12-2010

Pattern extraction from the world wide web

Praveena Mettu
University of Nevada, Las Vegas

PATTERN EXTRACTION FROM THE WORLD WIDE WEB

by

Praveena Mettu

Bachelor of Technology, Computer Science and Engineering
Jawaharlal Nehru Technological University, India
May 2008

A thesis submitted in partial fulfillment
of the requirements for the

Master of Science Degree in Computer Science
School of Computer Science
Howard R. Hughes College of Engineering

Graduate College
University of Nevada, Las Vegas
December 2010

THE GRADUATE COLLEGE

We recommend the thesis prepared under our supervision by

Praveena Mettu

entitled

Pattern Extraction from the World Wide Web

be accepted in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science
School of Computer Science

Kazem Taghva, Committee Chair

Ajoy K. Datta, Committee Member

Laxmi P. Gewali, Committee Member

Muthukumar Venkatesan, Graduate Faculty Representative

Ronald Smith, Ph. D., Vice President for Research and Graduate Studies
and Dean of the Graduate College

December 2010

ABSTRACT

Pattern Extraction from the World Wide Web

by

Praveena Mettu

Dr. Kazem Taghva, Examination Committee Chair
Professor, Department of Computer Science
University of Nevada, Las Vegas

The World Wide Web is a source of huge amount of unlabeled information spread across different sources in varied formats. This presents us with both opportunities and challenges in leveraging such large amount of unstructured data to build knowledge bases and to extract relevant information.

As part of this thesis, a semi-supervised logistic regression model called “Dual Iterative Pattern Relation Extraction” proposed by Sergey Brin is selected for further investigation. DIPRE presents a technique which exploits the duality between sets of patterns and relations to grow the target relation starting from a small sample.

This project built in JAVA using "Google AJAX Search API" includes designing, implementing and testing DIPRE approach in extracting various relationships from the web.

Keywords: Pattern Extraction, Machine Learning, DIPRE

TABLE OF CONTENTS

ABSTRACT iii

LIST OF FIGURES v

ACKNOWLEDGEMENTS vi

CHAPTER 1 INTRODUCTION 1
1.1 Aims and Objectives 2
1.2 Thesis Organization 3

CHAPTER 2 LITERATURE OVERVIEW 4
2.1 Information Extraction Systems 5
2.2 DIPRE Algorithm 9

CHAPTER 3 IMPLEMENTATION 12
3.1 Choosing the search engine 12
3.2 Choosing the programming Language 13
3.3 Overview of Java Program 14
3.3.1 User Interface 14
3.3.2 Find Occurances 15
3.3.3 Get the next set of tuples 18
3.3.4 Output 18

CHAPTER 4 RESULTS AND EVALUATION 25

CHAPTER 5 CONCLUSION AND FUTURE WORK 34
5.1 Conclusion 34
5.2 Future Possibilities 35

BIBLIOGRAPHY 38

VITA 40

LIST OF Figures

Figure 1	Pattern Learner	5
Figure 2	DIPRE Algorithm	10
Figure 3	Input for parameter1	14
Figure 4	Input for parameter2	14
Figure 5	Input for SearchSite	15
Figure 6	Json operations to fetch results from internet	16
Figure 7.1	Result Set1	19
Figure 7.2	Result Set2	20
Figure 7.3	Result Set3	21
Figure 7.4	Result Set4	22
Figure 8	Google Query Restriction	36

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank Dr. Kazem Thagva, my research advisor for his support and guidance. Through-out the project, he provided with a lot of ideas and sound advice.

I am also grateful to all the faculty, librarians and secretaries at University of Nevada, Las Vegas.

It is a pleasure to offer my regards to my friends and students at the University for a stimulating and fun-filled environment to learn and grow. Thanks a ton for the camaraderie and entertainment.

Lastly and most importantly, I would like to thank my family for all the love and sacrifices they made for my well-being. To them, I dedicate this thesis.

CHAPTER 1

INTRODUCTION

The World Wide Web, with its masses of unstructured and inconsistently coordinated information is easier to construe by humans than by machines.

Imagine what could be done if we can train systems to analyze text and mine the enormous amount of data created by people on the Internet. It would become the source of unprecedented amount of information easily interpreted by machines, thus maximizing both people and computer resources. It would include the largest and most diverse databases of people, products and academic work constantly growing with thousands of new web pages appearing everyday in the World Wide Web [1].

Information Extraction (IE) can be dated back to the early days of Natural Language Processing (NLP) in the 70’s. IE is a technique to retrieve focused target relations from unstructured machine-readable documents by means of NLP [2]. Most information extraction systems are based on rule-based methods that employ some form of machine learning. They can generate rules based on labeled-data which unfortunately may require labor-intensive hand-coding of search patterns. Data may also need to be massaged specific to the desired target relation which we want to extract. One approach to automating

the information extraction process is a class of techniques called semantic bootstrapping. In general, these techniques use some form of machine learning to recognize semantic patterns in large amounts of unlabelled data. The recognized patterns are then applied iteratively to extract more semantics. Effectively, semantic bootstrapping applies semantic labels to the data based on what it already knows and then labels even more data based on the new labels it applied [3].

1.1 Aims and Objectives

This thesis focuses on extracting entities from World Wide Web which have a relationship between them that is similar to the one existing between the two entities provided. There have been a number of algorithms proposed in recent years aimed at extracting relationships between entities. In this thesis, by implementing one of those algorithms, we intend to evaluate its validity and proficiency.

In his experiment, Sergey Brin focused on extracting (author, title) pairs from web pages. Given the nature of the data, there must have been heavy use of HTML formatting tags to automatically generate patterns consisting of authors and titles from various book-related sites, often in tables or lists. However, in our experiment we want to enlarge the data-set on which DIPRE can be executed and evaluate its effectiveness on a more free-form natural language text in internet [3].

1.2 Thesis Organization

Chapter 2 covers the Literature Review, which discusses a general approach to various research topics in information extraction. This chapter also covers some techniques for extracting factual information from the web and their applications that relate to this thesis.

Chapter 3 covers the approach implemented in this thesis (Java program, Google search API). It analyses the algorithm and design decisions made for the project. Explanation of various parts of the algorithm and how they function together is provided. Also, the technical challenges faced during project implementation and how they were overcome are mentioned.

Chapter 4 lists the results from the execution of the java program built based on DIPRE algorithm when provided with a seed pair. It also discusses the effectiveness and limitations of the algorithm.

Finally, Chapter 5 concludes the entire thesis and suggests how this utility program can be improvised for future work.

CHAPTER 2

LITERATURE OVERVIEW

This chapter presents relevant background knowledge that is closely related to this thesis. This will help readers to easily understand the analysis.

Machine Learning allows computer programs to automatically improve by interpreting the obtained results. This concept can be applied in Natural Language Processing by either annotating examples of mapping we are interested in or by applying a machinery to learn from examples. Since Information Extraction systems have to process large bodies of information, machine learning is an integral part of such algorithms.

Reducing the human workload in developing extraction rules is a huge challenge in Information Extraction. Depending on the level of human involvement most adaptive information extraction systems use different machine learning algorithms. However, many IE systems use very complex pre-defined rules rather than using any machine learning technique at all to accomplish the extraction task [4]. In the first section of this chapter, each type of IE system is reviewed and their advantages and disadvantages are discussed briefly. In the second section, we discuss Sergey Brin’s DIPRE algorithm.

2.1 Information Extraction Systems

Information Extraction Systems usually rely on a set of Seed-Patterns that they use to compare and retrieve relevant information from plain text. The new-patterns retrieved can be used as both extractors and discriminators.

