



UPPSALA
UNIVERSITET

IT 13 050

Examensarbete 30 hp
Juli 2013

Visualization and Detection of Multiple Aliases in Social Media

Amendra Shrestha

Institutionen för informationsteknologi
Department of Information Technology



UPPSALA
UNIVERSITET

**Teknisk- naturvetenskaplig fakultet
UTH-enheten**

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
<http://www.teknat.uu.se/student>

Abstract

Visualization and Detection of Multiple Aliases in Social Media

Amendra Shrestha

Monitoring and analysis of web forums are becoming important for intelligence analysts around the globe since terrorists and extremists are using forums for spreading propaganda and communicating with each other. Various tools for analyzing the content of forum postings and identifying aliases that need further inspection by analysts have been proposed throughout literature. But a problem related to web forums is that individuals can make use of several aliases and conceal their identity.

In this thesis work we propose a number of matching techniques for detecting forum users who make use of multiple aliases. By combining different techniques such as time profiling and stylometric analysis of messages the accuracy of recognizing users with multiple aliases increases, as shown in experiments conducted on the (ICWSM) International Conference on Weblogs and Social Media Dataset Boards.ie Dataset.

Handledare: Lisa Kaati
Ämnesgranskare: Mohamed Faouzi Atig
Examinator: Ivan Christoff
IT 13 050
Tryckt av: Reprocentralen ITC

This thesis work is dedicated to those innocent men, women and children who losses their life in any kind of terror attacks.

Acknowledgements

I would like to express my sincere gratitude to my thesis supervisor, **Mrs. Lisa Kaati** for her expert guidance and valuable feedback throughout the research. Her guidance helped me in research and writing of thesis report. I deeply thank my parents, family members and friends for their continual encouragement and support.

Finally, many thanks to Uppsala University and all of those who has helped me in any respect during the completion of this thesis and my computer science master degree.

Contents

Contents	iv
List of Figures	vi
List of Tables	vii
Nomenclature	viii
1 Introduction	1
1.1 Detecting users with multiple aliases	2
2 Background and Related Work	4
3 Detecting Multiple Aliases	6
3.1 Detecting the use of multiple aliases	6
3.1.1 Case 1: Multiple aliases case	6
3.1.2 Case 2: Alter ego case	7
3.2 Techniques for Detecting Multiple Aliases	8
3.2.1 String-based matching	9
3.2.1.1 Jaro-Winkler	9
3.2.2 Stylometric Matching	11
3.2.3 Time-based matching	13
3.2.4 Social-network based matching	14
3.3 Matching of aliases	15
3.3.1 Approaches for comparing aliases	15
3.3.2 Approaches for using matching techniques	16
4 Design and Implementation	18
4.1 Visualization	18
4.2 Scientific Visualization (SciVis)	19
4.2.1 Information Visualization (InfoVis)	19
4.2.2 InfoVis Pipeline	20

4.3	XML Parser	21
4.3.1	Parsers	21
4.3.1.1	SAX Parser	21
4.3.1.2	DOM Parser	22
4.3.2	Analyze the Feed	22
4.3.3	Instantiate the Parser	22
4.3.4	Parse XML data	22
4.4	Database	23
4.5	prefuse	24
4.5.1	Loading Data	24
4.5.2	Visual Abstraction	25
4.5.3	Views and User Interaction	26
4.6	System	26
5	Experiments	29
5.1	Dataset	29
5.1.1	SIOC Architecture	29
5.2	Tests	35
5.2.1	Machine specification	35
5.2.2	Methodology	35
5.3	Result and Discussion	36
6	Conclusion and Future Work	41
7	Appendix A	43
7.1	Function words	43
	References	47

List of Figures

3.1	Social-network based matching	15
4.1	The InfoVis Pipeline	20
4.2	prefuse	25
4.3	System	27
5.1	The SIOC architecture	30
5.2	Graph for Top 1 matches	39
5.3	Graph for Top 3 matches	39

List of Tables

3.1	Features used for stylometric matching	11
5.1	Test result for Top 1	37
5.2	Test result for Top 3	38

Nomenclature

Greek Symbols

Γ Gamma

Acronyms

API Application Programming Interfaces

DML Data Manipulation Language

DOM Document Object Model

FOAF Friend of a Friend

I/O Input/Output

ICWSM International Conference on Weblogs and Social Media

IP Internet Protocol

JUNG Java Universal Network/Graph

RDB Relational Database

RDBMS Relational Database Management System

RDF Resource Description Framework

SAX Simple API for XML

SIOC Semantically-Interlinked Online Communities

SQL Structured Query Language

URL Uniform Resource Locator

XML Extensible Markup Language

Chapter 1

Introduction

Internet is a platform for any individuals who want to express and share imaginary and visionary theories. It serves as an open meeting place to share ideas, philosophy, problem and personal judgments related to any subject matter. For many extremist groups and terrorists, Internet is a vital motivator for their ideology and assists them for exchanging and reinforcing their beliefs [25], which increases the risk of committing violent activities towards the society. A couple of years ago extremist groups mainly used printed magazines and centralized websites for spreading their views and information, but this has to a large degree been replaced by interactive discussion forums (such as the Ansar Al-Mujahidin Jihadist Forum) and social media platforms such as YouTube, Twitter and Facebook [35]. Monitoring and analysis of web forums (also called discussion boards, discussion forums, message boards, Internet forums, etc.) is therefore becoming an important task for analysts in order to detect individuals that might pose a threat towards society.

Terrorist activities on the Internet can potentially be detected by monitoring the traffic to websites and forums associated with terrorist organizations under surveillance. In this manner, users accessing these websites can be identified based on their unique IP (Internet Protocol) addresses. Unfortunately, it is difficult to monitor those sites since they do not use single fixed IP addresses and URLs (Uniform Resource Locators) [4]. In addition, these sites frequently change their geographical location of Web hosting servers in order to prevent the use of such techniques. Similarly, anonymization techniques like Onion Routing [9] and Crowds [21] have made it easier for users to hide their identity and activity on the Internet.

A discussion board is a Web application that is used to publish user-generated content under the form of a discussion. Discussions considering particular sub-

jects are called threads or topics. Discussion boards have an important social aspect since they might be active for a long period of time and attract a group of users that builds a community. Several different discussion boards exist that are dedicated to almost all possible aspect of human activity. This allows Internet users to find a board that suits their interests and needs.

A user on a discussion board can make multiple accounts in same forum using altered aliases¹. There are several different reasons and objectives for users to have multiple aliases. One reason is that a user has been banned by an administrator because of posting some extreme posts on forum, another reason is that a user has forgotten his old password. In some cases, an old alias might has lost the trust of the group or still exist but needs alter ego to support his arguments. It could also be the case that users have altered aliases to disguise themselves. In some cases an administrator bans aliases due to inactivity (in these these cases the aliases have not been used for a long time).

1.1 Detecting users with multiple aliases

The objective of this work is to develop techniques that can be used to detect users with multiple aliases. Some of the techniques are implemented and tested in a prototype to ensure that the theory works in practice.

There are several reasons for using multiple aliases, we have identified two cases; concealed case and non-concealed case. For each case, some assumptions are done regarding the behavior of the user and the assumptions are reformulated to detect multiple aliases. We attempted to determine multiple aliases of a user in a group by using a number of different matching techniques. The matching techniques uses different attributes such as: network of friends, time of posting messages, choice of alias name, the actual message text and activity in certain threads. The result from each matching technique can be combined and used to identify users with multiple aliases. The experimental results suggest that the combination of matching techniques can give significantly better results than if the techniques are applied individually. The experimental results show that the achieved accuracy is largely dependent upon the number of aliases under consideration.

More specifically we have:

- Parsed a subset of the ICWSM (International Conference on Weblogs and

¹Two or more username are mapped to the same entity

Social Media), *Boards.ie Dataset*¹

- Visualized and structured the dataset
- Implemented stylometric analysis of messages
- Developed and implemented time profile matching
- Developed and implemented algorithms for combining different matching techniques

The rest of this thesis is structured as follows. In Chapter 2 related work is presented. In Chapter 3 various cases in which people may use several aliases is described and techniques for how the use of multiple aliases can be detected in these cases are suggested. The suggested matching techniques have been implemented into a testbed which is used for the experiments with the ICWSM *Boards.ie Dataset*. The implementation is described in Chapter 4 and the experimental results with a discussion is presented in Chapter 5 The thesis is concluded in Chapter 6, together with thoughts for future work.

¹Data is available on <http://icwsm.cs.umbc.edu/>

Chapter 2

Background and Related Work

Alias matching is not a new topic in field in computer science. In [23], the problem of "anti-aliasing" is studied, i.e. to link multiple aliases to known individuals based on their postings in public fora such as bulletin boards, weblogs, and web pages. More specifically, the technique used for matching the aliases is based on the used vocabulary (i.e., which words that are used in the postings). The results are promising, but the similarity of aliases relies heavily on the topic that they write about since the users vocabulary is used as discriminating features. For this reason the method is not suitable for people writing about heterogeneous topics.

Writing style is also used in [22], but in this work stylometric features which are not topic dependent are used. By using such stylometric features (e.g. function words and the use of syntactic category pairs) Internet-scale authorship identification is conducted. The experiments were made on a large collection of blog posts written by 100 000 different authors. By using a small sample of blog posts they tried to identify the rest of the posts written by the same author, mixed in with the 100 000 other blog posts. Their algorithms ranked the possible authors in descending order of probability and the top guess was correct about 20% of the time. In 35% of the cases, the correct author was in the top 20 guesses. The precision was improved to 80% by lowering the recall to 50%. The results in [22] indicate that the method is scalable and applicable to large amount of data from e.g the Internet. We have in this article used many of the stylometric features suggested in [22], but have also used other classifiers than stylometric matching.

