https://doi.org/10.1145/2588555.2588559>
- Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single shot multibox detector. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9905 LNCS. https://doi.org/10.1007/978-3-319-46448-0_2
- Liu, Weiyang, Wen, Y., Yu, Z., Li, M., Raj, B., & Song, L. (2017). SphereFace: Deep hypersphere embedding for face recognition. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017-January*. <https://doi.org/10.1109/CVPR.2017.713>
- Lu, J., Ma, C., Li, L., Xing, X., Zhang, Y., Wang, Z., & Xu, J. (2018). A Vehicle Detection Method for Aerial Image Based on YOLO. *Journal of Computer and Communications*, 06(11). <https://doi.org/10.4236/jcc.2018.611009>
- Lung, C. H., & Zhou, C. (2010). Using hierarchical agglomerative clustering in wireless sensor networks: An energy-efficient and flexible approach. *Ad Hoc Networks*, 8(3). <https://doi.org/10.1016/j.adhoc.2009.09.004>
- Ma, M., & Wang, J. (2019). Multi-View Face Detection and Landmark Localization Based on MTCNN. *Proceedings 2018 Chinese Automation Congress, CAC 2018*. <https://doi.org/10.1109/CAC.2018.8623535>
- Malhotra, A., Totti, L., Meira, W., Kumaraguru, P., & Almeida, V. (2012). Studying user footprints in different online social networks. *Proceedings of the 2012 IEEE/ACM*

- International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2012*. <https://doi.org/10.1109/ASONAM.2012.184>
- Malisiewicz, T., Gupta, A., & Efros, A. A. (2011). Ensemble of exemplar-SVMs for object detection and beyond. *Proceedings of the IEEE International Conference on Computer Vision*. <https://doi.org/10.1109/ICCV.2011.6126229>
- Mao, Q. C., Sun, H. M., Liu, Y. B., & Jia, R. S. (2019). Mini-YOLOv3: Real-Time Object Detector for Embedded Applications. *IEEE Access*, 7. <https://doi.org/10.1109/ACCESS.2019.2941547>
- Marcec, R., & Likic, R. (2021). Using Twitter for sentiment analysis towards AstraZeneca/Oxford, Pfizer/BioNTech and Moderna COVID-19 vaccines. *Postgraduate Medical Journal*. <https://doi.org/10.1136/postgradmedj-2021-140685>
- Mei, B., Xiao, Y., Li, R., Li, H., Cheng, X., & Sun, Y. (2020). Image and Attribute Based Convolutional Neural Network Inference Attacks in Social Networks. *IEEE Transactions on Network Science and Engineering*, 7(2). <https://doi.org/10.1109/TNSE.2018.2797930>
- Meng, Y., Liang, J., Cao, F., & He, Y. (2018). A new distance with derivative information for functional k-means clustering algorithm. *Information Sciences*, 463–464. <https://doi.org/10.1016/j.ins.2018.06.035>
- Mislove, A., Viswanath, B., Gummadi, K. P., & Druschel, P. (2010). You are who you know: Inferring user profiles in online social networks. *WSDM 2010 - Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*. <https://doi.org/10.1145/1718487.1718519>
- Mohammed, A. A., Minhas, R., Jonathan Wu, Q. M., & Sid-Ahmed, M. A. (2011). Human face recognition based on multidimensional PCA and extreme learning machine. *Pattern Recognition*, 44(10–11). <https://doi.org/10.1016/j.patcog.2011.03.013>
- Motoyama, M., & Varghese, G. (2009). I seek you: Searching and matching individuals in social networks. *International Conference on Information and Knowledge Management, Proceedings*. <https://doi.org/10.1145/1651587.1651604>
- Nagrath, P., Jain, R., Madan, A., Arora, R., Kataria, P., & Hemanth, J. (2021). SSDMNV2: A

