IDENTITY MATCHING IN SOCIAL MEDIA PLATFORMS

by

Reza Soltani
BSc. (Hons), York University, 2010

A thesis
presented to Ryerson University
in partial fulfillment of the
requirement for the degree of
Master of Science
in the program of Computer Science

Toronto, Ontario, Canada, 2013

AUTHOR’S DECLARATION

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my thesis may be made electronically available to the public.

REZA SOLTANI

IDENTITY MATCHING IN SOCIAL MEDIA PLATFORMS

Reza Soltani
Master of Science, Computer Science, 2013
Ryerson University, Toronto, Ontario, Canada

Abstract

Identity matching is the process of mapping profile information from disparate data sources to one single entity; this is a crucial task for many businesses and governments. Introduction of Web 2.0 and the ever increasing number of social media platforms has led to an explosive amount of user participation and collaboration on web. An ordinary user has more than one social media profile, each of which has a unique set of properties and features. This thesis proposes a framework that uses syntactic and semantic based identity matching approaches among Facebook, Linkedin and Twitter user profiles. The framework accomplishes this task by collecting available profile data and performing analysis and comparison using a set of methodologies. These methods consist of weighted string matching techniques, Google Maps, YouTube and NLP web APIs. Extracted Profiles with a similarity score above a pre-computed threshold value are considered a match.

ACKNOWLEDGEMENT

I would like to express my gratitude to my supervisor Dr. Abdolreza Abhari for his useful feedbacks, remarks and engagements throughout the learning process of this master thesis. Without his patience, dedication and invaluable guidance completion of this work would not be possible.

I also wish to express my gratitude to my family, for their understanding and support, through the duration of my studies. I am truly grateful for their encouragements and advices.

I also like to thank the department of computer science at Ryerson University for providing me the facilities and services for collaboration and completing my research.

DEDICATION

To my Loving Family; my source of strength and pride

TABLE OF CONTENTS

CHAPTER 3

CHAPTER 4

CHAPTER 5

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

ACRONYMS

ACRONYM	**DEFINITION**
API	Application Programming Interface
CDI	Customer Data Integration
JSON	JavaScript Object Notation
MPI	Master Person Index
MDM	Master Data Management
NLP	Natural Language Processing
PI	Personal Identity
PIFS	Personal Identity Final Similarity Score
RI	Relational Identity
RIFS	Relational Identity Final Similarity Score
SI	Social Identity
SIFS	Social Identity Final Similarity Score
TF.IDF	Term frequency, Inverse document frequency
XML	Extensible Markup Language

CHAPTER 1

1. INTRODUCTION

As Deepak Chopra once said “The world is not a collection of things. It is a network of relationships”, user profiles too are connected with other profiles and entities. The concept of Identity matching is a crucial topic that spans across many fields including artificial intelligence, statistics, probability and database management. Nowadays social media has become an integral part of online users. Each social media site has a diverse set of features and functionalities. A common user has multiple social media accounts [1] and in most cases these accounts are not publicly connected to each other.

1.1. Motivation

Identity matching is a non-trivial and challenging research topic that serves crucial purposes for governments and businesses. Identity matching can also be used to supplement recommendation systems and assess privacy and digital footprints of users. In addition identity matching in social media can be used to detect spam and malicious users across multiple social media sites [2]. The following sections elaborate on the benefits of the general concept of identity matching to businesses and governments.

1.1.1. Importance of Identity Matching for Businesses

As we move forward, companies are required to deal with substantial amount of heterogeneous and unprocessed data. These data need to be intelligently analyzed, categorized and stored to be beneficial to the business. This process is also known as business intelligence or BI. On the other hand the processed information should not lose their integrity or initial context. In almost all organizations there is a genuine need for developing an accurate, structured and de-duplicated list of clients and/or employees. Identity matching is a key element in the creation of such list. The use of an intelligent identity matching mechanism leads to reduced company expenses, frauds, risks and conflict of interests. It also aids with improved marketing campaign and sales.

The reduction of expenses is due to reduced amount of manual labor work required in searching, sorting and de-duplicating client lists. Reduction of fraud and risks is partly due to more accurate profile consolidation which leads to faster and more streamlined access to clients' record and history.

For example a business would like to compile a complete list of its clients from every branch. This list is also referred to as Master Person Index or MPI. In this process there should be a method of figuring out if a person named "Joe H Smith" is the same person as "J Howard Smith", so that there aren't two separate records created for one individual. On the other hand a client company named "ABC Trades" should not marked as the same as "ABC Law firm". This process of removing duplicates and maintaining the client data is also known as Master Data Management or MDM.

Furthermore with the identity matching in place, the business would obtain 360 degrees, consolidated view of each customer. The consolidation process is also knows as Customer Data Integration or CDI. In the case where the business uses its client list to send out marketing and sales offers, if there are duplicate records within the database, a client will get multiple copies of the same offer. This is costly for the business and irritating for the client.

1.1.2. Importance of Identity Matching for Governments

Governments are another keen user of identity matching software. All levels of the government, from police force to border control to intelligence agencies need to have access to accurate and complete data. Identity matching increases the accuracy and integrity of government data by enhancing the searching, grouping and profile matching processes. This in turn lowers operational costs and provides a more comprehensive security defense against criminals and terrorists.

Border control and security agencies rely on uniform, synchronized and reliable database that can be accessed by every authorized employee at any time. The authorized employee may provide little and incomplete details of their subject person due to the lack of time or information. A system that incorporates a smart identity resolution framework is able to quickly process and identify the person based on the scarce input. This system could match and consolidate identities based on not just simple personal attributes such as name and date of birth, but also take advantage of person's social behaviors, social interactions and criminal history. Anti-terrorism and security agencies such as FBI or CSIS must be able to determine whether an individual is in fact the person they describe to be. This is an essential ingredient in finding potential terrorists and preventing acts of terrorism [3].

In the healthcare sector, access to accurate and structured data of patient is of imperative importance. At all levels of healthcare service such as emergency department, walk in clinics and labs there needs to be an intelligent tool for matching patient's approximate information with existing database. Different hospitals at different cities should be able to access a patient's universal profile and past records fast and with ease without missing any details previously added by other departments in other locations.

1.2. Objective and Scope

The objective of this thesis is to find a practical solution for the task of matching the similar profiles across heterogeneous social media sites. Figure 1.1 visualizes the result of identity matching among Facebook[4], Linkedin[5] and Twitter[6]. The thesis problem statement is as follow.

Problem Statement

Assuming user *X* is an online user. If User *X* has an account on Facebook, and if User *X* also has an account on Social media site Linkedin and Twitter.

Given the Facebook profile of user *X*, The objective of the proposed framework is to return the profiles of user *X* from Linkedin and Twitter.

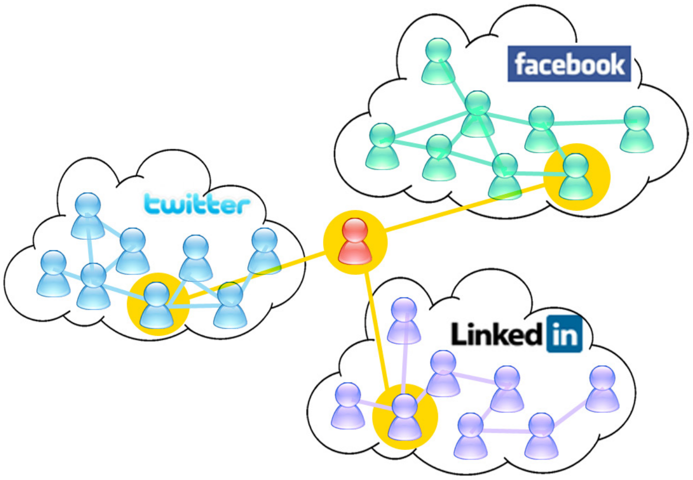


Figure 1.1 Matching profiles of the same person from different social media sites Facebook, Twitter and Linkedin

To mathematically state the theory and objective behind identity matching, consider the two populations A and B whose elements will be denoted by a and b respectively. Some elements (profiles) are assumed to be common to A and B. Therefore the set of ordered pairs can be stated as:

$$A \times B = \{(a, b) \mid a \in A, b \in B\} \tag{1.1}$$

$A \times B$ is the union of two disjoint sets named M and U. M (also referred to as matched set) is defined as:

$$M = \{ (a, b) \mid a = b , a \in A, b \in B\} \tag{1.2}$$

and Set U (also known as mismatched set) is defined as:

$$U = \{ (a, b) \mid a \neq b , a \in A, b \in B\} \tag{1.3}$$

Set *U* is the complements of set *M.* Each profile pair (a, b) within the matched set *M* belong to the same physical person. The objective of this thesis is to find out whether a profile pair (*a, b*) belongs to set *M* or *U*. Deciding on whether a pair of profiles (a, b) is a match or non-match comes down to comparing the similarity of the profile attributes.

1.3. Thesis Contributions and Challenges

This thesis provides a practical solution to the problem of identity matching across multiple social media platform. It introduces a framework that performs identity matching among Facebook, Twitter and Linkedin. The contributions of this thesis can be listed as follow:

- The proposed framework performs automatic search and identity matching among Facebook, Linkedin and Twitter user profiles by extracting and comparing profile attributes, online posts and user networks using syntactic and semantic methods.
- Furthermore this thesis shows the effect of the availability of information on performing identity matching.

Matching a pair of social media profiles is not a trivial task. User profiles do not necessary have a global and unique identifier (such as SIN number) that can assist in matching profiles. Even profile attributes such as *email* cannot be used as a universal identifier across all social media profiles because users may chose to use different email addresses on each of their profiles. (e.g. business email on Linkedin and personal email on Facebook). In addition social media profile data is always polluted, meaning it has typographical, missing, abbreviations and out of date values. The following points are some of the major challenges in matching social media profiles:

- Identity matching among social media profiles implies that beside simple profile attributes such as *name* and *location*, there are other profile features such as shared posts and users' network that need to be accounted for; The objective of this thesis is to propose a novel framework that goes beyond the basic profile attributes and considers other features of social media profiles. Table 1.1 represents the list of profile information that is extracted from each social media site by the proposed framework.

- In addition as opposed to conventional data sources such as database files, social media sites are dynamic and periodically introduce new profile attributes and new methods of accessing the information (e.g. changes to the API). These routine changes affect the framework execution and its performance. Therefore the proposed framework is developed in a modular approach which makes it easy to add and remove algorithms and code libraries.
- Furthermore conventional identity matching frameworks have direct access to profile information. On the other hand, due to the nature of social media sites, the proposed framework must tackle the access difficulties caused by privacy policies and permission rules.
- Moreover In the case of social media profile matching, finding correct profile matches to be used as the ground truth or as training data is not a trivial task as there is no public and trustable database of connected social media profiles. The data sets used in this thesis to perform framework validation are manually checked by us to ensure for correct matches.
- To obtain similar profiles from Linkedin and Twitter based on a given Facebook profile the framework has to perform a search using the search APIs provided by the social media sites. The search parameters and their values have a drastic affect on the result returned by the social media sites and consequently on the result of the framework. While performing the search, the assumption of the framework is that users use the same or at least very similar names across social media sites.
- Finally every identity matching package must have mechanism in place to protect the privacy and identity of the profile owners. The current version of the framework is password protected and removes all profile information after identity matching performance. Future iteration of the proposed framework can incorporate hashing and more advance security measures to protect the identity of profile owners.

Table 1.1 Data fields extracted from social media sites

Social Media site	Data fields
Facebook	First name, last name, id, email, username, location, place of birth, date of birth, occupation, education, recent posts, friendships
Twitter	Name, username, tweets
Linkedin	First name, last name, username, education, occupation, connections

1.4. Outline

The rest of our thesis is structured as outlined below:

Chapter 2 explains the past and recent research works on different methods of identity matching, string matching algorithms and natural language processing. Chapter 3 discusses the proposed identity matching framework. Chapter 4 presents the evaluation results of the framework and provides discussions on NLP APIs and privacy issues. Chapter 5 provides a summary of the thesis and concluding remarks about the future of identity matching in social media as well as possible improvements to the proposed framework.

CHAPTER 2

2. BACKGROUND AND RELATED WORKS

The topic of identity matching goes by many names. Among them *data deduplication* [7], *name matching* [8] and *record linkage* [9] are some of the more popular terms given to identity matching. In this thesis the term *identity matching* is used. Section 2.1 provides an overview of the history and main approaches to identity matching. Section 2.2 will then explain the related work on identity matching in the context of social media. Section 2.3 will examine the underlying components of identity matching in social media, and finally section 2.4 describes some of the existing open source and commercial identity matching packages.

2.1. History and General Approaches to Identity Matching

The concept of identity matching started by a paper called ‘Record Linkage’ authored by Halbert L. Dunn in 1946 [10]. Dunn starts his article by defining identity matching as following “Each person in the world creates a Book of Life. This Book starts with birth and ends with death. Its pages are made up of the records of the principal events in life. Record linkage is the name given to the process of assembling pages of this Book into a volume”. In 1969 Fellegi and Alan Sunter [11] laid the mathematical foundation of many probabilistic identity matching frameworks that are used up to now. The first step of performing identity matching involves the collection and preparation of profile information.

2.1.1. Profile Data Preparation

The first phase of identity matching is the preparation of data; the data can be from a database, files, social media websites or any other source. Profile attributes may be defined differently across different data sources. For example the *name* attribute could be labeled *name* in one data source, and in another source it can be composed of separate labels for each part of the name such as *first-name*, *last-name* and *middle-name*. Moreover the data sources may have different format for the value of a profile attributes. This means that the name of attributes may be consistent across data sources but the values are not identical even when they refer to the same entity. For example the name 'Joe M Howard' and 'Dr. J Mike Howard Sr.' refer to the same person but are syntactically different. The profile information can be semantically stored using variety of existing standards and formats including Friend Of Friend (FOAF) [12] and Microformats [13].

Facebook, Twitter and Linkedin for the most part have a consistent structure for the attributes that are extracted by the proposed framework. For example all three social sites have separate attribute names for *first-name* and *last-name* and one attribute name for *location* which holds the city and province in which the user lives in. Therefore the focus of the proposed framework in this thesis is more on matching profile attribute values rather than attribute names.

There are two major approaches in performing identity matching. The first approach is called domain knowledge and rule based approach and the second approach is called probabilistic and machine learning approach. Each approach has its own benefits and disadvantages.

2.1.2. Domain Knowledge Approaches to Identity Matching

The first approach of identity matching is by use of comparison rules. In this approach the matching program attempts to pass a pair of profile through a set of comparison rules that result in profile match or mismatches. This approach in solving identity matching yields accurate results but requires the involvement of a knowledge expert to create the rules. The comparison rules designed are specific to the requirements and the data sources used to perform identity matching. Pairs of attributes among profiles are compared using either simple string comparison functions with Boolean output or with more complex string matching algorithms. Papers such as [14] perform identity matching by strict comparison between *first name, last name* an *date of birth* attributes. This method lacks the flexibility of considering for cases where some of the attributes are missing or have minor differences. Therefore this technique may lead to false negatives, meaning that the profiles are wrongly considered to be mismatched.

2.1.3. Probabilistic and Machine Learning Approaches to Identity Matching

Probabilistic methods originate by Fellegi and Sunter [11]. Fellegi and Sunter introduced a probabilistic based identity matching framework that marks pairs of profiles as match or non-match depending on the similarity among different profile attributes. Probabilistic approaches use weights and approximate string matching algorithms to compare each pair of profile attributes. The development of new classification approaches has lead to the creation of new identity matching techniques. The new approaches utilize training data to learn how to compare and match profiles. In these methods the decisions are based on the training dataset with known identity matching cases. These methods require very little user involvement depending on the method of learning; however the training data sets are selected by human experts. In frameworks that rely on data to train, the quality of the data has an important effect in the performance of the frameworks.

J. Li *et al.* [15] uses a probabilistic relational model (PRM) based approach to resolve identity matching among data records. The data records belong to citizens and data comparison is performed based on personal and social activity features. Personal identities include features such as name and location. Social activities features include previous events attended or interactions with other people. Their experiments show that using social activities featuring improves the matching performance.

2.2. Related Work: Identity matching in the Context of Social Media

Methods of matching social media profiles can be divided into two categories. In the first category there are research works that focus on the syntactic differences between profile attributes, and in the second category there are research works that use semantic similarity metrics to match social media profiles. The proposed framework utilizes both syntactic and semantic approaches to identity matching.

2.2.1. Syntactic Profile Matching

Various papers focus on syntactic comparison of profile attributes among many different social media sites. However none of the papers focus on all three aspects of profile data covered in this thesis, namely profile attributes user posts and user network. Motoyama and Varghese [16] perform syntactic identity matching by using a MySpace[17] profile as reference and finding similar Facebook profiles using the Facebook Search API. This paper also uses OCR engines to extract email addresses from Facebook profiles to be used during profile comparison. Martin Szomszor *et al.* [18] performs profile correlation among Delicious[19] and Flickr[20] social media sites. Through this approach this paper discovers important user interests, locations and events based on the existing overlaps between detected user profiles. Tereza Iofciu *et al.* [21] uses social tags and user IDs to match profiles among Flickr, Delicious and StumpleUpon [22] and achieves an accuracy of 90%. Danesh Irani *et al.* [23] uses names and pseudonyms (nicknames) to find and construct social network profiles of users. This work is able to construct up to 40% of users social footprint based on their names and pseudonyms. Cuneyt Akcora and Barbara Carminati [24] propose methods of comparing users' profile attributes and users' network of friends. They also propose a new method of inferring values for missing user profile attributes based on the most common value found among the user's online friends. Paridhi Jain *et al.* [2] focus on matching Facebook and Twitter profiles and introduces various methods of searching for profiles on social media (example: self-mention search, network search , content search and profile search). They also introduce an identity matching system that rely on syntactic comparison of *name* and *username* profile attributes in addition to image comparison among users' photos. Perito *et al.* [25] show that using the *username* attribute alone, it is possible to link profiles from separate social media platforms. In their work they show that this technique is almost always applicable due to the public availability of usernames.

Vosecky *et al*. [26] introduce a vector based comparison algorithm where each user profile is represented as a vector. This approach is similar to the methods used by search engines where a document is represented as vector of words. In the same sense a profile can be represented as a vector of profile attributes such as *name, location,* etc. In [26] each profile attribute is compared with the corresponding attribute of other profiles. The resulting similarity score is a value between 0 and 1. There is a specific weight for each attribute. This weight is multiplied with the similarity score to compute the weighted similarity score for a pair of attributes. In the final step all weighted similarity scores are summed to develop the similarity score for a pair of profiles. The profile similarity scores that are above a threshold value are considered a match.

Vosecky *et al*. introduces three methods of attribute comparison techniques: exact matching for attributes such as *gender*, partial matching for comparing parts of profile attributes in cases where there are abbreviation, or multiple terms; and finally a fuzzy matching method named VMN which is used to provide an approximate similarity value between two strings.

The data set used for this work is obtained by crawling real social media profiles from Facebook and StudiVZ which is a German language social network. The dataset is manually checked to ensure that part of the data consist of overlapping users who have publicly available profile information on both social media websites. The experiments performed by Vosecky *et al.* are two folds. The first step is the training phase which results in evaluating the optimal attribute weights and attribute vector. The attribute vector is the list of all attributes that are considered during profile comparisons. The second step is the testing phase using the calculated weights while only comparing the attribute vector obtained from the training phase.

Vosecky *et al*. only compare profile attributes and does not take users' posts and network into consideration. The proposed framework in this thesis evolves beyond syntactic comparison of profile attributes and considers uses' network and posts. More over in contrast to Vosecky's work, the proposed framework utilizes semantic approach to analyze and compare users profile information.

2.2.2. Semantic Profile Matching

There are a variety of papers that utilize different methods of semantic string matching to compare and match profiles. Cortis *et al.* [27] perform semantic profile matching among social media platforms using DPBedia[28]. DPBedia provides variety of semantic services including text annotations and multi-level ontology based on the information available on Wikipedia. Golbeck and Rothstein [29] propose a framework that performs profile matching using publicly available FOAF[12] information from multiple blogs and social media sites.

