A PATTERN-BASED ANNOTATION APPROACH: AN ONTOLOGY-DRIVEN ROTE EXTRACTOR FOR PATTERN DISAMBIGUIATION

by

SHENG YIN

(Under the Direction of Ismailcem Budak Arpinar)

ABSTRACT

One difficulty that prevents a machine from searching, retrieving and processing web content through the World Wide Web (WWW) is that most web content is presented in natural language, which cannot be processed by a machine. The current pattern-based annotation approaches can generate patterns for a given relation from unrestricted text, and they can use those generalized patterns to extract related concepts, that have the same relation, from other text. However, those approaches all have one problem unsolved: the pattern ambiguity problem. Our approach can generate lexical patterns for a particular relation from unrestricted text. Then patterns can be used to recognize concepts which have the same relation in other text. We proposed an ontology-driven pattern disambiguation process. This process can dramatically improve the performance of existing pattern-based annotation approaches.

INDEX WORDS: Edit-Distance, Lexical Pattern, OWL, Ontology Query, Pattern disambiguation, Pattern Generalization, POS tagging, NER and Semantic Web

A PATTERN-BASED ANNOTATION APPROACH: AN ONTOLOGY-DRIVEN ROTE EXTRACTOR FOR PATTERN DISAMBIGUIATION

by

SHENG YIN

B.S., University of Shanghai for Science and Technology, China, 2001

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2009

A PATTERN-BASED ANNOTATION APPROACH: AN ONTOLOGY-DRIVEN ROTE EXTRACTOR FOR PATTERN DISAMBIGUIATION

by

SHENG YIN

Major Professor:	Ismailcem Budak Arpinar
Committee:	Khaled Rasheed Prashant Doshi

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
December 2009

ACKNOWLEDGEMENTS

I would like to thank everyone who helped make this thesis possible, especially, Dr. Ismailcem Budak Arpinar for his guidance and direction throughout my research and his efforts to make the project better. Thanks also to Dr. Khaled Rasheed and Dr. Prashant Doshi for their suggestions and recommendations, which enhanced many aspects of this work. I would also like to thank Samir Tartir, for his patience and help throughout my experience with the Large Scale Distributed Information Systems Lab (LSDIS). Finally, I would like to thank all my friends at the Computer Science department for making it fun to spend time with them.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

CHAPTER 1

INTRODUCTION

Due to the increasing amount of information contained in the WWW, an automatic process is needed to fulfill different tasks, such as to search, retrieve and process web content. However, most web content is written in natural language for humans to read. A machine cannot process web content automatically because there is a lack of implicit knowledge for a machine to solve language ambiguities. Various techniques including Machine Learning [1], Memory-based Language Processing [2], Word Sense Disambiguation [3] and Empirical Technique [4] are proposed for natural language processing (NLP). However, these techniques lack semantics, which restrict them in a limited scale.

The Semantic Web is an extension of the current web [5], which provides machine readable content by explicitly adding semantic annotations to certain words, pages and other web sources. The semantic annotations can be understood by machines, which allow machines to process web content automatically. The semantic annotations also enable automatic machine-to-machine interaction.

For example, Swoogle [6] is an indexing and retrieval system for semantic web. It computes a rank to measure the importance of a semantic web document, by using extracted metadata from each discovered document. The Semantic Web can also be used in information retrieval [7] and information inference [8].

The Semantic Web community developed several annotation languages, which can be used to tag concepts and relations explicitly. Two of them, the Resource Description Framework (RDF[1]) and OWL Web Ontology Language (OWL[2]), are accepted as standard. The annotation standard is used to model an ontology, which is "an explicit specification of a conceptualization" [9]. But adding annotation tags into vast amounts of existing web content and upcoming content is an expensive and time consuming task, especially when it has to be done manually by specialized experts. In an effort to annotate web content automatically, several solutions are proposed.

One of the proposed systems is KIM [10], which could annotate and add hyperlinks to named entities (NE) in textual documents. In order to do that, KIM uses an entity knowledge base to retrieve NE instead of words from documents. The knowledge base can be created automatically by KIM. A seed ontology is created which models real-world entity classes, and it is complemented with extensive instance knowledge. Then KIM will maintain and extend the seed ontology through information extraction process.

The on-line encyclopedia Wikipedia[3] is the largest and most popular general reference source on the Internet. Millions of articles on Wikipedia have been written collaboratively by volunteers around the world, and almost all of its articles can be edited by anyone who can access the site. Wikipedia lowers the technical barriers for adding semantic annotations to documents. Wikipedia provides the semantic annotations to users by simply assigning a type to

1 http://www.w3.org/RDF/
2 http://www.w3.org/TR/owl-guide/
3 http://www.wikipedia.org/

hyperlinks. Based on the type of hyperlink, Wikipedia can transform those hyperlink annotations into Semantic Web annotations, such as RDF or OWL.

Although KIM and Wikipedia achieve great success, they still suffer several limitations. KIM uses a basic assumption that a document is characterized by a bag of tokens, which constitute its content. It disregards the document's structure information. Wikipedia needs volunteers around the world to contribute time and effort to add and maintain articles. But Wikipedia cannot force volunteers to delve into a specific domain which they are not interested in. Therefore, Wikipedia was a passive annotation approach.

