International Journal of Applied Engineering Research ISSN 0973-4562 Volume 10, Number 14 (2015) pp 34954-34961

A Web Robot for Extracting Personal Name Aliases

Thangaraj M

Associate Professor, Department of Computer Science
Madurai Kamaraj University, Madurai-625 021, India
Email-Id thangarajmku@yahoo.com

Sivagaminathan P G

Research Scholar, Department of Computer Science
Bharathiar University, Coimbatore-641 046, India
Email-Id sai.nathan@rocketmail.com

Abstract
Popular Personalities have multiple name aliases addressed in different documents of the web. An exact textual web identification of a person is useful in information retrieval, sentiment analysis, relation extraction and name disambiguation. Accurate alias extraction of a person would never be completed unless there must be a formal mechanism to collect various alias names dispersed over the entire web spider. This is a difficult task since web is an unstructured electronic medium, and also web content writers have the freedom to write names with different spell characters but with the same pronunciation. Hence this task is possible only by a focused crawler spanning the web corpora for a large amount of time. Proposed algorithm eliminates noisy aliases to a greater extent. In this paper, proposed approach has shown significant improvement in F-Score compared to existing alias extraction method.

Keywords: Regular Expression, Information Retrieval, Machine Learning, Lexical Ambiguity, Referential Ambiguity, Relevance Feedback, Precision, Recall, F-Score

I. INTRODUCTION

Searching about people [24] in the web has become the daily activity among Internet users. This searching task increased tremendously due to the mobility of smart phones, tablet PC etc. However, there is no provision in the search engine to extract aliases about a specific celebrity pertaining to a particular domain of expertise. For Instance, Cricket Celebrity '*Sachin Tendulkar*' is referred by different names like '*Little Master'*, '*Tendlya*', '*God of Cricket*' etc., We will not be able to retrieve complete information about a player unless the actual alias list is determined and ranked[37] using page counts and other accurate statistical measures. There are two issues which crops in researchers mind (1) Lexical Ambiguity - Different entities can share the same name. For example, Former Indian President *Dr.A.P.J Abdul Kalam* being called by different names such as '*Missile Man*', '*People's President of India*', '*Common Man*' and '*Wings of Fire*' (2) On the other hand, a single entity can be designated by multiple names called Referential ambiguity. An example would be '*Little Master*' is aliases for both '*Sachin Tendular'* and '*Sunil Gavaskar'* in the same domain. Another example, an alias name 'Badshah' refers to both '*Rajinikanth'* and '*Shah Rukh khan*'in the same domain of movies.

This problem is solved by semantic Meta data for entities and automatic extraction of Meta data [25] can accelerate the process of semantic annotation. For named entities, automatically extracted aliases can serve as a useful source of Meta data. These Meta data provides a means to disambiguate an entity.

Traditional information extraction normally takes advantage of Natural Language Processing [NLP] techniques such as lexicons and grammars, whereas web information extraction [31] usually applies machine learning and pattern mining techniques to exploit the syntactical patterns.

Machine learning is a subfield of computer science that evolved from the study of pattern recognition and computational learning theory in artificial intelligence. Machine learning induces the study and formulation of algorithms that can learn from and to make predictions on data. Applications of machine learning techniques in IR are information extraction, relevance feedback, information filtering, text clustering and classification.

This research is aiming towards developing an efficient web extraction system which retrieves correct aliases initially eliminating unwanted, misspelled or junk names. Using the mismatched output content preferably displayed in a GUI, a user can select misspelled words of the same phonetic sound to induce the machine to learn on its own. Through subsequent iterative search it produces better result. Further, this web robot research tool can be applied invariably in all domains.

Web extraction is an information retrieval task in an un-structured medium containing large number of personal name aliases pertaining to different celebrities conforming to various expertise area like scientist, social reformers, literature, engineering, politics, music, cinema, sports etc., Moreover, it adds further complexity due to the social networking sites like *facebook*, *twitter*, *linked In* and so on.

Alias extraction is basically a sub-set of web extraction with an intention to look only for similar, preceding, succeeding, adjacent consecutive set of words in a large set of corpora.

The term-weighting measure [3] is used to test the accuracy of retrieval in a statistical perspective. Two measures are normally used to assess the ability of a system to retrieve the relevant and to reject the non-relevant items of a collection which is known as Recall and Precision respectively.