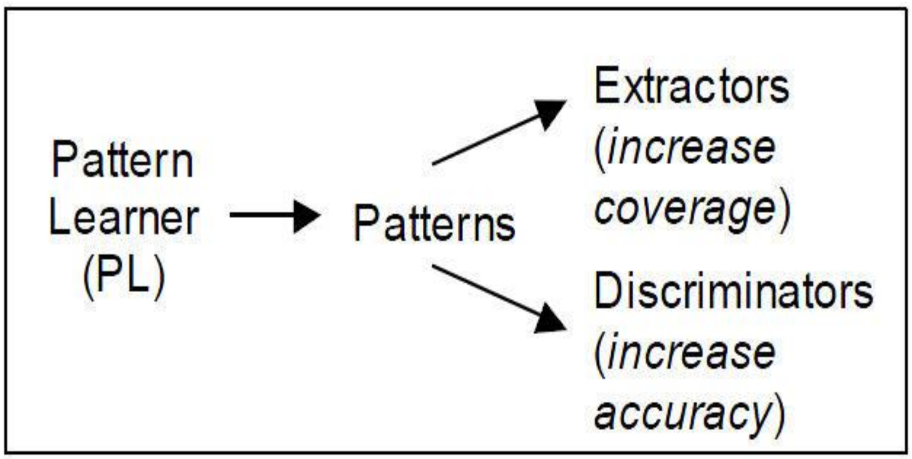

Figure 1: Pattern Learner [5]

Extractors can be further used as seed-patterns to extract new patterns.Not every extracted pattern may be relevant for our study. By adding those to the *Discriminators* list (negative list), we improve the overall accuracy of the algorithm. These discriminators prevent the algorithm from issues of semantic drift, where erroneous new patterns cause the system to run off track.

Many different IE approaches have been developed in the recent past for various IE tasks depending on the goal of study and the nature of

information source. However, no matter which data sources are utilized in relation extraction, all Information Extraction Systems have to meet three requirements:

1) A source which comprises of entities of our interest hidden in plain text. A group of such databanks will serve as data sources from which semantic relations will be extracted

2) A semantic or linguistic source in which the context for relations between entities is provided

3) Algorithms for automatic execution of processing operations.

How well a relation extractor performs is determined mainly by the contextual information sources and algorithms [6]. A heavy rule-based extractor with a good context source need not always contain “pattern learning algorithms”. However, this involves humungous human effort. Pattern Learning algorithms generate their rules from examples thus reducing human effort.The early IE systems were primarily rule-based. Skillful engineers embed the extraction rules in these systems using their knowledge in the domain. Unfortunately, natural languages are extremely flexible and it is impossible to hand code a set of rules to extract accurate information from a variety of web pages with varying structures. Hence, more intelligent and scalable approaches were sought after.

Machine Learning techniques can help train IE systems to discover new extraction rules automatically with experience. This is much faster

than the rule-based approach and has been used more often in recent IE system developments. Depending upon the amount of human intervention required, Machine Learning (ML) approaches can be roughly divided into 3 categories [4].

Supervised IE systems: These methods use a set of manually labeled data to construct extraction rules. This approach is relatively more automated compared to a rule-based approach, however needs human supervision.AutoSlog is a relationship extraction approach based on this method [7]. It needs texts and tagged noun phrases to generate patterns. For example, let us consider this sentence:

“A pedestrian was attacked by a dog”

Manually, “pedestrian” is tagged as a victim and “dog” as the perpetrator of a murder.AutoSlog will propose “<X> was attacked” as a pattern where <X> is a victim. It can use this pattern to find more victims.It will also propose “was attacked by <Y>” as a pattern where <Y> is the perpetrator. It can find more perpetrators by matching new sentences with learned pattern.The laborious job of manually annotating training documents makes this approach unfavorable in many practical scenarios.

Non-Supervised IE systems: These methods use highly automated algorithms hence; do not require any training documents or any human supervision. However, as a drawback they tend to suffer from inaccuracies in results.AutoFeed [8], Yangarber [9] and KnowItAll [10] are

examples of this method.Although these methods are still “young”, they suggest futuristic possibilities of building automated information extraction systems that can be used in many practical applications like search engines, knowledge-based AI systems, etc [10].

Semi-Supervised IE systems: These methods are somewhere in-between supervised and unsupervised IE systems.The advantage of a semi-supervised method over supervised method is that, given a small set of annotated examples, these methods can generate their own training data through “active learning”. These are also the most practical methods used in IE systems.In the process of active learning, a small amount of labeled training set is fed to the algorithm. It applies these rules to classify unlabeled data. The most trusted newly classified sets along with their labels are added to the training set. The algorithm re-learns from this new training set. These rules are again applied to classify unlabeled data. This procedure is applied repeatedly until unlabeled data set is exhausted or the trust-worthiness of the results drops below a threshold.

DIPRE (Dual Iterative Pattern Relation Extraction) [1] and Snowball [11] are two examples of Semi-Supervised IE systems which are used in extracting patterns from unstructured text.Traditional IE systems tend to extract all semantic information from each document. However, DIPRE and Snowball help create knowledge base from the web by reading only relevant content of the documents. They can extract pairs of related

entities, like authors and their books, from the web using a small group of seed pairs. They are especially effective in extracting commonly occurring relationships like (authors, books), (acronyms, meanings), etc.

The relation extraction approach described in DIPRE [1] is the core algorithm involved in a lot of other semi-supervised algorithms as well. Hence, it is of utmost importance for any IE system architect to understand this algorithm. In the next section, we shall discuss DIPRE [1] in detail.

2.2 Dipre Algorithm

DIPRE algorithm by Brin [1] is one of the earliest examples of discovering binary relationships using bootstrap learning approach. This is a technique where the duality between sets of relations is exploited to grow the target relations starting from a small sample.

Brin demonstrated this principle by finding authors and titles in the WWW. Let's say we look for "Mario Puzo" and "The Godfather" in the internet. There could be a few repeated patterns when you observe the citations of this book, for example "The Godfather was written by Mario Puzo". By looking for these "repeated patterns" in the internet, there is a good chance of uncovering other authors and titles. For example, the pattern/relation "was written by" could lead to various citations like "Catch-22 was written by Joseph Heller", "Gone with the wind was written by Margaret Mitchell", etc. These new authors and titles can again be used to find new patterns.

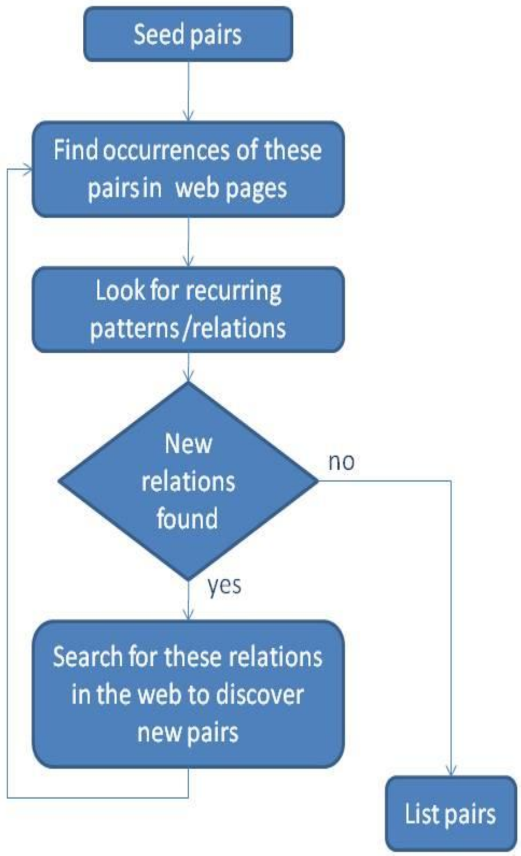

Figure 2: DIPRE Algorithm

Steps in DIPRE:

1. Start with a small set of trusted pairs (authors, books)
2. Find occurrences of these pairs in the WWW or in a defined set of Websites/WebPages
3. Identify generalized patterns from the citations fetched in step 2.
4. Use these identified patterns to find more pairs in the web.
5. Repeat steps 2 to 4 until no new patterns can be learnt.

