Empirical Evaluation of Web Based Alias Extraction Techniques

P. Selvaperumal*, A. Suruliandi, Dhiliphan Rajkumar
Department of Computer Science and Engineering, Manonmaniam Sundaranar University, Tirunelveli.

* Corresponding author. Email: selvaperumal.p@gmail.com
Manuscript submitted January 10, 2015; accepted August 6, 2015.
doi: 10.17706/ijcee.2015.7.4.261-274

Abstract: Alias name extraction is the process of extracting surnames for the known name in the web. Alias names are useful in a wide range of processes like information retrieval, opinion mining and terrorist activities detection and so on. Owing to significance of the alias name extraction work, different techniques for web based alias name extraction and validation have been proposed. These methods use a range of techniques like Ranking SVM, Supervised classifiers, Lexical patterns and Latent semantic analysis etc. In this scenario, it is essential to compare these techniques and review its performances. In this work, different alias name extraction and validation techniques are compared. Experiments were conducted on Name alias data set. Experimental Results show that the feature based alias extraction technique that uses ranking SVM outclasses other alias extraction techniques.

Key words: Alias name extraction, information extraction, name disambiguation, web mining.

1. Introduction

Alias name extraction also called as surname extraction is the process of extracting surname for the given primary name in the web. In the web an entity may be referred by more than one name. Identifying all the names is a challenging task of knowledge extraction in the web. Accurate identification of aliases of a given person name is useful in various web related tasks such as information retrieval, sentiment analysis, personal name disambiguation, and relation extraction [1]. Identifying and extracting name and alias name documents are difficult because of many-to-many mapping between them [2]. A person may be referred by different names and different persons may share the same name. While former is called alias name (i.e., referential ambiguity), latter is known as personal name ambiguity (i.e., lexical ambiguity). Two types of alias names are prevalent in web pages. One is String variation of its primary name (for example Barack Obama can be called as Barrack obaamaa) and another is the alias name that doesn't share any string similarity with the primary name (for example Barry soetoro for Barack Obama) [3].

1.1. Related Work

Alias name extraction involves extraction of surname for the interested primary name from the web. Alias names associated with a primary name are available in the web in web pages URLs, Meta data, web page contents etc. Over the past few years, researchers started to propose different ways to extract alias information from the web. While string matching algorithms can detect only names which are string variant of the primary names [4], [5], co-reference resolution [6] can only detect different words that refers to an entity. A variant of co-reference resolution is cross document co-reference resolution, where different words across a set of documents that refers to an entity is extracted [7], [8]. All the words that refers to a name need not be alias name. Thus most of the words in the co-reference chains extracted in the process of cross document co-reference resolution are not alias names. Name-Entity recognition recognises all the names in a document and is different from the alias name recognition process. In entity matching problem, two entities are compared and matched if there exists a similarity between two [9]. For instance, Barack Obama, B.obama and Obama are different variations of the same entity. Although different variations of the same entity are alias names, but most alias names do not share any string similarity between them.

Table 1 shows a list of various web based alias name extraction techniques. It is evident from the Table 1 that most of the method considers alias validation as a classification process and they employs conventional classifiers to classify whether a name is alias name of a known name or not. Bollegala *et al.*, [1] first extracted lexical patterns for English personal and location names. They then extracted candidate aliases from using the extracted lexical patterns. They trained a ranking SVM using 23 features like co-occurrence measures, page count based association measure and frequency of lexical pattern etc. They then validated candidate aliases using the trained ranking SVM. T. Hokama, [10] extracted candidate alias names by the query "koto name" where koto in Japanese refers to “be called". Prefix and suffix patterns were then extracted and are used to validate candidate alias names. Vinaybhat, [11] showed that Latent semantic analysis performs poor in alias identification in various circumstances and they rectified it by proposed a two stage algorithm based on LSA and showed that the algorithms performs better than using conventional LSA. (Paul Hsiung, 2005) proposed an algorithm that leverages on orthographic and semantic information to find alias names in link data sets. Using a supervised training model, they trained a classifier that classifies whether two names in a data set is alias or not. Ning an, [12] used Lingpipe to extract candidate aliases. Using subset based comparison method, they grouped entities and aliases. They used logistical regression classifier to get probability value representing how likely they are aliases. E. Sapena, [13] compared the use of lexico-orthographic similarity functions and classifiers for alias classification problem. On one hand they used Character, Token, structural and semantic similarity functions and on the other hand Hill climbing and SVM classifiers for alias classification. Patrick, [14] used Point-wise mutual information for evaluating features significance. He used cosine similarity measure for measuring the similarity between name and potential aliases. Ralf H¨ olzer, [15] and Meijuan Yin, [16] used different techniques to find aliases in emails. While the former constructs a social network for finding aliases, latter finds aliases between email address section and body section. Y. Meijuan, [17] proposed a novel alias ranking technique based on email communication relation analysis and clustering. They used alias breadth, alias frequency and the importance of correspondence name along with clustering of similar aliases using a novel agglomerative alias hierarchical clustering algorithm to evaluate authority of alias names. Tarique Anwar, [18] constructed a name graph for finding aliases of a name. Given a name, it retrieves web pages relevant to name and constructs a graph. They then use clustering to disambiguate web pages. Relevant clusters are mined for alias names. They also used Associative score, similarity score and Dice score for web count based alias name extraction [19].