E. Raad *et al.* [30] perform a heuristic approach on the data and compare the available profile information using weighted similarity score. Profile pairs that have a similarity value above a certain threshold are considered a match. The weights associated with profile attributes can be set manually or automatically. The automatic weight assignment process is as follow: In the first step, profiles that belong to the same person (i.e. contain same email address) are grouped together. In the next step, the similarity between each pair of profile attributes is computed and stored. In the last step the weight of each attribute *A* is set to the aggregated value of *all* similarity scores obtained for attribute *A* during the previous steps. The aggregation function can be any classic aggregation function such as the *Average* function.

Raad *et al.* introduce various syntactic and semantic similarity metrics of analyzing profile attributes. In terms of syntactic metrics they use Jaro metric [31] to compare single word attributes and uses SoftTFIDF [8] for comparing multi word attributes. Moreover they use edit-distance [32] to compare URI, image captions, phone number, email and many other text based attributes. In terms of semantic comparison, this paper uses the Explicit Semantic Analysis (ESA) [33] technique which uses Wikipedia to compute the relatedness of two terms. The threshold value is computed based on the weights associated with the attributes. Again to combine the weight into a single value as the threshold, various aggregation functions can be used. The dataset used on Raad's work consists of 50 automatically generated profiles. The prototype developed by Raad, stores each profile as FOAF [12] documents in a uniform attribute naming structure.

In contrast to Raad's work, in this thesis with the exception of a few attributes, all attributes are compared using the edit distance approximate string matching algorithm. Moreover the framework introduced in this thesis goes beyond profile attributes and compare users' posts and users' networks. Moreover the proposed framework in this thesis uses YouTube API to categorize video posts, and Google API to compare users' location, neither of which is used in Raad's research.

The paper presented by Soltani and Abhari [34] and published in IEEE related conference provides the groundwork of the framework proposed by this thesis. The experiments performed by Soltani and Abhari paved the way for the framework proposed by this thesis.

The next section focuses on the underlying algorithms and concepts for performing identity matching in social media platforms.

2.3. Underlying Components of Identity matching in Social Media

The task of comparing two social media profiles essentially comes down to comparing their attributes and elements. Section 2.3.1 identifies the social media sites used in this thesis along with their unique attributes and features. Section 2.3.2 briefly explains the syntactic methods of comparing profiles attributes; and section 2.3.3 explains the NLP approaches to analyzing social media posts.

2.3.1. Introduction to Web 2.0 and Social media sites

Web 2.0 was publicly introduced during the O'ReillyMedia and MediaLive conference [35]. The term does not represent a technical upgrade to the web, but rather it describes new methods of using the web/Internet. Web 1.0 is associated with centralized business where the business provides content or product to the rest of the Internet users [36]; on the other hand Web 2.0 shifts the power of the web to the hands of users. This means the web user contribute in the creation of the product. An example of such business model is Wikipedia where their service is based on the content created by different individuals. Web 2.0 has many identifying features. For example websites have the ability to offer their functionality as a service usually through a set of APIs. Moreover in Web 2.0 technologies such as JavaScript and Flash are used pervasively to develop interactive websites. Another important feature of Web 2.0 is its collaborative nature which places user in the center of the business models. In other words users' contents participate in the growth of the websites and the businesses.

Since the introduction of Web 2.0, countless number of social media websites have emerged. Social media sites such as Facebook [4], Twitter [6], Linkedin[5] and YouTube[37] have established themselves as some of the dominating user-centered information sharing platforms.

Social media platforms in general can be viewed as graphs composed of nodes and edges. Nodes may represent users of the platform and edges represent the connection between the users. Nodes can also represent events, companies, groups or any other entity for which a user can be in association. Figure 2.1 illustrates the difference between a traditional data record and a graph representation of a user profile.

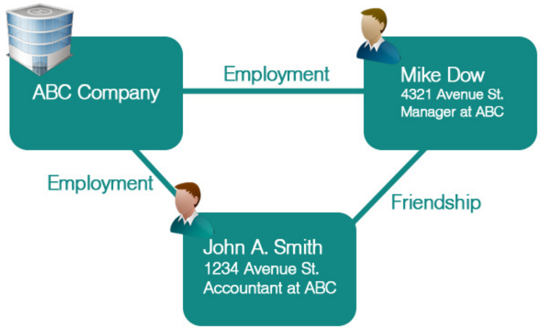
Traditional Record table	
Name	John A Smith
Location	1234 Avenue street
Date of birth	January 1st, 1980
Occupation	Accountant
Graph representation of entities	
	

Figure 2.1 Visualizing the difference between traditional data record (top) versus a graph of entities (bottom)

Facebook, Linkedin, Twitter and YouTube sites each have been designed with a specific market in mind and therefore provide a particular set of features and services. The following sections describe the aforementioned social media sites:

2.3.1.1. Facebook

Facebook's membership exceeds 1 billion accounts which make it the biggest and one of the most influential social media website. Users of Facebook are people of all ages, ethnicity, location and professional backgrounds. Facebook's interface is available in many different languages including English, French, German and Russian. Table 2.1 displays some of the statistics of Facebook [4]:

Table 2.1 Facebook Statistics

Year of creation	2004
Current number of members (Dec 2012)	More than 1 billion
Daily active users (Dec 2012)	618 million

Users of Facebook have personal profiles which consist of many fields/attributes. The following table 2.2 contains the majority of attributes found on each profile:

Table 2.2 Facebook profile fields

Attribute Name	**Type**	**Example**
First name	Text	John
Last name	Text	Smith
Middle name	Text	K
ID	Text	Jksxyz
Email	Text	John.smith@email.com
Website	Text	www.website.com
Date of birth	Date	05/10/1965
Place of birth	Text	Ottawa, Canada
Location	Text	Toronto, Canada
Education	Text	BSc Computer Science
Occupation	Text	Software developer at ABC Co.
Language	Text	English
A paragraph about the user	Text	'I am a tech junky'
Network	Text	Ryerson University, ABC Co.
Quotations	Text	'The universe is made of stories, not atoms.'
Cover Photo	Image	
Profile picture	Image	

Facebook users are also able to share materials with other users and groups. The materials include texts, URLs, pictures, videos, location check-ins and recently feelings/moods in the form of text and smiley faces. Users are also able to join specific groups and fan pages on Facebook and purchase products. Finally Facebook users are able to use (or author) applications and games built on top of the Facebook platform.

2.3.1.2. Twitter

Currently Twitter is the second largest social media website. Twitter offers its users the ability to broadcast short messages (up to 140 characters) to the public (or just followees, depending on the user settings). These short messages are known as tweets. Twitter contrasts with Facebook and Linkedin in that it does not provide as many profile fields. Consequently more emphasis is given to the tweets. Twitter is also available in many different languages. Table 2.3 displays the general statistics taken from [38] about Twitter and table 2.4 shows the profile fields available to Twitter users.

Table 2.3 Twitter statistics

Year of creation	2006
Current number of members (Sept 2012)	More than 500 million
Monthly active users (Sept 2012)	200 million

Table 2.4 Twitter profile fields

Attribute Name	**Type**	**Example**
Name	Text	John
Username	Text	Jksxyz
Language	Text	English
Email	Text	John.smith@email.com
Website	Text	www.website.com
Date of birth	Date	05/10/1965
Header Photo	Image	
Profile picture	Image	

2.3.1.3. Linkedin

Linkedin serves a specific demographic which include students and professionals who intent to broaden their knowledge and network with other professionals. Linkedin has about 200 million members [5] as of now and offers similar features as Facebook including having a personal profile and the ability to post material. In addition it allows for users to display their professional and technical skills, experiences and educational degrees in detail and topic based methods. Linkedin is available in many languages including English, French, Italian and German. Users can upload resume or design their profile to showcase their work and skills. Users can also follow companies and receive notifications about news and job offerings. Employers on the other hand can find potential employees by providing the job descriptions. Table 2.5 displays the statistics about Linkedin taken from [5] and table 2.6 shows the profile fields available to its Linkedin users.

Table 2.5 Linkedin statistics

Year of creation	2002
Current number of members (Sept 2012)	More than 200 million
Monthly active users (Sept 2012)	160 million

Table 2.6 Linkedin profile fields

Attribute Name	**Type**	**Example**
First name	Text	John
Last name	Text	Smith
Middle name	Text	K
ID	Text	Jksxyz
Email	Text	John.smith@email.com
Website	Text	www.website.com
Date of birth	Date	05/10/1965
Organization	Text	IEEE, Ryerson Alumni
Education	Text	MSc Ryerson University
Skills and Expertise	Set of Texts	BSc Computer Science

Language	Text	English
Courses	Text	Security fundamentals
Volunteer Experience	Text	Local hospital
Experience	Set of Texts	Ryerson University, ABC Co.
Summary	Text	Specialties and past experiences in a paragraph
Projects	Set of Texts	Location aware mobile application
Publications	Set of Texts	‘security in local ad-hoc networks’
Honors and Awards	Set of Texts	NSERC Grant
Test Scores	Set of Texts	80% in QX test
Patents	Set of Texts	
Certifications	Set of Texts	
Profile picture	Image	

2.3.1.4. YouTube

YouTube on the other hand allows the registered users to publish their own videos to the YouTube website and make them visible to the entire population of web users. YouTube users also have personal profiles which include username and name. In addition users can have channels which showcase their own videos. YouTube users can follow other user’s YouTube channels. YouTube is owned by Google [39]. YouTube offers multiple languages for its user-interface. Table 2.7 displays the statistics [37] about YouTube and table 2.8 shows the profile fields available to its users.

Table 2.7 YouTube statistics

Year of creation	2005
Current number of members (Sept 2012)	More than 500 million
Monthly active users (Sept 2012)	800 million

Table 2.8 YouTube statistics

Attribute Name	**Type**	**Example**
Name	Text	John
Username	Text	Jksxyz
Language	Text	English
Email	Text	John.smith@email.com
Website	Text	www.website.com
Date of birth	Date	05/10/1965
Header Photo	Image	
Profile picture	Image	

The amount of information shared by each user to the public and to the user's online friends is determined by the privacy terms of the platform and the user's personal settings. The number of social media platforms such as the ones mentioned is ever increasing. For many of these web sites a separate (i.e. unique) login account is required. This means that a user registered in three different social media sites such as Facebook, Linkedin and YouTube needs to have three separate accounts and therefore three separate profiles. Another important component of social media identity matching is the collection of string matching methods and techniques employed to compare profile attributes.

2.3.2. String matching techniques

In order to syntactically compare profile attributes the framework relies on string comparison techniques. In this thesis, three different types of attribute matching functions are used:

1. Exact Matching: The first type is the exact matching function that returns *true* if the two input string are identical and *false* otherwise. An example use of such attribute would be the *gender* attribute which is either 'male' or 'female'.
2. Geolocation proximity comparison: This type of string comparison is used solely for *location* and *place of birth* attributes. This type of comparison makes use of Google Maps API to compute the approximate distance between two locations.

3. Approximate String Matching: This type of comparison function returns a numeric value that specifies the syntactical difference between two input strings, this type of comparison is used for majority of profile attributes such as *name*, *education*, and *occupation*. This type of comparison function also checks for short forms variations of the strings. This extra feature is primary used for comparing first and last names. A name such as ‘John Howard Smith’ can have the short form of ‘J. H. Smith’ and ‘John. H. S.’.

There are many approaches to perform approximate string matching including Phonetic based, Token based and Pattern matching [8]. Edit distance is the primary approximate string matching algorithm used in this thesis and its formula is explained in chapter 3. Extended details about popular string matching techniques for identity matching and their variations can be found on the book called ‘An introduction to duplicate detection’ [40].

The next component of social media identity matching is the methods of semantically analyzing and comparing social media streams. Social media streams consist of online video, image or textual posts including tweets. The following section reviews some of the existing methods of analyzing this category of social media information.

2.3.3. Natural Language Processing Techniques

The topic of NLP (natural language processing) is a field of study in artificial intelligence, linguistics as well as human-computer interactions. The fundamental concept behind NLP is the understanding of the natural language by the computer to create meaningful and structured information. Modern NLP libraries use machine learning algorithms. They can process a sentence, a paragraph, or an entire page of natural text and perform many operations such as tokenization, chunking, sentence segmentation, named entity extraction and parsing. There are various NLP libraries such as OpenNLP and Stanford NLP that can be used for the purpose of the proposed framework. Aside from libraries that must be integrated within the framework, there are many NLP engines that are available through the web in the form of web services (i.e. web APIs). These engines offer similar semantic analysis and are able to convert unstructured texts (such as tweets, paragraphs, web pages and articles) into named entities (people, location, companies), topics, keywords and more. Table 2.9 list 5 NLP web services that have been reviewed for this thesis. Delegating the categorization task to a third party natural language processing service has the following advantage and disadvantages:

Advantages of Using Third Party NLP web APIs

Advantages:

- There are different use licenses (including a free license) available with NLP web API services
- Multiple NLP APIs can be used in conjunction to provide more accurate analysis of texts
- NLP web services provide more than just the category(topic) names. They can return key terms, language and other entities such as person names, company names, tags and locations.
- APIs Requires less time and effort to use and maintain compared to local NLP libraries

Disadvantages:

- Because API calls are done through the web, there is a latency involved with each call
- The technical details and implementation of the NLP web APIs are not fully disclosed due to competitive interests
- Based on the license being used there is typically limitation on the number of calls per day/per month as well as limits on the number of simultaneous calls.

Table 2.9 NLP Web services

Web service Name	**License types available**	**Free/trial license details**
AlchemyAPI[41]	Free/Commercial	30,000 API calls a day
OpenCalais[42]	Free/Commercial	50,000 API calls a day and 4 calls per second
Pingar[43]	Free/Commercial	Not available for free
Semantria[44]	Free/Commercial	10,000 API calls a month, 500 per minute
Wikimeta[45]	Free/Commercial	100 calls a day (maximum 10 mbytes a day)

The APIs used in this framework are AlchemyAPI and OpenCalais, this is based on the fact they both offer free use license with a high number of API calls per day. Moreover AlchemyAPI claims to be the most popular NLP service on the web used by many companies such as Shutterstock and PR Newswire.

Saif *et al.* [46] use AlchemyAPI[41] to add semantic annotation to tweets. Semantic annotation or tagging is the process of attaching names and attributes to a document or select part of a text. This process provides additional information (metadata) about that text. Abdel *et al.* [47] use OpenCalais[42] API to detect named entities in news related tweets. These entities are mostly company names, people, locations, products, etc. Steiner *et al.* [48] use AlchemyAPI and OpenCalais to extract name entities from Facebook posts. The results of name entities are delivered to the user in the form of mash-up like API.

2.4. Existing identity matching software and standards

There are existing technologies that allow users to use the same login information on different social media sites; however these solutions are not universally accepted among social media sites, and require the direct involvement of the user to register. Having fixed login information (i.e. same id) among all social media sites lead to easy recognition and profile matching by the identity matching framework. An example of similar technology is OpenID[49] which aims to centralize and unify the login process with a single username and password. This concept provides security because username and passwords are securely stored; as well as convenience because users are only required to remember one username and password.

There are also specific third party social media applications such as Lifestream [50] and FriendFeed [51] that allow the users to manage multiple social media profiles through a single interface. For example using Lifestream one would post the same post to both Facebook and Twitter while only submitting it once.

There are also specific options available on each social media site that allows linking of social media sites. For example Facebook allows its users to connect their profile to their Twitter account. This way all tweets sent by user’s Twitter account are also posted on their Facebook wall.

These concepts all depend upon the level of user involvement as the request to connect the profiles comes directly from the user. Third party applications and cross social media communications provided by the social media platforms all must be reviewed and approved by the users to be activated. The solution provided by this thesis is not concerned with whether the users have any direct involvement in interconnecting their social media accounts or utilizing any central authenticating system such as OpenID.

Open Source Software

Tailor [52] is an identity matching application written in Java which also provides performance results upon performing the comparison. This product is developed in modular fashion which separates the comparison function from the logic. Tailor supports a variety of string matching algorithms including Soundex, Phonex and Jaro. The input to this application is a plaintext file with the extension “.DTA”. DTA files are created using Stata which is a data analysis and statistical software.

Febrl which stands for Freely Extensible Biomedical Record Linkage [53] is an open source software with a graphical user interface authored by Peter Christen. This tool is written in Python and works on Windows, Linux and MacOS. It offers data standardization, deduplication and profile matching with numerous changeable parameters including the support for multiple string matching algorithms such as Jaro, Q-gram, and Edit distance. The input to this software is text files such as .CSV files.

Commercial Software

In terms of available commercial products, IBM InfoSphere® Global Name Management [54] is one of the leading commercial product for managing identity records. This tool also provides rule-based identity matching procedure where the user can provide the rules and choose different string matching techniques. Informatica Identity resolution[55] solution is another leading commercial product that provides identity matching and real time identity data search from different data sources. This software supports multiple languages and claims that it can overcome nicknames, spelling mistakes, abbreviations, and phonetic errors to find all relevant profile matches. Another company that focuses on identity matching is InfoGlide [56] which provides Identity resolution and social link discovery products. InfoGlide claims that their software can find, match and link similar individuals, locations and other entities. This is done through their patented similarity search algorithms. Their product also detects intentional or accidental misspellings and can search across multiple profile attributes rather than a single profile attribute which leads to reduced false positives. InfoGlide offer their identity matching service as a web service as well. Finally WizSame by Wizsoft [57] is an identity matching product that identifies records that are identical or similar. This software detects misspellings as well as short forms. The profile records used in comparison and the comparison rules are determined by the user. This program accepts any text file as input or it can connect directly to many database systems such as MySQL or Oracle database.

Majority of the existing software do not consider social media profiles as a standard source format input. Therefore in order to use social media profiles, as an intermediate step the profile information have to be first extracted using the social media APIs and then fed to the identity matching software. Moreover the proposed identity matching software is among the few programs that considers user’s posts and networks as well.

In summary this chapter provided an overview of existing methods of identity matching as well its underlying methodologies. Moreover it examined some of the existing open source and commercial software for identity matching.

CHAPTER 3

3. METHODOLOGY AND IMPLEMENTATION

The objective of this thesis is to provide a practical solution to find and match the profiles of a single person among multiple social media sites (figure 3.1). Inspired by the works of Vosecky *et al*. [26] and E. Raad *et al.* [30] which are described in chapter 2.2, the proposed framework uses both syntactic and semantic metrics to compare user profiles. In section 3.1 the overall design of the framework is explained. Section 3.2 describes each part of the framework in details and finally section 3.3 summarizes this chapter.

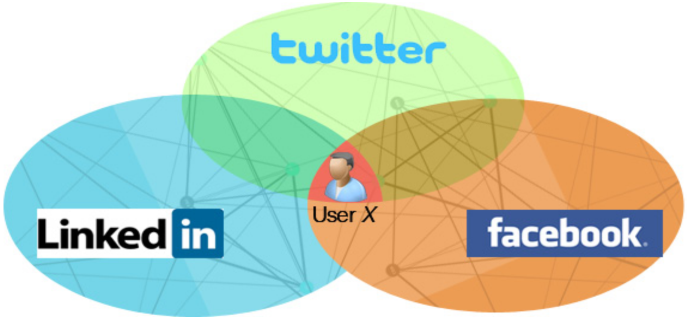


Figure 3.1 Identity matching among Facebook, Twitter, and Facebook profiles

3.1. Framework Overview

The general approach of the proposed framework is to accept a Facebook ID as input, and return corresponding Linkedin and Twitter profile URLs. To enhance the user experience, there is an add-on to the framework that allows the user of the framework to search for a Facebook user based on first and last name. Upon providing the name to the framework, the framework performs a search on Facebook and then presents a list of potential Facebook profiles to the user. The user then selects a profile from this list. The reason that Facebook is considered to be the first social networking website for input is due to two main reasons:

- Facebook has the highest number of users, meaning that it is very likely for the person in search to have a Facebook account
- Facebook profiles usually have a large set of profile attributes that are accessible by third party Facebook applications

The framework is divided into 3 major phases. They are 1): Input analysis, 2): search and data extraction, and 3): data analysis, comparison and decision making. Phase 1 is responsible for retrieving all Facebook information related to the input Facebook user (labeled as user *X* in this thesis*)*. Phase 2 is responsible for searching and retrieving Linkedin and Twitter profiles that are similar to user *X*. Finally phase 3 is responsible for analyzing and comparing the extracted profiles to obtain matching profiles. Figure 3.2 illustrates the three phases of the framework.