Pattern Generation (Step 3) is the most critical step in this algorithm. The accuracy of results and performance can get hugely impacted depending on how the implementation of this step is designed.Many complex variations of DIPRE algorithm have been proposed in the recent past; each with different ideas of pattern generation (Step 3). However, in this project we will focus on the well-known DIPRE algorithm originally suggested by Brin.

CHAPTER 3

IMPLEMENTATION

There were two major decisions that had to be made before beginning the development of the application.

1. Search Engine
2. Programming Language

3.1 Choosing the search engine

There are various APIs available to retrieve information from the internet. Famous ones are listed below:

Google AJAX Search API [12]: This is a JavaScript Library that returns JSON encoded results for easy processing. It has a limit of 64 results per query.

Bing API [13]: Also known as the Live Search API, this API was built by Microsoft on standards that are based on SOAP, XML and WSDL technologies. It provides an XML web service through a SOAP API. Historically, this API has been more tested using C# projects. Hence, is more suitable to the .NET platform.

Yahoo BOSS API [14]: BOSS (Build Your Own Search Service) is Yahoo!'s open search web services platform. For each query, this API can return up to 100 results in XML or JSON format. It also has the ability to restrict search in a pre-defined set of websites.

NewYork Times Article Search API [15]: This is another interesting API which searches NYTimes articles to retrieve headlines, abstracts, lead paragraphs and links to associated multimedia. This API also returns results in JSON format.Before using any of these APIs, we need to adhere to their Terms of Service.There is not much of a difference between these APIs. Almost all of them retrieve results in JSON format which is I believe is an easy-to-use form of representing data similar to XML. We chose Google AJAX Search API for this implementation simply because it is the current market leader and has been researched extensively.

3.2 Choosing the Programming Language

JSON (JavaScript Object Notation) [16] is a lightweight data-interchange format. It is based on a subset of the JavaScript Programming Language, Standard ECMA-262 3rd Edition - December 1999. JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others.

We considered coding the algorithm either in C#, java or Perl. Eventually, we chose java as it is comparatively much faster than the others.

3.3 Overview of Java Program

In this section, we explain the UI and various sections of the algorithm.

3.3.1 User Interface

As per DIPRE, we intend to take two parameters as input from user. In Brin's example, the study was focused on Books & Authors. However, in our experiment, we intend to apply DIPRE algorithm on any related parameters.

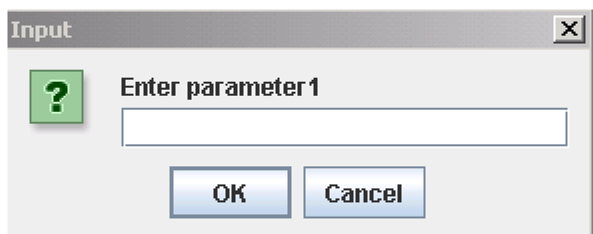

Figure 3: Input for parameter1

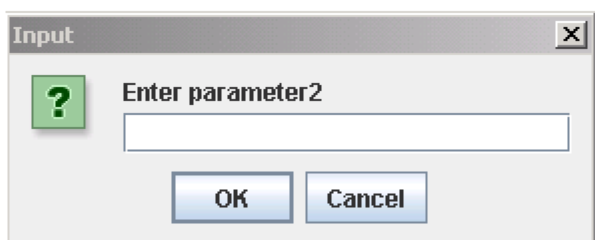

Figure 4: Input for parameter2

After providing the parameters, user can also specify any site where the search has to be restricted to. By default, we intend to search in

www.wikipedia.org because that's where we found most of our results. However, user is free to change it to any other specific site. If the SearchSite is not specified, the search will be similar to a standard "Google Search" where it looks for these two related parameters all over the internet.

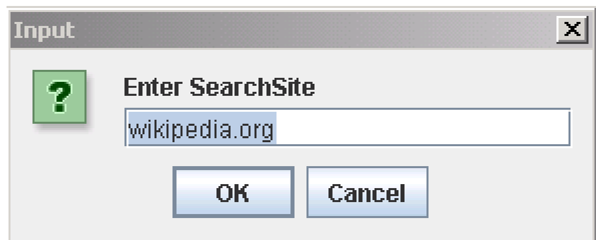

Figure 5: Input for SearchSite

3.3.2 Find Occurrences

Using these details, we generate the string that needs to be passed to the Google AJAX Search API. The following is done to build the search string.

1. Replace all spaces with + in parameters.
2. If SearchSite is empty, replace it with www.google.com
3. SearchString = "%22 " + parameter1 + "%22" + "+" + "%22 " + parameter2 + "%22" + " site:" + searchsite

Then the URL is built using this searchstring which can be passed to the search API.

URL:"http://ajax.googleapis.com/ajax/services/search/web?start=0&rsz=large&v=1.0&q=" + searchstring

The API result set is in the form of a JSON object. Within this object, there are eight occurrences inside an array. The program loops on the array and each result string is subdivided into “prefix”, “middle term” and “suffix” sections by looking up parameter1 and parameter2 in the string.

```
URLConnection connection = url.openConnection();
connection.addRequestProperty("Referer", HTTP_REFERER);

// Get the JSON response
String line;
StringBuilder builder = new StringBuilder();
BufferedReader reader = new BufferedReader(
  new InputStreamReader(connection.getInputStream()));
while((line = reader.readLine()) != null) {
 builder.append(line);
}

String response = builder.toString();
JSONObject json = new JSONObject(response);

JSONArray jarray = json.getJSONObject("responseData")
.getJSONArray("results");

if (count == 0) {
System.out.println();
System.out.println(" Results:");
System.out.println();
}

for (int i = 0; i < jarray.length(); i++) {
 System.out.print((i+count+1) + ". ");
 JSONObject jobj = jarray.getJSONObject(i);
 /*
 System.out.println(jobj.getString("titleNoFormatting"));
 System.out.println(jobj.getString("url"));
 System.out.println(jobj.getString("content"));
 */

 substring1 = jobj.getString("content");
```

Figure 6: json operations to fetch results from internet

Before working with json objects, json library has to be downloaded from json.org and added to the library list of the project.The above code describes how the results are fetched from the search API.The connection is opened for the specific URL which was built using the parameters passed by user.The inputstream is read line by line and appended to String Builder. This is then converted into a string “response”. The data in “response” is put into a JSON object “json”.The array list of results in json object can be looped on. As per Google AJAX Search API documentation, Title,Titlenoformatting,Url and Content tags can be used to get specific information from each result in Json object. By fetching the content into a string, it can be subdivided into various sections by looking for the parameters in the string.Once these sub-strings are stored, we can get the next set of eight results from the search API by passing the following string.

"http://ajax.googleapis.com/ajax/services/search/web?start=8&rsz=large&v=1.0&q=" + query

Parameter “start” tells the API where to start returning the results from. For example, if “start” is set as 8, then results 9-16 are passed back to the program.Parameter result size “rsz” can be set as small or large. Small gives 4 results and large gives 8 results at a time.The query is repeated by passing 8,16,24,32,40,48,56 as “start” parameters. Eventually, we get up to 64 results.

3.3.3 Get the next set of tuples

Most of the times, we get bad results because all the results need not always be relevant to the experiment. Hence, of the 64 results only a few maybe relevant for our analysis. The valid results are stored.Parsing through the valid results, most repeated “middle term” is fetched. Also, the longest common substring is fetched from “prefix” and “suffix”.

We prepare a query string by using these substrings.