1.2. Motivation and Justification of the Proposed Work

Danushka, [1] showed that usage of alias names along with primary names increases the information retrievals accuracy. Tarique Anwar, [18] used alias names for suspect tracking on the web. Extracted alias names from the web have numerous applications. While the problem of disambiguating different persons who are sharing the same name is researched much, extracting same person having multiple names has received little attention. It is therefore essential to do an empirical evaluation on robustness of different web based alias extraction techniques. Justified by this, in this work, performance of different web based alias extraction techniques are compared and its results are discussed.

Table 1. Different Web Based Alias Name Extraction Techniques

Researcher	**Features**	**Method**	**Advantages**	**Limitations**
Tomoko Hokoma	Words in the web documents	Prefix and suffix strings of primary name are extracted. alias names are evaluated based on extracted strings.	Simple to implement and no training data needed.	Not suitable for all the languages. Can detect only alias names that are in standard lexical patterns.
Bollegala *et al.*	Statistical similarity features, Web based features	Ranking SVM	Robust method to identify alias names that are in standard lexical pattern.	Fails to identify syntactically similar aliases (String variant, abbreviation aliases) and aliases that are not in standard lexical patterns.
Vinaybhat *et al.*	Words in the document	Latent semantic Analysis	Identifies both syntactically and semantically similar aliases.	Less robust i.e., retrieves more false aliases than other methods
Paul Hsiung	Orthographic and Semantic features	Classifier(SVM and Logistical regression)	Identifies both syntactically and semantically similar aliases.	Suitable only for link data set. Creating link data set is laborious for each and every name. Training the classifier is costly.
Ning an	Co-occurancerelevance, Social relevance and Alias relevance.	Classifier(Logistical regression)	Identifies both syntactically and semantically similar aliases. Robust method for finding alias names.	Training the classifier is costly.
Tariq anwar	associative, Similarity and Co-occurrence Scores	Ranking score calculated by utilizing the feature	Identifies alias that are in standard lexical patterns	Fails to identify syntactically similar aliases (String variant, abbreviation aliases) and aliases that are not in standard lexical patterns.
EmilliSapena *et al.*	Character, Token, Structural and Semantic features	SVM and Hill climbing	Can detect both syntactically and semantically similar aliases	Training the classifier is costly.

1.3. Outline of the Proposed Approach

In this paper, efficiency of the different alias name extraction techniques was evaluated. Thirty Name and alias pairs were taken as data set. Different alias name extraction techniques were evaluated for the primary names in the data set. Finally, the efficiency of different techniques in extracting alias names were evaluated in terms of precision, recall and f score.

1.4. Organization of the Paper

The remaining part of the paper is organized as follows: Section 2 describes the various Alias name extraction techniques in detail. In Section 3, Data set and performance metrics involved in the work are discussed. Section 4 deals with experiment results and performance analysis. Section 5 concludes the paper.

2. Alias Name Extraction Methods

2.1. Prefix and Suffix String Based Method

Tomoko Hokama, [10] used a Japanese term pronounced "koto" (be called) to extract prefix and suffix strings for the known primary name from the Japanese web pages. The method is based on the notion that prefix string and suffix string of primary name and alias names may be same. This method involves three steps. First, extraction of candidate alias name, then prefixes and suffix pattern extraction and finally alias name evaluation. Fig. 1 shows the process involved in extracting alias names.

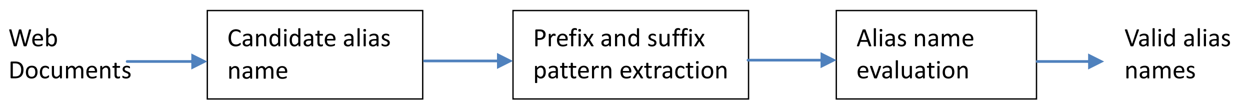

Fig. 1. Prefix and suffix string based process of alias name extraction and validation.

Hokomo *et al.*, used Japanese name data and Japanese web pages for extracting aliases. It also puts a caveat that such algorithm can work well only for Japanese web pages and Japanese name data set. The following algorithm explains the process involved in alias extraction from the web.