II. RELATED WORK

The first alias extraction research [29] [33] [34] [36] discovered a supervised machine learning approach to determine set of correct aliases of a given name from the web. A set of known names and their aliases were stored in a training dataset to help algorithm to fetch the lexical words and to induce machine learning with known value. Every Information Retrieval task will definitely contain noisy or ad-hoc output, which needs to be discarded while retrieval. This depends on the ability of the algorithm to discriminate valid versus invalid outputs. The existing method architecture is given as in Figure 1. It encompasses two main components

(1) Pattern extraction, and Candidate extraction
(2) Candidate ranking algorithm.

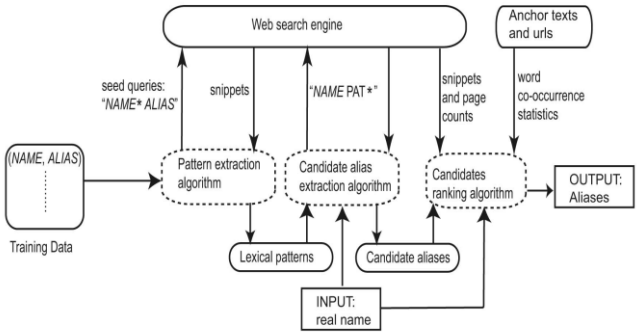

Figure 1- Outline of Previous Alias Extraction Method

2.1 Extracting Lexical Patterns from Snippets

Most of the modern search engines give a brief text snippet for each search result by selecting the text that appears in the web page in the proximity of the query given. It fetches few lines prior and few lines successive to the input pattern snippet contain the valuable information related to the local context of the query. Snippets convey useful semantic clues that are frequently used in web documents to mention a celebrity name in conjunction with alias name. It is obvious that when wildcard character is used in Google, "*Sachin Tendulkar * Tendlya*", query matches with one or more words in a snippet.

Another form of clue can be like aka (i.e also known as) which indicates the fact that *Sachin Tendulkar* also known as *Tendlya*. Further, proven ten clues for retrieving aliases are mentioned in the Table 1.

As a result, Figure 3 follows the shallow pattern extraction method to capture the different ways in which information about aliases of names are expressed on the web. Lexico-syntactic patterns have been used in numerous related tasks such as extracting hyponyms [6] and metonyms [14].

TABLE 1- Formal Patterns used in Existing Method

Slno	Formal Patterns
1	[name] nickname *
2	* was born [name]
3	[name] better known as *
4	[name] also known as *
5	[name] alias *
6	* aka [name]
7	[name] aka the *
8	* whose real name is [name]
9	* nee [name]
10	[name] aka *

657 matches - *Sachin Tendulkar*, is one of the most *famous* cricketer's of Indian cricket... *know* that for the Master Blaster, *also known* affectionately as *Tendlya*...

Figure 2 - Snippet returned by Google for the clue 'also known as' in Existing Method

Algorithm 3.1: EXTRACTPATTERNS(S)

comment: S is a set of (NAME, ALIAS) pairs

$P \leftarrow null$
for each $(NAME, ALIAS) \in S$
do { $D \leftarrow$ GetSnippets("$NAME * ALIAS$")
for each snippet $d \in D$
do $P \leftarrow P +$ CreatePattern(d)
return (P)

Figure 3- Given a set of (NAME, ALIAS) instances, extract Lexical Patterns

Given a set S of (Name, Alias) pairs are considered as input dataset for the *ExtractPatterns( )* function. This function returns a list of lexical patterns that frequently connect names and their aliases in web snippets. For each Name-Alias pair in S, the *GetSnippets( )* function downloads snippets from a web search engine for the query "Name * Alias". Then, from each snippet, the *CreatePattern ( )* function extracts the sequence of words that appear between the name and the alias.

Finally, the real name and alias in the snippet are replaced respectively by two variables [Name] and [Alias] to create patterns. The lexical pattern also includes words, symbols and punctuation markers. From the snippet *Figure* 2, patterns were extracted using the alias marker "[Name] aka *" by the algorithm Figure 3. To make the retrieval effective the same query is reversed to "* aka [Name]" to extract patterns in which alias precedes the name. Algorithm ignores stop words like 'a', 'an', 'the' during pattern extraction if present in the snippets. Thus, lexical patterns are extracted for each name.