Search query = Longest_prefix + "+*" + most_repeated_middle_term + "*+" + Longest_suffix + " site:” + searchsite

By passing this query to the search API, we can again get up to 64 results of new tuples. However, very few of them will be valid tuples.The occurrences of these valid tuples can again be searched in the internet to get valid relations. Such iterations can continue until the search API stops returning valid results.There has to be a filtering algorithm to determine valid tuples. They can also be manually checked before using to get next set of relations. This is a common problem in a lot of semi-supervised learning algorithms.

3.3.4 Output

The results are displayed on the console using a simple “System.out.println” statement.Let’s say I search for “Lebron James” and “NBA” in “Wikipedia.org”. Following is the result retrieved.

Querying for : %22 lebron+james%22+%22 nba%22 site:wikipedia.org

1. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/LeBron_James-D-YAHOO NEWS HTTPSPORTSYAHOOCOM-E-NEWSSLUGAPLEBRONSBOOKAMPPROVAPAMPTYPE LGNS LIVINGSTON BILL JULY 22 2009 QUOTNEW-F- BOOK TELLS OF A

2. LEBRON JAMES-A-NBA-B-1-C-http://en.wikipedia.org/wiki/Double_(basketball)-D-YOUNGEST PLAYER -E-CLEVELAND CAVALIERS AGED 20 YEARS AND 20 ROSENBLUTH CHRIS NOVEMBER 22 2006 QUOTAROUND THE ASSOCIATIONQUOT-F-COM

3. LEBRON JAMES-A-NBA-B-1-C-
http://en.wikipedia.org/wiki/List_of_career_achievements_by_LeBron_James-D-RECORDED 32 POINTS 11 REBOUNDS AND 11 ASSISTS ON APRIL 22 2006 VS LEBRONISREALLYGOODCOM WHY -E-IS THE 2009-F- MVP ACCESSED APRIL 14

4. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Upper_Deck_Company-D-UPPER DECK PREMIERED ITS -E-EXQUISITE COLLECTION LINE IN THE 20032004 SEASON LINE INCLUDE THE AUTOGRAPHEDPATCH ROOKIE CARDS NUMBERED TO 99-F- 22 AND FIRST EDITION SEPT 29 EACH OF THE CARDS WILL INCLUDE MJ39S

5. 6. LEBRON JAMES-A-NBA-B-1-C-
http://en.wikipedia.org/wiki/List_of_career_achievements_by_Kobe_Bryant-D-SURPASSED BY -E-AMP0000000000000002500000025 YEARS RETRIEVED JUNE 22 2009 QUOT-F- ATHLETE OF THE DECADE KOBE BRYANT SG LAKERSQUOT

7. LEBRON JAMES-A-NBA-B-1-C-http://en.wikipedia.org/wiki/Carmelo_Anthony-D-HE MADE SHOTS IN THE FINAL 22 SECONDS AGAINST THE CLEVELAND CAVALIERS ON QUOT -E-WINS 200304 GOT MILK ROOKIE OF THE YEAR AWARDQUOT-F-COM

8. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/2003_NBA_Draft-D- WON THE -E-DRAFT LOTTERY ON MAY 22 AND CLEVELAND CHAIRMAN GORDON GUND SAID AFTERWARD HIS TEAM WOULD SELECT-F- THE DETROIT PISTONS AND THE

9. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/2010_NBA_Playoffs-D-HOME COURT ADVANTAGE IN THE -E-FINALS DOES NOT NECESSARILY BELONG TO THE SCORING BY QUARTER 2231 2323 2720 1523 PTS-F- 22

10. LEBRON JAMES-A-NBA-B-1-C-http://en.wikipedia.org/wiki/O._J._Mayo-D-DATE OF BIRTH NOVEMBER 5 1987 19871105 AGE 22 MUCH LIKE ANOTHER HIGH SCHOOL STAR FROM OHIO ST VINCENTST MARY HIGH SCHOOL -E-ON JUNE 26 2008 O J MAYO WAS SELECTED 3RD OVERALL IN THE 2008-F- DRAFT BY

11. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Boston_Celtics-D-IT WOULD BE 22 YEARS BEFORE THEY WOULD REACH THE -E-FINALS AGAIN THE SECOND ROUND PITTED BOSTON AGAINST-F- AND THE CLEVELAND CAVALIERS

12. LEBRON JAMES-A-NBA-B-1-C-http://en.wikipedia.org/wiki/Dwyane_Wade-D--E-HIMSELF DESCRIBED THE DUNK AS QUOTGREAT PROBABLY TOP 10 ALLTIMEQUOT 2009-F- ALLSTAR GAME NBACOM ACCESSED JANUARY 22 2009

13. LEBRON JAMES-A-NBA-B-1-C-http://en.wikipedia.org/wiki/Kevin_Durant-D-ALTHOUGH DURANT HELD HIS OWN IN SCORING 22 POINTS FOR THE BLUE TEAM IN ONE DURANT JOINED -E-AS THE FORWARDS ON THE 2010 ALL-F- FIRST TEAM

14. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Dwight_Howard-D-HOWARD TOPS BALLOTING FOR 2009 -E-ALLSTAR GAME NBACOM JANUARY 22 2009 JAMES LEADS US SQUAD PAST ARGENTINA TO CLAIM GOLD NBACOM KELVIN TORBERT 2002-F- 2003 LEBRON JAMES 2004 DWIGHT HOWARD 2005

15. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Kevin_Garnett-D-ON APRIL 22 2008 GARNETT WAS NAMED THE -E-DEFENSIVE PLAYER OF THE YEAR FOR THE 200708 199495 AND-F- CLEVELAND CAVALIERS 200809

16. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Mark_Cuban-D-HE IS THE OWNER OF THE DALLAS MAVERICKS AN -E-BASKETBALL TEAM OWNER OF LANDMARK THEATRES AND CHAIRMAN OF ON MAY 22 CUBAN WAS FINED 100000 FOR COMMENTS HE MADE DURING A TELEVISION INTERVIEW ABOUT TRYING TO SIGN-F-