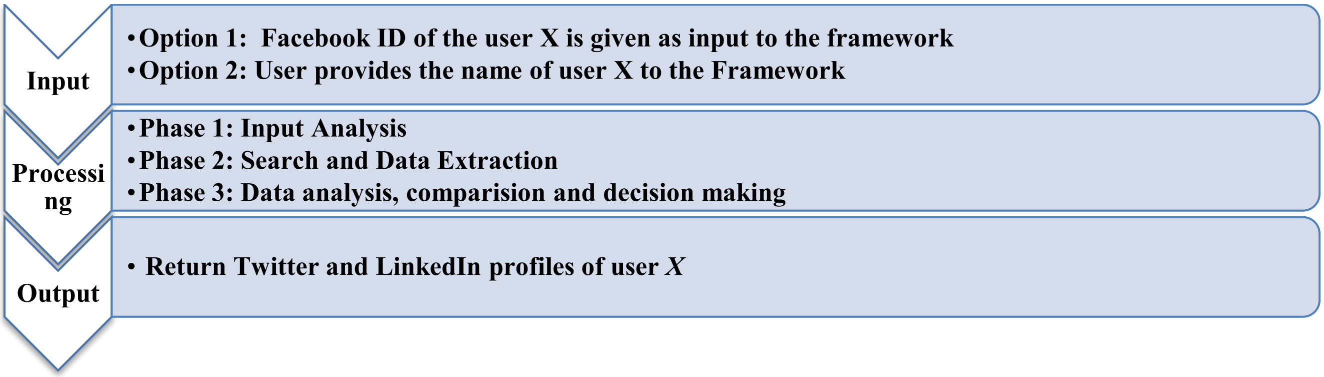

Input
•Option 1: Facebook ID of the user X is given as input to the framework
•Option 2: User provides the name of user X to the Framework
Processi ng
•Phase 1: Input Analysis
•Phase 2: Search and Data Extraction
•Phase 3: Data analysis, comparision and decision making
Output
• Return Twitter and LinkedIn profiles of user X

Figure 3.2 Flow of the Framework

To store, handle and process the information accurately and efficiently, the extracted data from social media sites are categorized into 3 different classes of identities (i.e. categories of profile data). They are *Personal Identities*, *Social Identities* and *Relational Identities*. Table 3.1 presents the 3 classes of identities along with their corresponding data fields on Facebook, Linkedin and Twitter.

Table 3.1 Three classes of data and their corresponding data fields

Class	Types of data contained in each class
Personal Identities (PI)	Name, location, date of birth, occupation, education
Social Identities (SI)	Facebook image, video, link and text posts., Twitter image and tweets, Linkedin text, image, video and link posts
Relational Identities (RI)	Facebook friendships, Linkedin Connections, Twitter followers, followees, Facebook and Linkedin Group memberships, Facebook fan page memberships

For each class of data, a specific set of algorithms are used to perform analysis and comparison. The followings section 3.2 describes the three phases of the framework in more details.

3.2. Three Phases of Framework

As mentioned the framework is comprised of three major phases. The following sections discuss the three phases.

3.2.1. Phase 1: Input Analysis

The first phase is responsible for fetching all available information for input user *X* from Facebook. The process of fetching information from Facebook is done by creating a Facebook application and calling the proper Facebook APIs. Section 4.3 explains about social media applications in more details. The input of the framework is essentially a Facebook ID. Assuming that this id is 'xyzfb', the framework connects with Facebook server and retrieves all 3 classes of identities for user 'xyzfb'.

3.2.2. Phase 2: Search and Data Extraction

In this phase of the framework, with the help of Linkedin and Twitter APIs, potential Linkedin and Twitter users that have a similar *name* as the Facebook user *X* (i.e. first name and/or last name) are searched and extracted.

Assumption

There is a limitation on using the search API to find similar profiles on Linkedin and Twitter based on the *name* profile attribute. The assumption is that users use similar names on Facebook, Linkedin and Twitter to identify themselves. In cases where the user chooses to use completely different names on social sites, the search API is not able to return profiles similar to Facebook User *X*.

The following algorithm 3.1 demonstrates the steps for performing the search on Linkedin and Twitter:

Algorithm 3.1 Search and data extraction algorithm for Linkedin and Twitter

Searching on Linkedin

```
Assuming User X's name is 'John Smith'

begin
results = SearchLinkedin(“john smith”, full-name)
If results ≠ Ø then
    return results
else
    results = SearchLinkedin(“smith”, last-name)
    if results ≠ Ø then
        return results
    else
        results = SearchLinkedin(“john”, first-name)
        if results ≠ Ø then
            return results
        else
            return null
        end
    end
end
end
```

Searching on Twitter

```
Assuming User X's name is 'John Smith'

begin
results = SearchTwitter (“john Smith”, full-name)
```

```
If results ≠ Ø then
    return results
else
    results = SearchTwitter (“smith”, last-name)
    if results ≠ Ø then
        return results
    else
        results = SearchTwitter (“john”, first-name)
        if results ≠ Ø then
            return results
        else
            return null
        end
    end
end
end
```

As algorithm 3.1 shows, if the search APIs are unable to find the person using the full Facebook names, then only the last name is chosen as the search query. If there are no results returned with only the last name, the first name alone will be used to perform the search. Finally if there are no results after the third attempt, *null* signal is returned to notify the framework that the Facebook user *X* cannot be found using the given name on Facebook. The quantity of search results requested from Linkedin and Twitter are variable and can be set through the framework's settings. Figure 3.3 illustrates the profile retrieval and search steps on Facebook, Linkedin and Twitter.

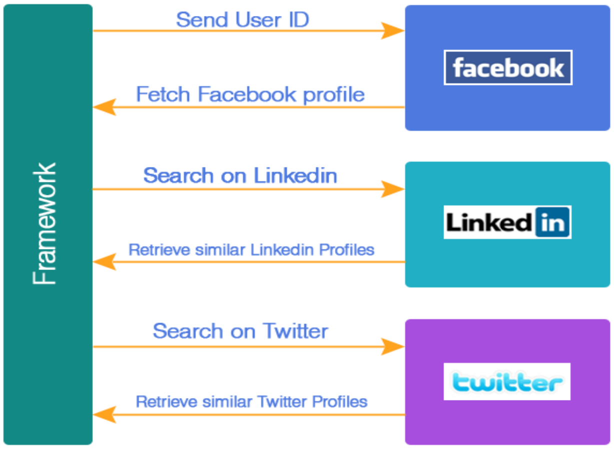


Figure 3.3 Flow of phase 1 and phase 2 of the framework. Interaction among the framework and the social media sites.

Use of search API has the following advantage and disadvantages:

Advantages:

1. The search APIs accept explicit attribute parameters such as ‘first name’ and ‘last name’
2. Indexing, sorting and searching are outsourced to the social media site
3. Pagination for search results is available
4. The search API has the ability to return only the specified fields such as name, location, gender, education. This feature increases the transmission efficiency.

Disadvantages:

1. Not all social media data is available to the framework at once, and the social media site decides on which profiles are to be returned first.
2. There are limitations on the number of search API calls that can be made within a day

3.2.3. Phase 3: Data Analysis, Comparison and Decision making

Phase 1 of this framework is responsible for collecting Facebook profile details; phase 2 of framework is responsible for collecting profile data from Linkedin and Twitter. The third and final phase of the framework is the crucial part of this thesis. Figure 3.4 displays the major steps of phase 3.

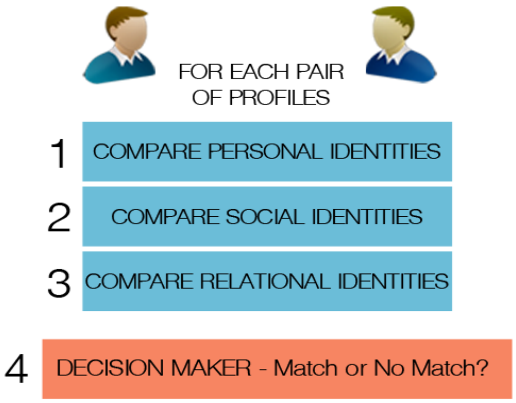

FOR EACH PAIR
OF PROFILES
1
COMPARE PERSONAL IDENTITIES
2
COMPARE SOCIAL IDENTITIES
3
COMPARE RELATIONAL IDENTITIES
4
DECISION MAKER - Match or No Match?

Figure 3.4 Flow of phase 3 of the framework which is responsible for comparing each pairs of profiles and decision making

This phase is responsible for comparing *Personal Identities*, and categorizing and comparing *Social Identities.* In addition it is responsible for comparing friendships and connections of retrieved user profiles which make up *Relational Identities*. Each class of identities is analyzed, categorized and compared differently. Algorithm 3.2 states the overall process of this phase.

Algorithm 3.2 Phase 3 of framework

```
Input:
fbX: the profile data of input user X ,
L: {l1...ln} Potential Linkedin profiles,
T: {t1...tn}  Potential Twitter profiles
τ: Profile matching threshold
Output:
MatchingSet: Matching Linkedin and Twitter profiles
Local Variables:
PIFS = Personal Identities Final Similarity Score,
SIFS = Social Identities Final Similarity Score,
RIFS = Relational Identities Final Similarity Score
Begin
    For fbX and every L profile:   // Finding Matching Linkedin Profiles
        PIFS = MatchPersonalIdentities(fbX, L)  // (Algorithm 3.3)
        RIFS = MatchRelationalIdentities(fbX, L) // (Algorithm 3.6)
        AveragedSimilarityScore = Average(PIFS, RIFS)
        if (AveragedSimilarityScore > $\tau$)
            MatchingSet = MatchingSet + L;
    End
    For fbX and every T profile: // Finding Matching Twitter Profiles
        SIFS = MatchSocialIdentities(fbX, T) // (Algorithm 3.5)
        if (SIFS > $\tau$)
            MatchingSet = MatchingSet + T;
    End
    return MatchingSet
End
```

3.2.3.1. Matching Personal Identities

Personal Identities are composed of strings and numbers (i.e. text based profile attributes). There are three methods of string matching used to compare *Personal Identities:*

1. Exact String Matching: This category compares attributes that are either exactly the same or completely opposite. *Gender* is an example of such attribute.
2. Location Matching: This category of comparison utilizes Google Maps API to compare attributes such as *Location* and *Place of Birth.* Google Maps API can return the Province and Country name of any given city. This feature can be used to find out if two particular cities are within a province or country.
3. Approximate String Matching: This category handles all other attributes including *name, occupation,* and *education.* A simple string matching algorithm that returns a Boolean response is not sufficient for these attributes. This is because the attribute values are in many cases misspelled, incomplete, abbreviated or reordered. For example a user may have the name 'John H Smith' on Facebook and then have 'J Howard Smith' on Linkedin, a simple Boolean string matching algorithm will return a false (mismatch) response but it is very likely that the two names may actually be for the same person.

As described in chapter 2 there are three main approaches to string matching algorithms namely: phonetic encoding, token based and pattern matching. Pattern matching includes the edit distance approach to string comparison. The general form of edit distance or Leveshtein Distance[32] attempts to measure the number of character changes required to equate two strings. For example the strings 'John' and 'Johnny' have the edit distance (or L-distance) of 2 because there needs to be 2 new characters added to the end of the name. On the other hand the edit distance of 'adam' and 'adan' is only 1 because there is need for one substitution to make the two strings equal. Based on previous experiments edit distance is a popular choice for identity matching frameworks as it provides accurate result [8], [15]. The improvements and add-ons to the edit distance algorithm are the following:

1. Normalize the resulting edit distance cost to be between 0 and 1 inclusive. 1 representing equal strings and 0 for completely different strings.
2. Consideration for short form names such as 'J Smith' or' J.S.' for 'John Smith'. If short forms and abbreviations are not considered a name such as 'Sarah K Johnson', 'Sarah K.J.' would not have the proper similarity score.
3. Increase of edit distance costs among shorter strings. This change puts more emphasis on edit distance values between shorter strings. For examples strings 'John' and 'Joun' have higher edit distance cost than 'Richardson' and 'Richardsun' . This is because accidental misspells are more prone in longer words.
4. Common words such as 'university', 'high school', 'middle school' and 'college' are omitted during comparison of some attributes such as *education*.

The string comparison algorithm is applied to every pair of profile attributes (i.e. *Personal Identity*). For example the *first name* attribute from the Facebook profile and the *first name* attribute from the Linkedin profile are passed to the string matching algorithm to obtain a similarity score. Algorithm 3.3 demonstrates the steps taken to compare two strings *A* and *B* for similarity.

Algorithm 3.3 Comparing two strings *A* and *B* using modified version of Edit distance

```
Input:
A, B:  Attribute values,
type: Attribute name,
$\tau$: Framework threshold value between 0 and 1
Output:
AttributeScore: Decimal value between 0 and 1
Local Variables:
editD: The edit distance value between two strings,
editDNew: The new edit distance value affected by the length of strings,
Score: Normalized score,
AttributeScore: Weighted similarity score between two attributes
begin
    if (type == "name") and (shortForm(A, B))  then
           score = compareShortForm(A, B)
    else
           editD = computeEditDistance (A, B)
           editDNew = readjustDistanceBasedOnLength(A,B, editD)
           Score = normalize(editDNew)
            if (type == "location") and (Score < $\tau$) then
                 Score = compareLocation(A, B, Score)
            end
    end
    AttributeScore = assignWeight(Score, weight(type))
    return AttributeScore
end
```

Within algorithm 3.3 the string matching function *compareEditDistance()* is the Leveshtein edit distance algorithm [32]. This recursive implementation of this algorithm is defined as follow:

Algorithm 3.4 Edit distance string matching algorithm for two strings A and B

```
Input:
A, B: strings to be compared
Output:
Edit distance cost between string A and B
Local Variables:
ALength: Length of String A
BLength: Length of String B
Cost: Cost associated with an operation
Begin function LevenshteinDistance(A, B)
  ALength = length(A);
  BLength = length(B);
  if (ALength == 0)   /* test for empty strings */
```

```
    return BLength;
  if (BLength == 0)   /* test for empty strings */
    return ALength;
  if (A[ALength - 1] == B[BLength - 1])/* test if last characters of strings match */
    cost = 0;
  else
    cost = 1;
  return minimum(LevenshteinDistance(A[0..ALength - 1], B) + 1,
                 LevenshteinDistance(A, B[0..BLength - 1]) + 1,
          LevenshteinDistance(A[0..ALength -1], B[0..BLength -1]) + cost)
End
```

The following paragraph will explain the functions within the algorithm 3.3. The function *readjustDistancekBasedOnLength()* on algorithm 3.3 is one of the new modifications introduced in this framework. After computing the edit distance of two string, the function *readjustDistancekBasedOnLength()* adds extra cost value depending on the length of the shorter string. As an example, consider two pairs of strings; in the first pair there are two strings 'Reza' and 'RezaS'. In the second pair there are 'Michael' and 'MichaelS'. The function *readjustDistancekBasedOnLength()* will cause the first pair of strings to have a higher edit distance cost because the length of the shorter string in the first pair (i.e. 'Reza') is smaller than shorter string in the second pair (i.e. 'Michael'). In other words this function gives more significance to typographical differences in shorter strings. This is because accidental misspellings are more prone in longer texts.

The function *readjustDistancekBasedOnLength(A,B, editD)* on algorithm 3.3 is defined as follow:

$$editDNew = \begin{cases} editD + \frac{1}{min(|A|,|B|)}, & editD > 0 \\ 0, & otherwise \end{cases} \qquad (3.1)$$

$$editDNew' \in \mathbb{R}$$

Given a set of numbers the function *min()* and *max()* in this thesis are mathematical functions that return the minimum and the maximum numbers of the sets respectively. The term $\frac{1}{\min(|A|,|B|)}$ is the inverse of the length of the shorter string. This formula produces a value which decreases as the length of the shorter string increases. If the value of the original edit distance $editD$ is 0, it means that the two strings are equivalent and the length of strings is no longer relevant and therefore the output $editDNew$ is also set to 0. The effect of *readjustDistancekBasedOnLength()* function with sample inputs is available on appendix A.

In algorithm 3.3 function *normalize()* maps the computed edit distance value to a value between 0 and 1. This is done by dividing the edit distance value by the maximum edit distance value possible. The maximum edit distance value possible is the sum of the length of longer string plus the highest possible cost obtained from the length of the shorter string (see formula 3.1). The function *normalize()* on algorithm 3.3 is defined as follow:

$$Score = 1 - \frac{editDNew}{max(|A|, |B|) + \frac{1}{min(|A|, |B|)}} \qquad (3.2)$$

$$Score \in [0,1]$$

In the second term of equation 3.2, the nominator is the number that will be normalized as a score. The nominator $editDNew$ is the value obtained from *readjustDistancekBasedOnLength()* on algorithm 3.3. The denominator has two terms. The first term of the denominator $max(|A|, |B|)$ is largest possible edit distance costs between string *A* and *B*. The second term in the denominator: $\frac{1}{min(|A|,|B|)}$ is the cost calculated based on the length of the strings as shown in formula 3.1. Without the second term in the denominator, the normalized value would not be correct if the length of shorter string is 1. The first term of the formula 3.2 reverses the value evaluated in the second term. This causes the formula to have a higher score when there is fewer edit distance cost and lower score when there is higher edit distance cost.

In algorithm 3.3 the function *compareLocation()* uses Google Maps API to compare the location of the two users in terms of distance. For example if string *A* is ‘Ottawa’ and *B* is ‘Toronto’, even though they are syntactically different, both cities are within Ontario and are geographically close to each other. If the values of the *location* attribute are not similar enough (i.e. score is below framework's threshold τ). The framework will use Google API to derive the Province name of the locations. The algorithm will then compare the province names of the pairs of locations, if they are exactly the same, a specific score value is given. If province names are not matched, the country names of the locations are compared. If the country names are exactly the same a specific score (lower than province score) is returned, otherwise the original score produced by the edit distance algorithms in algorithm 3.3 is considered as the score. The function *compareLocation(A, B, Score)* in algorithm 3.3 is defined as follow:

$$Score' = \begin{cases} 0.8 & if\ A\ and\ B\ are\ within\ same\ Province \\ 0.7 & if\ A\ and\ B\ are\ within\ same\ Country \\ Score & otherwise \end{cases} \qquad (3.3)$$
$$Score' \in [0,1]$$

As formula 3.3 shows, assuming *A and B* are two locations, if the two locations *A* and *B* are within the same province a specific score of 0.80 is returned. However if the locations are not within the same province but are within the same country a lower score of 0.70 is returned. If the two locations are not within the same country either, the score obtained from the previous steps of algorithm 3.3 is considered as the new score. The above score values associated with identical province and country names are managed by framework's settings.

Weight Assignment

In the final step of algorithm 3.3, by using *assignWeight()* the score obtained from comparing two strings is combined with a weight value to evaluate the similarity score between a pair of attributes. The weight assignment places more importance on certain attributes than others. For example if for the attribute *name,* the weight is 0.85, then for the attribute *location* the weight is something lower such as 0.50. This is because the *name* is a more unique attribute than *location*. In other words the *name* attribute can more precisely identify an individual, compared to the person's *location*. On the other hand a weight higher than that of *name* is assigned to the *email* attribute because emails are more unique than names and locations. All weight values are real numbers between 0 and 1 inclusive. Assigning a weight to a score value is done using the following formula:

$$AttributeScore \ = Score \times w_i \ \in [0,1] \tag{3.4}$$

$$AttributeScore \ \in [0,1]$$

where w_i is the weight associated with a specific attribute i.