2.2 Extracting Candidate Aliases

From the lexical patterns available on hand, candidate aliases are extracted for each name as described in procedure *ExtractCandidates( )* as in Figure 4. Finally, the *GetNgrams( )* function extracts continuous sequence of words (n-grams) from the beginning of the part that matches the wildcard operator. Experimentally, first five consecutive words are selected as candidate aliases. Later, it is trimmed off referring to the training dataset if required to bring meaningful output. For Instance, retrieved Snippet *Figure* 2 for the query "Sachin Tendulkar aka *", the algorithm Figure 4 extracts 'the master blaster' and "Tendlya" as output candidate aliases.

For efficiency reasons, this method restricted number of snippets downloaded by the function *GetSnippets( )* to a maximum of 100 in both algorithms Figure 3 and Figure 4.

2.3 Ranking of Candidates

Noisy candidate aliases may bring further problem in analyzing results hence that need to be removed. From these candidate aliases it is required to identify which are most likely to be correct aliases of a given name. These candidate aliases were modeled using ranking, which is most likely to be correct alias for a name that were given a higher rank. In order to bring accuracy in ranking, existing method used various ranking scores to measure the association between a name and a candidate alias using three different approaches listed below.

(1) Lexical pattern frequency
(2) Word co-occurrences in an anchor text graph
(3) Page counts on the web.

Algorithm 3.2: EXTRACTCANDIDATES($NAME, P$)

comment: P is the set of patterns

$C \leftarrow null$
for each pattern $p \in P$
do { $D \leftarrow$ GetSnippets("$NAME\ p\ *$")
for each snippet $d \in D$
do $C \leftarrow C +$ GetNgrams($d, NAME, p$)
return (C)

Figure 4- Given a name and a set of lexical patterns, extract candidate Aliases

III. PROPOSED WORK

3.1 The Method

The Proposed method is outlined in the below diagram automatic alias extraction abbreviated as *A2E* Figure 6. This web robot tool has been designed and developed under Visual Studio IDE VB.NET with backend as SQL Server. This web crawler named *A2E* requires a seed URL to initiate the crawling task. Online crawling were done for desired period of time, takes a copy of each and every page that is visited and stored in a binary offline storage for pattern matching, candidate alias extraction, and ranking. In the existing method, with the help of search engine returned snippets were stored in an array, undergoes pattern matching, candidate alias extraction and ranking.

Instead of storing snippets the entire web page is stored in a dataset and this is the major difference between proposed method and existing method. Another distinguishing feature is that regular expressions are used to fetch alias quickly, the complete co-occurrence of name-pattern pair along with preceding or succeeding alias names. Existing method remains in contact web till the end of alias extraction for all 50 names. Here, web crawling and alias extraction processes are independent of each other.

Once a seed URL is given, simultaneously ten threads are instantiated to crawl ten levels across ten different paths in a web graph. Hence, number of crawlers used in this method is ten. Moreover, the directory limit for crawl depth at any arbitrary site is also 10. It is experimented that proposed work crawls 6 pages per minute. Therefore, approximately it crosses 30 to 35 web pages in five minutes of time. This can be increased further by increasing the number of threads for a seed crawl. All sites are accessible to this method for crawling except Wikipedia web resources. There is no limitation on number of URL's from a website. Keyword search limit query is expanded using the predefined markers shown in Table 2.

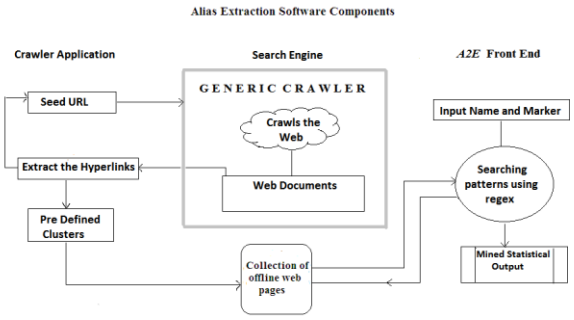

Figure 5 – Integration of Software Components in *A2E*

A2E comprises of three software components (1) Crawler Application (2) Generic Crawler/Search Engine (3) Automatic Alias Extraction Front End as in *Figure* 5. Crawler Application accepts a starting web address. Generic crawler identifies the next level of a page and feedback as new seed to crawler application. This feedback process is continued as long as there are no more links in the web.

Likewise, *A2E* search engine creates a maximum of ten different threads simultaneously. Both crawler application and generic search engine works in conjunction with each other till all the web pages are made visited or explicitly closed. It is obvious that this process seems to be a never ending task as web linked with enormous servers and it tends to grow rapidly due to dynamism. The uniqueness of this method is to speed up text processing by directly catching the matched text values in the web corpora.