Step 1: Perform "AKA Name" search in search engine.
Step 2: Extract the Strings precede to "AKA Name" which are Noun.
Step 3: Eliminate the strings that occurs very few times.
Step 4: Perform"Name AND obj"
Step 5: Extract the Strings adjacent to the "Name AND obj"
Step 6: Calculate weight of prefix, suffix pattern
Step 7: Select prefix and suffix pattern whose weight exceeds the threshold
Step 8: Evaluate the Mnemonic name by

- Set initial score cand to 0.
- Perform "Prefix Alias" for prefix pattern
- Perform "Alias Suffix" for Suffix pattern.
- Find the total number of web pages.
- Total = Total + Prefix (or suffix) pattern weight.

Step 9: Select top k candidates as Alias names for the person.

- Weighs of prefix and suffix are calculated using (1) and (2).

$$\text{weight(Prefix)} = \frac{\text{"Prefix Name"}}{\text{"Prefix"}} \tag{1}$$

$$\text{weight(suffix)} = \frac{\text{"Name suffix"}}{\text{"suffix"}} \tag{2}$$

where "prefix Name" is the number of results returned by the query "prefix name". "Prefix is the number of results returned by the query "prefix".Final score for every alias name can be calculated as follows.

Score (alias) = (results for the Query "prefix alias" * weight (prefix/suffix)).

This method is more suitable for Japanese names, as their language has the pattern of having prefix and suffix of primary name and alias will be similar.

2.2. Feature Based Method

Danushka Bollegala, [1] proposed an alias extraction technique similar to Hokama *et al*. They used Japanese web pages URL and anchor texts for finding aliases. Their method considered co-occurrence of name and alias name in both URL and web pages. Their method uses 23 Features extracted from URL and web data. Table 2 shows the list of features used by this method. These features were given as input for training the Ranking SVM [20]. The trained ranking SVM assigns a rank score between 0 to 1 for every name-alias pair. Aliases with the highest score are selected as valid alias for the given name. Fig. 2 shows the process involved in this method.

The proposed algorithm for extracting valid alias name for primary name is give below.

Step 1: Find the lexical pattern that occur between Name and Alias using queries in the Google like “Primary Name * Alias name” and “Alias name * Primary Name” like “aka”,be called, nicknamed etc.

Step 2: Find the lexico-syntactic pattern structure using extracted lexical patterns by using queries like “Primary name lexical pattern *” and “* lexical pattern Primary name”.

Step 3: Use top ‘n’ lexico-syntactic pattern structure to extract potential alias names by issuing queries like “Primary name lexical pattern *” and vice-versa.

Step 4: 23 features for primary name and potential alias names were taken as a feature for every name and alias pair.

Step 5: Feature vectors are normalized to the range between [0, 1].

Step 6: Feature vectors are given as input to a trained ranking SVM.

Step 7: Top scoring alias name for a primary name is considered as valid alias name.

Table 2. Features Used to Train a Ranking SVM for Validating Alias Names

Statistical Features	Co-occurrence frequency(CF), Term frequency-Inverse document frequency(tf-idf), Chi-squared measure(CS), Log-Likelihood ration (LLR), Pointwise Mutual Information, Hyper Geometric distribution(HG), cosine, overlap, and Dice(with and without weighing for hubs)
Web based Features	Web Dice, Web PMI , Conditional probability
Frequency based Feature	Frequency of lexical pattern

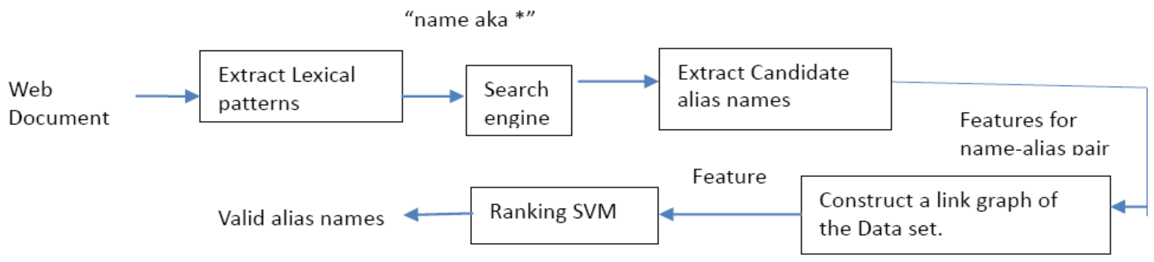
“name aka *”
Web Document
Extract Lexical patterns
Search engine
Extract Candidate alias names
Features for name-alias pair
Feature
Construct a link graph of the Data set.
Ranking SVM
Valid alias names

Fig. 2. Feature based process of alias name extraction and validation.

2.3. Two Stage Latent Semantic Analysis Based Method

Latent semantic indexing is used to find semantically similar words in a document collection. This means that words which do not have any string similarity but are similar in meaning can be found using latent semantic analysis. Vinaybhat, [3] showed that usage of latent semantic analysis for finding alias names in the web pages produces poor results and they tweaked the concept and proposed a two stage Latent semantic analysis algorithm for finding aliases in web pages. LSA attempts to project the document in to a lower dimensional space.