Several pattern-based annotation procedures [11-15] have been described to identify concepts from free text for a given relation. They can learn lexical patterns for a specific relation from a free textual corpus, and then apply the learned patterns to other free textual corpora to extract related concepts which have the same relation. Those procedures can be used for different purposes: Riloff and Schmelzenbach (1998) use it to extract concepts from unannotated text; Soderland (1999) uses it to do information extraction from semi-structured and free text; and Brin (1999) uses it to extract concept pairs for a given relation. Compared with KIM, pattern-based annotation approaches consider not only tokens, but also document structure. Compared with Wikipedia, they can actively learn patterns for any relation. Therefore, they can be used in a widespread domain.

We redefined a new pattern representation for pattern-based annotation approaches. We address the problem caused by Named Entity Recognition (NER). NER helps pattern-based annotation to identify named entities, such as persons, organizations and locations. Therefore, the

annotation approach can generate patterns based on named entities instead of specific instances. Currently, NER can identify persons, organizations and locations very well. But you have to train NER separately to identify any other named entities (dates, address, books and songs). Even so, NER's accuracy for identifying some named entities is still low. Therefore, existing pattern-based annotation approaches would make mistakes if NER identifies named entities incorrectly. The new pattern representation allows current annotation approaches to work correctly with wrong identified named entities.

In current approaches, ambiguities can be caused by two kinds of patterns: patterns which contain wildcards, and patterns which can be used to indicate several different relations. We propose an inference procedure, in which an ontology is used as an inference base, to help both kinds of patterns to solve ambiguity problems. This inference procedure can be used to find any relation existing between two related concepts. Our approach can use the found relation to solve ambiguity problems.

We proposed a new process to generalize patterns from a collection of patterns. Our process records an applied number for each pattern, and this process tries to generate a pattern from those patterns which can be applied to more sentences. We found that the patterns generated by this process are more general. Consider patterns generated for the birth-year relation, which are shown in APPENDIX B. The top eight patterns cover 52% sentences used in the training corpus.

Compared with the most recent approach [21], our approach achieved three improvements. The first is that our approach can identify four data formats correctly. NER cannot recognize the year part and the month and day part in a data. This problem could increase generalized patterns,

especially for those patterns related to data. Our approach pre-defined two named entity classes: YYYY and MMDD. By using those two named entities, our approach can create fewer patterns for data related relations, such as the birth-year relation.

The second is that our new pattern representation can reduce mistakes caused by NER. In the most recent approach, the pattern generalization process and the pattern application process are based on extracted named entities information. However, NER cannot work well for some un-defined classes, such as books. Our pattern representation requires that the content window size must be greater than 0 (Section 3.3.3.2). This requirement can guarantee that patterns can be generated and applied correctly even named entities are not correctly extracted by NER.

The last improvement is that our patterns disambiguation process can achieve good precision rate based on a well defined ontology. There are two existing solutions for patterns disambiguation. One is to discard those patterns; the other is to choose one possible relation based on extracted named entities. Our approach created an ontology for five relations, and the result shows that the precision rate can be improved by using the ontology.

CHAPTER 2

BACKGROUND

This chapter introduces fundamental techniques used in this project, particularly those related to the Semantic Web and lexical pattern-based annotation approaches. We begin with an overview of the Semantic Web itself, followed by semantic languages and technologies. Natural language processing and current lexical pattern-based annotation approaches are briefly explained.

2.1 Semantic Web

The Semantic Web is an extension of the current web in which "the meaning of information and services on the web are defined, making it possible for the web to understand and satisfy the requests of people and machines to use the web content"[4]. It enhances information sharing, searching and retrieving among software applications.

The rise of the Semantic Web is due to limitations of the current web. One limitation is that most web content is written in natural language. The web content can be understood by humans easily, but a machine cannot understand it. This limitation challenges information search and retrieval. For example, in order to answer the question "Show me all Oil and Energy companies around the world, whose stock price is higher than $ 80, have branches in India, but don't have branches in Iran," a machine needs to know some knowledge, such as stock value, Oil and

[4] http://en.wikipedia.org/wiki/Semantic_Web

Energy companies and country information. However, the web content cannot provide this kind of knowledge directly because the existing web infrastructure is not designed for that purpose. For instance, the Hypertext Markup Language (HTML) is only used for representing web content and linking existing online documents for human readability and understanding. Therefore, some expressive languages are needed to provide machine readable content.

2.1.1 Languages and Ontology

The Semantic Web community has proposed a lot of formal specifications, data interchange formats and notations to provide a formal description of concepts, terms and relationships within a given knowledge domain. RDF, RDF Schema (RDFS) and OWL are introduced in the following paragraphs.

RDF is an official Semantic Web specification for metadata models. It gives a standard way, like other conceptual modeling approaches such as Entity-Relationship and Class diagrams, to specify resources. RDF makes statements about resources in the form of *subject-predicate-object* expressions. Those expressions are called triples. The subject and the object denote two resources, and the predicate denotes an aspect of resources and expresses a relationship between the subject and the object. For example, the sentence "The Earth has a satellite the moon," can be represented in a triple: *the Earth-has a satellite-the moon*. There are two data interchange formats used to describe RDF models: one is the Extensible Markup Language (XML) and the other one is Notation 3. [5]

[5] http://www.w3.org/DesignIssues/Notation3

The XML language is often called simple RDF because it was introduced among other W3C specifications defining RDF. It is important to distinguish the XML language from RDF. XML allows users to define arbitrary tags for expressing structure in data. But RDF can only use pre-defined tags to represent triple statements. Figure 1 shows three XML representations for the statement “The Earth has a satellite the moon”. Figure 2 shows the RDF representation, in which the Earth is an instance of the class Planet and the Moon is an instance of the class Satellite, for the same statement.

```
<planet name="Earth">
    <satellite>Moon</satellite>
</planet>

<satellite name="Moon">
    <planet>Earth</planet>
</satellite>

< PlanetSatellite>
    <planet>Earth</planet>
    <satellite>Moon</satellite>
</ PlanetSatellite>
```