3.1.1 Regular Expressions

Regular expressions are a pattern matching standard for string parsing and replacement. They are used on a wide range of platforms and programming environments. Regular Expression or *Regex* in short, are a way to match text with patterns. They are a powerful way to find and replace strings that take a defined format. They are language independent it

can be written both case sensitive and non-case sensitive. Hence, regular expression parses large amount of text documents quickly to identify a specific co-occurring string pattern. The custom designed regular expressions were used in this method for a set of formal patterns in Table 2. Dot net representation of Regular Expression [38][40] for one such pattern is given below

[NAME] aka * can be written in regular expression as

\b[NAME]\b.{0,30}?\b[aA].{0,3}?[kK].{0,3}?[aA]\b

The algorithm maintains a queue to keep track of all unvisited pages as per First In-First-Out principle. All visited pages are copied on to offline storage in a compressed form and marked for identification to avoid repetition. In this experiment, web crawler component were run by supplying seed URL for each name and thus had a offline storage collection of 34,126 pages of size 3.2GB. Generic crawler in this method allows unwanted web pages occupying huge offline storage space which becomes unavoidable.

TABLE 2- Formal Patterns used in Proposed Method

Slno	Formal Patterns
1	[name] aka*
2	[name] aka the *
3	[name] popularly known as *
4	[name] nickname *
5	[name] popularly called *
6	[name] better known as *
7	[name] also known as *
8	[name] alias *
9	*aka [name]
10	*was born [name]
11	*whose real name is [name]
12	*nee [name]
13	[name] popularly referred to as *

Once the web pages are crawled from all threads it must be classified in to four different clusters similar to the previous alias extraction method like *Sports*, *Science*, *Politics*, and *Movies*. For simplicity this experiment considers only *Sports* domain data for alias retrieval. Exactly fifty five sports celebrity name-alias details were given as training dataset to the tool. This training dataset initially induces supervised learning to the machine. Later it fetches new aliases automatically with little human intervention. Therefore, this method follows semi supervised machine learning technique from known to unknown in extracting aliases online. When an unusual alias is found for a person, the tool allows the user to view all noisy eliminated aliases. However, there can be few valid aliases in the eliminated list for the concerned person which can be manually selected by the user and updated for future reference. After updating training dataset, offline storage needs to be searched again for the second time to see whether the machine learning is fulfilled. It is experimented that software shows tremendous results after automating machine learning task.

Name disambiguation [24] [26] is another problem which is handled in the same way for accurate F-Score results. For Instance, there are two persons named '*Harbhajan Singh*' belonging to two different professions (1) Cricketer (2) Politician. It is obvious that though their names are same, their aliases need not be the same. The code must ensure correct alias belonging to the same name with specific domain. In general, nick names emerges in connection with the profession. Some time people recognize a person easily with nick names rather than their real name. Hence, this task is also successfully implemented for accurate precision, recall and F-Score.

Another implementation issue is that web content writers use different spell characters as per their convenience. It is the responsibility of the code to fetch all misspelled aliases conforming to same phonetic sound. For instance, Tennis player 'monica seles' alias name is 'monica'. Its common that few documents spell the same name differently like 'moanica', 'monika'etc., These problem were also handled efficiently in new implementation without changing the alias count.

3.2 Data Set

To train and evaluate new method, this work requires two datasets (1) storing crawled web pages in compressed binary form (2) Sport personal names dataset of 55 celebrities pertaining to cricket, chess, boxing, tennis, and badminton. Name aliases were manually collected after referring various information sources like Wikipedia and official home pages. Machine learning is achieved by interactive screens in the tool to train which alias name is really incorrect. The sports personal name training dataset is provided with adding a new alias in to a new row if it doesn't exist. For each name-alias pair stored in dataset it has four alternate spell characters in order to reduce the number of noisy aliases in the output, and thus increases precision to a higher level. During the co-occurrence retrieval if there is a mismatch with stored alias and read alias [39] from pages, it will get displayed on the list box. The user can add those read alias as an alternate alias for further references by the code. Since web is very dynamic, any new website may crop up with new alias for a person, in such cases *A2E* allows user to add as a new row in the training dataset. If the alias is added as a new row, the same is reflected in alias count and ultimately in recall computation. Otherwise, alias count is not incremented. For the same fifty five sport personal names and for each pattern the algorithm is executed, obtained the result as in Table 4.