- real time DNN-based face mask detection system using single shot multibox detector and MobileNetV2. *Sustainable Cities and Society*, 66.
<https://doi.org/10.1016/j.scs.2020.102692>
- Nguyen, H. V., & Bai, L. (2011). Cosine similarity metric learning for face verification. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6493 LNCS(PART 2).
https://doi.org/10.1007/978-3-642-19309-5_55
- Nguyen, T. (2022, April 11). *YOLOFace*. GitHub. <https://github.com/sthanhng/yoloface>
- Oliseenko, V. D., & Abramov, M. V. (2021). Identification of user profiles in online social networks: A combined approach with face recognition. *Journal of Physics: Conference Series*, 1864(1). <https://doi.org/10.1088/1742-6596/1864/1/012119>
- OpenCV. (2015). *OpenCV: Cascade Classifier*. Docs.opencv.org. Retrieved 2022, from https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html
- OpenCV (2018). *opencv/samples/dnn at master · opencv/opencv*. GitHub. Retrieved 2022, from <https://github.com/opencv/opencv/tree/master/samples/dnn>
- Pal, A. (2021, August 11). K-Means Clustering Use-Cases in Security Domain. Retrieved 2022, from Medium website: <https://ashish-pal.medium.com/k-means-clustering-use-cases-in-security-domain-8d20948c8eff>
- Park, G. Y., Kim, H., Jeong, H. W., & Youn, H. Y. (2013). A novel cluster head selection method based on k-means algorithm for energy efficient wireless sensor network. *Proceedings - 27th International Conference on Advanced Information Networking and Applications Workshops, WAINA 2013*. <https://doi.org/10.1109/WAINA.2013.123>
- Parkhi, O. M., Vedaldi, A., & Zisserman, A. (2015). *Deep Face Recognition*.
<https://doi.org/10.5244/c.29.41>
- Peng, P., Tian, Y., Wang, Y., Li, J., & Huang, T. (2015). Robust multiple cameras pedestrian detection with multi-view Bayesian network. *Pattern Recognition*, 48(5).
<https://doi.org/10.1016/j.patcog.2014.12.004>

- Peng, S., Zhou, Y., Cao, L., Yu, S., Niu, J., & Jia, W. (2018). Influence analysis in social networks: A survey. *Journal of Network and Computer Applications*, Vol. 106. <https://doi.org/10.1016/j.jnca.2018.01.005>
- Perito, D., Castelluccia, C., Kaafar, M. A., & Manils, P. (2011). How unique and traceable are usernames? *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6794 LNCS. https://doi.org/10.1007/978-3-642-22263-4_1
- Preotiuc-Pietro, D., Hopkins, D. J., Liu, Y., & Ungar, L. (2017). Beyond binary labels: Political ideology prediction of twitter users. *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers), 1*. <https://doi.org/10.18653/v1/P17-1068>
- Priadana, A., & Habibi, M. (2019). Face detection using haar cascades to filter selfie face image on instagram. *Proceeding - 2019 International Conference of Artificial Intelligence and Information Technology, ICAIIT 2019*. <https://doi.org/10.1109/ICAIIIT.2019.8834526>
- Raad, E., Chbeir, R., & Dipanda, A. (2010). User profile matching in social networks. *Proceedings - 13th International Conference on Network-Based Information Systems, NBIIS 2010*. <https://doi.org/10.1109/NBiS.2010.35>
- Rathore, S., Sharma, P. K., Loia, V., Jeong, Y. S., & Park, J. H. (2017). Social network security: Issues, challenges, threats, and solutions. *Information Sciences*, 421. <https://doi.org/10.1016/j.ins.2017.08.063>
- Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-December*. <https://doi.org/10.1109/CVPR.2016.91>
- Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, faster, stronger. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017-January*. <https://doi.org/10.1109/CVPR.2017.690>
- Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. arXiv preprint

arXiv:1804.02767.

Ren, H., & Li, Z. N. (2014). Object detection using edge histogram of oriented gradient. *2014 IEEE International Conference on Image Processing, ICIP 2014*.