Similarity of two words can be found by (3)

$$sim(x,y) = \frac{\sum_{i=1}^{k} x_i y_i}{\sqrt{\sum_{i=1}^{k} x_2^i}\sqrt{\sum_{i=1}^{k} x_2^i}} \tag{3}$$

where, x_i, y_i are the words and k is their length produced by SVD.

Finding aliases using Latent semantic indexing involves the following steps

Step 1: build a term document matrix using the Document collection

Step 2: Compute Singular valued decomposition ($U\sum V^T$) and retain k largest singular values

Step 3: Determine all the semantically similar words

(Vinaybhat, 2004) proposed a two stage algorithm in which the first step is performing Latent semantic analysis and the semantically similar words are then passed to the second stage algorithm which considers words that are adjacent to alias names. Their second stage involves finding similarity between potential aliases. First, for every potential aliases, a document is added in the new document collection D. Thus if there are n number of potential aliases then n number of documents will be there in D. For every occurrence of alias in the document in D, words surrounding the alias are added to a new document S.

The algorithm works as follows:

Step 1: Build a term document matrix using the Document collection

Step 2: Compute Singular valued decomposition and retain k largest singular values

Step 3: Determine all the semantically similar words

Step 4: For every alias returned in Step 3, create a document D in the same name and add the text that occurs within a window (say '*n*' words) of web pages to document D.

Step 5: Build a term document matrix using the Document collection.

Step 6: Again run SVD in the new document collection and find semantically similar words.

Thus applying LSA twice increases the precision of aliases obtained. Fig. 3 shows the process involved in this method.

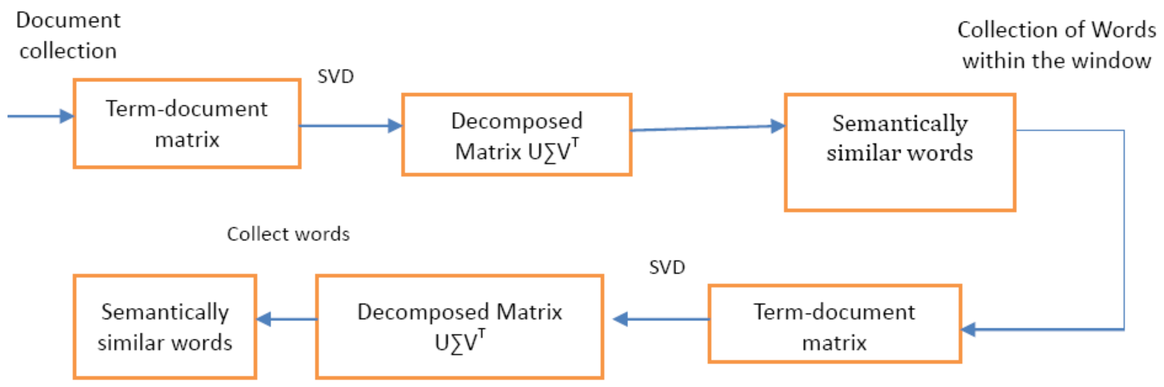

Fig. 3. Two-phase LSA based alias detection method.

2.4. Supervised Learning Method

Paul Hsiung, [11] proposed an alias extraction method for extracting aliases from link data set using supervised learning method. This method finds alias names from a link data set. A link data set consists of a set of names and links associated with that name. Links are names of a person, organization or any similar names associated to the name of a person. For finding orthographically non similar names (non-string variant names) they exploited local social network structure of these names. They used orthographic measures and semantic measures as features to train a classifier. Classifier was given few set of positive and negative examples (whether a pair of names is alias or not). Fig. 4 is the block diagram that shows the process involved in this method. Different measures used in the models are as follows.

Orthographic measure

String edit distance (SED)

Minimum number of insertions, deletion and substitutions required to transform one string to another.

Normalized string edit distance (NSED)

$$\mathrm{NSED}(s1, s2) = \frac{\mathrm{SED}(s1,s2)}{\max(\mathrm{length}(s1),\mathrm{length}(s2))} \tag{4}$$

where,

If s1, s2 are strings.

Discretized **string edit distance (DESD)**

If NSED is less than 0.7 then DESD is else 1.

Exponential **string edit distance(ESED)**

$$ESED(s1,\ s2) = \exp(SED(s1,\ s2)) \tag{5}$$

Semantic measures

Dot product

Number of occurrences of name and alias name with a common term.

Common friends

Friends that co-occur with both the names.

KL Distance

KL distance measures the similarity between two normalised friends lists.

$$\sum o_i \log\left(\frac{f_i}{s_i}\right) + p_i \log(\frac{f_i}{s_i}) \tag{6}$$

where f_i and s_i are i^{th} values in the normalized friends list of first and second name.