3.3 Proposed Architecture

The Proposed Architecture is given in the Figure 5 and Figure 6. This method comprises of an offline webpage collection crawled from the web. There are two search engines used in this research tool (1) Generic Search Engine (2) Alias extraction Search Engine. Architecture is designed in such a way that both search engines should work independently. However, Alias extraction search engine functionality depends on the output of the former one. Name and search pattern are given as input to the alias extraction engine. Alias extraction engine comprises of modules for catching co-occurring lexical patterns '[name] aka' using regular

expressions for each pattern. Once the co-occurring patterns are caught, candidate alias extraction is performed as in Figure 6.

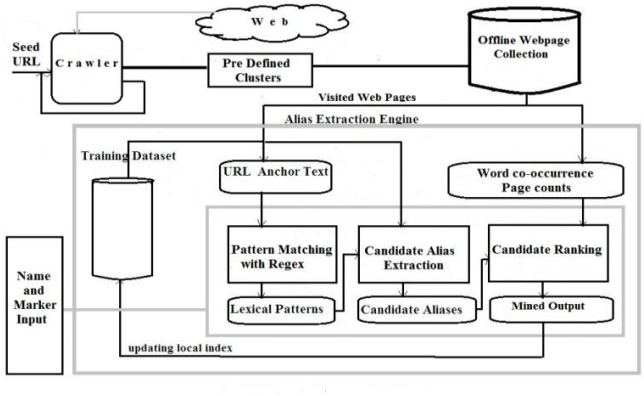

Figure 6 – Architecture of A2E bot

This method produces result based on the availability of total number of crawled web pages in the offline storage. An open source.NET code library [39] named HtmlAgilityPack [HAP] has been used to parse documents, which is in out of the web HTML files. The merit of this architecture is that it reduces net access time instead of simultaneous crawling and extraction.

3.4 Efficiency Consideration

Generic crawler Figure 5 allows all kind of web pages which contains audio, video and textual matters stored in a database. However, this method ignores web pages with images and audio files for pattern recognition. Experiment is conducted for fifty five personal names given as input to the *A2E* tool as a dataset and the derived F-Score are observed. Another, demerit of this method is that it ignores non-formal patterns which are described for aliases in social networking sites like *facebook*, *twitter*, *you tube* etc. It also ignores multi-lingual markers or colloquial aliases which are partly in English. In addition to the existing alias markers in the existing method, few markers like *popularly known as* *, *popularly called* *, *popularly referred to as* * are added and it adds value to proposed method results.

3.5 *Algorithm/Functions*

```
Algorithm 1.0  ExtractPatterns(S)
// S is a set of name-alias pairs//
P ← null
For each (NAME,ALIAS) ϵ S do
begin
      D←LoadPage(pattern)
      For each page d ϵ D do
      Begin
         P ← P+PatternMatch(d)
      End
End
```

```
Function 1.1  LoadPage(selected pattern)
  X←dot net implementation of regular expression co-occurrence
  Y←load an offline web page into cache from dataset
  T ← Start a new thread to perform pattern matching
return(Y)
```

```
Function 1.2 PatternMatch(X)
 //binary form web page is converted in to HTML document form using html
agility pack//
          corr =0,incorr=0
     If( regex(X) exactly matches in webpage)
          corr=corr+1
     elseif (regex(X) partially match in webpage)
        check for ambiguous alias names given in training dataset
     elseif (regex(X) doesnot match in webpage)
          incorr=incorr+1
//pattern co-occurrences are clearly classified in to two sets//
return
```

```
Algorithm 2.0 ExtractCandidates(Name,P)
   // P is the set of patterns
   C ←null
   For each pattern p ϵ P do
   Begin
       D ← LoadPage(pattern)
       For each page d ϵ D do
       Begin
            C ←C + GetNgrams(d,Name,P)
            Display mismatched list, candidate aliases, computed results
       End
  End
End
```

The above algorithm 2.0 list all mismatched output, user can change eliminated output into valid output. The reason for being eliminated is due to the same pronunciation or minor typographical errors or entirely a new alias which might be a recent arrival. Mismatched output can be added as *NewRow* or *SelectedRow* in the training dataset. It is required to search iteratively through relevance feedback mechanism [4] in order to see the enhanced machine learned result.

IV.PERFORMANCE EVALUATION AND COMPARISON

For the set of nine patterns common to both methods are tabulated and obtained the following values. It is evident from the graph that this proposed F-Score is much better than existing method of automatic alias extraction.