<https://doi.org/10.1109/ICIP.2014.7025824>

Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6). <https://doi.org/10.1109/TPAMI.2016.2577031>

Riederer, C., Kim, Y., Chaintreau, A., Korula, N., & Lattanzi, S. (2016). Linking users across domains with location data: Theory and validation. *25th International World Wide Web Conference, WWW 2016*. <https://doi.org/10.1145/2872427.2883002>

Roedler, R., Kergl, D., & Rodosek, G. D. (2017). Content driven profile matching across online social networks. *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2017*.

<https://doi.org/10.1145/3110025.3110095>

Sajana, T., Sheela Rani, C. M., & Narayana, K. V. (2016). A survey on clustering techniques for big data mining. *Indian Journal of Science and Technology*, 9(3).

<https://doi.org/10.17485/ijst/2016/v9i3/75971>

Sankaranarayanan, S., Alavi, A., Castillo, C. D., & Chellappa, R. (2016). Triplet probabilistic embedding for face verification and clustering. *2016 IEEE 8th International Conference on Biometrics Theory, Applications and Systems, BTAS 2016*.

<https://doi.org/10.1109/BTAS.2016.7791205>

Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 07-12-June-2015*.

<https://doi.org/10.1109/CVPR.2015.7298682>

Sharifara, A., Mohd Rahim, M. S., & Anisi, Y. (2015). A general review of human face detection including a study of neural networks and Haar feature-based cascade classifier in face detection. *Proceedings - 2014 International Symposium on Biometrics and Security Technologies, ISBAST 2014*. <https://doi.org/10.1109/ISBAST.2014.7013097>

- Sharif, M., Naz, F., Yasmin, M., Shahid, M. A., & Rehman, A. (2017). Face recognition: A survey. *Journal of Engineering Science and Technology Review. Eastern Macedonia and Thrace Institute of Technology*. <https://doi.org/10.25103/jestr.102.20>
- Shen, Y., & Jin, H. (2014). Controllable information sharing for user accounts linkage across multiple online social networks. *CIKM 2014 - Proceedings of the 2014 ACM International Conference on Information and Knowledge Management*. <https://doi.org/10.1145/2661829.2661985>
- Shu, K., Sliva, A., Wang, S., Tang, J., & Liu, H. (2017). Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, 19(1), 22–36. <https://doi.org/10.1145/3137597.3137600>
- sklearn.cluster.AgglomerativeClustering. (2019). Retrieved 2022, from scikit-learn website: <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html>
- sklearn.cluster.KMeans — scikit-learn 0.21.3 documentation. (2019). Scikit-Learn.org. <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html>
- sklearn.svm.SVC — scikit-learn 0.22 documentation. (2019). Retrieved from Scikit-learn.org website: <https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>
- Sokhin, T., Butakov, N., & Nasonov, D. (2019). User Profiles Matching for Different Social Networks Based on Faces Identification. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11734 LNAI. https://doi.org/10.1007/978-3-030-29859-3_47
- Soltani, R., & Abhari, A. (2013). Identity matching in social media platforms. *Proceedings of the 2013 International Symposium on Performance Evaluation of Computer and Telecommunication Systems, SPECTS 2013 - Part of SummerSim 2013 Multiconference*.
- Statista. (2022, January). Global Social Media Ranking 2022. Statista. <https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/>
- Surasak, T., Takahiro, I., Cheng, C. H., Wang, C. E., & Sheng, P. Y. (2018). Histogram of