In [24] detecting multiple aliases is done by combining behavioral features such as emails, phone calls and transactions. To detect multiple aliases on the Internet language use and style is suggested. In [5], methods for combining output from several matching techniques such as field matching, graph matching, and text-based matching are described. The combination of these methods is supposed to

improve identification of multiple aliases. However, no experimental results are shown.

Chapter 3

Detecting Multiple Aliases

When trying to detect individuals who are using multiple aliases on a web forums, there are several kinds of features and techniques that may be considered. To be clear on the terminology used, we will use the terms *matching techniques* or *classifiers* when referring to the algorithms used for identifying multiple aliases, while the term *features* will be used for the more low-level attributes which are used within the classifiers.

3.1 Detecting the use of multiple aliases

In this work we have identified two scenarios where multiple aliases are used. The first is when an individual is active on two or more discussion boards and uses different aliases on different forums (case 1). The second is when an individual make uses of several aliases on the same discussion board (case 2).

3.1.1 Case 1: Multiple aliases case

In this case, the user does not try to disguise him or herself. Possible reasons for using multiple aliases are:

- The user might be banned from using social media if he posted anything which are against the laws of discussion board. For example: if you post the links for downloading songs in Facebook¹ then your facebook account might get banned.
- If the user has not logged in into his account since long time then the administrator might ban his/her account because of inactivity.

¹social networking service. For more info: <https://www.facebook.com/>

-
- Some of the discussion boards set rules for alias name. For example the alias name cannot be longer than certain length or should use some characters. Hence the user needs to create account with different alias name in different social media.
 - If a user forgets his password then he needs to create another account in same social media with different alias.
 - If the alias is already in use then he has to choose another alias which might be different from his previous social media alias.

Assumptions:

- The alises have similar friend network.
- The different aliases don't reply on same thread in case of multiple aliases case because everyone knows the identity of user. Hence, there is no need of writing in same thread with different accounts. Nevertheless, if an alias has replied in a thread and if he forgets password. He can create another alias and can follow the thread again. In this case, the different alias reply in same thread.
- The user might choose alias name similar with the previous alias.
- They have same time profile.
- There is a similarity in writing style.

3.1.2 Case 2: Alter ego case

In this case, the user might try to disguise him or herself. Possible reasons for using alter ego aliases are:

- The user might be banned from using social media if he posted anything which are against the laws of discussion board.
- If the old alias lost the trust of the group then he will create new account and convey his views.
- Due to some reasons if a user develops bad personal relationships with members of the group then he might create new account and again join the group with different alias name.

-
- The author requires an alter ego to support his arguments. For example, people buy books by reading reviews given by the previous readers. If an individual creates multiple accounts and gives multiple positive review for a book then people will buy the book.

Assumptions:

- Since the user has created different alias to hide his identity then the common friends between two different accounts will be less.
- The different aliases write in at least one common thread which helps to promote or support their views.
- The user might not choose alias name similar with the previous alias.
- They have same time profile.
- There is a similarity in writing style.

3.2 Techniques for Detecting Multiple Aliases

Most existing works on alias matching focus on techniques for finding similarities in usernames (see e.g. [27]). Such techniques may work well for non-concealed cases (e.g. when the user is using similar usernames on several social media services and is not deliberately trying to hide that several aliases belong to the same individual). However, an individual may choose very dissimilar aliases (deliberately or not) and for such cases techniques that simply rely on similarities of usernames will not be fruitful. Moreover, two usernames may be very similar without belonging to the same individual. Hence, string matching techniques will not always be enough. For those reasons, the string matching technique is implemented with other techniques for alias matching. Similarly, we have also proposed a technique to combine the results in order to come up with better alias matching possibilities.

We present a set of matching techniques that can be used to identify multiple aliases created by a single author/individual. The matching techniques are:

- **String-based matching**
- **Stylometric matching**
- **Time profile-based matching**
- **Social network-based matching**

3.2.1 String-based matching

Aliases usually consist of text strings. As has been discussed before, the similarity of two aliases can be a useful feature to consider when trying to find users making use of multiple aliases, at least for the non-concealed case. There are several ASM (Approximate String Matching Algorithms) that have been suggested for matching names in the record linkage community [3]. Each algorithm has a particular use depending on the kind of data that are compared. The character based similarity metrics Jaro-Winkler algorithm [34] is used which is suitable for our application area [7, 33]. Jaro-Winkler is a suitable choice of algorithm [3] for finding similarity in short strings.

3.2.1.1 Jaro-Winkler

Jaro-Winkler distance is a degree of similarity between two strings. It is particularly efficient when there is a comparison between short strings, such as names. This is due to its relative robustness against letter transpositions and its weighting of similarity toward the beginning of the string. The two characters from two strings $s1$ and $s2$ are matched only when the variance between them is not greater than $\left\lfloor \frac{\max(|s1|, |s2|)}{2} \right\rfloor - 1$. In this calculation, each character of first string is compared with all matching characters of second string, and is divided by 2 to give the number of required transpositions.

Since we are comparing user's name in a discussion board, Jaro-Winkler distance is a suitable choice of algorithm. The similarity between two strings is directly proportional to Jaro-Winkler distance, i.e. higher the distance the more similar the strings are. The result is always normalized between 0 and 1 where 0 represents no match and 1 represents perfect match.

Jaro-Winkler distance is composed of two parts, Jaro's original algorithm [20] and Winkler's extension. The Jaro measure first computes the number of *matches* and compute the number of *transpositions* needed between two strings. Here, *transpositions* is the number of common characters which are not in sequence between two comparing strings. Then Winkler increased this measure by matching initial characters, then rescaled it by a piecewise function, whose intervals and weights depend on the type of string.

The Jaro distance for two strings $s1$ and $s2$ is given by:

$$d_j = \begin{cases} 0 & \text{if } m = 0 \\ \frac{1}{3} \left(\frac{m}{|s1|} + \frac{m}{|s2|} + \frac{m-t}{m} \right) & \text{Otherwise} \end{cases}$$

where:

-
- d_j is the Jaro distance
 - m is the number of matching character
 - t is half the number of transposition

Jaro distance consists average of three different sub-calculations:

1. ratio of matching characters to the length of first string $s1$.
2. ratio of matching characters to the length of second string $s2$.
3. ratio of non-transposition to the number of matching character.

The Jaro distance is used to calculate the Jaro-Winkler distance. Winkler states that if the prefix is common in two strings then the similarity score between them should be increased [27]. To compute the Jaro-Winkler distance, the number of matched prefix at the starting of the string l is used and rescales with a constant scaling factor p which increases the rating of string match. The standard value for the constant p is 0.1. We have use the same value in our implementation. The Jaro-Winkler distance d_w is given by:

$$d_j + (lp(1 - d_j))$$

Example

Given the strings $s1 = \text{MARTHA}$ and $s2 = \text{MARHTA}$

Matching characters $m = 6$

Length of first string $|s1| = 6$

Length of second string $|s2| = 6$

The mismatched characters transposition T/H and H/T is given by:

$$t = \frac{2}{2} = 1$$

Since the matching characters m is not equal to 0. So, the Jaro distance is given by:

$$d_j = \frac{1}{3} \left(\frac{6}{6} + \frac{6}{6} + \frac{6-1}{6} \right) = 0.944$$

The first 3 characters are common on both strings so, $l = 3$
Let us take standard value $p = 0.1$.
Thus Jaro-Wrikler distance is:

$$d_w = 0.944 + (3 * 0.1(1 - 0.944)) = 0.961$$

The above example shows that the similarity between two strings, MARTHA and MARHTA is 0.961.

3.2.2 Stylometric Matching

Stylometry is the statistical analysis of writing style [36]. It is one of the most likely and noteworthy mechanism for the cases where the alias names cannot be reliably used. With this technique, the author’s writing style is analyzed by constructing a ”writeprint”, which in many ways resemble a fingerprint and can be used in a similar way. A lot of algorithms and features for stylometry-based author identification have been proposed throughout the literature [29, 1, 14]. Most work has however been focused on closed-world problems with a small number of potential authors and a rather large quantity of text to build the stylometric profiles from (such as long literary books). Less research effort has been devoted to problems with a large number of potential authors and small quantities of text material, or to the cyberspace domain in general [36]. We have used a subset of the features used in the recent article by Narayanan et al. [22] and an extra feature: the frequency of sentence lengths. Our algorithm also shows that it is possible to identify users by looking just small portion of written text.

Category	Description	Count
Word length	Frequency of words with 1-20 characters	20
Sentence length	Frequency of sentences with various lengths	6
Letters	Frequency of <i>a</i> to <i>z</i> (ignoring case)	26
Digits	Frequency of 0 to 9	10
Punctuation	Frequency of characters . ? ! , ; : () ” - ‘	11
Function words	Frequency of various function words	293

Table 3.1: Features used for stylometric matching

Table 3.1 gives an overview of feature sets used for stylometric analysis. The features are extracted from the posts and transferred into feature vectors which are inputs for stylometric classifier. We have analyzed the frequency of english alphabets and numbers as well as the word length distribution from post while creating feature vectors. We look at only those words which have characters not more than 20 length. We have taken in consideration the punctuation used by author in their post like . , ? etc. We have also checked the frequently using verbs, pronouns, prepositions and question words in the post named as function words. Function words are the list of syntactic features which have been shown to be highly effective differentiators of authorship, since the usage variations of such words are a strong reflection of stylistic selection [18]. The function words are listed in Appendix 7.1.

Several other features could have been used, including lexical features such as vocabulary richness (e.g. using frequency of hapax legomena¹ or Yule’s K measure). Yule’s Characteristic (K) is a word frequency measurement for large blocks of text. It measures the likelihood of two nouns, chosen at random from the text because of being the same. Thus, it is a measurement of the complexity of the text, as well as its repetitiveness. Yule’s Characteristic measurements are given in the form of a positive integer which represents the ratio of misses to hits of the block of text, i.e. K value of 100 describes that for any pair of nouns chosen at random from the given text, there is 1 in 100 chance that they will be the same. There is also the possibility of using syntactic features such as part-of-speech tag n-grams, and idiosyncratic features such as misspelled words. POS (Part-of-Speech) tagging is a technique for assigning each word of a text with an appropriate parts of speech tag. The significance of POS for language processing is the large amount of information they give about a word and its neighbor. POS tagging can be used in TTS (Text to Speech), information retrieval, shallow parsing, information extraction, linguistic research for corpora [15] and also as an intermediate step for higher level NLP (Natural Language Processing) tasks such as parsing, semantics, translation, and many more [10]. We are not arguing that we have used the richest set of features possible, but rather that we have incorporated a lot of useful features that reasonably fast can be extracted from forum posts.