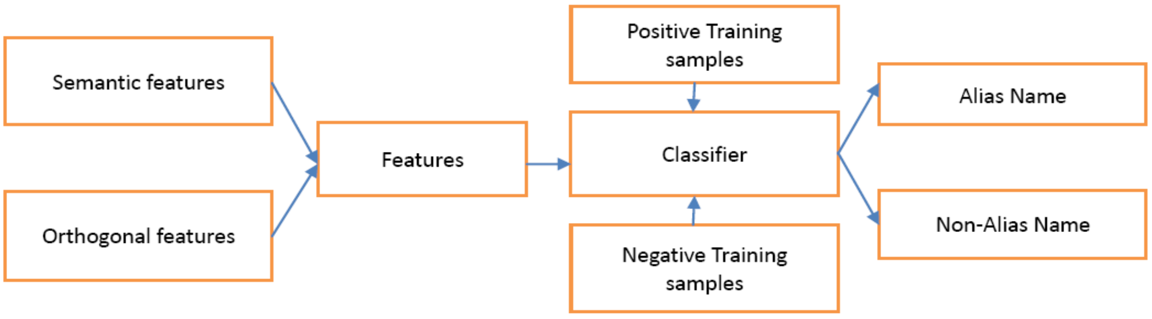

Fig. 4. Supervised alias detection method using semantic and orthogonal features.

3. Data set and Performance Metrics

3.1. Data set

Name-alias pairs were collected from web pages using the method said in D. Bollegala [1]. A set of queries like “Primary name aka *”, “* aka primary name” were given to Google to extract web snippets. For extracting candidate aliases, top 100 web snippet results returned by the Google were considered. This can be achieved by setting results per page value to 100 in Google's search settings. Extracted candidate aliases were then validated using different source of information. The reason for using query pattern like "Primary name aka *" is most of the alias names present in the web pages and URL's are present in this structure. Secondly, the alias name should be prevalent in the web. Alias name which are obsolete and which doesn't normally occur in web pages can't be extracted even by an efficient alias extraction technique. For this work, alias names for 30 names were extracted and validated and are taken as data set. Table 3 shows a partial list of name-alias pair extracted by using the aforementioned method.

Table 3. Partial List of Name Alias Data Set

S.No	Name	Alias Names
1.	Arnold schwarzenegger	the Governator, terminator, Arnie
2.	*Nelson Mandela,*	Madiba,Black Pimpernel
3.	Barack obama	barrysoetoro
4.	*Lance Armstrong*	*Juan Pelota ,The Boss*
5.	Warren Buffett	The Oracle of Omaha,Sage of *Omaha*
6.	Rajinikanth	Shivaji Rao Gaekwad, Thalaivar , superstar
7.	Sachin tendulkar	Sachin Ramesh Tendulkar(SRT), The Little Master , Master Blaster.
8.	Sonia Gandhi	*Antonia Maino,sonia*Maino,Madam
9.	Mother Teresa	Agnes Gonxha*Bojaxhiu,The Saint of the Gutters,*
10.	*Mahendra Singh Dhoni*	*Mahi, MS Dhoni,* Captain Cool

3.2. Performance Metrics

Three performance metrics used for evaluating effectiveness of alias extraction techniques are precision and recall. F-Score represents mean of precision and recall values. Precision, Recall and F-score are calculated using (7) (8) and in (9).

$$\text{Precision} = \frac{\text{Number of correct aliases extracted}}{\text{Total number of aliases extracted}} \tag{7}$$

$$\text{Recall} = \frac{\text{Number of correct aliases extracted}}{\text{Total number of actual aliases in the data set}} \tag{8}$$

$$\text{Fscore} = \frac{2*\text{Precision}*\text{Recall}}{\text{Precision}+\text{Recall}} \tag{9}$$

4. Experimental Results and Performance Analysis

4.1. Prefix and Suffix String Based Method

To implement this method, instead of "koto name", "aka name" was given since it is the most productive lexical pattern for extracting alias names (F-Score 0.335) D. Bollegala, [1] for every queries in this experiment, results of Google's top 100 web snippets were taken for consideration. Initially queries "aka name" were given to search engine and pos tagger was used to identify nouns that occurs left to the lexical pattern"aka name". Frequently occurred names before the "aka name" were taken as candidate alias names. Next, adjacent patterns were extracted as follows. Queries Name Object were given to extract list of prefix and suffix strings. Table 4 shows different object names chosen for this. These prefix and suffix strings were then weighed and heavy weighed prefix and suffix patterns were added to adjacent patterns list. Finally, alias names extracted previously were evaluated by scoring each alias names. Top scoring alias names were considered as valid aliases.