Figure 1: “The Earth has a satellite the moon.” represented in XML

```
<rdf:Description rdf:about="Planet">

    <rdf:has_name>Earth</rdf:has_name>

    <rdf:hasSatellite rdf:resource="#Moon"/>

</rdf:Description>
```

Figure 2: “The Earth has a satellite the moon.” represented in RDF

A Uniform Resource Identifier (URI) is used by RDF to identify a resource. URI is the universal identification used by other software applications. Figure 3 shows the RDF graph model with URI for the previous example.

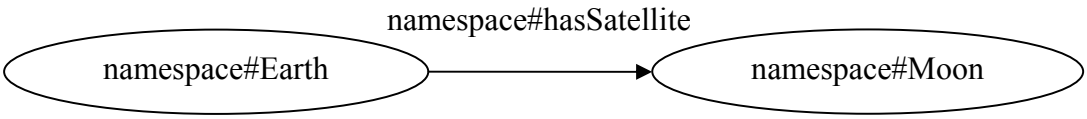

Figure 3: RDF graph model

RDFS is an extensible knowledge notation language designed to structure RDF resources by providing basic tags to describe classes. The benefit of an RDF Schema is that it facilitates inference on RDF data and enhances data searching. RDFS uses XML data format to define classes, subclasses and properties associated with classes. Besides XML tags, RDFS still has several pre-defined tags. Those tags are explained in the next paragraph (the specification for RDFS can be found in W3C.org).

rdfs:Class is used to declare a resource as a class, and *rdfs:subClassOf* is used to declare class hierarchy. The RDFS specification describes *rdf:Property* as the class of RDF properties. *rdfs:range* is used to indicate a property's value type. *rdfs:domain* is used to indicate the classes with which a property will be used. Other RDFS tags include: *rdfs:seeAlso*, which is an instance of *rdf:Property*, used to indicate a resource that might provide additional information about the subject resource; *rdfs:label,* which is an instance of *rdf:Property*, used to provide a human-readable name of a resource; *rdfs:comment*, which is an instance of *rdf:Property*, used to provide a human-readable description of a resource.

OWL is another knowledge representation language. The purpose of OWL is identical to RDFS - to provide an XML vocabulary to define classes, class properties and class relationships. Compared with RDFS, OWL has three new features: (1) OWL provides a property, *owl:sameIndividualAs*, for indicating that two resources are the same. (2) Also OWL provides elements to construct class hierarchies, which can be used to dynamically discover relationships. (3) OWL provides the capability to specify that a subject can have only one value by using a cardinality value. Currently, OWL has three sublanguages designed for different users to use.

- *OWL Lite* provides a classification hierarchy and simple constraint features to users. For example, *OWL Lite* only permits cardinality values of 0 or 1.
- *OWL DL* supports those users who want the maximum expressiveness. All elements in *OWL DL* are guaranteed to be computed, and all computations will finish in finite time for reasoning systems.
- *OWL Full* provides maximum expressiveness and the syntactic freedom of RDF. But it

In the Semantic Web, an ontology is a formal representation of a set of concepts within a domain and the relationships among those concepts. It defines the domain knowledge by providing concepts, properties associated with those concepts, and relations among concepts. For example, taxonomies on the web, catalogs for on-line shopping and domain-specific standard terminologies are ontologies. Ontologies share common understanding of the structure of information among people or among software agents, enable reuse of domain knowledge, and separate domain knowledge from the operational knowledge. In reality, ontologies have a high amount of information, and they undergo a rapid evolution. Therefore, the automatic acquisition of the ontologies is highly desired.

Several approaches are proposed for automatic or semi-automatic acquisition of the ontologies from unstructured sources. OntoGen [16] is a system which extracts topics covered by large corpora of documents and helps users to organize them into a topic ontology. In that ontology, topics are connected with different relations, and each topic includes all related documents. A new mechanism [17] is proposed by using the existing self-organizing tree algorithm (SOTA) to construct a hierarchy, and an automatic concept selection algorithm is used to find an appropriate concept for each node in the hierarchy. Another method [18] can automatically generate classifications of gene-product functions using bibliographic information.

2.2 Natural Language Processing

Natural language processing (NLP) is a field of computer science and linguistics concerned with the interactions between computers and human languages. It includes two directions: one is how to convert computer readable information into readable human language, and the other is to convert human language sentences into more formal representations for computer programs to manipulate. In reality, many problems apply to both directions. For example, a computer must be able to model the structure of words for understanding an English sentence, and a grammatically correct English sentence can be created based on any given structure of words. Speech segmentation, text segmentation, part-of-speech tagging, named entity recognition and word sense disambiguation are problems existing in NLP. We explain two most used tools, part-of-speech tagging and named entity recognition, in the following sections.

2.2.1 Part-of-speech Tagging

Part-of-speech tagging (POS) is also called grammatical tagging or word-category disambiguation. It is the process of marking up the words in a text as corresponding to a particular part of speech, based on both word definition and its context. It is similar to the process taught to children, which identifies each word in a sentence as nouns, verbs, adjectives, adverbs, but POS is implemented by computers, not humans.