Many modern algorithms [6, 19, 13] for author identification are based on machine learning, such as SVMs (Support Vector Machines) and decision trees. Such algorithms can be used for learning classifiers to generalize from training data in order to make good classifications on (previously unseen) test data, but

¹once-occurring words

are in general not appropriate for determining how similar the writeprints of two aliases are. We are therefore using the more basic approach to compare how similar the (normalized) stylometric feature vectors are for two aliases by simply calculating the cosine of the angle between them. The cosine of an angle is given by,

$$\cos(p, q) = \frac{p \cdot q}{\|p\| \|q\|} = \frac{\sum_{i=1}^n p_i \times q_i}{\sqrt{\sum_{i=1}^n (p_i)^2} \times \sqrt{\sum_{i=1}^n (q_i)^2}} \quad (3.1)$$

where p_i and q_i are two stylometric feature vectors of alias p and q respectively. n is the number of feature vectors.

3.2.3 Time-based matching

We have proposed to look at the point in time when different aliases have created their forum posts. This can give important clues to whether two different aliases refer to one individual or not. However, comparing the creation time of two posts is not reliable enough since it is likely that two individuals create their posts during the same time period without any other reason than pure chance or living in the same time zone. In this work we create time profiles based on the relative distribution of the time of day when the postings have been made, where the time of day is discretized into intervals of equal size (each interval corresponds to one hour). For example, assume that *AliasX* has written 12 posts in total, with the following times of posting:

7:01, 7:25, 7:29, 7:40, 8:05, 8:55, 9:27, 10:17, 10:43, 13:11, 14:19 and 14:59.

The first step is now to construct a feature vector corresponding to the frequency with how many posts that have been written each hour:

$$< 0, 0, 0, 0, 0, 0, 4, 2, 1, 2, 0, 0, 1, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0 >$$

Since we are interested in how the number of posts are distributed throughout the day rather than in the exact numbers of posts (some aliases write posts more frequently than others) we are in the next step normalizing the feature vectors, resulting in:

$$< 0, 0, 0, 0, 0, 0, 0.33, 0.16, 0.083, 0.16, 0, 0, 0.083, 0.16, 0, 0, 0, 0, 0, 0, 0, 0 >$$

In this way, a normalized time profile is created for each alias. We use the time interval one hour, we have also experimented with binary feature vectors and different larger time intervals but this yielded slightly worse results in general. In next step, the Euclidean distance is used for calculating how far away two time

profiles are from each other. The smaller the distance between the time profiles for two aliases, the more likely it is that the two aliases belong to the same user. Formally, the Euclidean distance between two vectors p and q is given by:

$$d(p, q) = \sqrt{\sum_{i=1}^n (q_i - p_i)^2} \quad (3.2)$$

where n is the time of day and p_i and q_i are numbers of messages posted at each hour of day by user p and q respectively.

The fact that two aliases have similar (dissimilar) time profiles does not mean that the individuals are necessarily the same (different) individuals, but it can be used as evidence for or against such hypotheses.

3.2.4 Social-network based matching

The underlying idea of using social networks is that a mapping and comparison of the social network of two aliases can reveal if those aliases are similar in the sense of whom they are connected to. The social network can be based on various information, depending on what the discussion forum look like. In some forums (such as the forums we have used in our experiments), there are friend or "buddy" lists available, in which the user can mark other users as friends. In many forums such friend lists are lacking, but it is also possible to use other kind of information to create social networks. These networks can for example consist of threads (connecting users who have made postings in the same thread) or topics (connecting users who have written about the same topic).

We use vertex similarity to calculate how similar two aliases are in terms of their social network either the social network is constructed based on friend-, thread- or topic information. The vertex similarity can be calculated as a function of the number of neighbors in common for two aliases. If the total number of neighbors should not impact the results too much, a normalization process in which the node degrees are taken into account is needed.

Let Γ_p be the neighborhood of vertex (alias) p in the network and Γ_q be the neighborhood of vertex (alias) q . Now, the number of common neighbors is calculated as $|\Gamma_p \cap \Gamma_q|$. The normalization can be done in various ways (such as with dice or cosine similarity), but in our implementation we make use of the Jaccard similarity coefficient. The Jaccard similarity coefficient is a statistic which helps in finding similarity and variety between two collections. It is the ratio of size of the intersection and the size of the union of the collections. The

Jaccard similarity coefficient $J(p, q)$, where:

$$J(p, q) = \frac{|\Gamma_p \cap \Gamma_q|}{|\Gamma_p \cup \Gamma_q|} \quad (3.3)$$

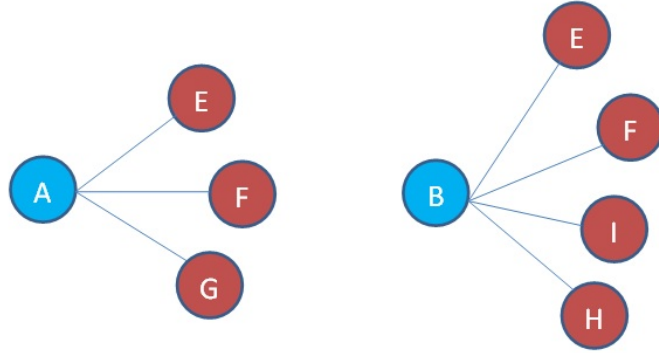


Figure 3.1: *Friend Equality*. Alias A and Alias B has 2 common friends (E, F) and friend equality is 2/5.

3.3 Matching of aliases

Each matching technique can be used to compute the similarity between two aliases, which technique to include depends on what case are consider (concealed or non-concealed case). If we are dealing with a non-concealed case all matching techniques may be used, while the string-based technique probably will be of little or no value for the concealed case.

The result of the matching techniques can be combined in various ways. If we want to decide whether two aliases should be merged or not, a straightforward approach is to combine the results from the used matching techniques into a (weighted) average.

3.3.1 Approaches for comparing aliases

There are several approaches for performing alias matching such as:

- **Match an alias against a list of other aliases:** Here a list of aliases is created and the first alias of list is compared with rest of the aliases. If the user chooses N users from a graph then there are (N-1) results for each classifier. We have implemented this approach in our prototype.

-
- **Match two aliases against each other:** In this approach two aliases are compared using the matching techniques described in Section 3.2. This approach is least computationally demanding since only two aliases are compared.
 - **Match all aliases in a list against each other:** A list of aliases is created and each of the aliases in list are analyzed and compared with each other. This method is the most computationally demanding since if N aliases are selected, then there are

$$\sum_{x=1}^{N-1} (N - x) \quad (3.4)$$

results for each matching technique.

3.3.2 Approaches for using matching techniques

After choosing a set of aliases a suitable set of matching techniques needs to be selected. The matching techniques can be applied all at once, in a sequence or by choice.

- **Apply at once:** When the matching techniques are applied at once, all matching techniques are computed simultaneously. This process is computationally demanding since all matching techniques are computed for all aliases.
- **Apply sequentially:** When using sequential matching only one matching technique at time is computed. Aliases that are considered as similar enough are kept and a different classifier is applied to these. The order of the matching techniques and can be defined dependent on the case and computational resources. Sequentially applying matching techniques can be less computationally demanding than applying all techniques at once.
- **Apply by choice:** Allowing a selection of matching techniques that can be used in any order to any given aliases is the least computationally demanding approach. This approach allows more control over the decision making process. The ability to flexibly choose sequence of matching techniques is highly effective in alias matching task [16] and therefore we use this approach in our experiments.

In some occasions it might be interesting to find out which alias in a set of aliases $A = \{a_1, \dots, a_n\}$ a certain selected alias a_0 is most similar to. Depending

on the size of the set, various approaches can be used. If the set is reasonably small it makes sense to output a rank from each matching technique (where the rank is based upon the computed similarities for each alias in the set). In this case a (weighted) average of the rankings can be calculated.

Apply by choice approach is used in our experiments which are presented in Chapter 5. If there are many aliases to compare and the alias matching has to be performed in a near real-time application, it may take too long to apply all matching techniques in parallel. In such cases it is better to apply the matching techniques in sequence, starting with the computationally cheap techniques for making a first coarse filtering. After the filtering only the k best matches are kept and the remaining techniques can be applied on the filtered subset of aliases.

Chapter 4

Design and Implementation

Visualizing and analyzing large relational data requires an interface that provides a tight coupling between interface design, algorithms, meaningful appearance and interactive view that approve or contradict an end users speculations. We have implemented a prototype that can be used to visualize and analyze a discussion board. The prototype visualizes a data set using a graph layout where the nodes represents users and an edge between two users is present if the users are friends. A subset of the data set can be selected in the prototype for further analysis.

4.1 Visualization

Information visualization is the use of computer-supported, interactive and visual representations of abstract data to amplify cognition [2]. The visual sense is the most sophisticated ability of human-beings which they use to obtain information about what they had seen.

Information visualization presumes that:

"visual representations and interaction techniques take advantage of the human eye's broad bandwidth pathway into the mind to allow users to see, explore, and understand large amounts of information at once. Information visualization focused on the creation of approaches for conveying abstract information in intuitive ways".[32]

Visualization is the scientific way of presenting abstract data in a convenient, clear, and comprehensible way that can be perceived by humans easily [8]. It is the process of creating meaningful visual structure of data where users can visualize, analyze and interpret information as they desire. The important aspect

of visualization is interactivity between computer applications and users. Users may not only want to view information, they might want to interact with the application. A user should not only be able to explore and search data, they should also be able to put up the visualization for their current needs, for instance to zoom in to reach more details or zoom out to view whole data at once. They should be able to highlight important aspects or expand and collapse composite data structures or finding the relation between data. Visualization is the formal way of delivering an overview of data to an observer and the aim of visualization is to gain cognition, deeper understanding and a properly new insight into data or processess.