Values of weights can be set manually [26] based on relative uniqueness of the attribute values within a defined a context. They can also be automatically set by algorithms such as the one discussed in [30]. In this framework the weights associated with each attribute are empirically evaluated based on the ratio of number of unique attribute values to the total number of attribute values extracted from a sample size of 1200 Facebook profiles. In other words the weight associated with each profile attribute is computed based on the following formula 3.5:

$$Weight\ of\ Profile\ Attribute = \frac{Total\ number\ of\ unique\ profile\ attribute\ values}{Total\ number\ of\ profile\ attribute\ values} \tag{3.5}$$

For example the *Middle Name* attribute is 95% unique across the sample size of 1200 Facebook profiles. The following table 3.2 is the list of weights associated with accessible profile attributes.

Table 3.2 Profile Attributes and the weight value associated with them

Attribute name	Evaluated Weight ∈ [0,1]	Attribute name	Evaluated Weight ∈ [0,1]
Email	0.95	**Telephone**	0.95
First name	0.70	**Address**	0.95
Last name	0.85	**Website**	0.95
Middle name	0.95	**Occupation**	0.84
Location	0.11	**Education**	0.47
Birthday	0.67		

For inaccessible attributes such *email, address* and *phone number* an estimated weight based on the highest computed weight is given. Mathematical evaluations of attribute weights are available on Appendix A.2. The randomness of the dataset and the source of the dataset (in this case Facebook) have an impact on the correctness of the weights.

Algorithm 3.3 is applied to all accessible profile attributes of every pair of profiles. If either of the profiles do not have a specific attribute (e.g. unavailable or inaccessible by framework), then that pair of attributes will be omitted and will not be considered by the comparison algorithm 3.3. In the end, for each pair of profiles, the weighted average of all attribute scores evaluates to *Personal Identities Final Similarity Score* or **PIFS**.

3.2.3.2. Matching Social Identities

This section explains the methods used to analyze, categorize and compare *Social Identities*. *Social Identities* are entities that are shared among users in social media sites. As table 3.1 shows they include tweets, text posts, image posts, links and video posts.

> **Assumption:** Analyzing *Social Identities* assist the framework in evaluating a more accurate similarity score among profiles. User's behavior in terms of materials being shared reflects the mentality, mood and personality of the user. Moreover for certain social media sites such as Twitter, *Social Identities* are the primary source of data for identity resolution. Twitter profiles do not have as many *Personal Identities* as Linkedin, Google+ and other social media platforms.

The approach used to analyze social media posts is not the same as the methods utilized for analyzing *Personal Identities.* Even though Facebook posts and tweets are also text based and typically composed of meaningful words, it is not possible to use a syntactical string matching algorithm as the primary source of comparing similar posts. For example if user *X* posts on his Facebook profile that he likes to play 'soccer', and on his Twitter profile he tweets that he is interested in 'sports', even though syntactically the words are different, semantically they are similar as soccer is a subset of sports. To state it more formally conventional syntax based string matching algorithms will not work for the following reasons:

1. Posts and tweets across different social media sites are different in terms of grammar and words. String matching algorithms will fail at comparing the posts because they compare characters rather than semantics.
2. Pictures and videos are not texts and in many cases do not have a consistent caption associated with them

Therefore the approach adopted by the framework is as follow: For every pair of user profiles, the framework extracts and analyzes the users profile posts by categorizing them into a finite set of category names (topics). It then computes the overlap between the category names. Higher overlap in category names corresponds to higher similarity between the posts shared by the users. Higher similarity between the online posts results in higher similarity scores between profiles.

To be able to categorize the posts, the framework uses natural language processing techniques. However instead of integrating NLP libraries within the framework, it outsources the categorization process to two NLP web APIs (web services) namely AlchemyAPI[41] and OpenCalais[42]. For example a text post such as "I voted for obama in 2012 election because he is a better candidate" is sent to AlchemyAPI via the web. AlchemyAPI categorizes the text and responds with the terms 'Culture & Politics".

In the case of non-textual posts on Facebook such as videos, images or links, a different approach is taken. In the case of a video, if the video is from YouTube, by using the YouTube API, the category name of the video is extracted. A YouTube video shared on Facebook includes the terms "youtube.com" or "youtu.be" in its web address (i.e. URL); that is how posts are recognized to be from YouTube. In the case of links and images only the available caption association with the post is categorized. When categorizing the posts and YouTube videos, the obtained category names are from a finite set of names, which means that when comparing the categories, a simple string matching algorithm with Boolean response is sufficient.

To enhance the categorization result, the proposed framework utilizes two NLP web APIs: AlchemyAPI and OpenCalais as opposed to just one. Because in many cases the first NLP API does not return any category for a given text yet the second NLP API is able to detect the category. The comparison result of using one NLP API versus two NLP APIs is available in section 4.2.3.2. Table 3.3 displays the full list of category names from OpenCalais, AlchemyAPI and YouTube. As it shows, the category names are not the same across different APIs, therefore each obtained category named is only compared with other category names from the same API.

Table 3.3 Available category names (i.e. topic names) on NLP services and YouTube

Web service	Supported topics/categories extracted from each web service
OpenCalais[42]	Business_Finance, Disaster_Accident, Education, Entertainment_Culture, Environment, Health_Medical_Pharma, Hospitality_Recreation, Human Interest,Labor, Law_Crime ,Politics, Religion_Belief,Social Issues, Sports, Technology_Internet, Weather, War_Conflict, Other
AlchemyAPI[41]	Arts & Entertainment, Business, Computers & Internet, Culture & Politics, Gaming, Health, Law & Crime, Religion, Recreation, Science & Technology, Sports, Weather, Unkown
YouTube[37]	Comedy, Entertainment, Film & Animation, Gaming, Howto & Style, Nonprofits & Activism, People & Blogs, Pets & Animals, Science & Technology Sports, Travel & Events, Education, Music, News & Politics

In order to compare YouTube videos with textual posts, YouTube category names are mapped to corresponding AlchemyAPI category names. Algorithm 3.5 states the steps taken to compare a Facebook profile with a Twitter profile in terms of *Social Identities*. In this framework analysis of *Social Identities* is only applied between Facebook and Twitter profiles. This is because Twitter profiles rely heavily on tweets and only include too few *Personal Identities*.

Algorithm 3.5 Categorization algorithm for each pair of Facebook and Twitter profiles

```
Input:
X: Facebook profile,
T: Twitter profile
Output:
finalSimilarityScore: Decimal value between 0 and 1
Local Variables:
$p_i$: Online Post i for Facebook user X,
$t_i$: Tweet i for Twitter user T
begin
foreach $p_i$ in X.posts  do
    if $p_i$.type == text then
                 facebookTopicNames[i] = categorize($p_i$.content)
    else if $p_i$.type == image or $p_i$.type == link then
                 if ($p_i$.caption $\neq \emptyset$) then
```

```
                       facebookTopicNames[i]= categorize($p_i$.caption)
                 end
    else if $p_i$.type == YouTube then
                  youTubeTopicName = categorizeVideo($p_i$.url)
                  facebookTopicNames[i]= mapTopic(youTubeTopicName)
    end
end
foreach $t_i$ in T.posts  do
    if $t_i$.type == text then
                  twitterTopicNames [i] = categorize($t_i$.content)
    else if $t_i$.type == image or $t_i$.type == link then
                 if ($t_i$.caption $\neq \emptyset$) then
                      twitterTopicNames[i]= categorize($t_i$.caption)
                end
    else if $t_i$.type == YouTube then
                 youTubeTopicName = categorizeVideo($t_i$.url)
                 twitterTopicNames[i]= mapTopic(youTubeTopicName)
    end
end
commonCategories = countCommon(facebookTopicNames,twitterTopicNames)
score = normalizeCommon(commonCategories);
finalSimilarityScore = assignWeight(score, weight(“post”))
return finalSimilarityScore
end
```

As algorithm 3.5 shows, the first *foreach* loop is for collecting and categorizing Facebook posts. Once that's accomplished, the next *foreach* loop is responsible for analyzing and categorizing the most recent tweets of the Twitter profile (ex. last 10 tweets). The next step computes the number of overlapping category names using the function *countCommon.* This function evaluates the number of common category names by comparing each category name obtained from the Facebook posts, with every category name obtained from the Tweeter profile. The result of this function is stored in a variable called *commonCategories.*

In the next step the algorithm normalizes the quantity of common category names into a score value; the higher the number of common categories the higher the score will be. The function *normalizeCommon()* normalizes the score between 0 and 1. This function divides the number of computed common categories by the total possible number of common category names between a pair of profiles. The function *normalizeCommon()* is defined as follow:

$$Score = \\ = \frac{CommonCategories}{Extracted\ category\ names\ from\ Facebook\ profile\ \times Extracted\ category\ names\ from\ Tweeter\ profile} \tag{3.6}$$

Finally the function *assignWeight()* of algorithm 3.5 will give a certain weight value to the normalized number of common categories names. In this framework, for all types of posts a constant weight value of 1 is used. This means that all posts have the same importance. The resulting value will be the final similarity score between two profiles based on *Social Identities.* This final score is also known as *Social Identities Final Similarity Score* or **SIFS**. The number of tweets extracted from each Twitter profile as well as the number posts extracted from the Facebook profile is determined manually in the configuration of the framework. For example the framework can be configured to only extract the last 5 posts from each Facebook profile. As an example table 3.4 displays 5 posts that have been categorized by the NLP APIs and YouTube.

Table 3.4 Example of 5 Facebook posts and their corresponding topic names

Facebook Posts	**Category name obtained from OpenCalais**	**Category name obtained from AlchemyAPI**	**Category name obtained from YouTube**
"I do not like Romney, I will probably vote for Obama, he must be a better president."	Politics	Culture & Politics	-
A YouTube Video of Obama giving a speech. http://www.youtube.com/watch?v=ON2XWvyePH8	-	Culture & Politics	News & Politics
"Eating vegetables in the morning reduces the blood sugar!"	-	Health	-
"lets ski today"	-	Sport	-
"the next iphone is going to have a bigger screen"	-	Computers & Internet	-

3.2.3.3. Matching Relational Identities

The final class of information that are analyzed and compared are *Relational Identities.* RI are composed of Facebook friendships, Linkedin connections and Twitter followee and followers. The class of *Relational Identities* also includes group memberships and fan page participations.

However in the current state of this framework only Facebook friends and Linkedin connections are considered.

> **Assumption:** Analyzing *Relational Identities* assist the framework in evaluating a better similarity score among Facebook and Linkedin profiles. Given the Facebook and Linkedin user, this user must have at least one common friend among his Facebook and Linkedin networks.

Using this assumption it is possible to compare the network overlap (i.e. Facebook friends and Linkedin connections) of each pair of profiles, a larger overlap correlate to a higher similarity score between the profiles. To find commonality between profiles in terms of *Relation Identities*, the framework extracts user *X*'s Facebook friends in phase 1, and extract Linkedin connections of each retrieved profile in phase 2 of the framework. In phase 3 each pair of Facebook and Linkedin profile is compared in terms of common friends/connections. Algorithm 3.6 describes the steps taken to compare *Relational Identities* between a pair of Facebook and Linkedin profiles.

Algorithm 3.6 Finding common network among a pair of Facebook and Linkedin profiles

```
Input: X: Facebook profile,
L: Linkedin profile
Output: finalSimilarityScore: Decimal value between 0 and 1
begin
    foreach $u_i$ in X.friends  do
           facebookNetwork[i] = $u_i$
    end
    foreach $u_i$ in L.friends  do
          LinkedinNetwork [i] = $u_i$
    end
    commonNetwork =  FindCommon(facebookNetwork, LinkedinNetwork)
    score = normalizeCommon(commonNetwork)
    finalSimilarityScore = assignWeight(score, weight("network"))
    return finalSimilarityScore
end
```

The function *assignWeight()* of algorithm 3.6 will give a specific weight value to the calculated number of common friends/connections. In algorithm 3.6 the framework assigns a value of 1 as the weight. In other words the function *weight("network")* evaluates to 1. Based on algorithm

3.6 the *Relational Identities Final Similarity Score* or **RIFS** of each pair of profiles is obtained from the common friends/connections of Facebook user *X* and each extracted Linkedin profile. *Relational Identities* alone may not be considered a reliable factor in resolving the identity of a person, therefore in this framework both *Relational Identities* and *Personal Identities* are suggested for matching Facebook and Linkedin profiles. Figure 3.5 visualizes the meaning of common friends/connections between a Facebook and Linkedin user.

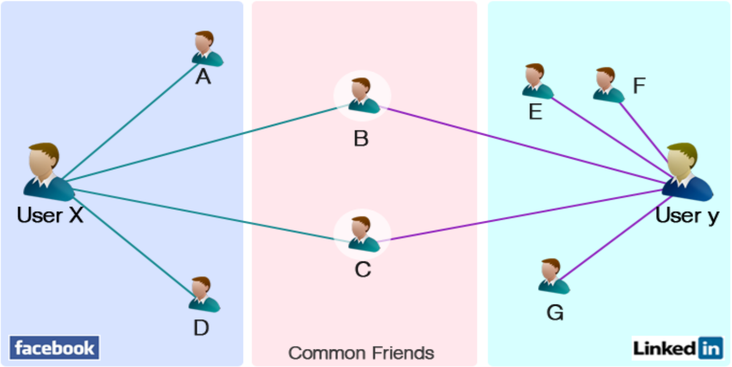


Figure 3.5 Similar friends among two social media platforms

Based on figure 3.5, Facebook user *X* has friends A, B, C, and D. User y who is one of the Linkedin profiles returned as part of the search result in phase 2 of the framework. User y has connections with user B, C, E, F and G. As the figure shows, the third phase of the framework is able to detect that users B and C are friends with user *X* and also connections to user y on Linkedin. As the number of common friends/connections increases the similarity score between user profile *X* and user y increases.

Computing the number of common friends/connections among two users *X* and y is done by the function *FindCommon* which is defined as:

$$FindCommon\left(N_X, N_y\right) = |\{(p_k, p_l) : p_k \equiv p_l, p_k \in N_X, p_l \in N_y\}| \quad (3.7)$$

Where (p_k, p_l) is a pair of profiles from Facebook and Linkedin; N_x is a network of friends of user *X* and N_y is the network of Linkedin connections of user y . $p_k \equiv p_l$ means that profiles p_k and p_l are sufficiently similar in terms of the *name* and *location* attributes.

In order to improve the accuracy of *Relational Identity* based comparison, next versions of the framework can also adopt a recursive process where the 3 classes of identities of each connection/friend of user *X* and user y are compared. In other words instead of solely relying on simple personal attributes, the framework can look at the online posts as well as second degree friends/connections.

3.2.3.4. Overall Process and Decision Making

The final step of the framework is to return the Linkedin and Twitter profiles that the framework considers to belong to the same Facebook user *X*. Depending on the type of social media sites and the amount of information available for extraction, there may be a combination of **PIFS**, **SIFS** and a **RIFS** values for each pair of candid profile and input user *X*. For each of the three final score, there are two methods of decision making. In both cases of decision making, manual review may be required.

The first method attempts to sort the candidate profiles based on the similarity scores where the first Q profiles with the highest similarity score are considered as matches. As the value of Q increases, the number of possible answers increases which allows for more potential profiles. Increasing the value of Q reduces the accuracy of the framework by letting in more profiles as match, but increases the number of responses and consequently increases the amount of manual work. A benefit of this method is the fact that in almost all cases, the framework returns at least one profile as match. In cases where there are no true matches, the returned result can be considered as profiles of users with identities *similar* to that of the input profile. This result can be used in various applications such as recommendation systems.

The second method of decision making is to return the profiles which posses a similarity score above a certain threshold τ. In some cases this method returns no answers at all because none of the profiles have a final score above τ. The value of τ can be determined by experiments or as the work of Raad *et al.* [30] describes it can be computed using the following formula:

$$\tau = f_{Aggregation}(w(I_0), w(I_1), \dots, w(I_n)) \tag{3.8}$$

Where

$\boldsymbol{\tau}$ is the profile matching threshold

$\boldsymbol{f_{Aggregation}}$ is the aggregation function used to produce a single value as the threshold

$\boldsymbol{I}$ is the attribute used. The attributes are from the list of attributes that are accessible from social media site and are used during profile comparison (ex. *name, location, occupation*). Attributes that are not available on both social media sites – such as username in the case of Linkedin – are not considered as input to this formula.

$\boldsymbol{w}$ is the function that return the weight value of the attribute (Weight calculations are discussed in section 3.2.3.1)

$\boldsymbol{n}$ is the number of available attributes

Each class of identities (i.e. PI, SI and RI) has its own threshold value. The aggregation function $\boldsymbol{f_{Aggregation}}$ can be a classical aggregation functions such as *Minimum*(min), *Maximum* (max) and *Average* (avg) or any other more complex data aggregation function. In this thesis all three of the above aggregation functions are used and their results are compared and shown in chapter 4.

3.3. Summary

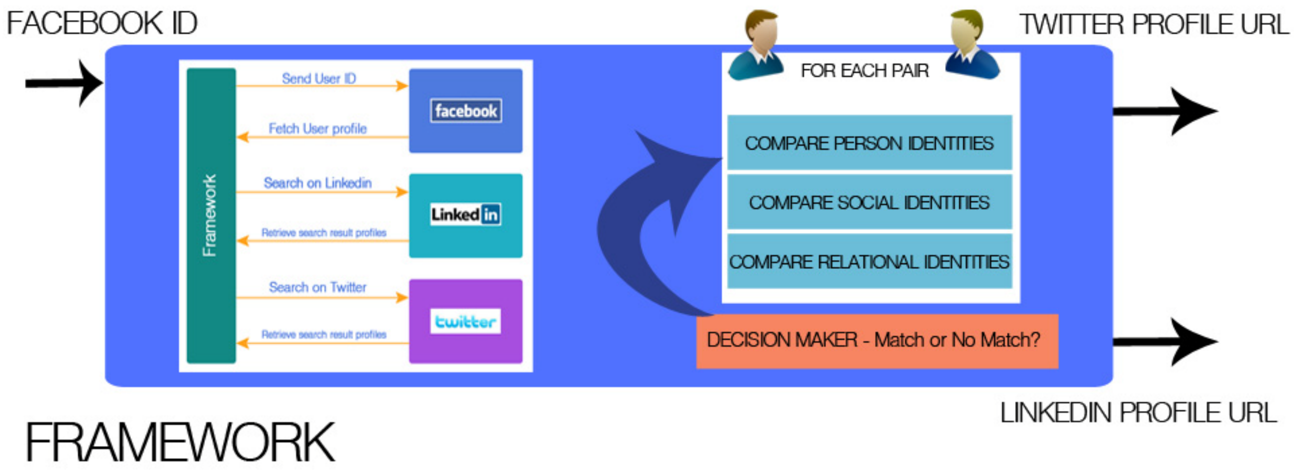

FACEBOOK ID
Framework
Send User ID
facebook
Fetch User profile
Search on Linkedin
Linked in
Retrieve search result profiles
Search on Twitter
twitter
Retrieve search result profiles
FOR EACH PAIR
COMPARE PERSON IDENTITIES
COMPARE SOCIAL IDENTITIES
COMPARE RELATIONAL IDENTITIES
DECISION MAKER - Match or No Match?
TWITTER PROFILE URL
LINKEDIN PROFILE URL
FRAMEWORK

Figure 3.6 Overall view of the proposed framework

This chapter proposed the overall framework that is divided into three phases (figure 3.6). The first phase extracts all available Facebook profile information for a particular user, named user *X*. In the second phase the framework performs a search on Linkedin and Twitter to obtain profiles similar to the reference Facebook User *X*. The third phase of the framework analyze, categorize and compares the profile data using NLP APIs, Google Maps API, string matching algorithms and weight assignment; Comparison of each Linkedin and Twitter profiles with the Facebook profile produces a similarity score. Profiles with a score above a certain pre-computed threshold are considered matches. The threshold value can be set manually or by using aggregation functions on weights associated with profile attributes.