Table 4. Object Names Chosen for Various Personal Names

Name	Object
Arnold schwarzenegger	cinema
Nelson Mandela	South Africa
Rajinikanth	cinema
Sonia Gandhi	congress
Sachin tendulkar	cricket

Table 5 shows the performance of the method for the alias name data set. This method relies mainly on the notion that prefix and suffix strings before and after the name and alias name will be same which is does not always the case.

Table 5. Precision and Recall for Data Set by Tomoko Hokoma Method

Name	Precision	Recall
Arnoldschwarzenegger	0.33	0.66
Nelson Mandela	0.25	0.5
Rajinikanth	0.28	0.66
Sonia Gandhi	0.33	0.66
Sachin tendulkar	0.28	0.66

4.2. Feature Based Method

To implement the above method proposed by bollegala *et al.*, top 100 web snippets and web pages returned by Google search engine for name and alias name queries were taken as data set. The reason for taking web pages and not anchor texts as data set is based on the premises that more alias names can be found in web page content than in the anchor text of web pages. First, query "name" * "alias" is issued to Google for each name and patterns that replaces the * in top 100 web snippets returned by the Google were then extracted. (for example aka, also called etc.,). These patterns were then used for generating query like "name pattern *" and "* pattern name". These queries were issued to Google and F-scores were then calculated separately for the taken alias name data set. The procedure followed by bollegala *et al.* [1] was followed to find lexical patterns for name alias data set in the web. Table 6 shows the Lexical patterns extracted for name alias data set.

Table 6. F-Score Values Obtained for Various Lexical Patterns in the Web

Sl.No	Lexical Pattern	F-Score
2.	"Primary name aka *"	0.484
1.	"* aka Primary name"	0.369
3.	"Name alias *"	0.289
4.	"Name also known as *"	0.263
5.	"Name nickname *"	0.234

This method mainly relies on anchor text graph mined from the web. For implementing the above work, instead of anchor text graph, web structure graph was constructed. For example web page containing real name, its immediate inbound and outbound links were extracted and alias information from the immediate inbound and outbound were taken for constructing web graph structure.

Features were extracted from the data set and given for ranking SVM with linear kernel. Top ranking alias names were considered as valid alias names. Performance of the method for alias name data set is as below is given in the Table 7.

Table 7. Precision and Recall Values Using Dansuhkabollegala Method

S.No	Name	Precision	Recall
1.	Arnoldschwarzenegger	1	1
2.	*Nelson Mandela,*	*0.66*	*1*
3.	*Warren Buffett*	0.33	0.5
4.	Sachin tendulkar	0.4	0.66
5.	Rajinikanth	0.5	0.66

4.3. Two Stage Latent Semantic Analysis Based Method

Query "nelson madela" AND "madiba" AND "Black Pimpernel" return Web pages containing real and alias names of nelson mandela. Top 10 web pages were then taken for consideration. The reason for taking these web pages is web pages should have primary and alias names in order to find names that are interrelated (VinayBhat, 2004). After pre-processing, each document contained 400 words in average. Term document matrix was constructed for document collection and SVD was computed for each matrix ($U\sum V^T$).Then singular values (diagonal values of $\sum$) were sorted in non-increasing order and k largest singular values were set to 0. This matrix is called $\sum_{k.}$ Then truncate U and V^T according to the $\sum_k$ matrix. Alias names can be found by comparing elements corresponding rows of the U matrices. Here k is the dimension of to which the matrix is reduced.

Each row of Umatrix corresponds to a word and each column of V^T corresponds to a document. Similarity can also be found by the projection of words or documents in a k dimensional space as dots. In such case, a distance measure like Manhattan or Finding angle of the points from the origin gives the similarity measure.

Table 8. List of Alias Names Obtained for Nelson Mandela by Two Stage LSA Based Algorithm (t=2.5, k=20)

Madiba
Nelson
Mandiba
freedom
African
Thembu
ANC
Black Pimpernel
Sisulu
president

Table 8 shows list of alias names extracted using the two stage LSA method. After eliminating non-personal names the precision improves substantially.

Table 9. Precision and Recall Values for Names in the Data Set Using LSA

S.No	Name	Precision	Recall
1.	Arnoldschwarzenegger	0.4	1
2.	Nelson Mandela,	0.33	1
3.	Warren Buffett	0.4	1
4.	Sachin tendulkar	0.25	0.66
5.	Rajinikanth	0.4	0.66

It can be inferred from the Table 9 that this method retrieves true alias names but along with that, it also retrieves a number of false alias names. This is because not all the semantically related names are alias names for a name.

4.4. Supervised Learning Method

To implement the above method of finding aliases, query "name aka *" and "* aka name" were given to Google and top 50 web pages were extracted. From this, link data set was created for the names in the data set.