The field of visualization has been developing with two different communities. Although they vary in their approaches and aim, they have a common goal i.e. to present the sophisticated data in efficient and meaningful way. The two different fields in visualization are:

4.2 Scientific Visualization (SciVis)

Scientific visualization focuses on visualizing results of research simulations, scientific and medical data and flowing patterns. They enhance use of visualization to explore and analyze data is used to demonstrate research results in a more understandable way. This type of visualization facilitates scientists to graphically illustrate scientific and gather sufficient and clear information from the data. Medical tomography data, weather forecast graph over a time, terrain rendering and molecular rendering are some of the field where scientific visualization is used.

4.2.1 Information Visualization (InfoVis)

Information Visualization is the way of communicating abstract data through the use of interactive visual interfaces [17]. The main goal of information visualization is to find an appropriate way for visualizing data and analyzing abstract data visually. Digital library, data mining, information graphics, financial data analysis, market study and crime mapping are some of the areas where InfoVis are being applied. The data from a discussion board is analyzed and we will mainly focus on InfoVis in our thesis work.

Visualizing abstract data in a systematic way is challenging and tricky to visualize. Information visualization applications are difficult to build since it requires highly skilled expertise in the field of mathematics as well as in programming in order to implement complex layout algorithm and dynamic graphics. After doing

a survey on existing visualization toolkits¹, the pre-built toolkit *prefuse*² was used in our prototype to create dynamic visualization of discussion boards. *Prefuse* provides theoretically-motivated abstract classes to design multi-functional visualization applications [11]. *Prefuse* has been widely used and accepted in the community for many years and is proven to work efficiently, well organized and it is possible to extend and customize the classes of *prefuse* to meet specific needs and specifications.

4.2.2 InfoVis Pipeline

The InfoVis Pipeline is the software architecture model that executes the visualization process by breaking down into series of discrete steps. The result of previous step is used by upcoming step to eventually transform a raw data into a tangible visual data. The InfoVis Pipeline describes the working mechanism of information visualization. The process is illustrated in figure 4.1.

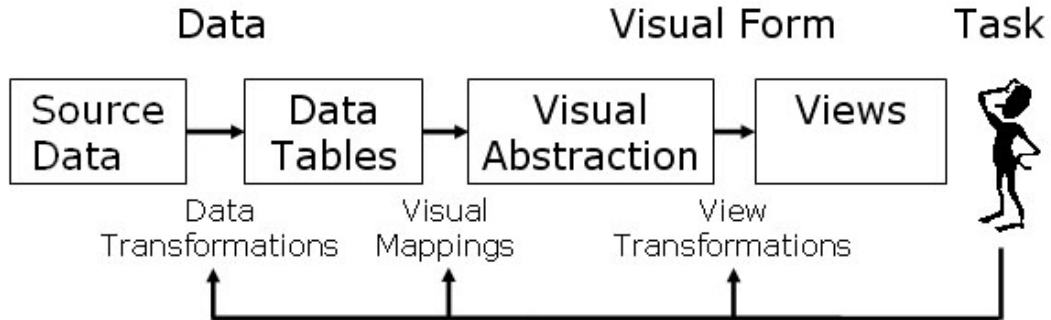


Figure 4.1: *The InfoVis Pipeline*. The source data³ is mapped into appropriate data tables structure which are used for visualization. These backing tables are then used for various purpose such as for the construction of visual object of data and for modeling other visual properties like defining position, font, color and shape and size of the object. This visual abstraction is then used to create interactive view of the data so that user can make changes at any point in framework[11].

¹JUNG, Piccolo, Graphviz and Gephi toolkit

²For more information about *prefuse* see <http://prefuse.org/doc/manual/> (accessed on August 14th, 2012).

³Source data can be the text file or relational database tables.

The process of visualization starts with the the collection of data that needs to be visualized. The *source data* could be in any format: list of figures, file directory structure, a social network graph or any other data set. These data set are then transformed into a generic table format called *data tables*¹. The *source data* with different format are transformed into a collection of tables or matrices with rows, columns and values representing references to the tables to support more complex graph and tree structures. Then, *data tables* are mapped into *visual abstractions* having different visual features like layout, position, color and geometry of the data. This state of visualization is responsible for defining all kind of information required for data visualization. The content of *visual abstractions* is prepared and displayed in the interactive *view* having features like dragging an item, panning and zooming of display via process of view transformations.

4.3 XML Parser

XML (Extensible Markup Language) is a markup language that sets rules for encoding documents in machine-readable and human-readable form. XML is a popular format for sharing data on the Internet. It is used to emphasize simplicity, generality, and usability over the Internet. Web pages which changes their web content frequently such as news sites or blogs, often provide an XML feed so that external programs can keep abreast of content changes. This section explains how we parsed XML documents and use their data.

4.3.1 Parsers

There are two types of parsers available in Java. They differ in terms of speed, memory consumption and there ability to process large XML files.

4.3.1.1 SAX Parser

The SAX (Simple API for XML) parser is an event based XML parser which does not create any internal structure; instead it takes the occurrences of components of an input document as events, and the parsing works accordingly. Since a SAX parser does not create internal structure, it is much more space efficient in case of a big input document. A SAX parser serves client application only with pieces of document at any given time and is therefore faster and uses less memory than a DOM parser. The SAX parser has less functionality, so users have to take care of creating and maintaining data structures themselves.

¹the internal representation of data as it should be visualized

4.3.1.2 DOM Parser

A DOM parser creates a tree structure in memory from an input XML file and waits for a client requests for further operations. A DOM (Document Object Model) parser is rich in functionality, it creates a DOM tree in memory and can also access any part of the document repeatedly and can modify the DOM tree at any instance. Since DOM parser loads the whole document in memory it is inefficient when the document size is large.

After parsing XML files, the data is stored in relational database tables. The XML files that are considered in this work are large in size and since no I/O (Input/Output) operations are necessary in the XML files the SAX Parser is a suitable parser. So, we have used SAX Parser for parsing large XML files.

4.3.2 Analyze the Feed

The first step in parsing a document is to decide which of the fields are necessary. The parser extracts data from those fields (using tags) and ignores the rest.

4.3.3 Instantiate the Parser

The next step is to instantiate the Java parser and start the process.

Listing 4.1: Instantiate the SAX Parser

```
SAXParserFactory spf = SAXParserFactory.newInstance();
    SAXParser parser;
    parser = spf.newSAXParser();
```

4.3.4 Parse XML data

In our case the XML file contains unusual characters in their filenames. Before parsing the files with the SAX parser the file needs to be renamed. After renaming the files, the files are parsed with a suitable parser¹.

Listing 4.2: Renaming XML filename

```
// decode file to get rid of unusual characters
```

¹Five different parsers were created for different types of XML files. For more information about XML files, see section [Dataset](#).

```
String decodedFileURL = URLDecoder.decode(fileName,
    "UTF-8");
// get last part of decoded url which will be the new
// file name to be renamed
String newFileName =
    decodedFileURL.substring(decodedFileURL.lastIndexOf("?")
        + 1, decodedFileURL.length());
// rename the file and return the same file which needs
// to be parse
File fileToParse =
    FileDirectoryHandler.renameFile(baseDir, fileDir,
        fileName, newFileName);
```

SAX parser uses callback function (`org.xml.sax.helpers.DefaultHandler`) to inform clients of the XML document structure. We have extended `DefaultHandler` and override a few methods to achieve correct parsing. The override methods are:

- **startDocument()** and **endDocument()** – Method called at the start and end of an XML document.
- **startElement()** and **endElement()** – Method called at the start and end of a document element.
- **characters()** – Method called with the text contents in between the start and end tags of an XML document element.

The parsed XML files are stored into MySQL tables. The database will be used by the prototype implementation.

4.4 Database

We could have worked directly with XML files but XML databases are not as efficient and feasible as RDB (relational database), when it comes to highly structured data. RDBs are appropriate for storing items that are related in some sense. The normalization feature of RDBMS (Relational Database Management System) uses these relations to reduce the redundancy of data in the database, by placing repeated data in related tables. Relational databases are set-oriented i.e. it can specify and retrieve many records in a single DML (Data Manipulation Language) statement.

After parsing XML files the parsed data are inserted into relational database tables. RDBMS has been extensively used over the past decades. It supports many features and has been optimized a lot. There are lots of existing applications that are built on top of an RDBMS like social media site Facebook[26].

RDBMS can be queried by using descriptive query language. It can also be extended with stored procedures to provide extra functionality. Moreover, many RDBMS provide support for data mining. We have used MySQL client version: 5.1.44 for handling database of our system and Sequel Pro¹ for development and administration of our application's database. The MySQL database is an open source RDBMS which serves as server providing access to multiple databases.

4.5 prefuse

Prefuse is a Java based toolkit that is used for developing interactive information visualization applications. The architecture of prefuse is based upon the InfoVis Pipeline model described in section 4.2.2. Particularly, it provides data structure, layout and visual encoding techniques for creating graph, trees and tables. It supports dynamic queries, integrated search, database connectivity and animation. It is flexible, simple and has developer-friendly API (Application Programming Interfaces) for developing custom processing, interaction and rendering components. Prefuse uses different packages and classes within the process of transforming unstructured raw data to structured view.