Figure 7.1: Result Set1

17. 18. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Rasheed_Wallace-D-WALLACE WAS NAMED AN -E-ALLSTAR IN 2000 AND 2001 AND LED THE TRAIL BLAZERS TO COMMITTING A FOUL ON-F- AND THEN RECEIVED TWO TECHNICAL FOULS 22 THE CELTICS MADE THE NBA FINALS IN 2010 BUT LOST THE SERIES TO THE LOS
19. LEBRON JAMES-A-NBA-B-1-C-http://en.wikipedia.org/wiki/Von_Wafer-D-DURING THE GAME HE SCORED 8 POINTS AND FINISHED SECOND TO -E-IN THE WAFER DECLARED HIMSELF AS AN EARLYENTRY CANDIDATE FOR THE 2005-F- DRAFT WAFER CHANGED HIS NUMBER FROM 22 WHICH HE WORE WITH THE NUGGETS TO 24
20. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Greg_Oden-D-GREGORY WAYNE ODEN JR BORN JANUARY 22 1988 IS AN AMERICAN BASKETBALL PLAYER AT THE CENTER POSITION ODEN IS A MEMBER OF THE PORTLAND TRAIL BLAZERS OF THE -E-BECOMING THE FIRST JUNIOR SINCE-F- TO BE NAMED SUCH
21. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Allen_Iverson-D-IVERSON IS AN ELEVENTIME -E-ALLSTAR WHICH INCLUDES WINNING THE ALLSTAR IVERSON AND-F- WERE BENCHED FOR A GAME FOR HAVING ARRIVED LATE AT FEBRUARY 22 2010 HTTPSPORTSESPNGOCOMNBANEWSSTORYID4936773
22. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Derrick_Rose-D-SHORTLY AFTER ROSE DECLARED FOR THE 2008 -E-DRAFT AND WAS SELECTED FIRST IN ROSE39S MUCHPUBLICIZED DEBUT HE HAD 22 POINTS 7 REBOUNDS AND 5 STEALS HE WAS ALSO THE FIRST OVERALL DRAFT PICK SINCE-F- TO WIN THE AWARD
23. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Chris_Paul-D-PAUL PARTICIPATED IN HIS SECOND -E-ALL STAR GAME STARTING FOR THE WESTERN ON MARCH 22 PAUL RETURNED TO ACTION SINCE JANUARY 29 AGAINST THE CHRIS PAUL AND-F- ARE BESTFRIENDS PAUL STATED THAT QUOTHE39S LIKE MY BROTHERQUOT
24. LEBRON JAMES-A-NBA-B-2-C-
http://en.wikipedia.org/wiki/Wikipedia:Mediation_Cabal/Cases/2008-05-10_LeBron_James-D-MAY 12 2008 THE -E-LISTS-F- AS 6398 250LBS AS DOES OTHER MULTIPLE RELIABLE REPUTABLE SOURCES BENDER235 TALK 2229 12 MAY 2008 UTC
25. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Kobe_Bryant-D-WHEN BRYANT WAS SIX HIS FATHER LEFT THE -E-AND MOVED HIS FAMILY TO ITALY TO IN GAME 4 BRYANT SCORED 22 POINTS IN THE SECOND HALF AND LED THE TEAM TO AN BRYANT WAS RUNNERUP IN THE MVP VOTING BEHIND-F- AND WAS
26. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/NBA_high_school_draftees-D-A B JAMES MICHAEL MARCH 22 1995 QUOTGARNETT IS NO GEM FOR -E-QUOT PREPS TO NBA BASKETBALL MORE AND MORE TRY THE LEAP BUT FOR EVERY-F- WHO
27. LEBRON JAMES-A-NBA-B-1-C-
http://en.wikipedia.org/wiki/List_of_nicknames_used_in_basketball-D- -E-QUOTTHE LTRAINQUOT QUOTKING JAMESQUOT QUOTTHE AKRON HAMMERQUOT QUOTLBJQUOT QUOTTHE CHOSEN ONEQUOT EARVIN JOHNSON QUOTMAGICQUOT QUOTBUCKQUOT QUOTEJQUOT THE BOOK OF BASKETBALL THE-F- ACCORDING TO THE SPORTS GUY ZICARELLI FRANK 201006 22 QUOTTIME FOR BOSH TO GOQUOT QUOTGARNETT JAMES LEAD ALONG DIFFERENT PATHSQUOT
28. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/23_(number)-D-23 TWENTYTHREE IS THE NATURAL NUMBER FOLLOWING 22 AND PRECEDING 24 MICHAEL JORDAN A STAR BASKETBALL PLAYER FOR THE -E-WORE THE NUMBER 23 ON HIS JERSEY FORMER CLEVELAND CAVALIERS FORWARD-F- ALSO WORE NO
29. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Michael_Jordan-D-IN LATER YEARS THE -E-SHORTENED ITS THREEPOINT LINE TO 22 FEET FROM 23 THEIR ROLE MODEL WHILE GROWING UP INCLUDING-F- AND DWYANE WADE
30. LEBRON JAMES-A-NBA-B-1-C-
http://en.wikipedia.org/wiki/List_of_National_Basketball_Association_season_scoring_leaders-D-RETRIEVED FEBRUARY 22 2009 QUOT-E-QUOT BASKETBALLREFERENCECOM QUOT TIGHT-F-SCORING RACE COMES DOWN TO JAMES AND DURANTQUOT
31. LEBRON JAMES-A-NBA-B-1-C-http://en.wikipedia.org/wiki/Ben_Gordon-D-GORDON ALSO FINISHED WITH 21 DOUBLEDIGIT FOURTH QUARTER POINT PERFORMANCES SECOND TO ONLY -E-39 22 IN THE-F- GORDON HELPED LEAD THE BULLS TO
32.

Figure7.2: Result Set2

33. LEBRON JAMES-A-NBA-B-1-C-
http://en.wikipedia.org/wiki/2007%25E2%2580%259308_Boston_Celtics_season-D-PAUL PIERCE AND -E-COMBINED FOR THE 2ND HIGHEST POINT TOTAL KEVIN GARNETT WINS KIA DEFENSIVE PLAYER OF THE YEAR-F-COM APRIL 22 2008
34. 35. 36. 37. LEBRON JAMES-A-NBA-B-2-C-
http://en.wikipedia.org/wiki/Fred_McLeod_(sportscaster)-D-HE JOINED THEM PRIOR TO THE 200607 -E-SEASON AS THE PLAYBYPLAY VOICE OF THE PRIOR TO JOINING FS OHIO MCLEOD SERVED 22 CONSECUTIVE SEASONS AS THE WHEN-F- MAKES A 3 POINT BASKET REFERENCING THE 330 AREA CODE OF
38. 39. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Chris_Bosh-D-IN THE 200607 SEASON BOSH LED THE RAPTORS TO THEIR FIRST -E-PLAYOFFS BERTH IN HOWARD DOMINATED THE GAME FINISHING WITH MORE THAN 22 POINTS THAN 25 SUCH AS-F- DWYANE WADE AND BOSH WOULD SIGN WITH NEW TEAMS
40. 41. LEBRON JAMES-A-NBA-B-1-C-
http://en.wikipedia.org/wiki/2007%25E2%2580%259308_Cleveland_Cavaliers_season-D-ON DECEMBER 17 2007 -E-BECAME THE YOUNGEST-F- PLAYER TO SCORE ON FEBRUARY 22 2008 LEBRON JAMES GRABBED HIS 2500TH REBOUND AS A CAVALIER
42. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Steve_Francis-D-ON FEBRUARY 22 2006 ONE DAY BEFORE THE -E-39S TRADE DEADLINE STEVE FRANCIS WAS TRADED AFTER THE MIAMI HEAT SIGNED ALLSTARS-F- CHRIS BOSH
43. 44. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Stephen_Curry_(basketball)-D-WITH -E-SUPERSTAR-F- IN ATTENDANCE CURRY SCORED 33 POINTS OVER 22 HE SET THE RECORD IN THE NEXT GAME AGAINST THE KANSAS JAYHAWKS WITH HIS
45. LEBRON JAMES-A-NBA-B-2-C-
http://en.wikipedia.org/wiki/2006%25E2%2580%259307_Cleveland_Cavaliers_season-D-ON JUNE 14 THE CAVALIERS39 SEASON ENDED IN AN -E-FINALS SWEEP TO THE 41 JANUARY 22 ORLANDO 7990 CLEVELAND NA-F- 18 20562 2417
46. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/New_York_Knicks-D-THE LOSS DENIED NEW YORK THE DISTINCTION OF HAVING BOTH -E-AND NHL CHAMPIONSHIPS THE 199798 SEASON WAS MARRED BY A WRIST INJURY TO EWING ON DECEMBER 22 WHEN TOPFLIGHT PLAYERS SUCH AS-F- DWYANE WADE CHRIS BOSH
47. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Vince_Carter-D-HE WAS ONE OF ONLY THREE -E-PLAYERS ALONG WITH-F- AND KOBE BRYANT THIS PAGE WAS LAST MODIFIED ON 28 SEPTEMBER 2010 AT 2234
48. LEBRON JAMES-A-NBA-B-1-C-http://en.wikipedia.org/wiki/2008_NBA_All-Star_Game-D-CLEVELAND CAVALIERS SMALL FORWARD -E-AND MIAMI HEAT SHOOTING GUARD DWYANE MAY 22 2006 HTTPSPORTSESPNGOCOM-F-NEWSSTORYID2453969