TABLE 3- Comparison Table

Slno	Patterns	Proposed Method F-Score	Existing Method F-Score
1	aka *	0.48	0.32
2	aka the *	0.48	0.19
3	nickname *	0.36	0.22
4	better known as *	0.33	0.31
5	also known as *	0.30	0.28
6	alias *	0.24	0.29
7	* aka	0.19	0.34
8	* was born	0.18	0.15
9	* whose real name is	0.13	0.20

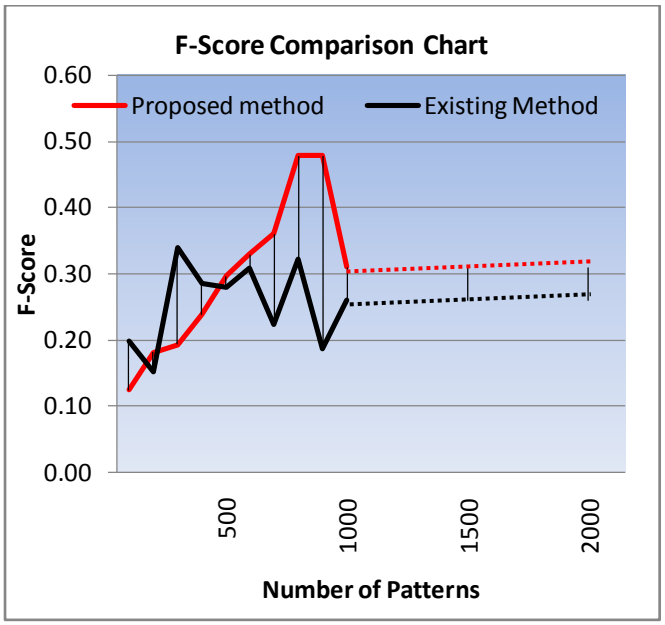

Figure 7- Improved F-Score obtained in Proposed Method

4.1 Performance Analysis

It is evident from Figure 8 precision values are maximum at four co-ordinate points, because it is observed that likelihood valid aliases exist from the mismatched set of output and thus few valid mismatched items were made as matched item during the process of machine learning.

For instance, celebrity 'magnus carlsen' has original alias name as Mr.Precision. While searching, the tool retrieved aliases like mister precision, Mr Precision, Mister P….. etc., Here, first two aliases were made as original, proposed method ignores the last one since it is incomplete.

These typographical errors were scrutinized to pinpoint correct alias and thus it transforms noisy aliases to correct alias through relevant feedback mechanism. At the same time, code rejects junk character output as people use to write in social networking websites. The accuracy in rejecting the non-relevant item is successful in proposed algorithm and hence the precision as in Figure 8.

TABLE 4- Top Ranked Patterns obtained in Proposed Method

Slno	Formal Patterns	F-Score
1	aka *	0.48
2	aka the *	0.48
3	*popularly known as* *	0.38
4	nickname *	0.36
5	better known as *	0.33
6	also known as *	0.30
7	*popularly called* *	0.24
8	alias *	0.24
9	* aka name	0.19
10	* was born name	0.18
11	* whose real name is name	0.13
12	* nee	Nil
13	*popularly referred to as* *	Nil

TABLE 5-Snapshot of Extracted Unranked Aliases from A2E bot

Real Name	Extracted Aliases
Sachin tendulkar	The little master, The master blaster, The god of cricket, Tendlya
Harbhajan Singh	The zen master of modern Cricket, bhajji, The turbanator
Rahul Dravid	The wall, Jammy
Magnus Carlsen	The prince of chess, The Mozart of Chess, Justin bieber of chess, Mister Precision, Maggie
Stefanie maria graf	Steffi Graf
Viswanathan Anand	Vishy, The tiger of madras, Lightning kid

4.2 Graphical Output

It has been crawled uniformly for all patterns. However, there are no co-occurring results in the web documents available in the offline storage. It could be the fact that 34,126 crawled web pages were not sufficient to bring results for two such patterns out of 13 patterns. They are * *nee [name]* and *[name] popularly referred to as* *. The below graph is drawn using the obtained result for precision, recall and F-Score of proposed method.

From the Figure 7 and Figure 8, it is evident that *A2E* tool produces a higher Precision and F-Score in an information retrieval task. The graph Figure 8 shows maximum precision value for four chosen patterns. It is observed that the first two patterns in Table 4 gave maximum yield and both patterns are ranked as top one.