Table 10. Link Data Set Used for Finding Aliases from the Web

Name	Number of link names
Arnold schwarzenegger	43
Nelson Mandela,	27
Warren Buffett	19
Sachin tendulkar	41
Rajinikanth	36

Table 10 shows name and link data set constructed from the web. With the built link data set, features were extracted. Tables 11 and 12 shows the co-occurrence and Normalized co-occurrence statistics of some words with Sachin Tendulkar (Primary name) and his alias names.

Table 11. Number of Occurrences Co-occurrence Table for Sachin Tendulkar

	Sachin Tendulkar	Sachin Ramesh Tendulkar	The Little Master	Master Blaster
"India"	3,15,00,000	1,68,000	18,30,000	9,46,000
"cricket"	2,31,00,000	1,74,000	16,80,000	9,71,000
"match"	1,14,00,000	82,200	13,30,000	4,96,000
"Cinema"	29,10,000	88,600	27,60,000	7,43,000
"university"	45,60,000	1,39,000	5,35,000	2,39,000

Table 12. Normalized Co-occurrence Table for Sachin Tendulkar

	Sachin Tendulkar	Sachin Ramesh Tendulkar	The Little Master	Master Blaster
"India"	0.47	0.002	0.027	0.014
"cricket"	0.35	0.002	0.025	0.014
"match"	0.17	0.001	0.020	0.007

From Table 12 it is clear that primary name Sachin Tendulkar has a semantic relationship with words "India", "cricket" and "match" but not with "cinema" and "university". Thus there is a possibility that strongly associated words like "cricket" may be alias names of the primary name Sachin Tendulkar. For each name and alias names, orthographic and semantic features were extracted. Table 13 shows feature set for Sachin Tendulkar.

Table 13. Orthographic and Semantic Feature Values for Sachin Tendulkar

Name	Orthographic measure				Semantic measure			
	SED	NSED	DSED	ESED	DP	NDP	CF	KL
Sachin Ramesh Tendulkar	7	0.33	0	1.09	1024	0.001	1	2.25
The Little Master	16	0.76	1	8.88	11161	0.024	7	1.07
Master Blaster	15	0.71	1	3.26	2808	0.012	3	1.22

Positive and Negative examples were given as training set for SVM classifier. Results of the classifier are tabulated in Table 14.

Table 14. Classifier's Accuracy in Predicting Alias Names of Sachin Tendulkar

S.No	Name	P(Alias|measures)
1.	Sachin Ramesh Tendulkar	0.3521
2.	The Little Master	0.7632
3.	Master Blaster	0.5265
4.	Prince	0.0187

The probability of "Little master" being alias name for Sachin Tendulkar is high compare to prince. A threshold is chosen and the alias names whose probability values above the threshold was chosen as valid alias names.

Table 15. Precision and Recall Values Using Supervised Learning Method

S.No	Name	Precision	Recall
1.	Arnold schwarzenegger	1	1
2.	Nelson Mandela,	0.66	1
3.	Warren Buffett	0.66	1
4.	Sachin tendulkar	0.5	0.66
5.	Rajinikanth	0.75	0.66

Table 15 shows efficiency of the proposed method for various names. Similar to taken features, various other features can be used as feature for classification [21], [22]. This method works well for finding both string variant and non-string variant aliases.

4.5. Performance Analysis

Experiments have been conducted for 30 Name alias data set. Each alias extraction technique is implemented and the results were evaluated against the name alias pair in the data set. Table 16 shows the overall average precision and average recall values of different alias extraction techniques for all the 30 names. The results were averaged out and tabulated. Results were evaluated in terms of precision, Recall and F-score.

It is evident from the Table 16 that feature based alias extraction technique that used ranking SVM has an upper edge over the other alias extraction techniques. It is also observed that Latent semantic analysis based alias extraction method has good recall score and supervised alias extraction technique has a balanced precision and recall value. Although PrefixSuffix string based method does not view alias validation as a classification or ranking process, it performs reasonably well even for English names because English too support referring alias name in the form of "Alias name Primary name" like in the case of "Governor Arnold Schwarzenegger".

Table 16. Average Precision and Recall Values for Different Alias Extraction Methods

Sl.No	Method	Average Precision	Average Recall	F-Score
1.	PrefixSuffix string Method	0.60	0.64	0.61
2.	Feature based Method (Linear Kernel)	0.85	0.78	0.81
3.	Supervised Method	0.74	0.76	0.74
4.	Latent semantic analysis based Method	0.68	0.88	0.76

5. Conclusion and Future Scope

In this work, performance of various alias extraction and validation techniques is compared. These methods employ various techniques like ranking SVM, supervised classifier, and Latent semantic analysis etc. Experiments were conducted to compare their efficiencies using Name alias data set. Results show that feature based alias extraction method that uses ranking SVM outclasses all the other methods. Future works includes usage of alias names for improving accuracy of sentimental analysis in the tweets, improving classification accuracy in tweet classification and improving accuracy of web information retrieval system. Similar to alias names, finding previous names of a named entity like a person or a city is an interesting problem.