Figure 7.3: Result Set3

49. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Michael_Beasley-D- ANTHONY WHO HAD 22 DOUBLEDOUBLES IN HIS ONLY SEASON AT SYRACUSE IN 200203 ON APRIL 14 2008 BEASLEY ANNOUNCED THAT HE WOULD ENTER THE -E-DRAFT AND THUS ALLOWING THEM TO SIGN FREE AGENTS-F- AND CHRIS BOSH
50. 51. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Carlos_Boozer-D-BOOZER DECLARED FOR THE 2002 -E-DRAFT RELINQUISHING HIS FINAL YEAR OF NCAA 155 PPG AND 114 RPG HIS SECOND YEAR WHILE PLAYING ALONGSIDE-F- THE NBA39S TOP TEN PERFORMERS IN FIELD GOAL PERCENTAGE SIX TIMES AND HAS
52. LEBRON JAMES-A-NBA-B-2-C-
http://en.wikipedia.org/wiki/2008%25E2%2580%259309_NBA_season-D-ON JANUARY 22 2009 ALONZO MOURNING RETIRED FROM THE -E-AFTER 15 SEASONS ON FEBRUARY 7 2009-F-39S 52POINT TRIPLEDOUBLE AGAINST THE NEW
53. LEBRON JAMES-A-NBA-B-1-C-
http://en.wikipedia.org/wiki/2007%25E2%2580%259308_NBA_season-D-ON FEBRUARY 28 2008 CLEVELAND CAVALIERS39 -E-BECAME THE YOUNGEST FROM JANUARY 29 2008 TO MARCH 18 2008 THE HOUSTON ROCKETS WON 22 CONSECUTIVE HORNETS ACQUIRE WELLS AND JAMESJACKSON REUNITED WITH ADELMAN-F-
54. 55. LEBRON JAMES-A-NBA-B-1-C-http://en.wikipedia.org/wiki/2009_NBA_All-Star_Game-D-PREVIOUS YEAR39S ALLSTAR GAME MVP -E-LED THE EAST WITH 20 POINTS BUT WAS UNABLE TO-F-COM TURNER SPORTS INTERACTIVE INC JANUARY 22
56. LEBRON JAMES-A-NBA-B-2-C-
http://en.wikipedia.org/wiki/National_Basketball_Association_Nielsen_ratings-D-WHILE THE 2007 -E-FINALS FEATURED-F- MAKING HIS FIRST APPEARANCE TNT39S RATINGS FOR SECOND ROUND PLAYOFF GAMES WERE UP 22 PERCENT FROM THE
57. 58. 59. LEBRON JAMES-A-NBA-B-1-C-
http://en.wikipedia.org/wiki/2005%25E2%2580%259306_Cleveland_Cavaliers_season-D-ON JANUARY 21 -E-BECAME THE YOUNGEST PLAYER IN-F- HISTORY TO SCORE 5000 CAREER POINTS 21 YEARS 22 DAYS ON MARCH 29 LEBRON JAMES BECAME THE
60. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/Shaquille_O%27Neal-D-IN HIS THIRD SEASON O39NEAL LED THE -E-IN SCORING WITH AN AVERAGE OF 293 MOTTO IS VERY SIMPLE WIN A RING FOR THE KINGQUOT REFERRING TO-F- HOWEVER ON JUNE 22 2008 O39NEAL FREESTYLED A DISS RAP ABOUT BRYANT IN A
61. 62. 63. LEBRON JAMES-A-NBA-B-1-C-http://en.wikipedia.org/wiki/List_of_2009_all-decade_Sports_Illustrated_awards_and_honors-D-SF -E-CAVALIERS PF TIM DUNCAN SPURS C SHAQUILLE O39NEAL LAKERSHEATSUNS QUOT2000S THE DECADE IN SPORTS-F-HIGHLIGHTS AND LOWLIGHTSQUOT QUOT2000S BEST NEW STADIUMSQUOT SPORTS ILLUSTRATED DECEMBER 22 2009
64.

Figure7.4: Result Set4

Repeated middle term: BECAME THE YOUNGEST

number of times repeated: 3

Longest common url:

http://en.wikipedia.org/wiki/2007%25E2%2580%259308_Cleveland_Ca

valiers_season

Longest common prefix: ON DECEMBER 17 2007

Longest common suffix: PLAYER TO SCORE ON FEBRUARY 22 2008 LEBRON JAMES GRABBED HIS 2500TH REBOUND AS A CAVALIER

Finding other parameters related to term: ON DECEMBER 17 2007 +*BECAME THE YOUNGEST*+ PLAYER TO SCORE ON FEBRUARY 22 2008 LEBRON JAMES GRABBED HIS 2500TH REBOUND AS A CAVALIER

site:http://en.wikipedia.org/wiki/2007%25E2%2580%259308_Cleveland_Cavaliers_season

Results: NIL

Explanation of output:

1. LEBRON JAMES-A-NBA-B-2-C-http://en.wikipedia.org/wiki/LeBron_James-D-YAHOO NEWS HTTPSPORTSYAHOOCOM-E-NEWSSLUGAPLEBRONSBOOKAMPPROVAPAMPTYPE LGNS LIVINGSTON BILL JULY 22 2009 QUOTNEW-F- BOOK TELLS OF A

The substrings are separated using identifiers –A-, -B-, -C-, -D-, -E- and –F-

Parameter 1: LEBRON JAMES

Parameter 2: NBA

Order of parameters in the result : 2 (1 implies parameter1 came ahead of parameter2 in the result and 2 implies the opposite)

Link to webpage where the content was found : http://en.wikipedia.org/wiki/LeBron_James

Middle Term : YAHOO NEWS HTTPSPORTSYAHOOCOM

Prefix :

NEWSSLUGAPLEBRONSBOOKAMPPROVAPAMPTYPE GNS LIVINGSTON BILL JULY 22 2009 QUOTNEW

Suffix : BOOK TELLS OF A

And the most common repeated term was “BECAME THE YOUNGEST” and obviously there were no results found for further iterations. This happens often with DIPRE because the principle used in determining the next tuples is not very robust.

CHAPTER 4

RESULTS AND EVALUATION

In his experiment, Brin used DIPRE [1] to build a database of {Authors, Titles}. He started off with an initial sample of 5 books. Eventually, after numerous iterations and a bit of manual intervention to remove bogus data retrieved, he claimed to have built a repository of around 15,000 unique books. He suggested that the same principles can be applied to other domains like movies, music, restaurants, etc. Brin also mentioned that a sophisticated version of DIPRE may also be able to extract people directories, product catalogs, and more.

We wanted to experimentally test and evaluate some of these claims about the abilities of Brin's DIPRE algorithm. As part of the experiment, we developed our own implementation of DIPRE in a java program which uses Google AJAX Search API as the web crawling tool.

This program developed as part of this project can be used to get relations between any two parameters. However, in this experiment we decided to focus on music and build a repository of {singers, songs}; similar to Brin's experiment which built a repository of {Authors, Titles}.

We started the experiment with only one set of related parameters.

Parameter1: Bryan Adams

Parameter2: song

SearchSite: Wikipedia.org

The query “"%22 " + BRYAN+ADAMS + "%22" + "+" + "%22 " + SONG + "%22" + " site:" + WIKIPEDIA.ORG” resulted in a list of songs by Bryan Adams.

The longest prefix was “is a”, most repeated middle term was “written by” and there was no longest suffix.

When the query “IS A +*WRITTEN BY*+ site:http://en.wikipedia.org/wiki/” was passed to the search API, it resulted in more singers and their songs.