Two problems make POS difficult. The first is that traditional grammar classifies words into eight parts of speech: noun, verb, adjective, preposition, pronoun, adverb, conjunction and interjection. However, in NLP, there exist more categories and sub-categories. For example,

Brown Corpus[6] uses 82 categories. The other reason is that a word can be different parts of speech in different sentences. In the sentence "*Books are made of ink, paper, and glue.*" *books* is a noun. But it is a verb in the sentence "*Deborah waits patiently while Bridget books the tickets.*" Therefore, in natural languages, a large percentage of word-forms are ambiguous.

2.2.2 Named Entity Recognition

Named entity recognition (NER) is a subtask of information extraction that seeks to locate and classify atomic elements in text into pre-defined categories such as persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc. In the 1990s, NER was aimed for extraction from journalistic articles. Since 1998, the molecular biology, bioinformatics, and medical natural language processing communities have had an increasing interest in entity identification. Now names of genes and gene products are the most common entities for NER to locate.

2.3 Rote Extractors

In order to process natural language, we have to use annotation languages to solve the language ambiguities problem and to provide domain knowledge. Currently, RDF and OWL are two annotation languages accepted as standard. However, these languages can only manually tag existing web content. This could be too expensive as the amount of information contained in the WWW increases. Therefore, an automatic approach is needed to tag web content.

---

6 http://en.wikipedia.org/wiki/Brown_Corpus#Part-of-speech_tags_used

Several procedures have been described that can be trained on unannotated free text, and then can be used to extract structured information, such as concepts and relations, from the web content. Once the structured information is extracted from the web content, the annotation tag can be added to the structured information automatically. For example, Riloff and Schmelzenbach (1998) propose a procedure to generalize lexical patterns from free text and then apply the generalized patterns to extract concepts from any other unannotated text [19]; Soderland (1999) proposes a pattern based information extraction process to deal with semi-structured and free text [11]; and Mann and Yarowsky (2005) propose an approach to extract a set of biographic facts about target individuals from a collection of Web pages [12].

We focus on one pattern based approach, the Rote method [20], in this paper. This method can train extractors, which are called rote extractors, to look for special patterns. And then, rote extractors can use the patterns to recognize a certain relation between two concepts. For example, Ruiz-Casado, Alfonseca and Castelss (2006) train rote extractors to recognize relations in Wikipedia [21].

According to the definition of rote extractors [12], the probability of a relation $r(p,q)$ given the relation's surrounding context $A1pA2qA3$ is calculated by formula (1) below. That means, with a training corpus T, the probability that two elements (p,q) have the relation r can be calculated based on the two elements' surrounding context $A1pA2qA3$. The probability equals the total number of times that two related elements (x,y), where x and y have the relation r, appear with context $A1xA2yA3$ divided by the number of times that x appears in the same context with any other element.

$$P(r(p,q) \mid A1\,pA2qA3) = \frac{\sum_{x,y \in r} c(A1xA2yA3)}{\sum_{x,z} c(A1xA2zA3)} \qquad (1)$$

In the following sections, x is called the *hook*, and y is called the *target*. Therefore, two elements $(hook, t\arg et)$ have the relation r. In order to apply the Rote method, a widely used process is shown in Figure 4.

1. For a given relation, create a list of concept pairs as a seed. For example, select <Jim Rogers, 1942>, <Dan Brown, 1964> as the seed for a birth-year relation.
2. For each concept pair <*hook, target*> in the seed, collect a number of sentences containing both *hook* and *target* as the training corpus; collect sentences only containing *hook* as the testing corpus.
3. Extract surrounding context $A1hookA2t\arg etA3$ from each sentence in the training corpus. Generalize those extracted surrounding contexts into patterns.
4. Apply the generalized patterns to extract new concept pairs in the testing corpus.
5. Repeat the procedure for other relations.

Figure 4: A common process for the Rote method

CHAPTER 3

PATTERN GENERALIZATION

Our goal is to identify semantic concepts which have a given relationship in a large textual corpus. To do so, we develop an approach which can learn lexical patterns for a given relation. This procedure starts with a list of related concept pairs (each pair of concepts has the same relation). A training corpus is collected from the web by using this list. NLP tools are applied to each sentence in the corpus. After that, lexical patterns are generalized based on the context surrounding the concepts. The patterns can be applied to any other corpus to extract new concept pairs which have the same relation. Figure 5 shows an overview of this procedure carried out on one relation. The following subsections elaborate on corpus extraction, natural language processing and pattern generalization steps.

3.1 Textual Corpus Extraction

For a given relation, such as the birth-year relation, we can have a concept pair <*Dan Brown, 1964*>. In this concept pair, *Dan Brown* is the *hook*, and *1964* is the *target*. We submit a search query "*Dan Brown 1964*" to the Yahoo search engine, and download those pages which have at least one sentence containing both *Dan Brown* and *1964*. From the downloaded pages, sentences containing both *Dan Brown* and *1964* are put into the training corpus, and sentences only containing *Dan Brown* are put into the testing corpus.

A list of concept pairs for the relation should be created for textual corpus extraction. Each pair in the list could follow the procedure described above to download pages and extract sentences. The list can be created manually or created from some data source automatically. Brin (1999) creates a list that only has five author-book pairs [13]. Mini-biographies are used by Mann and Yarowsky (2005) as a pair list [12]. In our approach, we create lists from an ontology for five relationships: birth-year, death-year, country-capital, writer-book and singer-song relation. How to create the ontology is described in Section 4.2.1.