The implementation of InfoVis Pipeline on prefuse is show in Figure 4.2:

4.5.1 Loading Data

The *prefuse.data* package includes memory efficient data structures like *Table*, *Graph* and *Tree* for representing data. The *Table* rows are represented by *Tuple* interface whereas the entities of *Graph* and *Tree* are represented by *Nodes* and *Edges*. The *Graph* and *Tree* classes are implemented using *Table* instance to store *node* and *edges* data. The *prefuse.data.io* package provides classes for reading and writing *Table*, *Graph* and *Tree* from formatted files. Similarly, *prefuse.data.io.sql* is responsible for issuing queries to SQL (Structured Query Language) database which returns a result within a prefuse *Table*. The classes supporting data queries can be found in the *prefuse.data.query* and *prefuse.data.search* packages.

¹For more information about Sequel Pro 1.0.1, see <http://www.sequelpro.com/>

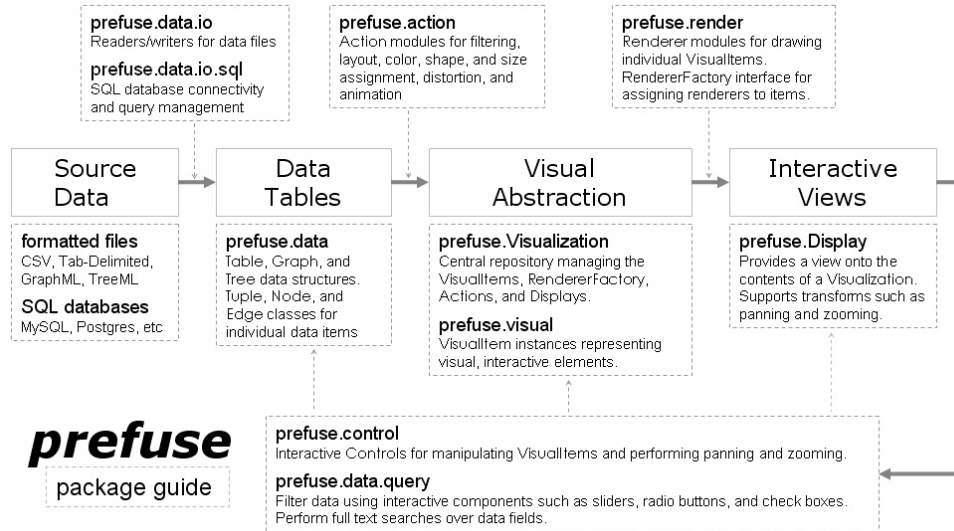


Figure 4.2: *The Prefuse package guide*. Guide describing the relation of different prefuse packages and classes to the infovis reference model [28].

4.5.2 Visual Abstraction

The *prefuse.visualization* class creates visual abstraction from provided data set. It creates new data structure by combining original data and visualization-specific data fields such as font, color, layout, position and geometry values. For any *Tuple*, *Node*, or *Edge* added to the Visualization, a corresponding *VisualItem*, *NodeItem*, or *EdgeItem* instance is created. These classes, found in the *prefuse.visual* package extend the *Tuple* interface and provide access to both the visual attributes and the primary data.

The visual mappings are performed through *Action* classes located in the *prefuse.action* package. These classes manipulate properties of *VisualItems* like visibility, position, size and color. Subclasses of *Actions* are typically layouts, animators or highlighters. A number of prebuilt implementations can be found in the *prefuse.action* package and its sub-packages.

The *prefuse.render* package contains *Renderers*, which read the properties of *VisualItems* and do the actual painting based upon available data. They are used for drawing various shapes, labels, and images out of the box and can be combined with any *VisualItems* as our desire.

4.5.3 Views and User Interaction

The *Display* draws interactive visible items by using appropriate *Renderers* which can be panned, zoomed, and rotated as our desire. Multiple *Display* instances can be associated with a single Visualization, enabling multi-view configurations, including Detail views within an Overview panel.

The *prefuse.controls* package provides *Controls* that can be registered with *Display* instances and be used to process mouse and keyboard actions performed by the user. Prefuse provides pre-built *Controls* for selecting items, dragging items around, and panning, zooming, and rotating the *Display*. It's easy to write custom *Control* implementations by extending the *ControlAdapter* class.

4.6 System

The methods described in Chapter 3 are in a prototype that is used to detect users with multiple aliases in a discussion board. The prototype is implemented in Java and uses prefuse for visualization . The graphical user interface consists of five coordinated panels as shown in Figure 4.3.

1. Display

This is the main frame of the application. The display visualize users and their friends network using a social network. The users are represented as nodes and their friend relations as edges. Depending on the structure and size of the network, different type of visualization layout algorithm can be used. In this case we are using a small subset of the original data set and therefore the force directed layout algorithm [30] is used for visualizing the network. A force-directed layout algorithm simulates a physical system where the edges acting as springs and nodes as repelling objects. The repelling forces of nodes push the nodes apart while the edges hold the graph together. By iteratively running the algorithm, after certain interval of time the graph will come to an equilibrium state in which the nodes do not change their position further more. Since the running time is $O(N \log N)$ or $O(E)$, whichever is greatest, where N is the number of nodes and E the number of edges, it is not usable for graphs larger than a few hundred nodes.

The display can be used to choose a list of users that should be considered for equality test. When a node is selected, the respective node as well as the nodes in its friend network changes their color. The nodes are selected

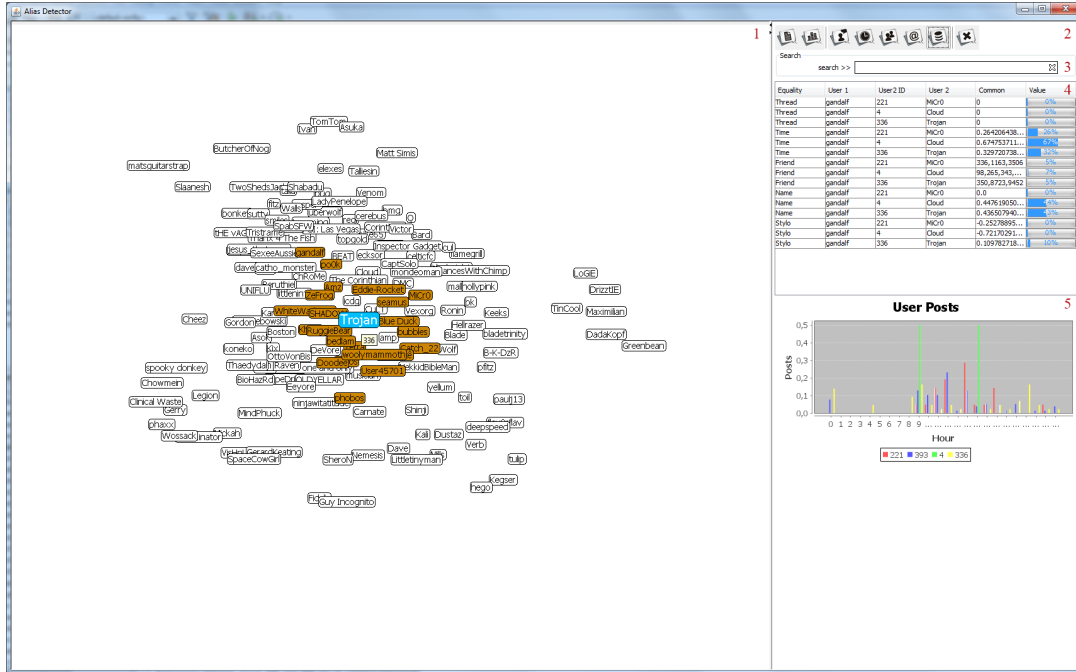


Figure 4.3: Screen shot of the alias matching prototype implementation.

with mouse left-button click and after selecting nodes of graph it is possible to select any one of equality measure from mouse right click pop up menu to find out the equality percentage between the selected users.

The network can be panned and zoomed. Panning the graph can be done by hold down the left mouse-button without pointing the mouse pointer over any node or edge. Similarly, the graph can be zoomed in and zoomed out by scrolling the scroll-wheel of mouse. The network can be centrally panned and zoomed by pressing the mouse-wheel.

2. Tool Bar

The tool bar panel consists set of icons having the functionality to perform five different equality test. After selecting a set of nodes from the display, one of tool bar icons for equality check can be used.

3. Search

The search panel helps to find users in the graph. As an individual types into search field, the graph nodes matching each of typed character are

highlighted with different color then previous one. Furthermore, the search panel shows total count of nodes matched with the search.

4. Table

The table shows the results of the equality checks. It contains 5 columns; *Equality*, *User1*, *User2*, *Common* and *Value*. The *Equality* column shows the type of equality that user has selected from popup menu. *User1* and *User2* column shows users those were selected from graph for doing equality test. *User1* shows the first user of the list, which is our test user whereas *User2* column gives the list of the users that we want to compare with. The *Common* column shows the result after doing the equality check between the test user and other users. Similarly, *Value* column shows the matched result between test user and other users in percentage.

5. Post Time Graph

The post times of user or users chosen by an individual user are plotted in graph and is shown in this panel. It gives us a view that when people tend to write messages during a certain time period of the day. The disparity in colors distinguish and represent each user in the graph.

Chapter 5

Experiments

In this chapter, we consider the performance and testing of our implementations of methods described in preceding chapters. First, the dataset used for our experiment is briefly explained. Then the test procedures are carried out to validate and verify our thesis specification are formulated.

5.1 Dataset

For the experiments a set of discussions forum data from the Irish forum site¹, structured in SIOC (Semantically-Interlinked Online Communities) [31] format is used. The data is in total around 9 million documents and takes about 50 gigabytes of disk space, but only data from one year is used in our experiments. The forum contains different types of data: site, forums, users and FOAF (Friend of a Friend) documents in RDF/XML data format. These dataset represents RDF (Resource Description Framework) graph data in XML syntax. The data set was made available for research purpose by ICWSM².