Iteration 1

Parameters:

Parameter1: Bryan Adams

Parameter2: song

SearchSite: Wikipedia.org

Analysis of Search Results:

Repeated middle term: WRITTEN BY

Longest common url: http://en.wikipedia.org/wiki/

Longest common prefix: IS A

Longest common suffix:

Result List of songs by Bryan Adams:

Everything I do I do it for you

Mysterious ways

Waking Up the Neighbours

Summer of 69

Heaven

Christmas Time

Run To You

Relevant parameters with similar relation:

Search Query to get more tuples: IS A + *WRITTEN BY* + site:http://en.wikipedia.org/wiki/

Tuple 2a: Parameter1: SONG, Parameter2: COCHRANE, searchsite: http://en.wikipedia.org/wiki/

Tuple 2b: Parameter1: SONG, Parameter2: KNIGHT AND MIKE CHAPMAN, searchsite: http://en.wikipedia.org/wiki/

Tuple 2c: Parameter1: SONG, Parameter2: DYLAN, searchsite: http://en.wikipedia.org/wiki/

Tuple 2d: Parameter1: SONG, Parameter2: EVANS, searchsite: http://en.wikipedia.org/wiki/

Tuple 2e: Parameter1: SONG, Parameter2: musician Tom Johnston, searchsite: http://en.wikipedia.org/wiki/

Tuple 2f: Parameter1: SONG, Parameter2: ADAMS AND JIM VALLANCE, searchsite: http://en.wikipedia.org/wiki/

Iteration 2

(2a)

Parameters:

Parameter1: COCHRANE

Parameter2: SONG

SearchSite: http://en.wikipedia.org/wiki/

Analysis of Search Results:

No identifiable Repeated middle term.

Result List of songs by COCHRANE:

Life is a highway

(2b)

Parameters:

Parameter1: KNIGHT AND MIKE CHAPMAN

Parameter2: SONG

SearchSite: http://en.wikipedia.org/wiki/

Analysis of Search Results:

No identifiable Repeated middle term.

Result List of songs by KNIGHT AND MIKE CHAPMAN:

Silent Wings

(2c)

Parameters:

Parameter1: DYLAN

Parameter2: SONG

SearchSite: http://en.wikipedia.org/wiki/

Analysis of Search Results:

No identifiable Repeated middle term.

Result List of songs by DYLAN:

I shall be released

All along the watch tower

(2d)

Parameters:

Parameter1: EVANS

Parameter2: SONG

SearchSite: http://en.wikipedia.org/wiki/

Analysis of Search Results:

Repeated middle term: FEATURED

Longest common url: http://en.wikipedia.org/wiki/

Longest common prefix:

Longest common suffix:

Result List of songs by EVANS:

Trapped in the closet

This is home

Relevant parameters with similar relation:

Search Query to get more tuples: Song + *FEATURED* + site:http://en.wikipedia.org/wiki/

Tuple 3a: Parameter1: SONG, Parameter2: AKON, searchsite: http://en.wikipedia.org/wiki/

Tuple 3b: Parameter1: SONG, Parameter2: SLEEPY BROWN, searchsite: http://en.wikipedia.org/wiki/

Tuple 3c: Parameter1: SONG, Parameter2: TIMBALAND, searchsite: http://en.wikipedia.org/wiki/

(2e)

Parameters:

Parameter1: musician Tom Johnston

Parameter2: SONG

SearchSite: http://en.wikipedia.org/wiki/

Analysis of Search Results:

No identifiable Repeated middle term.

Result List of songs by MUSICIAN TOM JOHNSTON:

Nil

(2f)

Parameters:

Parameter1: ADAMS AND JIM VALLANCE

Parameter2: SONG

SearchSite: http://en.wikipedia.org/wiki/

Analysis of Search Results:

Repeated middle term: WRITTEN BY BRYAN

Longest common url: http://en.wikipedia.org/wiki/

Longest common prefix: IS A

Longest common suffix:

Result List of songs by ADAMS & JIM VALENCE:

Summer of 69

Reckless

Relevant parameters with similar relation:

Search Query to get more tuples: Song +*WRITTEN BY BRYAN*+ site:http://en.wikipedia.org/wiki/

Tuple 3d: Parameter1: SONG, Parameter2: BRYAN-MICHAEL COX, searchsite: http://en.wikipedia.org/wiki/

Tuple 3e: Parameter1: SONG, Parameter2: BRIAN WILSON, searchsite: http://en.wikipedia.org/wiki/

Tuple 3f: Parameter1: SONG, Parameter2: BRIAN MOLKO, searchsite: http://en.wikipedia.org/wiki/

Tuple 3g: Parameter1: SONG, Parameter2: BRIAN MAY, searchsite: http://en.wikipedia.org/wiki/

Tuple 3h: Parameter1: SONG, Parameter2: BRYAN FERRY, searchsite: http://en.wikipedia.org/wiki/

Similarly, iteration 3 is also performed with these 8 tuples to get more songs and singers. Eventually after 4 iterations following 43 results were retrieved:

1.	Everything I do I do it for you	by Bryan Adams
2.	Mysterious ways	by Bryan Adams
3.	Waking Up the Neighbours	by Bryan Adams
4.	Summer of 69	by Bryan Adams
5.	Heaven	by Bryan Adams
6.	Christmas Time	by Bryan Adams
7.	Run To You	by Bryan Adams
8.	Reckless	by Bryan Adams
9.	Life is a highway	by Cochrane
10.	Silent Wings	by Knight and Mike
11.	I shall be released	By Bob Dylan
12.	All along the watch tower	By Bob Dylan
13.	Trapped in the closet	by Evans
14.	This is home	by Evans
15.	Freedom	by Akon

16.	Trouble Nobody	by Akon
17.	The Sweet Escape	by Akon
18.	I Wanna Love You	by Akon
19.	We Don't Care	by Akon
20.	Pot of Gold	by Akon
21.	Hypnotized	by Akon
22.	I Cant Wait	by Sleepy Brown
23.	Morning After Dark	by Timbaland
24.	Scream	by Timbaland
25.	Release	by Timbaland
26.	Good Vibrations	by Brian Wilson
27.	The Warmth of The Sun	by Brian Wilson
28.	Sleeping with Ghosts	by Brian Molko
29.	Battle for the Sun	by Brian Molko
30.	Pure Morning	by Brian Molko
31.	Without Im Nothing	by Brian Molko
32.	Tear The Signs Down	by Brian Molko
33.	Made in Heaven	by Brian May
34.	A Hard Rain	by Brian Ferry
35.	All Along the Watchtower	by Brian Ferry
36.	Avalon	by Brian Ferry
37.	Another Time Another Place	by Brian Ferry
38.	Positively 4th Street	by Brian Ferry
39.	You Wont See ME	by Brian Ferry
40.	Shes Leaving Home	by Brian Ferry
41.	Street Life	by Brian Ferry
42.	I Put a Spell On You	by Brian Ferry
43.	Jealous Guy	by Brian Ferry

Usually, the results tend to wander off after 4 iterations. Robust filtering techniques are required to make sure the results stay relevant to the experiment. Manual interventions are also mandated to make sure we get the right results.

For example, let's say after analyzing the results we get a most commonly repeated middle term but no longest_common_prefix or a longest_common_suffix. In such cases, since we are looking specifically for songs in this particular example, we can add "song" as a prefix or a

suffix in the search query and try to induce results related the experiment. Brin did similar adjustments to the algorithm and there have been many researchers who proposed similar enhancements to DIPRE.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

DIPRE has a simple generalization principle to control its expansion process based on longest matching prefix/suffix and most often repeated middle term. This does not always pick high quality tuples for next set of iterations. This leads to bad results and high recalls. Sometimes, there cannot be any iteration possible at all as the prefix, middle term and suffix found after the very first query can be erratic.

The primary difference between this experiment and the one did by Brin was that Brin's focus was on extracting author&books from the internet. This experiment applies the same principles on a broader set and looks to retrieve relations between any two parameters.