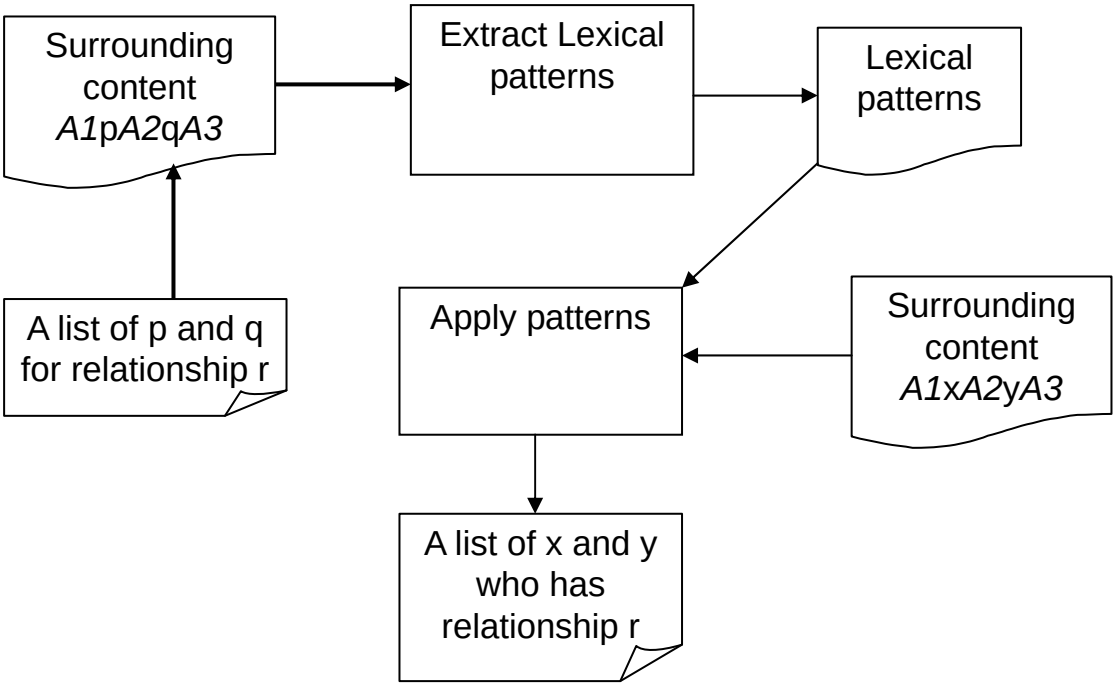

Figure 5: Overview of the procedure

Once the training corpus and the testing corpus are created, we apply two normalization processes to them. The first process is used to discard meaningless sentences. For example,

sentences used in an online advertisement, the web page's structure and programming language code are all classified as meaningless sentences. Figure 6 shows an example in which the *hook* (Michael Jackson) and the *target* (1958) are used as the web page menu. Therefore, this normalization process can guarantee that our corpus only contains readable natural language sentences.

Figure 6: Michael Jackson and 1958 are used as menu[7]

The other normalization process is to remove Unicode symbols. Unicode symbols are used widely, but their meanings are the same as ASCII symbols. Therefore, we only allow ASCII symbols to be used in our approach. By this limitation, we could reduce the total number of patterns. In order to do that, we try to convert Unicode symbols into their corresponding ASCII

[7] http://www.ifc.com/blogs/thedaily/2009/06/michael-jackson-1958---2009.php

symbols, and discard those sentences that are converted incorrectly. In the example below, “ can be converted into “ and ” can be converted into ” in sentence (1). But sentence (2) has to be discarded because there is no ASCII symbol that has the same meaning as ♥.

(1) “Kerry and other Vietnam veterans who went into politics were constantly worried about getting closure from Vietnam,” said Douglas Brinkley, author of “Tour of Duty: John Kerry and the Vietnam War, ” published this year.

(2) R.I.P ♥Michael Jackson♥ "DIED" (1958-2009)

3.2 Natural Language Processing

When the training corpus and the testing corpus are created, POS tool[8] and NER tool[9] are applied to them separately. For each sentence, the output created by POS and the output created by NER should be combined together. The combined output is used to generalize patterns in our approach. Sentence (3) below shows the output created by POS tool for the input “Janet Evanovich is an American writer, born in 1943, in New Jersey.” Each element in sentence (3) is represented by the format *word/POS tag*, where POS tag is the role of the word in the whole sentence. For example, *writer/NN* means the word *writer* works as a singular common noun. A small list of POS tags used by Stanford Parser is shown in Table 1. Beside those POS tags, we define a new tag, *Entity*, to label those named entities extracted by NER.

(3) Janet/NNP Evanovich/NNP is/VBZ an/DT American/JJ writer/NN ,/, born /VBN

[8] Stanford Parser 2009 (http://nlp.stanford.edu/software/lex-parser.shtml)
[9] Stanford NER 2009 (http://nlp.stanford.edu/software/CRF-NER.shtml)

in/IN 1943/CD ,/, in /IN New /NNP Jersey /NNP ./.

(4) <PERSON>Janet Evanovich</PERSON> is an American writer, born in 1943, in <LOCATION>New Jersey</LOCATION>.

Sentence (4) shows the output created by NER for the same input sentence. There are two named entities identified by NER: PERSON and LOCATION. Stanford NER could recognize persons, organizations and locations very well without any training. But it could not recognize different date formats. Therefore, we trained NER to recognize two named entities, YYYY and MMDD. They break a normal date into a year part and a month-day part. Table 2 shows examples for four different date formats supported by our approach. Any other date format is treated as invalid.