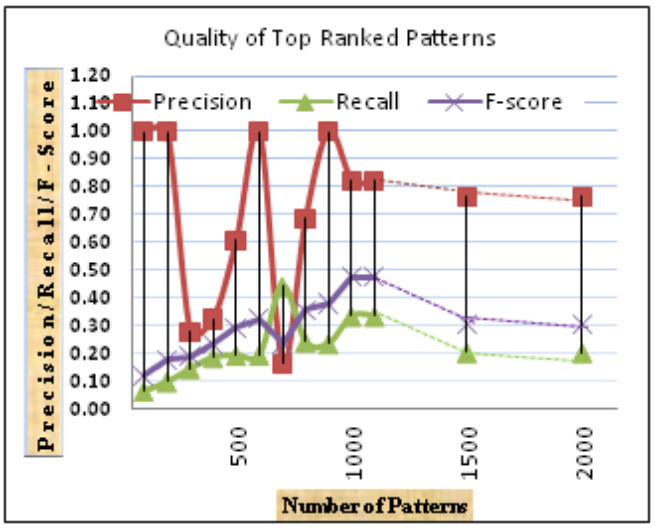

Figure 8- Top Ranked Patterns

V. CONCLUSION

The new architecture has been successfully implemented with two major features they are (1)Inducing unsupervised learning process (2)Name disambiguation in case of referential ambiguity. Also, it has been proved that new method outperforms existing method [37] of alias extraction. Proposed method is advantageous in bringing out higher precision and F-Score in a web search task.

VI. FUTURE WORK

As of now, celebrities belonging to sports alone are considered as input and results are shown above. Web page clustering is yet to be applied for all such domains. A suitable ranking method is to be adopted to enhance the work. Considering all categories of web sites which is inclusive of video and audio files can be the future enhancement. Information retrieval using ad-hoc, non-formal pattern seem to be interesting. Moreover, regional language distinguishing feature and colloquium in web documents would be further challenges in this area.

References

1. G.Salton A.Wong and C.S.Yang, "A vector space model for Information Retrieval", Communications of the ACM, 1975
2. G.Salton and M.McGill," Introduction to Modern Information Retrieval", Mc Graw-Hill 1986
3. G.Salton and C.Buckley, "Term-Weighting Approaches in Automatic Text Retrieval, "Information Processing and Management", 1988
4. Salton.G and C.Buckley, "Journal of American Society for Information Science", 1990
5. K.Church and P.Hanks, "Word Association Norms, Mutual Information and Lexicography", Computational Linguistics, 1991
6. M.Hearst, "Automatic Acquisition of Hyponyms from large text corpora," Proceedings International Conference Computational Linguistics, 1992
7. F.Smadja, "Retrieving Collocations from Text:Xtract", Computational Linguistics 1993
8. T.Dunning, "Accurate Methods for the Statistics of Surprise and Coincidence," Computational Linguistics, 1993
9. Y.Yao, "Measuring retrieval effectiveness based on user preference of documents", Journal of the American Society for Information science, 1995
10. A.Bagga and B.Baldwin, "Entity-Based Cross-Document Co-Referencing using the Vector Space Model," Proceedings of International Conference on Computational Linguistics, 1998
11. Dekang Lin, "Automatic retrieval and clustering of similar words", 17th Int'l conf. On computational Linguistics, 1998
12. M.Mitra, A.Singhal, and C.Buckley, "Improving Automatic Query Expansion," Proceedings of SIGIR '1998
13. C.Manning and H.Schutz," Foundations of Statistical Natural Language Processing", MIT Press, 1999
14. M.Berland and E.Charniak, "Finding parts in very large corpora", Proc.Ann.Meeting of the Association for Computational Linguistics (ACL '99), pp 57-64, 1999
15. R.Baeza-Yates and B.Riberio-Neto,"Modern Information Retrieval", ACM Press, 1999
16. D.Beeferman and A.Berger," Agglomerative clustering of a search engine Query log", In ACM SIGKDD, International conference on Knowledge Discovery and Data Mining (KDD), 2000
17. T.Joachims, "Learning to classify Text Using Support Vector Machines – Methods, Theory, and Algorithms, 2002
18. T.Joachims, "Optimizing Search Engines using Clickthrough Data,"Proceedings ACM SIGKDD 2002
19. Konchady M., and R.D'Amore, "Implementing a Smart Spider," Dr.Dobbs Journal, August 2002.
20. Jackson.P and I.Moulinier, Natural Language Prcoessing for online Applications:Text Retrieval, Extraction, and Categorization. John Benamins Publishing Company 2002
21. Y.Li.Zuhair, A.Bandar, D.M, "An approach for measuring semantic similarity between words using multiple information sources", IEEE Transactions on Knowledge and Data Engineering, 2003
22. G.Manna and D.Yarowsky, "Unsupervised Personal Name Disambiguation, "Proceedings of Conference on Computational Natural Language Learning, 2003
23. M.Bilenko and R.Mooney, "Adaptive Duplicate Detection using Learnable String Similarity Measures," Proceedings SIGKDD 2003
24. R.Guha and Garg, "Disambiguating People in Search", Technical Report, Stanford University, 2004