References

[1] Bollegala, D., Matsuo, Y., & Ishizuka, M. (June 2011). Automatic discovery of personal name aliases from the web. *IEEE Transactions on Knowledge and Data Engineering*, *23(6)*, 831-844.

[2] Wacholder, N., Ravin, Y., & Choi, M. (1997). Disambiguation of proper names in the text. *Proceedings of Fifth Conference on Applied Natural Language Processing* (pp. 202-208).

[3] Bhat, V., Oats, T., Shanbag, V., & Nicholas, C. (2004). Finding alias on the web using Latent semantic analysis. *Data and Knowledge Engineering*, *49(2)*, 129-143.

[4] Shaikh, M., Memon, N., & KoekWiil, U. (2011). Extended approximate string matching algorithms to detect name aliases. *Proceedings of Intelligence and Security Informatics IEEE International Conference* (pp. 216-219).

[5] Galvez, C., & Moya-Anegon, F. (2007). Approximate personal name-matching through finite-state graphs. *Journal of American Society for Information Science and Technology*, *58*, 1-17.

[6] Bagga, A., & Baldwin, B. (1998). Entity-based cross-document coreferencing using the vector space model. *Proceedings Int'l Conf.Computational Linguistics* (pp. 79-85).

[7] Ravin, Y., & Kazi, Z. (1999). Is Hillary Rodham Clinton the president? Disambiguating names across documents. *Proceedings of Coref. App '99, the Workshop on Coreference and its Applications* (pp. 9-16). Stroudsburg, USA.

[8] Lee, H., & Recasens, M. (2012). Joint entity and event coreference resolution across documents. *Proceedings of EMNLP-CoNLL '12* (pp. 489-500).

[9] Shen, W., Li, X., & Doan, A. (2005). Constraint-based entity matching. *Proceedings of AAAI* (pp. 862-867).

[10] Hokama, T., & Kitagawa, H. (2006), Extracting mnemonic names of people from the web. *Proceedings of Ninth Int'l Conf. Asian Digital Libraries (ICADL '06)* (pp. 121-130).

[11] Hsiung, P., Moore, A., Neill, D., & Schneider, J. (2005). Alias detection in Link data set. *Proceedings of International Conference on Intelligence Analysis*.

[12] Ning, A., Jiang, L., & Wang, J. (March 2014). Towards detecting of alias without string similarity. *Information Science Journal*, 89–100.

[13] Sapena, E., Padro, L., & Turmo, J. (2007). Alias assignment in Information extraction. *Spanish Society for Natural Language Processing*, *39*, 105-112.

[14] Pantel, P. (2006). Alias detection in malicious environments. *Proceedings of AAAI Symposium on Capturing and Using Patterns for Evidence Detection* (pp. 14–20).

[15] Holzer, R., Malin, B., & Sweeney, L. (2005). Email alias detection using social network analysis. *Proceedings of LinkKDD 2005*.

[16] Yin, M. J., & Luo, J. (April 2011). User name alias extraction in emails. MECS.

[17] Yin, M. J., *et al*. (2011). Ranking the authority of name aliases for email users. *Proceedings of International Conference on Multimedia Information Networking and Security* (pp. 425-430).

[18] Anwar, T., & Abulaish, M. (2014). Namesake alias mining on the Web and its role towards suspect tracking. *Information Sciences*, *276*, 123–145.

[19] Anwar, T., Abulaish, M., & Alghathbar, K. (August 2014). Web content mining for alias identification: A first step towards suspect tracking. *Proceedings of 9th IEEE Int'l Conf. on Intelligence and Security Informatics* (pp. 195–197).

[20] Joachims, T. (2002). Optimizing search engines using click through data. *Proceedings of ACM SIGKDD '02* (pp. 133-142).

[21] Zobel, J., & Dart, P. W. (1995). Finding approximate matches in large lexicons. *Software — Practice and Experience*, *25(3)*, 331–345.

[22] Zobel, J., & Dart, P. W. (1996). Phonetic string matching: Lessons from information retrieval. *Proceedings of the 19th International Conference on Research and Development in Information Retrieval* (pp. 166–172).

P. Selvaperumal has a strong passion in web mining, data mining, machine learning, NLP and artificial intelligence. He is currently pursuing his Ph.D. degree in computer engineering, at Manonmaniam Sundaranar University, India.

A. Suruliandi is a professor in the Department of Computer Engineering, Manonmaniam Sundaranar University, India. His areas interest includes knowledge mining, computer vision and image processing.

Dhiliphan Rajkumar has a strong passion in web mining, pattern recognition and social networking. He is currently pursuing his Ph.D. degree in computer engineering, at Manonmaniam Sundaranar University, India.