Table 1: POS tags

Tag	Definition
JJ	adjective
RB	adverb
CC	coordinating conjunction
DT	determiner/demonstrative pronoun
FW	foreign word
NN	common noun, singular
NNS	common noun, plural or dual
NNP	proper noun, singular

NNPS	proper noun, plural or dual
RP	particle
VBP	imperfect verb (***nb: imperfect rather than present tense)
VBN	passive verb (***nb: passive rather than past participle)
VBD	perfect verb (***nb: perfect rather than past tense)
UH	interjection
PRP	personal pronoun
PRP$	possessive personal pronoun
CD	cardinal number
IN	subordinating conjunction (FUNC_WORD) or preposition (PREP)
WP	relative pronoun
WRB	wh-adverb
,	punctuation, token is , (PUNC)
.	punctuation, token is . (PUNC)
:	punctuation, token is : or other (PUNC)
-LRB-	punctuation, token is ((PUNC)
-RRB-	punctuation, token is) (PUNC)

Table 2: Date format

Date format	YYYY and MMDD Entities

YYYY-MM-DD[10]	2009-10-01 <YYYY>2009</YYYY> - <MMDD>10-01</MMDD> 2006-2-1 <YYYY>2006</YYYY> - <MMDD>2-1</MMDD>
MM/DD/YYYY	10/01/2009 <MMDD>10/01</MMDD> / <YYYY>2009</YYYY> 2/1/2006 <MMDD>2/1</MMDD> / <YYYY>2006</YYYY>
U.S.A. style	March 8(th), 2008 <MMDD>March 8(th) </MMDD> , <YYYY>2008</YYYY> Mar. 8,2008 <MMDD>Mar. 8</MMDD> , <YYYY>2008</YYYY>
European style	8(th) March, 2008 <MMDD>8(th) March</MMDD> , <YYYY>2008</YYYY> 8 Mar.,2008 <MMDD>8 Mar. </MMDD> , <YYYY>2008</YYYY>

After POS and NER are applied to each sentence, two outputs are created. The next step is to combine them together. All extracted named entity tokens are set to *named entity/entity*. For example, *Janet/NNP Evanovich/NNP* will be changed into *Person/entity*; and *New /NNP Jersey*

[10] ISO 8601:2004

/NNP will be changed into *Location/entity*. The content for each named entity is also recorded. Figure 7 shows the result which combines sentence (3) and (4) together.

3.3 Pattern Generalization

In order to extract related concepts from the textual corpus, we need to learn patterns from the training corpus. Our approach uses a seed list to create the training corpus, and NLP tools are applied to the corpus. After that, we need to extract surrounding context around the related concepts from each sentence, and then each surrounding context is represented by a pattern. A modified edit-distance algorithm is used to guide the pattern generalization process. The following subsections explain surrounding context extraction, pattern representation and pattern generalization.

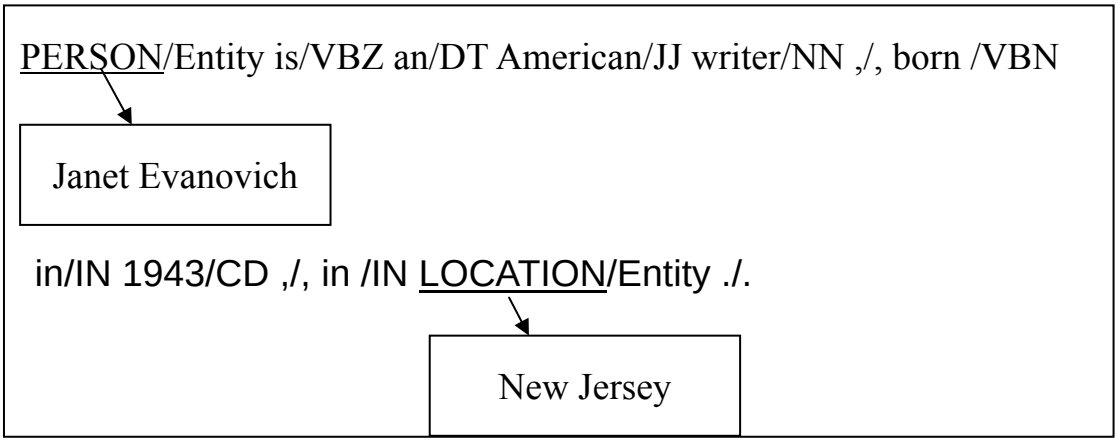

Figure 7: POS tag combines with named entities

3.3.1 Surrounding Context Extraction

In Section 2.3, we mention that the probability of two concepts (x, y) that have a relation r

can be calculated by their surrounding context $A1xA2yA3$. Therefore, for a training corpus T, in which each sentence contains two related concepts $(hook, target)$, the surrounding context $A1hookA2targetA3$ would provide clues to indicate which relation those two concepts have. For example, a small training corpus for the relation birth-year has sentences (5-13). Some patterns for the birth-year relation are obvious.

(5) James Patterson was born in 1947.

(6) Herbie Hancock was born in 1940.

(7) LaVern Baker was born in 1929.

(8) James Patterson was born in New York in 1947.

(9) LaVern Baker was born in Chicago in 1929.

(10) Max Lucado was born in San Angelo, Texas in 1955.

(11) James Patterson was born in 1947 in Newburgh New York.

(12) James Patterson was born in March 22, 1947 and is one of the biggest bestselling authors and novelists of all times and an award winning American Author.