CHAPTER 4

4. EXPERIMENTS, RESULTS AND DISCUSSION

This chapter includes the technical details of the proposed framework as well as the experiments performed to measure the performance and runtime of the framework.

Technical Details

The proposed framework is written in Java as a single process single thread application using Eclipse IDE (version Indigo). All tests are performed on a Sony Vaio – Intel Core i5 (M520) with clock speed of 2.4GHz with 8.00GB of RAM on Windows 7 Home Premium SP1 64bit. Internet connection speed at about 35Mbps/3Mbps.

The framework consists of 7000 lines of code, and relies on many underlying Java libraries. Internally the framework is composed of 3 main modules; these modules are composed of various Java classes. Figure 4.1 shows the three modules of the framework.

As figure 4.1 shows the three modules are named Social Media module, Central Module and Analysis and Comparison Modules. The Social Media module is composed of Facebook, Linkedin and Twitter SDKs (Software Development Kit) and libraries. The purpose of the SDKs is to streamline the communication between the framework and the API servers by providing simple to use Java classes and methods. The Analysis and Comparison module contain YouTube SDK, AlchemyAPI SDK, OpenCalais SDK, Google Maps SDK as well as string matching algorithms. String matching algorithms are written based on existing implementations and referenced papers. The Central Module is responsible for processing the input of the framework and providing output. It is also responsible for attribute weight assignments, evaluating the similarity scores and decision making. The Central module communicates with both Social Media Module and Analysis and Comparison Modules.

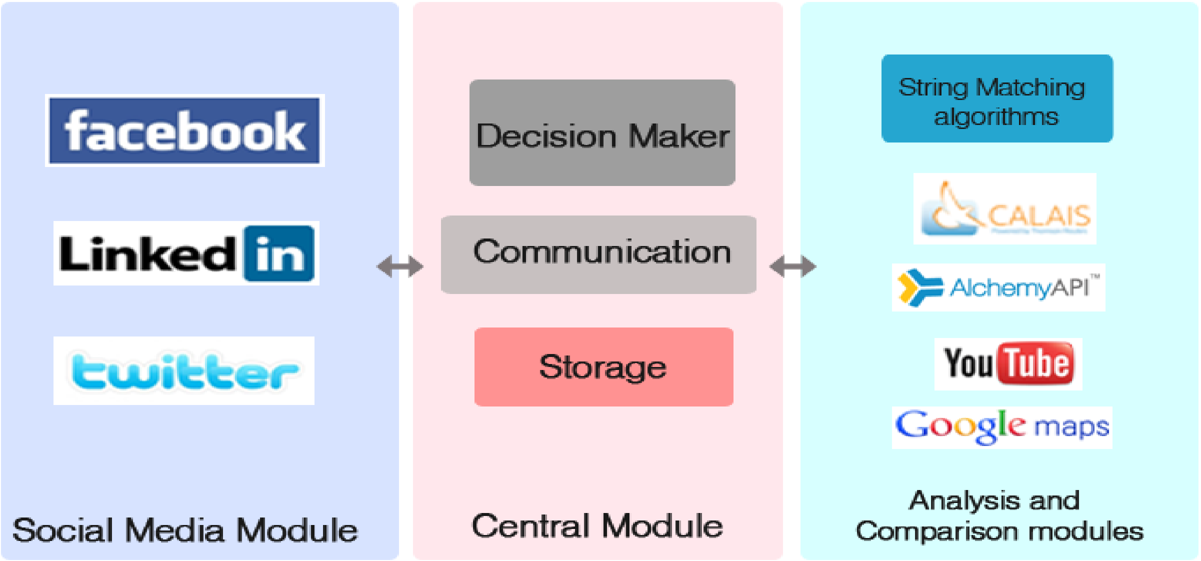


Figure 4.1 Overview of the framework's modules

The three modules perform the 3 phases of the framework as follow:

Phase 1: The central module begins by accepting a Facebook ID as input. As a user experience add-on, the framework can also accept a first name and a last name and perform a search on Facebook. It can then display the returned Facebook profiles to the user; the user then chooses the Facebook profile for which the identity matching will be performed.

Phase 2: The Central module is also responsible for sending the appropriate requests to the Social media module to extract the necessary information from each social media sites. The response retrieved from Facebook server in phase 1 and the responses retrieved from Twitter and Linkedin servers in this phase, are in JSON (Javascript Object Notation) or XML (Extensible markup language) format. The social media SDKs within the framework's Social Media module transforms the server response into Java objects. Each Java object consists of fields such as name, id, location and occupation. Each Java object also contains user's profile posts and user's online connections/friends. In other words each Java object contains all three classes of identities.

Phase 3: Once all the information is available to the Central module, they are passed to the

Analysis and Comparison modules to be analyzed and categorized. Upon the completion of categorization and comparisons, it is again the duty of the Central module to measure the similarity score of each pair of profiles and decide whether they are a match or not.

Since the framework is modular is it possible to add and remove new social media SDKs or NLP services without affecting the other parts of the framework. This feature allows for rapid growth and easier maintenances of the framework when upgrades are required.

In terms of memory consumption, the framework has the ability to store every extracted social media profile into MySQL database; or it can process and compare profile attributes while in memory (i.e. Java objects). Certain social media sites' privacy policies do not allow storage of other users profile information on client machines. The amount of memory required to execute the framework is dependent on the amount of space needed to store the profile of input user *X* (Facebook profile) and other extracted profiles (i.e. Linkedin and Twitter search results). By default the framework discards the profile information after evaluating the final similarity scores and displaying the result.

Scalability

In terms of scalability, the framework can be used in such a way that only *new* information such as new posts and videos are processed upon their retrieval. For example, once new posts (i.e. new *Social Identities*) are retrieved, only part of Algorithm 3.5 needs to be re-executed to perform categorization and comparison.

Mathematically the time complexity of comparing *Personal Identities* between a Facebook profile and *N* Linkedin profiles, with k common profile attributes is:

$$O(N \times k) \tag{4.1}$$

Where *N* is the number of profile search results; k is the number of *Personal Identity* attributes such as *name* and *location.* The time complexity of comparing a Facebook profile with *N* possible candidates based on *Social Identities* is as follow:

$$O(N \times l^2) \tag{4.2}$$

Where *N* is the number of Twitter profiles; *l* is the number of posts extracted from Facebook and Tweeter profiles. For every pair of profile each Facebook post is compared with each Tweeter tweet. If there are *l* posts/tweets on each profile, this creates a nested loop of $l \times l$. The time complexity of comparing a Facebook profile with *N* possible candidates based on *Relational Identities* is as follow:

$$O(N \times m^2) \tag{4.3}$$

Where *N* is the number of profile search results and *m* is the number of *Relational Identities* such as Facebook friends or Linkedin connections on every profile. Comparing m Facebook friend with *m* Linkedin connection creates a nested loop of $m \times m$.

Experiments

The experiments consist of running the framework with various framework settings and inputs and recording and analyzing the results of each execution of the framework. The following figure 4.2 displays the complete lists of different experiments performed to evaluate the performance of the framework. This chapter is divided into 3 sections. Section 4.1 displays the results of matching Linkedin profiles, and section 4.2 demonstrates the results for matching Twitter profiles. Section 4.3 also describes the observed performances differences between the NLP APIs used on the identity matching framework and discusses the challenges that were encountered in developing and testing this framework.

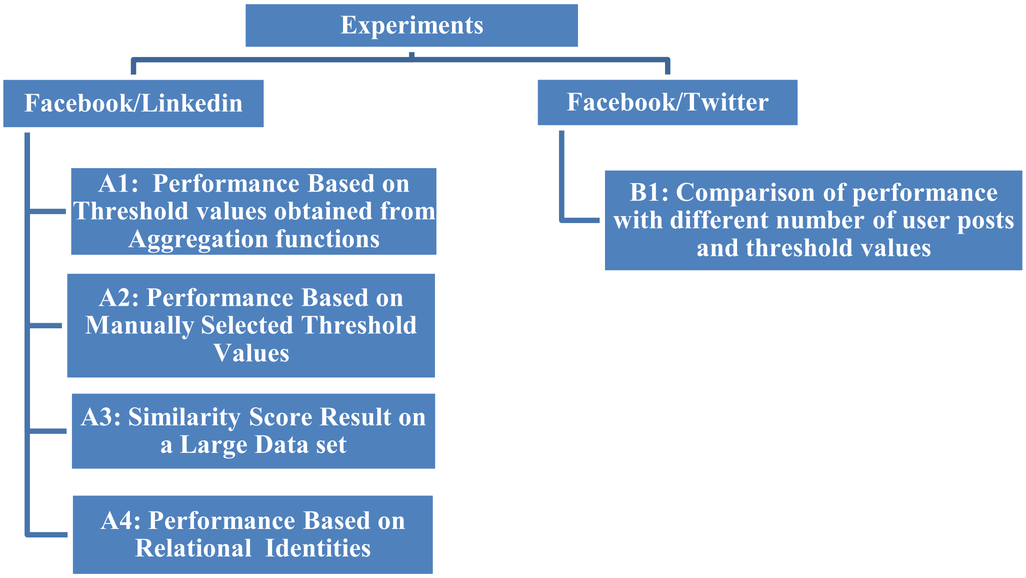

Experiments
Facebook/Linkedin
Facebook/Twitter
A1: Performance Based on Threshold values obtained from Aggregation functions
A2: Performance Based on Manually Selected Threshold Values
A3: Similarity Score Result on a Large Data set
A4: Performance Based on Relational Identities
B1: Comparison of performance with different number of user posts and threshold values

Figure 4.2 Hierarchical view of all experiments undertaken to measure the performance of the framework

4.1. Facebook and Linkedin Experiments

The flow of the framework execution is as follow. In the first step the Facebook ID of a user is given as input to the framework; the framework will then connect with Linkedin and extract similar profiles. The extracted Linkedin profiles are then analyzed and compared with the Facebook profile. For Facebook and Linkedin profile matching; only *Personal Identities* and *Relation Identities* are compared. However in the case of *Relational Identities,* only the Linkedin connections of users who have authorized the framework are accessed. Further details about privacy issues associated with Linkedin are available in the section 4.3. In the last step of the execution, Linkedin profiles with the similarity score above the threshold are returned as match.

4.1.1. Test bed

The test bed is composed of real Facebook and Linkedin profiles. This allows the framework to obtain more realistic results. For Facebook/Linkedin profile matching *three* separate data sets have been collected. The first data set is composed of 100 unique Facebook users, of which 77 have a confirmed Linkedin account. The remaining 23 Facebook users do not have a Linkedin profile. The first data set of 100 Facebook users is used to measure the performance of the framework in terms of finding correct matches and correct non-matches while focusing on *Personal Identities*. Experiments on sections 4.1.3.1 and 4.1.3.2 utilize the first data set as input.

The second data set is composed of 1200 unique Facebook profiles which may or may not have a corresponding Linkedin profile. This data set is used to evaluate the weights associated with the profile attributes (see section 3.2.3.1). This data set is also used to observe the distribution of the similarity scores. Experiments on section 4.1.3.3 utilize the second data set as input to find matching Linkedin profiles based on *Social Identities*.

The third data set is composed of 3 Facebook users who each have a Linkedin profiles as well. These users have authorized the framework so it has access to their Linkedin connections. This data set is used solely for verification of the framework in matching Facebook/Linkedin profiles based on *Relational Identities*.

4.1.2. Runtime

The total runtime of the framework depends on multiple factors including:

1. Facebook API call roundtrip latency
2. Linkedin API call roundtrip latency
3. Google Maps API call roundtrip latency
4. String comparison algorithm runtime

The amount of delays produced by the above factors is dependent on the number of Linkedin profile requested and processed. Table 4.1 shows the run-time for performing identity matching between Facebook and Linkedin with specified framework settings.

Table 4.1 Run-time of performing 100 executions of Facebook/Linkedin identity matching

Total number of framework executions – (each time with a different Facebook ID as input)	100
Number of profiles requested and processed from Linkedin on each execution	10
Average duration of each execution	**3121ms = 3.12 seconds**
Maximum duration of each execution	13177ms
Minimum duration of each execution	853ms

4.1.3. Performance

To formally evaluate the performance of the framework for matching Facebook and Linkedin profiles, certain information retrieval metrics are used. The precision and recall (i.e. sensitivity) of the framework, similar to the equations found on [30] are measured as follow:

$$Precision = \frac{\text{Number of correct Linkedin profile matches found by the Framework}}{\text{Total number of Linkedin profile matches found by the Framework}} \tag{4.4}$$

$$Recall = \frac{\text{Number of correct Linkedin profile matches found by the Framework}}{\text{Total number of correct Linkedin matches}} \tag{4.5}$$

The performance of the framework is also measured by the accuracy metric [2] which considers the rates of both correct mismatches and correct matches. Accuracy is defined as:

$$Accuracy = \frac{\text{Number of correct answers found by the Framework}}{\text{Total number of profiles}} \tag{4.6}$$

In other words precision represents the fraction of returned profiles that are actually correct matches. As precision value decreases, manual review of the result increases. Recall represents the fraction of correct answers that are returned by the framework. Accuracy is the percentage of correct answers returned by the framework. Precision, recall and accuracy can be rephrased as following:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{4.7}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{4.8}$$

$$Accuracy = \frac{True\ Positive + \ True\ Negative}{True\ Positive + True\ Negative + False\ Positive + False\ Negative} \tag{4.9}$$

Where *Positive* stands for profiles found by the framework as match and *Negative* stands for profiles found as mismatch by the framework. $True$ denotes actual correct matches and *False* denotes actual correct mismatches. Precision and recall in conjunction with accuracy demonstrate the performance of the framework in terms of finding match and mismatch profiles on Linkedin.

F-score [58] is a widely used measure in classification tasks that evaluate the performance by combining the precision and recall values into a score between 0 and 1. The traditional definition of F-score is a harmonic mean of precision and recall and is defined as follow:

$$F_1 = 2 \ \cdot \frac{Precision \ \cdot Recall}{Precision + Recall} \tag{4.10}$$

4.1.3.1. Performance Based on Threshold values obtained from Aggregation functions

As discussed in section 3.2.3.4 there are various aggregation functions that yield different threshold value for framework decision making. Profile pairs with a similarity score above the threshold value are considered matches. The following table 4.2 compare the resulting performance of using three different aggregation functions: *Minimum, Average* and *Maximum* to compute the threshold value τ. Figure 4.3 displays the performance of the aforementioned aggregation functions in one graph to better visualize their difference.

Table 4.2 Results of three experiments using Minimum, Average and Maximum as aggregation functions to produce the threshold values

Aggregation Function	Threshold Value	Precision	Recall	Accuracy	F-score
Minimum	0.11	0.18	**1.00**	0.19	0.31
Average	0.69	0.28	0.92	0.55	0.43
Maximum	**0.95**	**0.35**	0.85	**0.67**	**0.49**

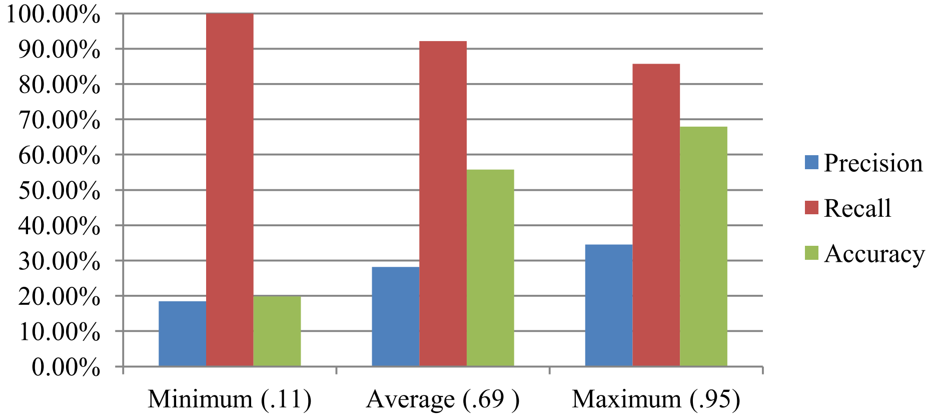


Figure 4.3 Recall, Precision and True Negative Rate of the framework using *Minimum*, *Average* and *Maximum* aggregation functions.

Based on the above results, the *Maximum* aggregation function computes a threshold τ that produces a recall value of 85% (ratio of correct matches returned) and a precision value of 35%. In comparison to *Minimum* and *Average* functions the *Maximum* aggregation function produces a moderate recall and highest f-score, precision and accuracy values. These set of experiments conclude that the framework produces the optimal results when the threshold value is the maximum value among the values obtained by the three aggregation functions.

4.1.3.2. Performance Based on Manually Selected Threshold Values

To evaluate the best value for the threshold, the framework is executed with different manually selected values of threshold τ; and the resulting recall and precision of each execution is calculated. The range of the threshold value is between 0 and 1 inclusive. This experiment focuses on 5 threshold τ values between 0.55 and 0.95, with the intervals of 0.10 points. These threshold values are labeled threshold 1 to 5. The following table 4.3 displays the performance of using these 5 different threshold values, in addition to the performance result of the previously computed threshold values in prior section 4.1.3.1.

Table 4.3 Performance of the framework with different threshold values; as shown the highest threshold values yields the best result

Method of Calculating the Threshold	**Threshold Value**	**Precision**	**Recall**	**Accuracy**	**F-score**
Minimum Aggregation function	0.11	0.18	**1.00**	0.20	0.31
Chosen Threshold value 1	0.55	0.23	0.99	0.41	0.37
Chosen Threshold value 2	0.65	0.28	0.95	0.54	0.42
Average Aggregation function	0.69	0.28	0.92	0.56	0.43
Chosen Threshold value 3	0.75	0.29	0.91	0.58	0.44
Chosen Threshold value 4	0.85	0.30	0.87	0.61	0.44
Maximum Aggregation function / Chosen Threshold value 5	**0.95**	**0.35**	0.85	**0.68**	**0.49**

The following figure 4.4 visualizes the result of the 3 computed threshold values using the aggregation function and the 5 manually chosen threshold values. As it is expected, as the value of threshold τ increases, the value of precision also increases but the value of recall decreases. Meaning that as the threshold is increased the framework returns less matching profiles, but more of the matching profiles returned are correct. In other words higher threshold value means more strict decision making which in turn causes the framework to output less result but higher ratio of correct results. In conclusion, based on the value of f-score and accuracy, the framework performs at the optimal level when the threshold value is set to 0.95. The complete list of values obtained from both sets of experiments is available on appendix B. In environments where it is admissible to manually check the frameworks results, it is more suitable to have a higher recall and lower precision value. However in places where manual labor is costly, it is better to have a reasonably lower recall but higher precision values.

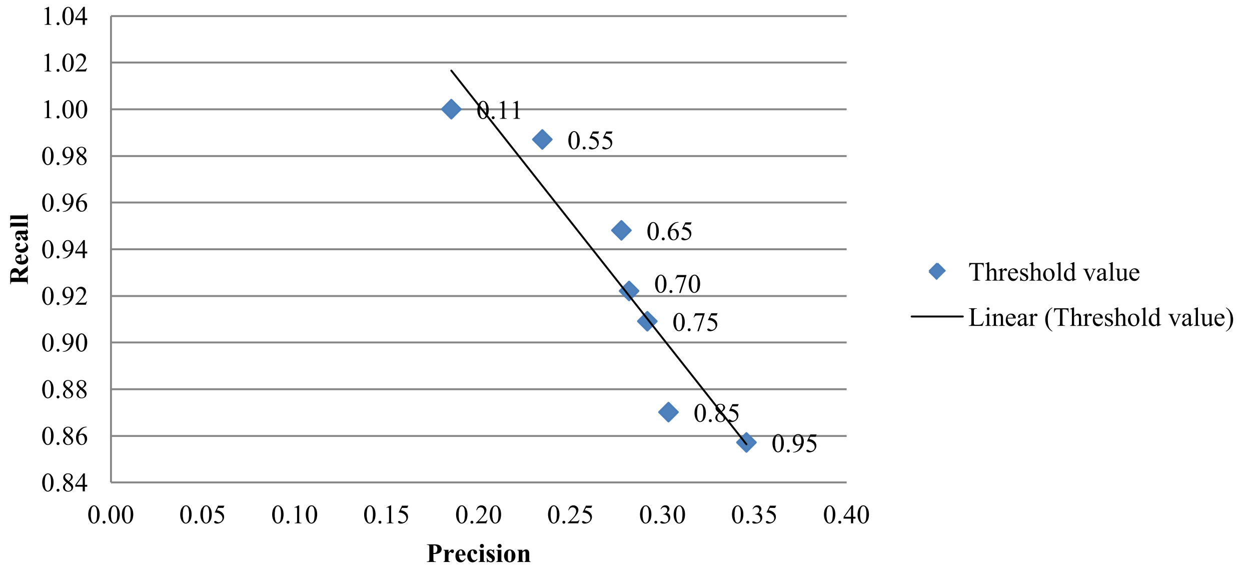


Figure 4.4 Graph of precision and recall of all Facebook/Linkedin threshold values. As the threshold increases, the precision value also increases while recall decreases.