25. P.Cimano S.Handshub and S.Staab, "Towards the Self Annotating Web", Proceedings of International World Wide Web Conference 2004
26. R.Bekkerman and A.Mccallum, "Disambiguating Web Appearances of People in a Social Network," Proceedings of International World Wide Web Conference 2005
27. J.Articles, J. Gonzalo and F.Verdejo, " A Testbed for people Searching Strategies in the World Wide Web", Proceedings of SIGIR 2005
28. T.Hokama and Kitagawa, "Extracting Mnemonic Names of People from the web," Proceedings of Ninth International Conference Asian Digital Libraries, 2006
29. D.Bollegala, Y.Matsuo, and M.Ishizuka, "Extracting keyphrases to disambiguate personal names on the web", In Proceedings of CICLing 2006
30. Y.Matsuo, J.Mori, M.Hamasaki, K.Ishida, T.Nishimura, H.Takeda, K.Hasida and M.Ishizuka, "Polyphonet: An advanced Social Network Extraction System", Proceedings of World Wide Web 2006
31. Chia-Hui Chang, Mohammed Kayed, Moheb Ramzy Girgis, Khaled F.Shaalan, "A Survey of Web Information Extraction Systems", IEEE Transactions on Knowledge and Data Engineering, 2006
32. C.Galvez and F.Moya Anegon, "Approximate Personal Name-Matching through Finite State Graphs, "Journal of American Society for Information Science and Technology, 2007
33. D.Bollegala, Y.Matsuo and M.Ishizuka, "Measuring Semantic Similarity between Words using Web Search Engines," Proceedings International World Wide Web Conference 2007
34. Danushka Bollegala, Taiki Honma, Yutaka Matsuo and Mitsuru Ishizukz, "Mining for Personal Name Aliases on the Web", In Proceedings of World Wide Web 2008
35. S.Sekine and J.Artiles, "Weps 2 Evaluation Campaign:Overview of the Web People Search Attribute:Extraction Task," Proceedings second Web People Search Evaluation Workshop(Weps 2009) at 18th International World Wide Web 2009
36. Danushka Bollegala, Yutaka Matsuo and Mitsuru Ishizuka, "Identifying People on the Web through Automatically Extracted Key phrases", In proceedings of World Wide Web
37. Danushka Bollegala, Yutaka Matsuo, Mitsuru Ishizuka, "Automatic Discovery of Personal Name Aliases from the Web", IEEE Transactions On Knowledge and Data Engineering, 2011
38. http://www.princeton.edu/~mlovett/reference/Regular-Expressions.pdf
39. http://htmlagilitypack.codeplex.com/
40. http://www.ultrapico.com/expresso.htm

Thangaraj Muthuraman received his Masters Degree in Computer Applications from Alagappa University, Karaikudi, M.Tech Degree in Computer Science from Pondicherry University and Ph.D Degree in Computer Science and Engineering from Madurai Kamaraj University, Madurai, Tamilnadu, South India in 2006. He is now the Associate Professor in Computer Science and Engineering Department at Madurai Kamaraj University. He is an active researcher in Web Mining, Semantic Web, Information Retrieval and Big-Data Analytics. He has published more than 60 papers in Journals and Conference Proceedings. He is a member in the Society of Digital Information and Wireless Communication (SDIWC) and a Senior Member of IACSIT. He has served as Program Chair and Program Committee member of many International Conferences held in India for about one decade.

Sivagaminathan Periyasamy Ganesan received his Masters Degree in Computer Applications from Alagappa University, Karaikudi. He is currently a Ph.D Research Scholar in the Department of Computer Science and Engineering, Bharathiar University, Coimbatore, India. He has been working in Education Industry for more than two decades. He worked in reputed Institutions in India and abroad Universities. Participated International Conferences in the field of computer science and engineering abroad. His area of interest include Information Retrieval, AI and Machine Learning, Neural Networks