(13) Janet Evanovich was born in 1943 in New Jersey and didn't begin writing until she was already married with children and in her thirties.

In this corpus, two related concepts for the birth-year relation are the *hook*, person, and the *target*, birth-year. Any words between *hook* and *target* are considered as $A2$. $A1$ is a small content window before *hook*, and $A3$ is a small content window after *target*. Alfonseca, Castells, Okumura and Ruiz-Casado (2006) propose five [20] as the length of $A1$ and $A3$. If the length of $A1$ is less than 5, BOS (Beginning of sentence) is used to identify the starting

point of one sentence. If the length of *A*3 is less than five, EOS (End of sentence) is used to show the end point of one sentence. The length of *A*1 and *A*3 is also called content window size (cWin).

In our approach, the surrounding content is extracted from the output created by NLP tools, which combines the POS tag and named entities. The POS tags and the extracted named entities can help pattern generalization process. If the length of the content between *hook* and *target* is greater than 2*cWin, *A*2 is the content after *hook*, whose length equals cWin and the content before *target* (length equals cWin) connected by a star wildcard. Otherwise, *A*2 is the content between *hook* and *target*. The definition for *A*1 and *A*3 is the same with Alfonseca's approach. Table 3 shows the extracted surrounding content for sentences (5-13).

Table 3: Surrounding content for sentences (5-13), cWin=2

Sentence	*A*1	*A*2	*A*3
5	BOS	was/VBZ born/VBN in/IN	./.
6	BOS	was/VBZ born/VBN in/IN	./.
7	BOS	was/VBZ born/VBN in/IN	./.
8	BOS	was/VBZ born/VBN * LOCATION/Entity in/IN	./.
9	BOS	was/VBZ born/VBN * LOCATION/Entity in/IN	./.
10	BOS	was/VBZ born/VBN * LOCATION/Entity in/IN	./.
11	BOS	was/VBZ born/VBN in/IN	in/IN LOCATION/Entity

12	BOS	was/VBZ born/VBN * MMDD/Entity ,/,	and/CC is/VBZ
13	BOS	was/VBZ born/VBN in/IN	in/IN LOCATION/Entity

3.3.2 Pattern Representation

Before the solution of Alfonseca, Castells, Okumura and Ruiz-Casado (2006), only lexical patterns are extracted from the training set. Ravichandran and Hovy (2002) have reported that wildcards might be harmful because patterns with wildcards are more general than normal lexical patterns. Wildcards improve the recall rate of patterns, but they also decrease the precision rate of them. For example, pattern (a) matches sentences (5-7), pattern (b) matches sentences (5-10), and pattern (c) can match all sentences (5-13). Therefore, wildcards are not allowed in the generalized patterns.

(a) PERSON was born in YYYY .

(b) PERSON was born * in YYYY .

(c) PERSON was born * in|, YYYY ,|.|in|and

In our approach, we proposed an ontology-driven pattern disambiguation process to solve the pattern ambiguity problem. Therefore, wildcards and disjunctions are allowed in our approach. The star is a wildcard (represents any word, has POS tag *) and the vertical bar | means a disjunction (means any adjectives can be found). Besides the wildcard and disjunction, there are other four special elements used in our representation:

1. BOS (beginning of sentence, with POS tag *BOS*)

2. EOS (end of sentence, with POS tag *EOS*)

3. <hook> (means this element is the *hook*, with POS tag <*hook*>)

4. <target> (means this element is the *target*, with POS tag <*target*>)

We use patterns to represent each surrounding context extracted from the training corpus. Table 4 shows the patterns for all surrounding context shown in Table 3. Therefore, before the pattern generalization process, the training corpus will be converted into patterns.

We need to point out that in our approach, the content window size, cWin, must be greater than zero. This improves pattern recall rate because NER cannot guarantee that all named entities can be recognized correctly. For example, after applying NER to sentence *"Vic Dickenson: Vic (Victor) Dickenson (August 6, 1906 - November 16, 1984) was an African-American jazz trombonist."*, only *Dickenson* is recognized as PERSON shown in sentence (14). Therefore, pattern (d) cannot match this sentence.

(14) Vic Dickenson: Vic (Victor) <PERSON>Dickenson</PERSON> (August 6, 1906 - November 16, 1984) was an African-American jazz trombonist.

(d) BOS/BOS PERSON/Entity :/: */* ,/, YYYY/Entity -/-

Table 4: Example patterns extracted from the surrounding content of sentences (5-13)

Sentence	Pattern
5	BOS/BOS <hook>/<hook> was/VBZ born/VBN in/IN <target>/<target> ./. EOS/EOS

6	BOS/BOS <hook>/<hook> was/VBZ born/VBN in/IN <target>/<target> ./. EOS/EOS
7	BOS/BOS <hook>/<hook> was/VBZ born/VBN in/IN <target>/<target> ./. EOS/EOS
8	BOS/BOS <hook>/<hook> was/VBZ born/VBN * LOCATION/Entity in/IN <target>/<target> ./. EOS/EOS
9	BOS/BOS <hook>/<hook> was/VBZ born/VBN * LOCATION/Entity in/IN <target>/<target> ./. EOS/EOS
10	BOS/BOS <hook>/<hook> was/VBZ born/VBN * LOCATION/Entity in/IN <target>/<target> ./. EOS/EOS
11	BOS/BOS <hook>/<hook> was/VBZ born/VBN in/IN <target>/<target> in/IN LOCATION/Entity
12	BOS/BOS <hook>/<hook> was/VBZ born/VBN * MMDD/Entity ,/, <target>/<target> and/CC is/VBZ
13	BOS/BOS <hook>/<hook> was/VBZ born/VBN in/IN <target>/<target> in/IN LOCATION/Entity

Because of the limitation, patterns (e-g) are considered as invalid patterns in our approach. Consider pattern (e), its cWin value is less than 1. There is one wildcard element adjacent to *hook* or *target* in pattern (f) and (g). Therefore, it is impossible to find the beginning of $A2$ for

pattern (f). Also it is impossible to find the ending of *A*2 for pattern (g). We also consider these two types of patterns as invalid patterns.