4.1.3.3. Similarity Score Result of a Large Dataset

For this experiment, the distribution of similarity scores for 1200 unique Facebook profiles are observed. There is no assurance that these Facebook users actually have a Linkedin profile or whether the profiles returned by the framework as match are in fact correct matches.

On figure 4.5, the horizontal axis represents the individual Facebook profiles. The vertical axis represents the highest similarity score of matching Linkedin profiles and the horizontal axis represent different Facebook profiles.

Figure 4.5 shows all similarity scores above the average threshold value 0.69. As the graph shows we can make the conclusion that the highest density of scores is located above the optimal threshold of 0.95. In other words most of the Linkedin profiles have a similarity score above the optimal threshold value. This suggests there is a good possibility that majority of the Facebook profiles in this data set have a corresponding Linkedin profile.

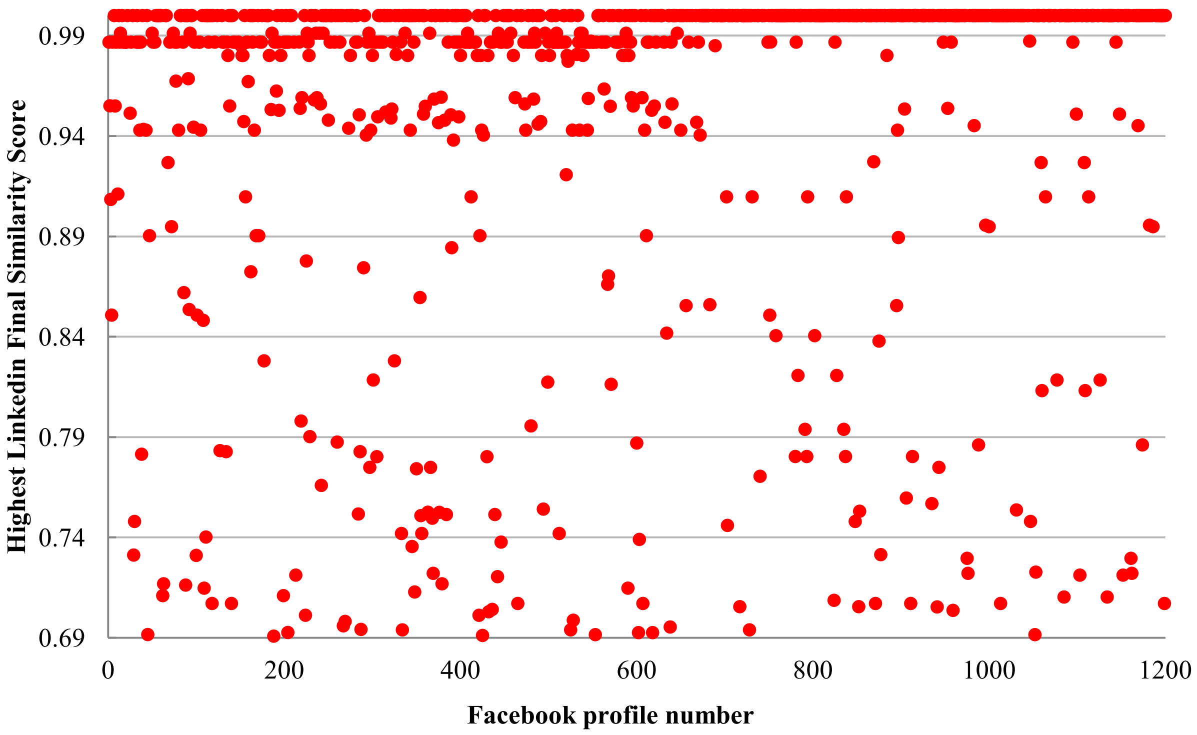


Figure 4.5 Highest Similarity score between 1200 Facebook profiles and Linkedin profiles

4.1.3.4. Similarity Score based on Relational Identities

To find matching Facebook/Linkedin profiles based on Relational Identities, on each execution of the framework the friends of the Facebook user *X* are compared with the connections of each potential Linkedin candidate y. Linkedin user y is returned by the phase two of the framework after performing a search on Linkedin. For every of pair of Facebook friend and Linkedin connection, the framework compares only the *name* attribute. If the number of matching friends/connections (i.e. size of overlapping networks) is above the threshold the framework considers Facebook *X* and Linkedin user y the same user.   In order to access friends/connections of users the framework must have the proper permissions. Unfortunately Linkedin privacy policy only allows *authorized* third party applications to access users' connections. This means that out of entire data sets that are used to verify the framework, only a small selection of users who have authorized the framework can be considered for this part of the experiment. Therefore the following experiments use 3 Facebook users who also have Linkedin profiles and have explicitly authorized the framework. These 3 users have specifically authorized the framework to have access to their network of connections. The following figure 4.6 compares the performance of the framework by using two different manually set threshold values 0.1 and 0.2 to find matching profiles. The number of common friends/connections for each pair of profiles is compared with the threshold value. During both experiments, a maximum of 100 friends/connections are extracted from each profile.

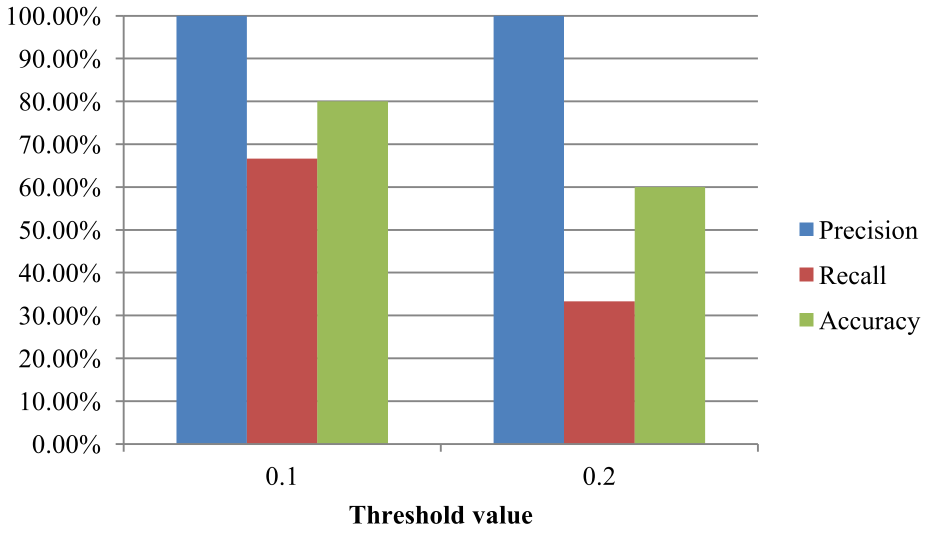


Figure 4.6 Performance results of finding matching Facebook/Linkedin profiles based on Relational Identities using two different threshold values

As figure 4.6 shows the precision value stays at 100% for the first and second experiment, this is because out of all the profiles found as match by the framework, all of them are actually correct matches. As the threshold value increases accuracy decreases because less correct matches are returned. The reason precious value stays at 100% for both first and second experiment is due to the privacy policies of the Linkedin website. Since only a small set of users have authorized the framework, the framework can only obtain connections of this set of users. This means that the framework can only compute the similarity score of users that have authorized the framework. This is why there are no false positive answers which make the precious value 100%. Appendix B.3 displays the numeric values obtained from this set of experiments.

4.2. Facebook and Twitter Experiments

In these experiments, similar to Facebook/Linkedin experiment, the Facebook ID is given to the framework as the input. In the case of Facebook and Twitter profile matching, only *Social Identities* of two profiles are compared, this is due to the limited number of profiles attributes (*Personal Identities)* available on Twitter profiles. Twitter profiles with the similarity score above the threshold are returned as match.

4.2.1. Test bed

The test bed for this experiment is comprised of 40 unique Facebook profiles/fan pages for which their owner also have a Twitter account. Out of the 40 Facebook profiles, most of them belong to famous celebrities. The Facebook profiles and Twitter profiles of celebrities are generally more open and easier to access than average Facebook profiles which are usually more private. Furthermore it is easier to find correct Facebook and Twitter profiles of celebrities than common Facebook/Twitter user. Twitter has a feature that allows celebrities to verify their Twitter profiles. Through this method its easy find the official Twitter profiles of celebrities. Facebook has recently introduced a similar feature. This feature is exercised to ensure that Twitter and Facebook profiles used as data set in the experiment are the true and official profiles of the celebrities, and not profiles of other individuals with the same name or interest.

4.2.2. Runtime

The runtime of the framework for finding matching Twitter profiles depends on multiple factors:

1. Facebook API call roundtrip latency

2. Twitter API call roundtrip latency
3. String comparison algorithm runtime
4. NLP web API call roundtrip latencies
5. YouTube API call roundtrip latency

The amount of delays produced by the above factors is dependent by the following framework settings:

1. Number of Twitter profiles to request and process
2. Number of Facebook posts to fetch
3. Number of Twitter posts to fetch

Table 4.4 shows the runtime for performing identity matching between Facebook and Twitter with the specified framework settings.

Table 4.4 Run-time of performing 40 executions of Facebook/Twitter identity matching

Total Number of executions – (each time with a different Facebook ID as input)	40
Number of profiles requested to process from Twitter on each execution	10
Number of Tweets fetched from each Twitter profile	20
Number of Facebook posts fetched from each profile	20
Average duration of each execution	**62136ms ≈ 1.04m**
Minimum duration of each execution	1012ms ≈ 0.016m
Maximum duration of each execution	291282ms ≈ 4.85m
Average duration of one API call to AlchemyAPI and back	135ms (Sums to ~15% of total execution time for 72 API calls)
Average duration of one API call to OpenCalais and back	464ms (Sums to ~54% of total execution time for 72 API calls)

4.2.3. Performance

To evaluate the performance of Facebook/Twitter identity matching framework, a series of experiments with different framework settings are performed. The focus of the following experiments is on one major framework setting which determines the number of extracted posts from each Facebook and Twitter profile. More specifically the following experiments observe the effect of change caused by the quantity of extracted profile posts.

In these experiments the precision and recall of the framework is computed similar to the previous section 4.1.3 and based on [30] as follow:

$$Precision = \frac{\text{Number of correct Twitter profile matches found by the Framework}}{\text{Total number of Twitter profile matches found by the Framework}} \tag{4.11}$$

$$Recall = \frac{\text{Number of correct Twitter profile matches found by the Framework}}{\text{Total number of correct Twitter matches}} \tag{4.12}$$

Similar to section 4.1.3 the performance of the framework is also measured by the accuracy metric [2] which considers the rates of both correct mismatches and correct matches. Accuracy is defined as:

$$Accuracy = \frac{\text{Number of correct answers found by the Framework}}{\text{Total number of profiles}} \tag{4.13}$$

As stated before precision represents the fraction of returned profiles that are actually correct answers. As precision value decreases, manual review work of the answers increases. Recall represents the fraction of correct answers that are returned by the framework. Precision, recall and accuracy can be rephrased as follow:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{4.14}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{4.15}$$

$$Accuracy = \frac{True\ Positive +\ True\ Negative}{True\ Positive + True\ Negative + False\ Positive + False\ Negative} \tag{4.16}$$

Where *Positive* stands for profiles found by the framework as match and *Negative* stands for profiles found as mismatch by the framework. *True* denotes actual correct matches and *False* denotes actual correct mismatches. Precision and recall in conjunction with accuracy can demonstrate the performance of the framework in terms of finding matching profiles on Twitter.

Similar to section 4.1.3, F-score [58] is used to measure the performance of the framework by combining the precision and recall values into a score between 0 and 1.

4.2.3.1. Performance with different number of user posts and threshold values

Tables 4.5 and figure 4.7 show performance values of the framework with different number of extracted user posts with a fixed threshold value of 0.1. Appendix C displays the complete list of values obtained from these experiments. As shown on figure 4.7 and table 4.5, during the three experiments as the number of extracted user posts increase, the value of recall increases from 50% to 65%, but the value of precision decrease from 11% to 10%. The best F-score is obtained when only 10 posts are obtained from each profile and in contrast, the best accuracy is obtained when there are only 5 post extracted from each profile. The increase in the number posts increases the number of correct matches returned (true positive), but it also increases the number of false positives (wrongly matched by framework). Based on the above experiments and the low value of precision, it is suggested that the result obtained require manual review to filter out false positive answers.

Table 4.5 Recall and precision values of the framework with different number of extracted posts

Threshold Value	Number of Posts Extracted from each profile	Precision	Recall	Accuracy	F-score
0.1	5	**0.11**	0.50	**0.53**	0.18
0.1	10	**0.11**	**0.65**	0.44	**0.19**
0.1	20	0.10	**0.65**	0.38	0.17

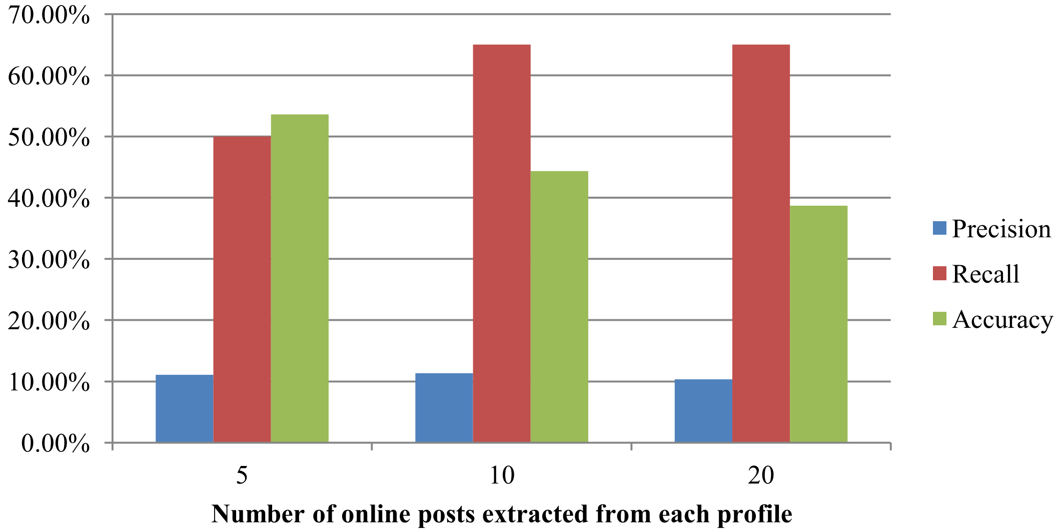


Figure 4.7 Framework's Facebook/Twitter profile matching performance with different number of posts, using a constant threshold value of 0.1

To analyze the effect of different threshold values on the performance of the framework, a series of experiments with manually chosen threshold values from 0.1 to 0.5 with intervals of 0.1 are undertaken. For the following experiments only 5 posts are extracted from each profile. Table 4.6 shows the results obtained from different threshold values.

Table 4.6 Performance of the framework in finding matching Twitter profiles while using different threshold values

Threshold Value	Precision	Recall	Accuracy	F-score
0.1	0.11	**0.50**	0.53	**0.18**
0.2	0.11	0.37	0.65	**0.18**
0.3	0.13	0.27	0.74	**0.18**
0.4	**0.14**	0.20	0.80	0.17
0.5	0.12	0.12	**0.81**	0.12

As table 4.6 shows the highest precision is obtained when the threshold value is set to 0.4. The highest recall is achieved when the threshold is set to 0.1 and the highest accuracy is achieved when threshold is set to 0.5. Based on the obtained value and pattern it is apparent that as the threshold value increases the accuracy of the framework increases but the recall value decreases. There is no single threshold value that produces the highest value for all metrics. The following figure 4.8 displays the obtained precious and recall measures for each threshold value. As figure 4.8 shows, as the threshold value increases, with the exception of threshold 0.5, the precision also increases, however recall decreases.

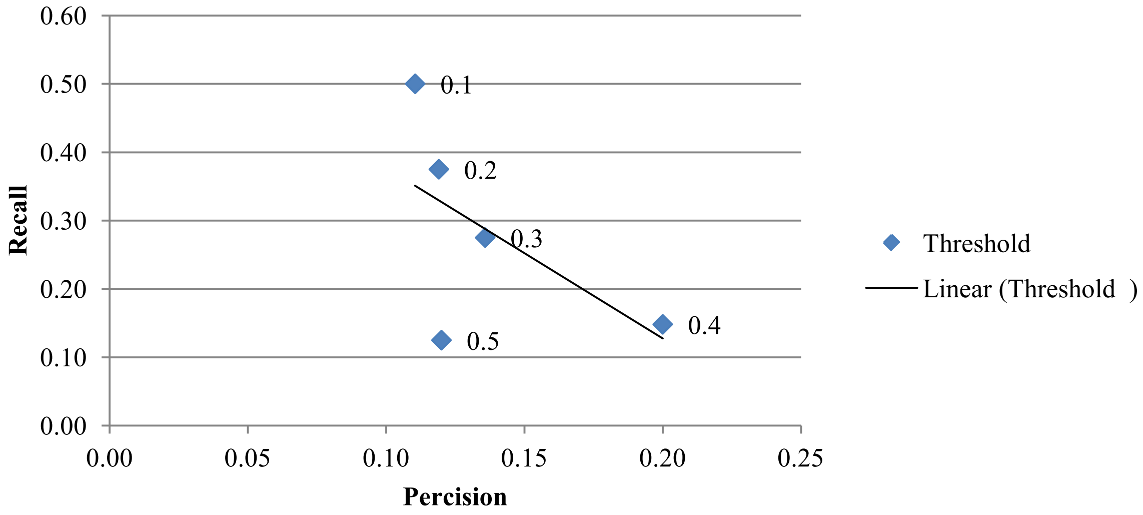


Figure 4.8 Graph of precision and recall of all Facebook/Twitter threshold values. As the threshold increases, the precision value also increases while recall decreases.

Twitter in contrast to Facebook and Linkedin does not offer adequate number of *Personal Identity* attributes and that's the focus in *Social Identities*. The result of Facebook/Twitter profile matching yields much lower recall and precision values compared to Facebook/Linkedin results, which leads to the conclusion that recognizing and comparing users simply based on categorized posts may not be as accurate as comparing profile attributes such as *name, location* and occupation. Relying on the posts alone may not be sufficient enough to perform accurate identity matching however comparing *Social Identities* in combination with *Personal Identities* should assist in reducing the number of false positive and false negative matches.

It should be mentioned that due to the nature of tweets and Facebook posts it is not always easy to categorize the texts properly. This issue is the result of many reasons including the fact the online messages are usually short in length and do not posses enough content to be semantically analyzed. To extend the length of online posts Abel *et al.* [47] enriches the tweets by associating news articles to them. For example if a tweet contains a URL Abel *et al.* will expand and analyze the content of the URL and associate its result to the user who published the tweet. If there are no URLs in the tweet Abel *et al.* extract named entities such as names, locations and hash-tags and search for news articles associated with the extracted entities. Another reason that online posts are not easy to analyze is because they sometimes contain smiley faces and abbreviation and typos that are not in the language dictionary. Han and Baldwin [59] propose a framework that converts abbreviation and typos found in tweets to their standard form. It should also be mentioned that online posts are time and context dependent which means that the true meaning of the post depends on where and when the user made the post and the identity of user's online friends/connections.