5.1.1 SIOC Architecture

The SIOC documents start with the top-level site of document which links to user and to FOAF files as well as to top-level forums. A user document contains a link to a FOAF file with information about the person that owns the user account. The forum dataset link to sub-forums and threads, which finally link to individual posts. The posts link to each other based on replying and quoting. The FOAF files also link to each other, describing a social network based on the users' friend

¹<https://www.boards.ie/>

²The dataset can be accessed from <http://icwsm.cs.umbc.edu/> by a registered user

lists. The architecture describing SIOC forum data is shown in Figure: 5.1.

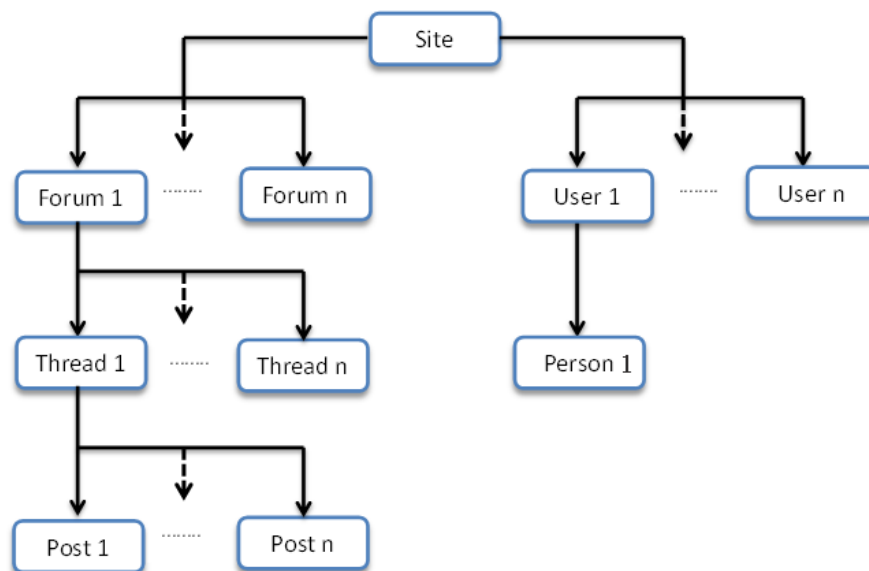


Figure 5.1: *The SIOC architecture*. Top-down link between the documents in SIOC format.

Some of the important component of RDF files are described below:

- **foaf:Document**

This section of RDF file gives general information about the title and description of the document. The **primaryTopic** tag gives a SIOC concept about site, forum, user and thread which is further described in the document later.

Listing 5.1: Sample of foaf:Document

```

<foaf:Document rdf:about="">
  <dc:title>SIOC profile for "boards.ie"</dc:title>
  <dc:description>A SIOC profile describes the structure and
    contents of a community site (e.g., weblog) in a
    machine processable form. For more information refer to
    the
  <a href="http://rdfs.org/sioc">SIOC project
    page</a></dc:description>

```

```

<foaf:primaryTopic
  rdf:resource="http://boards.ie/vbulletin/showthread.php?t=14661"/>
<admin:generatorAgent
  rdf:resource="http://wiki.sioc-project.org/index.php/
  PHPEXportAPI?version=1.01"/>
<admin:generatorAgent
  rdf:resource="http://sw.deri.org/svn/sw/2005/08/sioc/vbulletin"/>
</foaf:Document>

```

- **foaf:Forum**

The section describes about the list of threads it encloses and provides links to those threads. The **sioc:parent_of** and **sioc:has_parent** tags describe whether the forum is main forum or sub-forum of another one.

Listing 5.2: Sample of foaf:Forum

```

<sioc:MessageBoard
  rdf:about="http://boards.ie/vbulletin/forumdisplay.php?f=7">
  <rdf:type rdf:resource="http://rdfs.org/sioc/ns#Forum" />
  <sioc:link
    rdf:resource="http://boards.ie/vbulletin/forumdisplay.php?f=7"/>
  <dc:title>After Hours</dc:title>
  <dc:description>All things non-work
    related.</dc:description>
  <sioc:has_parent>
    <sioc:Forum
      rdf:about="http://boards.ie/vbulletin/forumdisplay.php?f=3">
      <rdfs:seeAlso
        rdf:resource="http://boards.ie/vbulletin/sioc.php?
        sioc_type=forum&sioc_id=3"/>
      </sioc:Forum>
    </sioc:has_parent>
  <sioc:parent_of>
    <sioc:Thread
      rdf:about="http://boards.ie/vbulletin/showthread.php?t=146">
      <rdfs:seeAlso
        rdf:resource="http://boards.ie/vbulletin/sioc.php?
        sioc_type=thread&sioc_id=146"/>
      </sioc:Thread>
    </sioc:parent_of>
  [... many more threads ...]

```

```

<rdfs:seeAlso
  rdf:resource="http://boards.ie/vbulletin/sioc.php?
sioc_type=forum&sioc_id=7&page=9"/>
</sioc:MessageBoard>

```

The above example shows that, the forum contains threads "146" and is sub-froum of forum "3".

- **foaf:Thread**

The thread data shows list of posts and the parent forum of respective thread in the discussion forum. The **sioc:container_of** gives the link of posts for the respective thread and **sioc:has_parent** gives the line for parent forum.

Listing 5.3: Sample of foaf:Thread

```

<sioc:Thread
  rdf:about="http://boards.ie/vbulletin/showthread.php?t=12648">
  <sioc:link
    rdf:resource="http://boards.ie/vbulletin/showthread.php?t=12648"/>
  <sioc:num_views>72</sioc:num_views>
  <dc:title>Hasta La Vista Internet</dc:title>
  <dcterms:created>1998-11-16T20:46:00</dcterms:created>
  <sioc:has_parent>
    <sioc:Forum
      rdf:about="http://boards.ie/vbulletin/forumdisplay.php?f=13">
      <rdfs:seeAlso
        rdf:resource="http://boards.ie/vbulletin/sioc.php?
sioc_type=forum&sioc_id=13"/>
      </sioc:Forum>
    </sioc:has_parent>
  <sioc:container_of>
    <sioc:Post
      rdf:about="http://boards.ie/vbulletin/showpost.php?p=93397">
      <rdfs:seeAlso
        rdf:resource="http://boards.ie/vbulletin/sioc.php?
sioc_type=post&sioc_id=93397"/>
      <sioc:next_by_date
        rdf:resource="http://boards.ie/vbulletin/
showpost.php?p=93398"/>
      </sioc:Post>
    </sioc:container_of>
  </sioc:Thread>

```

```

    <rdfs:seeAlso
      rdf:resource="http://boards.ie/vbulletin/sioc.php?
        sioc_type=thread&sioc_id=12648&page=2"/>
  </sioc:Thread>

```

- **foaf:Post**

This type of XML file gives information about the post posted by user in discussion forum. It contains information about the creator of post, link to the creator's user account page and link to a **foaf** file. It also contains images and date and time of post creation.

Listing 5.4: Sample of foaf:Post

```

<sioc:BoardPost
  rdf:about="http://boards.ie/vbulletin/showpost.php?p=23318">
  <rdf:type rdf:resource="http://rdfs.org/sioc/ns#Post" />
  <dc:title>So like why was I first???


---



```

- **foaf:User**

The **sioc:Person** holds an online account which is the **sioc:User**. So, a user is not a person, it is just an account of a person who can have more than one user account on discussion forum. A person uses **sioc:User** account to create and reply on posts. The **sioc:User** account provides link to the Foaf profile of user. The user may be "Registered User", "Moderator", "Banned User" or "Administrator".

Listing 5.5: Sample of foaf:User

```
<foaf:Person
  rdf:about="http://boards.ie/vbulletin/foaf.php?u=1#person">
  <rdfs:seeAlso
    rdf:resource="http://boards.ie/vbulletin/foaf.php?u=1"/>
  <foaf:holdsAccount>
    <sioc:User
      rdf:about="http://boards.ie/vbulletin/member.php?u=1#user">
      <sioc:name>Cloud-Warrior</sioc:name>
      <sioc:has_function>
        <sioc:Role>
          <sioc:name>Registered User</sioc:name>
        </sioc:Role>
      </sioc:has_function>
    </sioc:User>
  </foaf:holdsAccount>
</foaf:Person>
```

- **foaf:Person**

This XML file contains the list of person and their friends' network. The **foaf:knows** tag gives information about the friend of respective person.

Listing 5.6: Sample of foaf:Person

```
<foaf:Person
  rdf:about="http://boards.ie/vbulletin/foaf.php?u=4#person">
  <foaf:name>Cloud</foaf:name>
  <foaf:nick>Cloud</foaf:nick>
  <foaf:depiction
    rdf:resource="http://boards.ie/vbulletin/customavatars/avatar4_1.gif"/>
  <foaf:knows>
    <foaf:Person
      rdf:about="http://boards.ie/vbulletin/foaf.php?u=98#person">
      <foaf:nick>DeVore</foaf:nick>
```

```
<rdfs:seeAlso
    rdf:resource="http://boards.ie/vbulletin/foaf.php?u=98"/>
</foaf:Person>
</foaf:knows>
```

5.2 Tests

The prototype for alias detection has been implemented and tested to verify that the theory works in practice.

Software testing is any activity aimed at evaluating an attribute or capability of a program or system and determining that it meets its required results [12].

Software testing is about validating and verifying software specification but in this case we focus on to prove that our theories hold in practice. In short, internal mechanism of system working is ignored and we focus mainly on output generated against any input and execution of system. In this section, the experiments and experimental design are presented.

5.2.1 Machine specification

The test is carried out in the machine having following specification:

- Operating System : Mac OS X 10.8.2
- Processor: 2.66 GHz Intel Core 2 Duo
- Memory: 4 GB 1067 MHz
- Graphics: NVIDIA GeForce 320M 256 MB

5.2.2 Methodology

It is hard to find reasonable datasets that evaluate alias matching algorithms which we have presented here. We have not found any standard datasets where the ground truth is known and where all of features which we have suggested are available. So, we have limited our current experiment to only stylometric matching and time-based matching which can be used in our both cases: alter ego and multiple aliases case. The string-based matching and social network based matching have not been utilized in our experiments.