- oriented gradients for human detection in video. *Proceedings of 2018 5th International Conference on Business and Industrial Research: Smart Technology for Next Generation of Information, Engineering, Business and Social Science, ICBIR 2018*.
<https://doi.org/10.1109/ICBIR.2018.8391187>
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... Rabinovich, A. (2015). Going deeper with convolutions. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 07-12-June-2015*.
<https://doi.org/10.1109/CVPR.2015.7298594>
- Tadesse, M. M., Lin, H., Xu, B., & Yang, L. (2019). Detection of depression-related posts in reddit social media forum. *IEEE Access*, 7.
<https://doi.org/10.1109/ACCESS.2019.2909180>
- Taigman, Y., Yang, M., Ranzato, M., & Wolf, L. (2014). DeepFace: Closing the gap to human-level performance in face verification. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.
<https://doi.org/10.1109/CVPR.2014.220>
- Tan, S., Guan, Z., Cai, D., Qin, X., Bu, J., & Chen, C. (2014). Mapping users across networks by Manifold Alignment on Hypergraph. *Proceedings of the National Conference on Artificial Intelligence, 1*. <https://doi.org/10.1609/aaai.v28i1.8720>
- The Global State of Digital in July 2022. (2022, July). DataReportal – Global Digital Insights.
<https://datareportal.com/reports/digital-2022-july-global-statshot>
- Toor, A., Husnain, M., & Hussain, T. (2017). The impact of social network marketing on consumer purchase intention in Pakistan: Consumer engagement as a mediator. *Asian Journal of Business and Accounting*, 10(1).
- Tschiatschek, S., Singla, A., Gomez Rodriguez, M., Merchant, A., & Krause, A. (2018). Fake News Detection in Social Networks via Crowd Signals. *The Web Conference 2018 - Companion of the World Wide Web Conference, WWW 2018*.
<https://doi.org/10.1145/3184558.3188722>
- Vadakkepat, P., Lim, P., De Silva, L. C., Jing, L., & Ling, L. L. (2008). Multimodal approach to human-face detection and tracking. *IEEE Transactions on Industrial Electronics*,

55(3). <https://doi.org/10.1109/TIE.2007.903993>

Verma, M., Srivastava, M., Chack, N., Diswar, A. K., & Gupta, N. (2012). A Comparative Study of Various Clustering Algorithms in Data Mining. *International Journal of Engineering Research and Applications Wwww.Ijera.Com*, 2(3).

Vinh, T. Q., & Anh, N. T. N. (2020). Real-Time Face Mask Detector Using YOLOv3 Algorithm and Haar Cascade Classifier. *Proceedings - 2020 International Conference on Advanced Computing and Applications, ACOMP 2020*.
<https://doi.org/10.1109/ACOMP50827.2020.00029>

Vinay, A., Shekhar, V. S., Murthy, K. N. B., & Natarajan, S. (2015). Face Recognition Using Gabor Wavelet Features with PCA and KPCA - A Comparative Study. *Procedia Computer Science*, 57. <https://doi.org/10.1016/j.procs.2015.07.434>

Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1*. <https://doi.org/10.1109/cvpr.2001.990517>

Visual Geometry Group - University of Oxford. (2015). www.robots.ox.ac.uk. Retrieved 2022, from https://www.robots.ox.ac.uk/~vgg/data/vgg_face

Vosecky, J., Hong, D., & Shen, V. Y. (2009). User identification across multiple social networks. *2009 1st International Conference on Networked Digital Technologies, NDT 2009*. <https://doi.org/10.1109/NDT.2009.5272173>

Wang, L., Hu, K., Zhang, Y., & Cao, S. (2019). Factor Graph Model Based User Profile Matching across Social Networks. *IEEE Access*, 7.
<https://doi.org/10.1109/ACCESS.2019.2948073>

Wang, M., & Deng, W. (2021). Deep face recognition: A survey. *Neurocomputing*, 429, 215-244.

Wang, Y., & Zheng, J. (2018). Real-time face detection based on YOLO. *1st IEEE International Conference on Knowledge Innovation and Invention, ICKII 2018*.
<https://doi.org/10.1109/ICKII.2018.8569109>

Wei, X., & Stillwell, D. (2017). How smart does your profile image look? Estimating