During my experiments with various parameters, usually after a couple of iterations, there were a lot of patterns generated by many seemingly valid results. However, these patterns tend to generate bad results after a while making the system unstable. Unless we develop good filtering algorithms, it is difficult to scale up DIPRE algorithm to build practical knowledge databases.

Infact, research mentions that this instability is a common problem in many bootstrapping algorithms [3].

However, to Brin's credit, it has to be acknowledged that up to the point patterns started delivering divergent results; the data retrieved had

reasonable accuracy. His research has allowed others to come up with many models as enhancements to DIPRE; paving path for future research work in trying to make semi-supervised models better day by day.

5.2 Future Possibilities

There are a lot of practical applications that can be built based on machine learning. Examples could be to build knowledge bases, comparison-shopping [17], medical applications like counting redblood cells, recognition of cell tissues through microscopes to the detection of tumours in magnetic resonance scans and the inspection of bones and joints in X-ray images.

Recommendations for future enhancements to the current JAVA program.

1.Implement Snowball algorithm:This is an algorithm similar to DIPRE. While using DIPRE, system usually becomes unstable after a few iterations due to incorrect relations in results causing a snowball effect leading to bad patterns which introduce more bad relations. Snowball algorithm re-evaluates the patterns after each iteration and only the ones with highest confidence are kept for next iteration. Hence, Snowball gives more precise results. Unlabeled text as data: Feed unlabeled text to the algorithm instead of crawling web pages in the World Wide Web.

2.Using the algorithm, do a comparative study of results from various web search engines like Google AJAX Search API, Bing API and Yahoo BOSS API.Find a way to remove restrictions on usage of Google Search API.We can get only 64 results from a search query using Google AJAX Search API.Google sometimes suspects the queries from our application as automated queries from robots/spiders and stops responding.

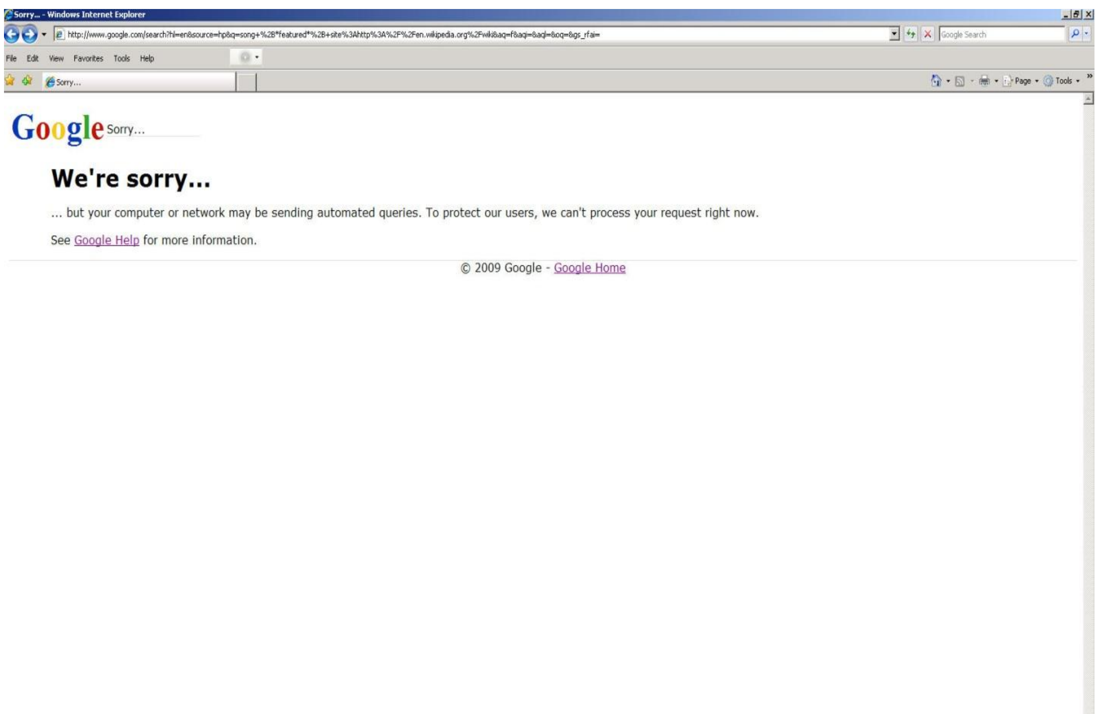

Figure 8: Google Query Restriction

Computer Science Department at UNLV should make a formal request to Google Inc. to remove these limitations for a certain "Google AJAX Search API Key". This way, we can pass this key in the search queries from our applications and Google would know that UNLV Computer

Science Department is sending the query for educational purpose. A few Universities have done the same and Google Inc. has good-heartedly provided them with greater access.

BIBLIOGRAPHY

[1]. Sergey Brin. A Thesis on *"Extracting Patterns and Relations from the World Wide Web"*, Computer Science Department, Stanford University.

[2]. Wikipedia, the free Encyclopedia on *"Information Extraction".* http://en.wikipedia.org/wiki/Information_extraction

[3]. Dmitri Bobrovnikoff. A Thesis on *"Semantic Bootstrapping with a Cluster-Based Extension to DIPRE",* Computer Science Department, Stanford University.

[4]. Lei Xia. A Thesis on *"Adaptive Relationship Extraction by Machine Learning"*, Department of Computer Science, University of Sheffield.

[5]. Doug Downey, Oren Etzioni, Stephen Soderland, and Daniel S. Weld. A thesis on *"Learning Text Patterns for Web Information Extraction and Assessment"*, Department of Computer Science and Engineering, University of Washington

[6]. Miao Chen, Xiaozhong Liu and Jian Qin. Proceedings from *"International Conference on Dublin Core and Metadata Applications"* in 2008.

[7]. Ellen Riloff. *"Automatically constructing a dictionary for information extraction Tasks",* in proceeding of the eleventh national conference on artificial intelligence

[8]. Bora Gazen and Steven Minton. *"Overview of AutoFeed: An Unsupervised Learning System for Generating Webfeeds".*

[9]. R Yangarber, R Grishman, Parsi Tapamainen and S Huttunen. *"Unsupervised discovery of scenario-level patterns for information extraction",* in Proceedings of Conference on Applied Natural Language Processing ANLP-NAACL, pages "282–289", Seattle, WA, 2002.

[10]. O. Etzioni, M. Cafarella, D. Downey, S. Kok, A. Popescu, T. Shaked, S. Soderland, D. Weld, and A. Yates. *"Unsupervised named-entity extraction from the web: An experimental study".*

[11]. Eugene Agichtein and Luis Gravano. *"Snowball: Extracting relations from large plaintext collections",* in Proceedings of the Fifth ACM International Conference on Digital Libraries, 2000.

[12]. Google AJAX Search API.
http://code.google.com/apis/ajaxsearch/documentation/
[13]. Bing API. http://msdn.microsoft.com/en-us/library/dd251056.aspx

[14]. Yahoo BOSS API.
http://developer.yahoo.com/search/boss/boss_guide/

[15]. NewYork Times Article Search API.
http://developer.nytimes.com/docs/article_search_api/

[16] JSON. http://www.json.org/

[17]. Line Ekvil. A Survey on *"Information Extraction from World Wide Web*", July 1999.

VITA

Graduate College
University of Nevada, Las Vegas

Praveena Mettu

Degrees:
Bachelor of Technology in Computer Science and Engineering, 2008
Jawaharlal Nehru Technological University, India

Thesis Title: Pattern Extraction from the World Wide Web

Thesis Examination Committee:
Chair Person, Dr. Kazem Taghva, Ph.D.
Committee Member, Dr. Ajoy K. Datta, Ph.D.
Committee Member, Dr. Laxmi P. Gewali, Ph.D
Graduate College Representative, Dr. Muthukumar Venkatesan, Ph.D