To improve the Facebook/Twitter identity matching framework the future version of the framework can have the following changes:

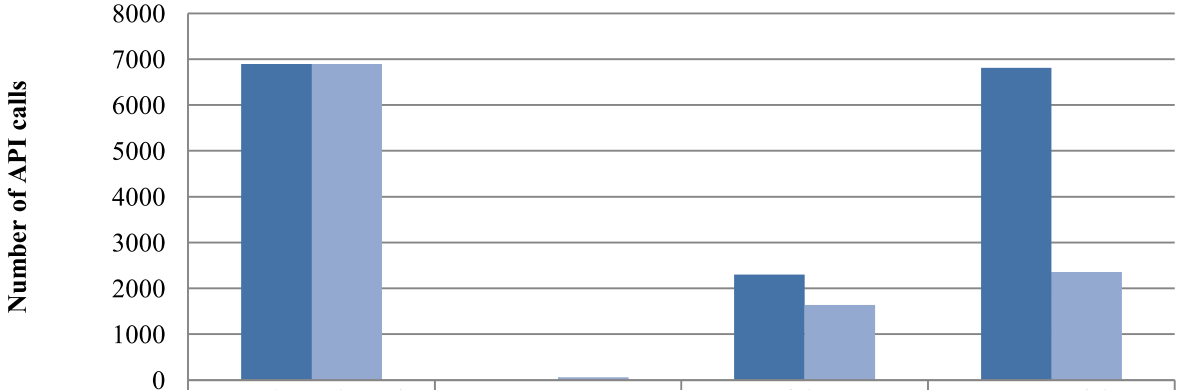
- Incorporating Twitter *Personal Identity* attributes while performing profile comparison.
- Incorporating semantic analysis methods that are specifically developed for the purpose of understanding social media streams which can also process non-English posts.

- Online posts should be further analyzed not solely through categorization but also based on named entities and hash tags and URLs embedded within the posts

4.3. Discussion

4.3.1. Comparison of NLP APIs during Facebook/Twitter experiments

AlchemyAPI shows a superior performance when compared to OpenCalais. This is based on thousands of API calls made during the Facebook/Twitter experiment while extracting 20 posts from each profile. As figure 4.9 shows, in the final Facebook/Twitter identity matching experiment where 20 posts were extracted from each profile, there were 6898 API calls to AlchemyAPI and OpenCalais servers. Out 6898 different user posts, AlchemyAPI was able to detect the language of the texts 98% of the times whereas OpenCalais was only able to detect the language 34% of the time. Moreover AlchemyAPI and OpenCalais were only able to categorize about half of the total number of posts. AlchemyAPI returned a category name for 33% of the total user posts which is 10% better than OpenCalais's results. If the category or language of the input text is not known, the NLP APIs return the string 'unknown' or 'other'. It is also observed that NLP APIs are able to categorize a larger number of Facebook posts than Twitter posts; this may most likely be due to limited length of tweets described in the previous section. Furthermore in this experiment OpenCalais server was not responsive to many of the API calls. In other words HTTP calls to the OpenCalais server were timed out 60 times out of the 6898 API calls.   Figure 4.9 also displays the number of server timeouts.



	Total Number of API calls	API timeout	Successful category detection	Successful language detection
■ AlchemyAPI	6898	0	2304	6811
■ OpenCalais	6898	60	1642	2359

Figure 4.9 Performance of AlchemyAPI versus OpenCalais API, while extracting 20 posts from each profile

This framework relies on both AlchemyAPI and OpenCalais to categorize users' posts. Based on the experiments, use of two NLP API instead of one NLP API improves the number of categorized texts. In the third Facebook/Twitter experiment where 20 posts were extracted from each profile, out of 6898 user posts/tweets, AlchemyAPI alone was able to categorize 2304 of them, but with the help of OpenCalais, total number of categorized posts reached to 3049 which is an improvement of 10%. Figure 4.10 breaks down the number of categorized user posts by both NLP APIs. As figure 4.10 shows out of 6898 posts, 3849 were uncategorized, 1407 were categorized by AlchemyAPI only and 745 by OpenCalais alone, and 897 were categorized by both AlchemyAPI and OpenCalais.

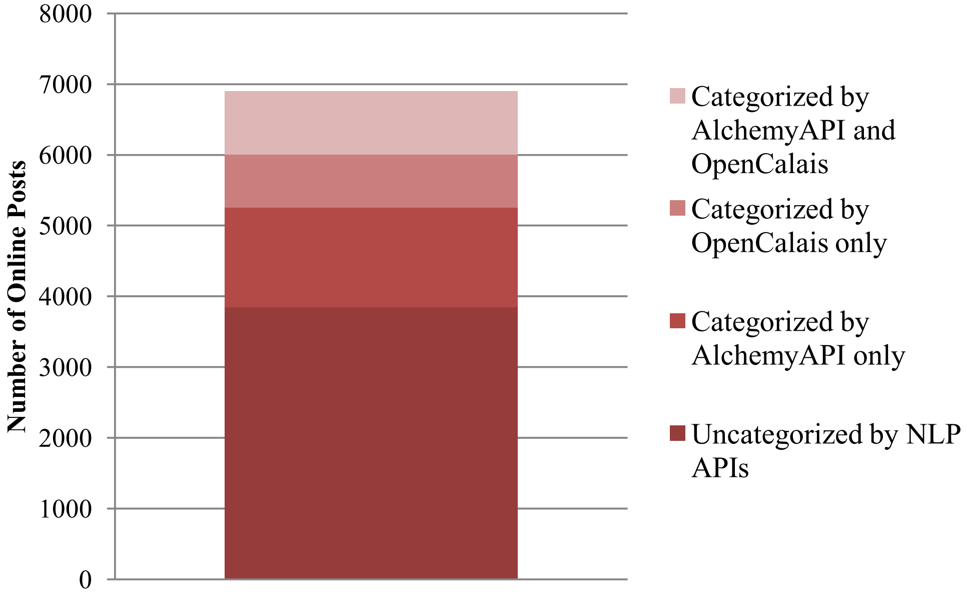


Figure 4.10 Ratio of categorized posts by NLP APIs

4.3.2. Data Availability, Permission and Privacy Issues

Creation of a practical framework that performs identity matching on real social media profiles has its own set of unique challenges, some of which are discussed in this section. The goal of this thesis is to provide a practical solution to identity matching among social media sites. Therefore real data from Facebook, Linkedin and Twitter are extracted and fed to the framework. Unlike artificial test beds, all testing data are from existing profiles extracted by our framework. Gaining access to social media information requires two components namely: application registration and user authorization, both of which are described in the following sections.

4.3.2.1. Framework's Social Media Applications

As figure 4.1 shows, the Social Media module is composed of Facebook, Linkedin and Twitter SDKs and libraries, each of which communicates with the corresponding social media site.

The job of the SDKs is to relay for the information requested by the framework to the social media sites, and direct the responses back to the framework. In order for the social media SDKs to communicate and retrieve profile information, they have to be registered with the social media site. Each SDK is registered as a *third party social media application*. Each application has a unique *access key* which identifies the application to the corresponding social media site. All communications from the SDKs to the social media server must accompany the *access key*. The applications are subject to the policies and terms of use provided by the social media sites: Facebook's platform policies [60], Linkedin's policies [61] and Twitter policies [62].

4.3.2.2. Social Media User Authorization

The three social media applications (i.e. Facebook Application, Linkedin Application and Twitter application) included in the framework must be authorized (i.e. approved) by at least one user of the corresponding social media site. Once authorized by a user, the information of that user and his/her network are available to the application. For the experiments performed in this thesis the framework is authorized by 3 Facebook and 3 Linkedin users. The framework is also authorized by one Twitter user. As more users authorize the framework's applications, more data becomes accessible to the framework, which leads to better identity matching result. In order to obtain more user authorizations, the applications can be bundled with popular social media applications and games, provided that users have full knowledge of the applications' privacy policies.

The authorization of the applications by the users is done through a protocol called OAuth which is explained on RFC5849 [63] and RFC6749 [64]. During the authorization process the application can ask for specific extended permissions that allow the application to access certain profile information that are otherwise inaccessible. Without asking for extended permissions, the application is only allowed to access basic profile information (such as name and photo) of the authorizing user.

4.3.2.3. Challenges Unique to Social Media sites

Each social media site gives certain permissions and API call limits to its applications. The level of data accessibility given to third party applications is sometimes lower than the level of accessibility given to a browser to fetch the social media sites. In other words some profile attributes may be viewable on the browser but not accessible by the APIs. For example in the case of Linkedin, the application can only retrieve the details of first level connections of the user who has authorized the application. However on the website, any Linkedin user can browse and view details of a user’s first degree connections. This is why the proposed framework is not able to retrieve *Relational Identities* (connections and group memberships) of every Linkedin user but only users who have authorized the framework’s application. Moreover Linkedin applications can only retrieve the basic profile information and current employment data of the first degree connections but the Linkedin website displays profile details beyond the basic information. Figure 4.11 displays the availability of Facebook and Linkedin profile attributes to our framework. In addition, generally social media sites place a API call throttle limit for third party applications. For example Linkedin allows 100 Search API calls per day per user.

Another challenge faced by the proposed framework is due to the evolving nature of the data source. Periodically social media sites introduce new profile attributes or remove certain profile attributes. Moreover these sites time to time modify the access permission of attributes based on users’ feedbacks or business decisions. For example the *Education* profile attribute on Linkedin profiles is not longer openly accessible by third party applications. In order to access this profile attribute the owner of the profile must explicitly authorize the application.

In general the availability of the data to the three social media applications is dependent on the following list:

1. Privacy policies and API call limits set by the social media sites for the applications
2. The specific privacy settings set by the users which include:
    - Whether the user’s profile should appear on search results
    - Whether the user has authorized the application
    - Whether any of friends/connections of the user have authorized the application
    - Which profile attributes can be retrieved by the applications (i.e. *name*, *location*)
    - Whether shared media can be extracted by applications (i.e. posts and videos)

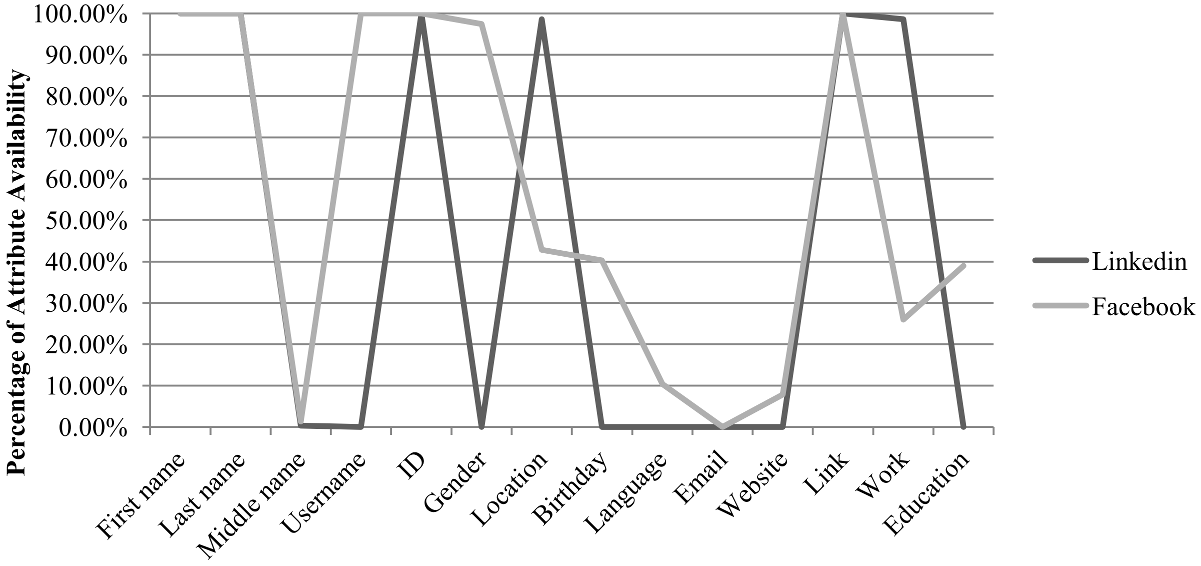


Figure 4.11 Availability of Facebook and Linkedin profile attributes to the framework (in percentage)

4.4. Summary

This chapter started by discussing the implementation structure of the framework and the testing environment. It also explained the two sets of experiments performed to measure the performance of the framework. In the first set of experiments, the performance of the framework in terms of matching Facebook and Linkedin profiles is measured. The data set for this set of experiments consists of profiles of 100 Facebook users out of which 77 also have a confirmed Linkedin profile; and 1200 Facebook users who may or may not have a Linkedin profile. In the case of finding matching Linkedin profiles based on *Relational Identities* a data set composed of 3 authorized Linkedin users is used. The second set of experiments focused on matching Facebook and Twitter profiles. The data set used for the second set of experiments is composed of profiles of 40 Facebook users who also have a a Twitter account. The performance of both sets of experiments is measured based on recall, precision, accuracy. This chapter also includes the comparison of the two NLP APIs (AlchemyAPI and OpenCalais) used in Facebook and Twitter profile matching experiments. In the end this chapter discusses the privacy and permission protocols associated with accessing social media information and their inherited challenges.

CHAPTER 5

5. CONCLUSIONS, SUMMARY AND FUTURE WORK

The purpose of this thesis is to tackle the crucial challenge of identity matching by providing new methods of processing and understanding user profiles. The ever increasing existence of social media website means that an average user has multiple social media profiles throughout the web. With the assistance of proposed identity matching frameworks it is possible to automatically detect and match different profiles of the same user across the web. Identity matching is essential in many intelligence and security applications as well as customer and employee management tasks. This chapter concludes the thesis by reviewing the proposed framework in section 5.1, stating the contributions made in section 5.2 and presenting future work in section 5.3.

5.1. Overview of Proposed Framework

The lack of a universal and unique identifier across different social media profiles makes identity matching a non-trivial job. This thesis proposes a framework that performs inter-social network analysis and comparison. The framework scans all available profile information from the users' Facebook, Linkedin and Twitter accounts and divides them into three classes of *Personal Identities*, *Social Identities* and *Relational Identities*. The framework compares and evaluates the similarity score of each pair of profiles in terms of the three classes of identities. For *Personal Identities* which include profile attributes such as *name* and *location*, edit distance string matching algorithm and Google Maps API is used. For *Social Identities* including textual posts, image and YouTube posts the framework uses third party APIs namely AlchemyAPI, OpenCalais and YouTube API to categorize and compare. Finally for *Relational Identities* which includes users' Facebook friends and Linkedin connections the framework will evaluate the network overlap between each pair of profiles. Comparing each pair of profiles based on each class of identities results in a similarity score. Profile pairs with the similarity score above a specific threshold are considered a match. The threshold value can be set manually or evaluated using weight aggregation formulas.

5.2. Contributions

This proposed framework performs automatic search and semi-automatic identity matching among three popular social media sites: Facebook, Linkedin and Twitter. This framework divides the social media profile information into classes of *Personal Identities*, *Social Identities* and *Relational Identities*.

In matching Facebook and Linkedin profiles by comparing *Personal Identities* the proposed framework achieves 85% recall and 35% precision values while using the highest threshold value of 0.95. See Table 4.3 for more details.

In terms of finding matching Facebook and Linkedin profiles based on *Relational Identities* the framework uses a small data set of Linkedin users who have authorized the framework. In these experiments the framework achieves 100% precision, 67% recall and 80% accuracy while using 0.1 as the threshold value (Figure 4.6). As the threshold value increases accuracy and recall values decrease.

In the case of matching Facebook with Twitter profiles based on *Social Identities* the proposed framework yields 10% precision and 65% recalls when extracting 20 posts from each profile while setting the threshold value to 0.1 (Table 4.6). The framework produces the highest accuracy of 53% and f-score of %18 when extracting 5 posts from each profile and maintaining the same threshold value of 0.1.

Based on the performed experiments comparing *Personal Identities* in comparison to *Social Identities* produces a higher accuracy for identity matching. On the other hand comparing *Relational Identities* produces a higher accuracy than the values obtained from comparing *Personal Identities* however the data set used for comparing *Relational Identities* is not large enough to make a conclusive comparison.

While performing the experiments the performance of AlchemyAPI and OpenCalais in terms of text categorization and text language detection is also compared. In addition to this, during the experiments the percentage of profile attribute availability from Facebook and Linkedin social media sites is calculated and compared.

A live demo of the proposed identity matching framework is available at Distributed Systems and Multimedia Processing (DSMP) research lab at http://dsmp.ryerson.ca/projects/

In summary the list of important contributions is as follows:

- The proposed framework collects all available information including profile attributes, user posts and online relationships. It then uses syntactic and semantic methods to compare each pair of profiles.
- This thesis demonstrates the performance comparison of identity matching among Facebook and LinkedIn versus Facebook and Twitter using *Personal Identities* and *Social Identities* respectively.
- The proposed framework shows the importance of using Personal, Social and Relational profile information for performing identity matching.

5.3. Future Works

The field of identity matching is dependent on various underlying concepts including string matching algorithms, natural language processing, data scalability and efficiency methods, privacy and security procedures. Improvement upon each of these concepts enhances the performance and practicality of the identity matching framework. In the words of Charles Eames "Eventually everything connects, people, ideas, objects.. the quality of the connections is the key to quality per se.". The current framework focuses on Facebook, Twitter and Linkedin, subsequent version of this framework can focus on other social media sites such as MySpace and Google+. The future version of the framework can be extended to incorporate machine learning approaches to evaluate the attribute weights and framework's threshold. In general this thesis can be extended in the following directions.

Search Methods

Improvement on the methods of performing search on social media sites can assist in retrieving more similar and narrowed down user profiles. For example besides *first-name* and *last-name* attributes such as *location* and *occupation* can also be used during the search phase. Moreover the framework can be extended to search not just based on profile attributes but also based on social posts. Sometimes users post the same material on multiple social media sites (ex. Facebook and Twitter). They either manually post the same text multiple times or use a third party application such as TwitterFeed [65] to sync their posts. The framework can use this trend to search for a particular post on Twitter which was also posted on Facebook. The result of the search is Twitter profiles of users who have shared the same post [2].

Improved Syntactic and Semantic methods of profile matching

More accurate and intelligent string matching algorithm can be used to handle name variations and short forms. In addition other string matching algorithms such as Jaro should also be reviewed and compared with the existing string matching algorithm.

In order to process tweets and social posts more accurately there can be improvements on the accuracy and performance of the NLP APIs. It is also possible to develop a custom local NLP library to use in conjunction with the existing third party NLP APIs. The framework can also improve its post analysis performance by relying not only on basic topic categorization but also on narrowed down categorization, keywords, time of post and hash-tags. Recently YouTube has introduced video tagging and Facebook has introduced hash-tags for posts, which can help to improve the categorization process.

Extended Profile Data

The current framework compares *Personal Identities*, *Social Identities* and *Relational Identities*. These classes can be extended to include images of the users. With the help of image analysis and comparison algorithms and face detection techniques the framework can perform image comparison across profiles. Papers such as [2] perform such comparisons.

Future versions of the framework can also compare shared links. For example if a user shares website links about soccer games, he/she can be categorized as someone who is interested in the sport of soccer.

Missing profile attributes can be inferred by methods such as the one described in [24]. There can also be research on the closeness of users to particular Facebook groups or Linkedin groups. This proximity helps to better understand the interests of users which leads to a better comparison among users.

Periodically Facebook and Linkedin introduce new profile attributes as well as new types of materials for users to share online, the subsequent iteration of this framework can extract these new profile details to gain a better understanding of the user. As more information becomes available to the framework, superior study of users can be achieved.