The following procedure is carried out in order to validate our theory.

-
1. The first 50 users are picked from database with the lowest user ID. We have taken only those users who has posted more than 60 messages in the discussion board.
 2. Each of the users are split into two users. Assume that we have picked user **UserX** from the database. We split **UserX** into **UserX_A** and **UserX_B**. The posts and time when the posts are written of **UserX** are also divided between **UserX_A** and **UserX_B**. The posts with odd numbers in the database are assigned to **User_A** and posts with even number messages are assigned to **User_B**.
 3. Now, the post and post time of **UserX_A** is compared with list of **UserX_B** users. After comparing users a list of rank is generated. We have given a rank from 1 to 50, i.e. for stylometric: higher the result higher the rank and for post time: lower the result higher the rank.
 4. After getting rank for both stylometric and post time between users, we have taken average of rank between stylometric and post time. And, the result with the users that have higher rank is placed in top and the users that have least rank are kept at bottom.
 5. The above test procedure is repeated for 1000 users, each time picking the users with lowest IDs.

5.3 Result and Discussion

To test how the matching techniques works in practice and to evaluate the prototype implementation, tests are carried out by following the procedure described in section 5.2.2. The experiments shows how much the results can be improved by combining stylometric matching and time profile matching. To do this, we have calculated stylometric matching and time profile matching for each user. The result of Top 1 and Top 3 matchings are taken in consideration for plotting the graph. The graph are shown in figure 5.2 and figure 5.3.

The test results from the experiments are shown in table 5.1 and table 5.2. The **No. of Users** column are the number of user with whom we are comparing our single test user. Similarly, the **Time** and **Stylometric** column give the percentage of users whose are ranked correctly to its alias by using time-profile based matching and stylometric matching respectively. The 4th column **Fusion** gives the percentage of matched user whose are ranked correctly to its alias by using combination of time-profile and stylometric matching.

No. of Users	Time	Stylometric	Fusion
50	60%	50%	70 %
100	61%	49%	66 %
150	48%	38%	62 %
200	47%	34%	59 %
250	45%	29%	55 %
300	44%	26%	53 %
350	42%	23%	50 %
400	40%	21%	47 %
450	38%	21%	46 %
500	38%	20%	47 %
550	36%	20%	45 %
600	36%	19%	45 %
650	36%	19%	44 %
700	35%	18%	44 %
750	34%	18%	44 %
800	34%	18%	44 %
850	34%	18%	43 %
900	33%	18%	43 %
950	33%	18%	44 %
1000	33%	17%	43 %

Table 5.1: Test result for Top 1

No. of Users	Time	Stylometric	Fusion
50	76%	66%	86 %

100	78%	65%	80 %
150	70%	54%	75 %
200	62%	48%	72 %
250	60%	42%	70 %
300	58%	38%	66 %
350	57%	36%	64 %
400	54%	32%	64 %
450	54%	32%	62 %
500	53%	32%	61 %
550	52%	31%	59 %
600	51%	31%	58 %
650	51%	30%	58 %
700	50%	29%	56 %
750	49%	29%	56 %
800	49%	29%	56 %
850	48%	29%	56 %
900	48%	29%	55 %
950	47%	29%	56 %
1000	47%	28%	56 %

Table 5.2: Test result for Top 3

In graphs we can see that as the number of users increases, the accuracy of matching decreases. It can also be seen that the time-based matching consistently performs better than the stylometric matching for both top-1 and top-3 ranking. The combination of the classifiers consistently perform better than the two classifiers individually. Studying the results in further detail, we can see that the correct alias is ranked first with over 70% accuracy when there is up to 50

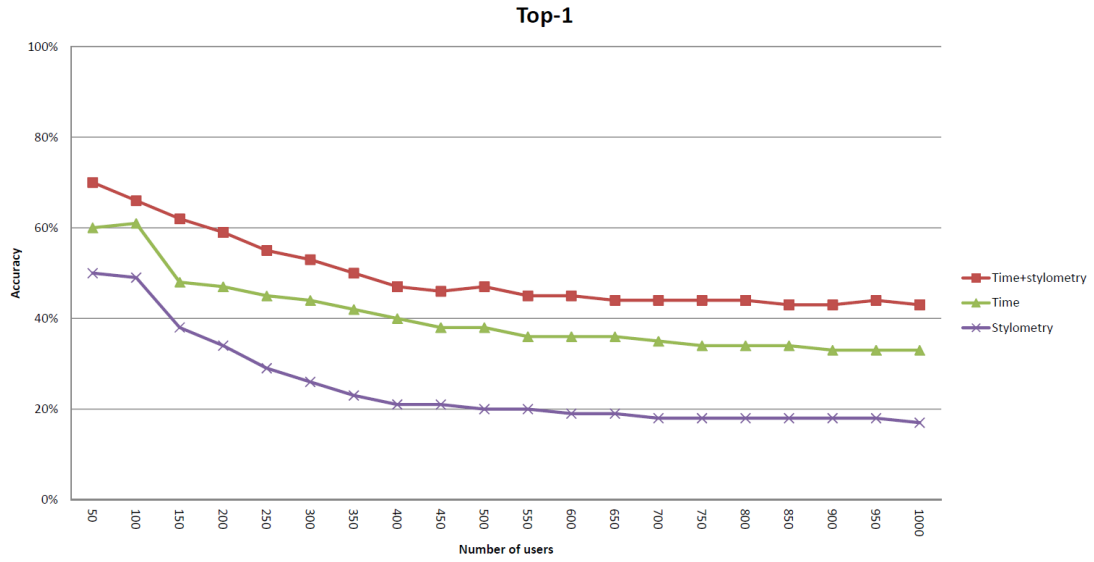


Figure 5.2: *Top 1*. Graph of users matched at top 1 while comparing with 1000 users.

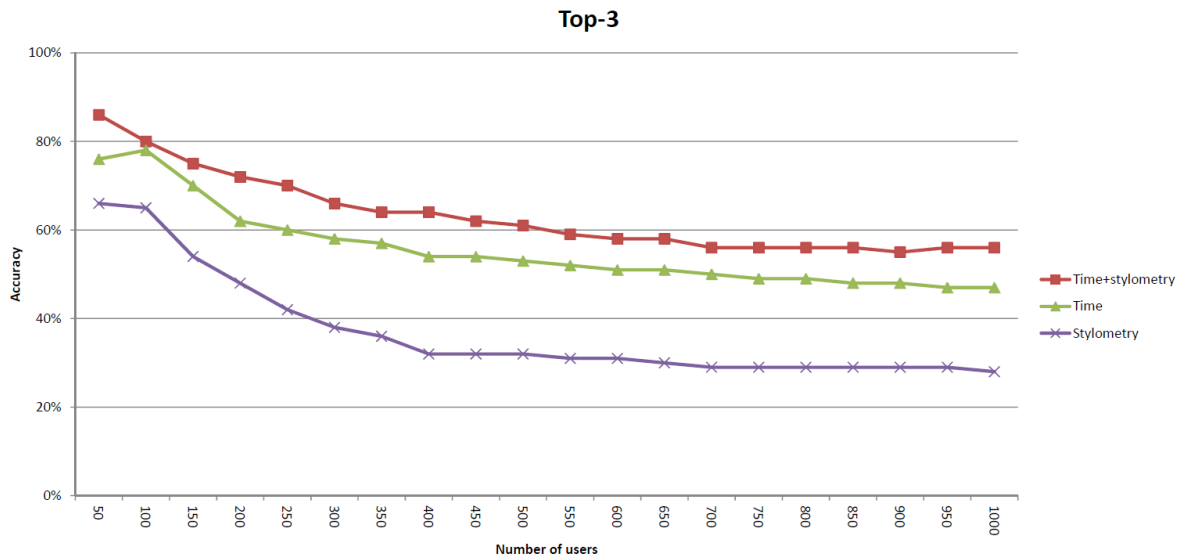


Figure 5.3: *Top 3*. Graph of users matched at top 3 while comparing with 1000 users.

users. The accuracy drops as the number of users is increased further, but it is still higher than 60% for up to 150 users and 55% for 250 users. The accuracy for the combined results thereafter becomes more stable, remaining at 43% for 1000 users. If we instead only demand that the correct user should be in the top-3, the combined classifiers yield accuracy over 80% for up to 100 users. The accuracy is still over 70% for up to 250 users. The accuracy is then slowly decreasing, resulting in an accuracy of 56% for 1000 users. The red line in the graph represents Time and Stylometric fusion result whereas green line and blue line represents time and stylometric respectively.

Chapter 6

Conclusion and Future Work

We have presented four different types of techniques for alias matching: string-based, stylometric-based, time profile-based, and social network-based matching. Several of those matching techniques have been proposed and used earlier, but there are no earlier attempts to use them in combination to find the use of multiple aliases within discussion forums. Moreover, we are not aware of any previous attempts to use time profile-based matching for alias matching purposes. In our experiments on forum data we have evaluated how accurate the stylometric and time-based techniques are on their own and in combination. The results suggest that our novel time-based matching technique yields better accuracy than stylometric matching, and that the combined result is always better than the individual classifiers alone. Furthermore, it is shown that quite good accuracy can be achieved also with limited amounts of posts and a large number of potential authors (e.g. 80% for 10 users and 55% for 250 users).

The presented results indicate that there is a possibility to use algorithms for detecting the use of multiple aliases on discussion forums using quite limited amounts of data. Although we have only tried the methods on data from a discussion forum, there are no reasons for why this kind of methods cannot be used also for blogs or other kinds of social media services. There is also a possibility that the same methods could be used for linking user accounts from various social media services to each other, although this is not as obvious. The use of this kind of techniques can also be generalized to other domains, such as record linkage of bibliographic data.