- intelligence from social network profile images. *WSDM 2017 - Proceedings of the 10th ACM International Conference on Web Search and Data Mining*.
<https://doi.org/10.1145/3018661.3018663>
- Xiao, C., Freeman, D. M., & Hwa, T. (2015). Detecting clusters of fake accounts in online social networks. *AISeC 2015 - Proceedings of the 8th ACM Workshop on Artificial Intelligence and Security, Co-Located with CCS 2015*.
<https://doi.org/10.1145/2808769.2808779>
- Xu, D., & Tian, Y. (2015). A Comprehensive Survey of Clustering Algorithms. *Annals of Data Science*, 2(2). <https://doi.org/10.1007/s40745-015-0040-1>
- Xue, D., Wu, L. F., Li, H. B., Hong, Z., & Zhou, Z. J. (2017). A novel destination prediction attack and corresponding location privacy protection method in geo-social networks. *International Journal of Distributed Sensor Networks*, 13(1).
<https://doi.org/10.1177/1550147716685421>
- Yang, S., Luo, P., Loy, C. C., & Tang, X. (2016). WIDER FACE: A face detection benchmark. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-December*. <https://doi.org/10.1109/CVPR.2016.596>
- Yang, S., Luo, P., Loy, C. C., & Tang, X. (2018). Faceness-Net: Face Detection through Deep Facial Part Responses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(8). <https://doi.org/10.1109/TPAMI.2017.2738644>
- Yassein, M. B., Aljawarneh, S., & Wahsheh, Y. A. (2019). Survey of online social networks threats and solutions. *2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology, JEEIT 2019 - Proceedings*.
<https://doi.org/10.1109/JEEIT.2019.8717381>
- Yoon, S., Kleinman, M., Mertz, J., & Brannick, M. (2019). Is social network site usage related to depression? A meta-analysis of Facebook–depression relations. *Journal of Affective Disorders*, Vol. 248. <https://doi.org/10.1016/j.jad.2019.01.026>
- Zafarani, R., & Liu, H. (2009). Connecting Corresponding Identities across Communities. *International AAAI Conference on Web and Social Media*.
- Zafarani, R., & Liu, H. (2013). Connecting users across social media sites: A behavioral-

- modeling approach. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Part F128815*.
<https://doi.org/10.1145/2487575.2487648>
- Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8689 LNCS(PART 1).
https://doi.org/10.1007/978-3-319-10590-1_53
- Žemgulys, J., Raudonis, V., Maskeliunas, R., & Damaševičius, R. (2018). Recognition of basketball referee signals from videos using Histogram of Oriented Gradients (HOG) and Support Vector Machine (SVM). *Procedia Computer Science*, 130.
<https://doi.org/10.1016/j.procs.2018.04.095>
- Zhang, F., Qi, L., & Chen, E. (2015). Face Recognition Based on Uncorrelated Multilinear PCA Plus Classical LDA. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 8(3), 347–356. <https://doi.org/10.14257/ijsp.2015.8.3.32>
- Zhang, K., Zhang, Z., Li, Z., & Qiao, Y. (2016). Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. *IEEE Signal Processing Letters*, 23(10).
<https://doi.org/10.1109/LSP.2016.2603342>
- Zhang, N., Luo, J., & Gao, W. (2020). Research on face detection technology based on MTCNN. *Proceedings - 2020 International Conference on Computer Network, Electronic and Automation, ICCNEA 2020*.
<https://doi.org/10.1109/ICCNEA50255.2020.00040>
- Zhang, S., Zhu, X., Lei, Z., Shi, H., Wang, X., & Li, S. Z. (2017). S3FD: Single Shot Scale-Invariant Face Detector. *Proceedings of the IEEE International Conference on Computer Vision, 2017-October*. <https://doi.org/10.1109/ICCV.2017.30>
- Zhao, X. M., & Wei, C. B. (2017). A real-time face recognition system based on the improved LBPH algorithm. *2017 IEEE 2nd International Conference on Signal and Image Processing, ICSIP 2017, 2017-January*. <https://doi.org/10.1109/SIPROCESS.2017.8124508>
- Zheng, W., Yuan, C. H., Chang, W. H., & Wu, Y. C. J. (2016). Profile pictures on social media: Gender and regional differences. *Computers in Human Behavior*, 63.

<https://doi.org/10.1016/j.chb.2016.06.041>

Zhou, X., Liang, X., Zhang, H., & Ma, Y. (2016). Cross-Platform Identification of Anonymous Identical Users in Multiple Social Media Networks. *IEEE Transactions on Knowledge and Data Engineering*, 28(2). <https://doi.org/10.1109/TKDE.2015.2485222>