Efficiency, Scalability and Privacy

In terms of efficiency the proposed framework can be improved by utilizing more efficient comparison algorithms. The framework can also incorporate stricter privacy and security procedures - such as the use of hashing - to hide the identity of the users.

APPENDIX A

The effect of length of strings on String Similarity Score

A. 1. Result of string matching algorithm 3.3 with sample terms *A* and *B* with a constant edit distance value of 3. As the graph A.1 shows, while keeping the edit distance value constant, as the length of the shorter term *B* grows the value of the edit distance is reduced to the minimum value of 3. This technique puts more emphasis on typographical differences between shorter strings.

Table A.1 Normalized score between two strings A and B and the effect of length of strings

Length of Longer term |A|	**Length of Shorter term |B|**	**1 / |B|**	**Assumed Edit distance cost**	**edit distance cost + 1/|B|**	**Normalized score**
3	**1**	1	3	4	**0**
4	**2**	0.5	3	3.5	**0.22**
5	**3**	0.33	3	3.33	**0.37**
6	**4**	0.25	3	3.25	**0.48**
7	**5**	0.2	3	3.2	**0.55**
8	**6**	0.16	3	3.16	**0.61**
9	**7**	0.14	3	3.14	**0.65**
10	**8**	0.12	3	3.12	**0.69**

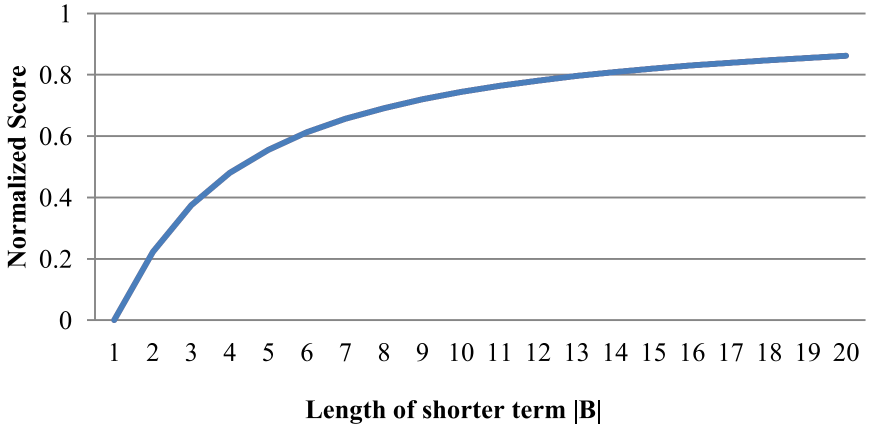


Figure A.1 Graph of the score obtained from comparing string *A* and *B*.

Mathematical evaluation for attribute weight assignments

A.2. As explained in section 4.2.3.1 the profile attribute weights are evaluated based on the data set of 1200 Facebook profiles. Empty or 'null' attribute values are omitted from the evaluation. The following table displays the mathematical process in calculating the weight of each profile attribute based on the following formula. For profile attributes that are inaccessible by the framework, the weights are manually set relative to the evaluated weights.

$$Weight\ of\ Profile\ Attribute = \frac{Total\ number\ of\ unique\ profile\ attribute\ values}{Total\ number\ of\ profile\ attribute\ values}$$

Table A.2 Mathematical evaluations for attribute weight assignment

Attribute name	Evaluated Weight $\in [0,1]$	Method of calculating the weight
First name	0.70	$\frac{930}{1316} = 0.70$
Last name	0.85	$\frac{1129}{1316} = 0.85$
Middle name	0.95	$\frac{95}{100} = 0.95$
Location	0.11	$\frac{96}{833} = 0.11$
Birthday	0.67	$\frac{586}{863} = 0.67$
Website	0.95	$\frac{139}{145} = .95$
Occupation	0.84	$\frac{408}{484} = 0.84$
Education	0.47	$\frac{410}{872} = .47$
Email	0.95	Manually set to the highest calculated weight
Telephone	0.95	Manually set to the highest calculated weight
Address	0.95	Manually set to the highest calculated weight

APPENDIX B

Numerical results of Facebook and Linkedin identity matching experiments; using threshold values computed by the aggregation functions

B.1. As discussed in section 3.2.3.4 there are various aggregation functions that yield different threshold values. The following tables are the numeric results of using different threshold values τ which are obtained from three different aggregation functions: *Minimum, Average* and *Maximum*. The data set used for these experiments is composed of 100 Facebook profiles.

Table B.1 Results of Facebook/Linkedin experiment using Minimum aggregation function

Aggregation function:		*Minimum (0.11)*
	Predicted as Non-Match	**Predicted as Match**
True Non-Match Cases	7	339
True Match Cases	0	77
Recall	0.19	
Precision	1.00	

Table B.2 Results of Facebook/Linkedin experiment using Average aggregation function

Aggregation function:		*Average (0.694)*
	Predicted as Non-Match	**Predicted as Match**
True Non-Match Cases	165	181
True Match Cases	6	71
Precision	0.28	
Recall	0.92	

Table B.3 Results of Facebook/Linkedin experiment using Maximum aggregation function

Aggregation function:		*Maximum (0.95)*
	Predicted as Non-Match	**Predicted as Match**
True Non-Match Cases	222	125
True Match Cases	11	66
Precision	0.35	
Recall	0.86	

Numerical results of Facebook and Linkedin identity matching experiments; using manually selected threshold values

B.2. As discussed in section 3.2.3.4 the alternative method of finding the optimal framework performance is to manually set the threshold value and observe the results. The following tables display the performance result of the framework in finding matching Linkedin profiles. The threshold values chosen are 0.55, 0.65, 0.75, 0.85 and 0.95. The data set used for these experiments is composed of 100 Facebook profiles.

Table B.4 Results of Facebook/Linkedin experiment using 0.55 as the threshold value

Threshold value:		*0.55*
	Predicted as Non-Match	**Predicted as Match**
True Non-Match Cases	98	248
True Match Cases	1	76
Precision	0.23	
Recall	0.99	

Table B.5 Results of Facebook/Linkedin experiment using 0.65 as the threshold value

Threshold value:		*0.65*
	Predicted as Non-Match	**Predicted as Match**
True Non-Match Cases	155	190
True Match Cases	4	73
Precision	0.28	
Recall	0.95	

Table B.6 Results of Facebook/Linkedin experiment using 0.75 as the threshold value

Threshold value:		*0.75*
	Predicted as Non-Match	**Predicted as Match**
True Non-Match Cases	175	170
True Match Cases	7	70
Precision	0.29	
Recall	0.91	

Table B.7 Results of Facebook/Linkedin experiment using 0.85 as the threshold value

Threshold value:		*0.85*
	Predicted as Non-Match	**Predicted as Match**
True Non-Match Cases	192	154
True Match Cases	10	67
Precision	0.30	
Recall	0.87	

Table B.8 Results of Facebook/Linkedin experiment using 0.95 as the threshold value

Threshold value:		*0.95*
	Predicted as Non-Match	**Predicted as Match**
True Non-Match Cases	222	125
True Match Cases	11	66
Precision	0.35	
Recall	0.86	

Numerical results of Facebook and Linkedin identity matching experiments based on Relational Identities

B.3. In order for the framework to access a user's network, the framework has to be authorized by the user. The following tables display the performance result of the framework in finding matching Linkedin profiles based on Relational Identities. The threshold values chosen are 0.1 and 0.2. The data set used for these experiments is composed of 3 Facebook profiles.

Table B.9 Results of Facebook/Linkedin experiment based on Relational Identities using 0.1 as the threshold value

Threshold value:		*0.1*
	Predicted as Non-Match	**Predicted as Match**
True Non-Match Cases	2	0
True Match Cases	1	2
Precision	1.00	
Recall	0.67	

Table B.10 Results of Facebook/Linkedin experiment based on Relational Identities using 0.2 as the threshold value

Threshold value:		*0.2*
	Predicted as Non-Match	**Predicted as Match**
True Non-Match Cases	2	0
True Match Cases	2	1
Precision	1.00	
Recall	0.33	

APPENDIX C

Numerical results of Facebook and Twitter identity matching experiments with different number of online posts extracted from each profile

C.1. The following tables display the obtained values from performing Facebook/Twitter profile matching, while extracting 5, 10 and 20 online posts and using 0.1 as the threshold value. The data set used for these experiments is composed of 40 Facebook profiles.

Table C.1 Result of Facebook/Twitter experiment while extracting 5 posts per profile

	Predicted as Non-Match	**Predicted as Match**
True Non-Matching Cases	189	161
True Matching Cases	20	20
Precision	0.11	
Recall	0.50	

Table C.2 Result of Facebook/Twitter experiment while extracting 10 posts per profile

	Predicted as Non-Match	**Predicted as Match**
True Non-Matching Cases	147	203
True Matching Cases	14	26
Precision	0.11	
Recall	0.65	

Table C.3 Result of Facebook/Twitter experiment while extracting 20 posts per profile

	Predicted as Non-Match	**Predicted as Match**
True Non-Matching Cases	125	225
True Matching Cases	14	26
Precision	0.10	
Recall	0.65	

C.2. The following tables display the obtained values from performing Facebook/Twitter profile matching while using 0.1, 0.2, 0.3, 0.4 and 0.5 as the threshold value. Only 5 posts are extracted from each profile. The data set used for these experiments is composed of 40 Facebook profiles.

Table C.4 Result of Facebook/Twitter experiment while setting 0.2 as threshold

	Predicted as Non-Match	**Predicted as Match**
True Non-Matching Cases	239	111
True Matching Cases	25	15
Precision	0.12	
Recall	0.38	

Table C.5 Result of Facebook/Twitter experiment while setting 0.3 as threshold

	Predicted as Non-Match	**Predicted as Match**
True Non-Matching Cases	280	70
True Matching Cases	29	11
Precision	0.14	
Recall	0.28	

Table C.6 Result of Facebook/Twitter experiment while setting 0.4 as threshold

	Predicted as Non-Match	**Predicted as Match**
True Non-Matching Cases	304	46
True Matching Cases	32	8
Precision	0.15	
Recall	0.20	

Table C.7 Result of Facebook/Twitter experiment while setting 0.5 as threshold

	Predicted as Non-Match	Predicted as Match
True Non-Matching Cases	314	36
True Matching Cases	35	5
Precision	0.12	
Recall	0.13	

RERERENCES

[1] T. Y. Sara Radicati, "Social Media Market, 2012-2016," The Radicati Group, Inc. 2012.

[2] P. Jain, P. Kumaraguru, and A. Joshi, "@ i seek'fb. me': identifying users across multiple online social networks," in *Proceedings of the 22nd international conference on World Wide Web companion*, 2013, pp. 1259-1268.

[3] J. Pistole, "Fraudulent identification documents and the implications for homeland security. Statement for the Record - Before the House Select Committee on Homeland Security.," in *House Select Committee On Homeland Security Washington DC*, ed, 2003.

[4] (Aug 20, 2013). *Facebook Inc.* Available: www.facebook.com

[5] (Aug 20, 2013). *Linkedin* Available: www.linkedin.com

[6] (Aug 20, 2013). *Twitter Inc.* Available: www.twitter.com

[7] S. Sarawagi and A. Bhamidipaty, "Interactive deduplication using active learning," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002, pp. 269-278.

[8] M. Bilenko, R. Mooney, W. Cohen, P. Ravikumar, and S. Fienberg, "Adaptive name matching in information integration," *Intelligent Systems, IEEE,* vol. 18, pp. 16-23, 2003.

[9] H. B. Newcombe, J. M. Kennedy, S. J. Axford, and A. P. James, "Automatic linkage of vital records," *Science (New York, N.Y.),* vol. 130, pp. 954-959, 1959.

[10] H. L. Dunn, "Record Linkage*," *American Journal of Public Health and the Nations Health,* vol. 36, pp. 1412-1416, 1946.

[11] I. P. Fellegi and A. B. Sunter, "A theory for record linkage," *Journal of the American Statistical Association,* vol. 64, pp. 1183-1210, 1969.

[12] D. Brickley and L. Miller, "FOAF vocabulary specification 0.98," *Namespace Document,* vol. 9, 2010.

[13] J. Allsopp, *Microformats: Empowering your markup for Web 2.0*. New York City, New York: friendsofED, 2007.

[14] B. Marshall, S. Kaza, J. Xu, H. Atabakhsh, T. Petersen, C. Violette*, et al.*, "Cross-jurisdictional criminal activity networks to support border and transportation security," in *Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on*, 2004, pp. 100-105.

[15] J. Li, G. A. Wang, and H. Chen, "Identity matching using personal and social identity features," *Information Systems Frontiers,* vol. 13, pp. 101-113, 2011.

[16] M. Motoyama and G. Varghese, "I seek you: searching and matching individuals in social networks," in *Proceedings of the eleventh international workshop on Web information and data management*, 2009, pp. 67-75.

[17] (Aug 20, 2013). *MySpace*. Available: https://myspace.com/

[18] M. N. Szomszor, I. Cantador, and H. Alani, "Correlating user profiles from multiple folksonomies," in *Proceedings of the nineteenth ACM conference on Hypertext and hypermedia*, 2008, pp. 33-42.

[19] (Aug 20, 2013). *Delicious* Available: https://delicious.com

[20] (Aug 20, 2013). *Flickr*. Available: http://www.flickr.com/

[21] T. Iofciu, P. Fankhauser, F. Abel, and K. Bischoff, "Identifying Users Across Social Tagging Systems," in *Conference on Weblogs and Social Media*, 2011.

[22] (Aug 20, 2013). *StumpleUpon*. Available: http://www.stumbleupon.com/

[23] D. Irani, S. Webb, K. Li, and C. Pu, "Large online social footprints--an emerging threat," in *Computational Science and Engineering, 2009. CSE'09. International Conference on*, 2009, pp. 271-276.

[24] C. G. Akcora, B. Carminati, and E. Ferrari, "Network and profile based measures for user similarities on social networks," in *Information Reuse and Integration (IRI), 2011 IEEE International Conference on*, 2011, pp. 292-298.

[25] D. Perito, C. Castelluccia, M. A. Kaafar, and P. Manils, "How unique and traceable are usernames?," in *PETS'11 Proceedings of the 11th international conference on Privacy enhancing technologies*, 2011, pp. 1-17.

[26] J. Vosecky, D. Hong, and V. Y. Shen, "User identification across multiple social networks," presented at the Networked Digital Technologies, 2009. NDT'09. First International Conference on, 2009.

[27] K. Cortis, S. Scerri, I. Rivera, and S. Handschuh, "Discovering semantic equivalence of people behind online profiles," in *In Proceedings of the Resource Discovery (RED) Workshop, ser. ESWC*, 2012.

[28] (Aug 20, 2013). *DBpedia*. Available: http://dbpedia.org/

[29] J. Golbeck and M. Rothstein, "Linking Social Networks on the Web with FOAF: A Semantic Web Case Study," in *AAAI'08 Proceedings of the 23rd national conference on Artificial intelligence*, 2008, pp. 1138-1143.

[30] E. Raad, R. Chbeir, and A. Dipanda, "User profile matching in social networks," presented at the Network-Based Information Systems (NBiS), 2010 13th International Conference on, 2010.

[31] M. A. Jaro, "Advances in record-linkage methodology as applied to matching the 1985 census of Tampa, Florida," *Journal of the American Statistical Association,* vol. 84, pp. 414-420, 1989.

[32] V. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," *Soviet Physics Doklady,* vol. 10, pp. 707-710, 1966.

[33] E. Gabrilovich and S. Markovitch, "Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis," in *In Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2007, pp. 1606-1611.

[34] R. Soltani and A. Abhari, "Identity Matching in Social Media Platforms," presented at the 2013 IEEE International Symposium on Performance Evaluation of Computer and Telecommunications Systems (SPECTS), Toronto, Canada, 2013.

[35] T. J. B. O'Reilly. (Aug 20, 2013). *Web 2.0 Opening Welcome:The State of the Internet Industry*. Available: http://itc.conversationsnetwork.org/shows/detail270.html

[36] G. Cormode and B. Krishnamurthy, "Key differences between Web 1.0 and Web 2.0," *First Monday,* vol. 13, 2008.

[37] (Aug 20, 2013). *YouTube Inc.* Available: www.youtube.com

[38] (Aug 20, 2013). *Twitter Statistics*. Available: http://www.statisticbrain.com/twitter-statistics/

[39] (Aug 20, 2013). *Google inc.* Available: www.google.com

[40] F. Naumann and M. Herschel, "An introduction to duplicate detection," *Synthesis Lectures on Data Management,* vol. 2, pp. 1-87, 2010.

[41] (Aug 20, 2013). *AlchemyAPI*. Available: http://www.alchemyapi.com/

[42] (Aug 20, 2013). *OpenCalais*. Available: http://www.opencalais.com/

[43] (Aug 20, 2013). *Pingar*. Available: http://www.pingar.com/

[44] (Aug 20, 2013). *Semantria*. Available: http://semantria.com/

[45] (Aug 20, 2013). *Wikimeta*. Available: http://wikimeta.com/

[46] H. Saif, Y. He, and H. Alani, "Alleviating data sparsity for twitter sentiment analysis," in *The 2nd Workshop on Making Sense of Microposts*, 2012.

[47] F. Abel, Q. Gao, G.-J. Houben, and K. Tao, "Semantic enrichment of twitter posts for user profile construction on the social web," in *The Semanic Web: Research and Applications*, ed: Springer, 2011, pp. 375-389.

[48] T. Steiner, R. Verborgh, J. G. Vallés, and R. de Walle, "Adding meaning to Facebook microposts via a mash-up API and tracking its data provenance," in *Next Generation Web Services Practices (NWeSP), 2011 7th International Conference on*, 2011, pp. 342-345.

[49] D. Recordon and D. Reed, "OpenID 2.0: a platform for user-centric identity management," in *Proceedings of the second ACM workshop on Digital identity management*, 2006, pp. 11-16.

[50] (Aug 20, 2013). *LifeStream* Available: http://lifestream.aol.com/

[51] (Aug 20, 2013). *FriendFeed*. Available: http://friendfeed.com/

[52] M. G. Elfeky, V. S. Verykios, and A. K. Elmagarmid, "TAILOR: A record linkage toolbox," in *Data Engineering, 2002. Proceedings. 18th International Conference on*, 2002, pp. 17-28.

[53] P. Christen, "Febrl-: an open source data cleaning, deduplication and record linkage system with a graphical user interface," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008, pp. 1065-1068.

[54] (Aug 20, 2013). *InfoSphere Global Name Management*. Available: http://www-03.ibm.com/software/products/us/en/infosphere-global-name-management

[55] (Aug 20, 2013). *Informatica Identity Resolutions*. Available: http://www.informatica.com/ca/solutions/enterprise-data-integration-and-management/identity-resolution/

[56] (Aug 20, 2013). *Infoglide Software*. Available: http://www.infoglide.com/

[57] (Aug 20, 2013). *WizSoft - Data and Text Mining*. Available: http://www.wizsoft.com/

[58] Y. Sun, M. Robinson, R. Adams, R. Te Boekhorst, A. G. Rust, and N. Davey, "Using feature selection filtering methods for binding site predictions," in *Cognitive Informatics, 2006. ICCI 2006. 5th IEEE International Conference on*, 2006, pp. 566-571.

[59] B. Han and T. Baldwin, "Lexical normalisation of short text messages: Makn sens a# twitter," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011, pp. 368-378.

[60] (Aug 20, 2013). *Facebook Platform Policies.* Available: https://developers.facebook.com/policy

[61] (Aug 20, 2013). *LinkedIn Platform Guidelines* Available: http://developer.linkedin.com/documents/linkedin-platform-guidelines

[62] (8/20/2013). *Developer Rules of the Road* Available: https://dev.twitter.com/terms/api-terms

[63] (Aug 20, 2013). *The OAuth 1.0 protocol*. Available: http://tools.ietf.org/html/rfc5849

[64] (Aug 20, 2013). *The OAuth 2.0 authorization framework*. Available: http://tools.ietf.org/html/rfc6749

[65] (Aug 20, 2013). *Twitter Feed*. Available: http://www.twitterfeed.com/