Although the presented results show an interesting potential to be used for counterterrorism purposes, they also raise privacy concerns since the same kind of techniques can be used also for more doubtful purposes. Applications that attack pseudonymity can pose a threat to the privacy of innocent people since the

linking of anonymous postings made by, e.g., a dissident in a totalitarian regime to other pieces of text where the author reveals his or her identity could have severe consequences. Also commercial companies may have an interest in such techniques due to advertising campaigns and similar applications.

For future work, we attempt to improve the implemented matching techniques by adding more features and applying feature reduction techniques such as principal component analysis. We also hope to find non-synthetic data on which all the implemented classifiers can be tested. It would also be interesting to test the methods on a large scale, where there are thousands of potential authors.

Chapter 7

Appendix A

7.1 Function words

a	eight	neither	them
able	eighth	nevertheless	themselves
aboard	either	next	then
about	enough	nine	thence
above	every	ninth	therefore
absent	everybody	no	these
according	everyone	nobody	they
accordingly	everything	none	third
across	except	nor	this
after	excepting	nothing	those
against	excluding	notwithstanding	though
ahead	failing	number	three
albeit	few	numbers	through
all	fewer	of	throughout
along	fifth	off	thru

alongside	first	on	thus
although	five	once	till
am	following	one	time
amid	for	onto	to
amidst	four	opposite	tons
among	fourth	or	top
amongst	from	other	toward
amount	front	ought	towards
an	given	our	two
and	good	ours	under
another	great	ourselves	underneath
anti	had	out	unless
any	half	outside	unlike
anybody	have	over	until
anyone	he	part	unto
anything	heaps	past	up
are	hence	pending	upon
around	her	per	us
as	hers	pertaining	used
aside	herself	place	various
astraddle	him	plenty	versus
astride	himself	plethora	via
at	his	plus	view
away	however	quantities	wanting
bar	i	quantity	was
barring	if	quarter	we
be	in	regarding	were

because	including	remainder	what
been	inside	respecting	whatever
before	instead	rest	when
behind	into	round	whenever
being	is	save	where
below	it	saving	whereas
beneath	its	second	wherever
beside	itself	seven	whether
besides	keeping	seventh	which
better	lack	several	whichever
between	less	shall	while
beyond	like	she	whilst
bit	little	should	who
both	loads	similar	whoever
but	lots	since	whole
by	majority	six	whom
can	many	sixth	whenever
certain	masses	so	whose
circa	may	some	will
close	me	somebody	with
concerning	might	someone	within
consequently	mine	something	without
considering	minority	spite	would
could	minus	such	yet
couple	more	ten	you
dare	most	tenth	your
deal	much	than	yours

despite	must	thanks	yourself
down	my	that	yourselves
due	myself	the	
during	near	their	
each	need	theirs	

References

- [1] Ahmed Abbasi and Hsinchun Chen. Writeprints: A stylometric approach to identity-level identification and similarity detection in cyberspace. *ACM Trans. Inf. Syst.*, 26(2):7:1–7:29, April 2008. ISSN 1046-8188. [11](#)
- [2] Stuart K. Card, Jock Mackinlay, and Ben Shneiderman. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann, 1999. [18](#)
- [3] W Cohen, P Ravikumar, and S Fienberg. A comparison of string metrics for matching names and records. *Communications*, 3:73–78, 2003. [9](#)
- [4] Jane Corbin. *Al-Qaeda: In Search of the Terror Network that Threatens the World*. Thunders Mouth Press / Nation Books, New York, 2002. [1](#)
- [5] J. Dahlin, F. Johansson, L. Kaati, C. Martenson, and P. Svenson. Combining entity matching techniques for detecting extremist behavior on discussion boards. In *Advances in Social Networks Analysis and Mining (ASONAM), 2012 IEEE/ACM International Conference on*, pages 850–857, August 2012. [4](#)
- [6] Joachim Diederich, Jörg Kindermann, Edda Leopold, and Gerhard Paass. Authorship attribution with support vector machines. *APPLIED INTELLIGENCE*, 19:2003, 2000. [12](#)
- [7] Ahmed K. Elmagarmid, Panagiotis G. Ipeirotis, and Vassilios S. Verykios. Duplicate record detection: A survey. *TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, 19(1):1–16, January 2007. [9](#)
- [8] Andreas Fellner. Treating temporal uncertainties of complex hierarchical data visually. Master’s thesis, Vienna University of Technology, 2006. [18](#)
- [9] David Goldschlag, Michael Reed, and Paul Syverson. Onion routing. *Commun. ACM*, 42(2):39–41, February 1999. ISSN 0001-0782. [1](#)

-
- [10] Yair Halevi. Part of speech tagging. In *Seminar in Natural Language Processing and Computational Linguistics (Prof. Nachum Dershowitz)*, 2006. [12](#)
- [11] Jeffrey Heer, Stuart K. Card, and James A. Landay. prefuse: a toolkit for interactive information visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '05, pages 421–430, 2005. ISBN 1-58113-998-5. [20](#)
- [12] William C. Hetzel. *The Complete Guide to Software Testing*. Wellesley, Mass, 2nd edition, 1988. [35](#)
- [13] Rohit R. Joshi, Rajesh V. Argiddi, and Sulabha S. Apte. Article: Author identification: An approach based on code feature metrics using decision trees. *International Journal of Computer Applications*, 66(4):34–39, March 2013. [12](#)
- [14] Patrick Juola. Authorship attribution. *Found. Trends Inf. Retr.*, 1(3):233–334, 2006. ISSN 1554-0669. [11](#)
- [15] Daniel Jurafsky and James H. Martin. *Speech and Language Processing*. Prentice Hall, 2nd edition, 2008. ISBN 0131873210. [12](#)
- [16] Hyunmo Kang, Lise Getoor, Ben Shneiderman, Mustafa Bilgic, and Louis Licamele. Interactive entity resolution in relational data: A visual analytic tool and its evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 14(5):999–1014, September 2008. ISSN 1077-2626. [16](#)
- [17] Daniel A. Keim, Florian Mansmann, Jorn Schneidewind, and Hartmut Ziegler. Challenges in visual data analysis. In *Proceedings of the conference on Information Visualization*, IV '06, pages 9–16, 2006. ISBN 0-7695-2602-0. [19](#)
- [18] Moshe Koppel, Navot Akiva, and Ido Dagan. Feature instability as a criterion for selecting potential style markers: Special topic section on computational analysis of style. *J. Am. Soc. Inf. Sci. Technol.*, 57(11):1519–1525, September 2006. ISSN 1532-2882. [12](#)
- [19] Ioannis Kourtis and Efstathios Stamatatos. Author identification using semi-supervised learning. *Notebook for PAN at CLEF 2011*, 2011. [12](#)
- [20] Jaro A. Matthew. *UNIMATCH: A Record Linkage System*. Bureau of the Census, Washington, 1978. [9](#)

-
- [21] Aviel D. Michael K., Reiter and Rubin. Anonymous web transactions with crowds. *Commun. ACM*, 42(2):32–48, February 1999. ISSN 0001-0782. 1
 - [22] Arvind Narayanan, Hristo Paskov, Neil Zhenqiang Gong, John Bethencourt, Emil Stefanov, Eui Chul Richard Shin, and Dawn Song. On the feasibility of internet-scale author identification. In *Proceedings of the 2012 IEEE Symposium on Security and Privacy*, SP '12, pages 300–314, 2012. 4, 11
 - [23] Jasmine Novak, Prabhakar Raghavan, and Andrew Tomkins. Anti-aliasing on the web. In *Proceedings of the 13th international conference on World Wide Web*, WWW '04, pages 30–39, 2004. ISBN 1-58113-844-X. 4
 - [24] P. Pantel. Modeling observation importance for alias detection. In *Proceedings of the DHS Conference on Partnerships in Homeland Security*, 2005. 4
 - [25] Elaine Pressman. Risk assessment decisions for violent political extremism 2009-02. *Public Safety Canada*, 2009. 1
 - [26] Prodromus. What database does facebook use?, 2011. URL <http://www.prodromus.com/2011/01/27/what-database-does-facebook-use>. 24
 - [27] M. Shaikh, N. Memon, and U.K. Wiil. Extended approximate string matching algorithms to detect name aliases. In *Intelligence and Security Informatics (ISI), 2011 IEEE International Conference on*, pages 216–219, july 2011. 8, 10
 - [28] Sourceforge.net. prefuse user’s manual, 2007. URL <http://prefuse.org/doc/manual/introduction/structure/>. 25
 - [29] Efstathios Stamatatos. A survey of modern authorship attribution methods. *Journal of the American Society for Information Science and Technology*, 60(3):538–556, 2009. ISSN 1532-2890. 11
 - [30] From Wikipedia the free encyclopedia. Force-based algorithms (graph drawing), 2012. URL [http://en.wikipedia.org/wiki/Force-based_algorithms_\(graph_drawing\)](http://en.wikipedia.org/wiki/Force-based_algorithms_(graph_drawing)). 26
 - [31] From Wikipedia the free encyclopedia. Semantically-interlinked online communities, 2012. URL http://en.wikipedia.org/wiki/Semantically-Interlinked_Online_Communities. 29
 - [32] James J. Thomas and Kristin A. Cook. Illuminating the path: The R&D agenda for visual analytics. *IEEE CS Press*, 2005. 18

-
- [33] Irma Veldman. Matching profiles from social network sites : Similarity calculations with social network support. Master's thesis, University of Twente, 2009. [9](#)
- [34] William E. Winkler and Yves Thibaudeau. An application of the fellegi-sunter model of record linkage to the 1990 u.s. decennial census. In *U.S. Decennial Census?. Technical report, US Bureau of the Census*, 1987. [9](#)
- [35] Aaron Y. Zelin and Richard Borow Fellow. The state of global jihad online. *New America Foundation*, 2013. [1](#)
- [36] Rong Zheng, Jiexun Li, Hsinchun Chen, and Zan Huang. A framework for authorship identification of online messages: Writing-style features and classification techniques. *J. Am. Soc. Inf. Sci. Technol.*, 57(3):378–393, February 2006. ISSN 1532-2882. [11](#)