(e) (/(<hook>/<hook> <target>/<target>)/)

(f) BOS/BOS <hook>/<hook> */* born/VBN in/IN <target>/<target> in/IN

(g) BOS/BOS <hook>/<hook> was/VBZ born/VBN */* <target>/<target> in/IN

3.3.3 Edit-Distance based Generalization

The edit-distance algorithm [22] is used to find the minimum number of edit operations needed to convert one string to another string. The edit operations are *Inserting* (I), *Removing* (R), *Replacing* (U) and *Equal* (E). Each time, the operation is allowed to change only one symbol. For example, *Inserting* can insert one symbol into the string. Figure 8 shows that the edit-distance between "abcde" and "abfe" equals two. That menas in order to convert string "*abcde*" into "*abfe*", symbol "*c*" should be replaced by "*f*" and symbol "*d*" should be removed. In our approach, we use edit-distance algorithm to calculate the distance between two patterns and guide the pattern generalization process.

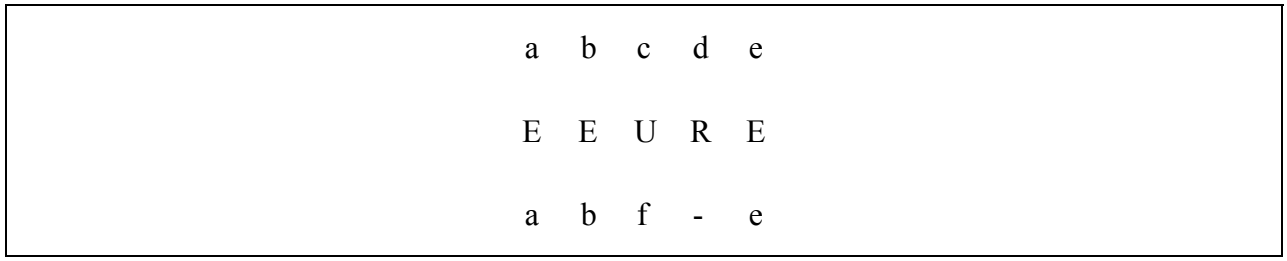

Figure 8: Convert "*abcde*" into "*abfe*"

In order to apply edit-distance algorithm to patterns, we allow the edit operation to change one element instead of one symbol each time. The algorithm will use dynamic programming. It would calculate two matrixes: one is the distance matrix (M) and the other one is the direction matrix (D). Based on matrix M and D, a change route, which is a sequence of edit operations, can be captured. For example, in Figure 8, "*EEURE*" is a change route.

We use pattern (h) and pattern (i) to show how the edit-distance algorithm generalizes patterns. The distance matrix (shown in Table 5) and the direction matrix (shown in Table 6) are created by the algorithm.

(h) <hook>/<hook> wrote/VBD the/DT very/IN old/JJ <target>/<target>

(i) <hook>/<hook> wrote/VBD the/DT classic/JJ <target>/<target>

Table 5: Distance matrix (M)

M	null	<hook>	wrote/VBD	the/DT	classic/JJ	<target>
null	0	1	2	3	4	5
<hook>	1	0	1	2	3	4
wrote/VBD	2	1	0	1	2	3
the/DT	3	2	1	0	1	2
very/IN	4	3	2	1	1	2
old/JJ	5	4	3	2	2	2
<target>	6	5	4	3	3	2

Table 6: Direction matrix (D)

D	null	<hook>	wrote/VBD	the/DT	classic/JJ	<target>
null		I	I	I	I	I
<hook>	R	E	I	I	I	I
wrote/VBD	R	R	E	I	I	I
the/DT	R	R	R	E	I	I
very/IN	R	R	R	R	U	I
old/JJ	R	R	R	R	R	U
<target>	R	R	R	R	R	E

Based on the distance matrix and the direction matrix, two possible change routes can be found (shown in Figure 9). The first route suggests replacing “very/IN” with “classic/JJ”, and removing “old/JJ”. The second route suggests removing “very/IN”, and replacing “classic/JJ” with “old/JJ”.

<hook>/<hook>	wrote/VBD	the/DT	very/IN	old/JJ	<target>/<target>
E	E	E	U	R	E
<hook>/<hook>	wrote/VBD	the/DT	classic/JJ	-	<target>/<target>
<hook>/<hook>	wrote/VBD	the/DT	very/IN	old/JJ	<target>/<target>
E	E	E	R	U	E

<hook>/<hook> wrote/VBD the/DT - classic/JJ <target>/<target>

Figure 9: Converting pattern (h) into pattern (i)

If we compare the two routes shown in Figure 9, we can see that the second one is more natural. With the same number of edit operations as the first route, it removes subordinating conjunction *very* and replaces adjective *old* with *classic*. Therefore, POS should be considered by the edit-distance algorithm.

In our approach, we modify the original edit-distance algorithm in the element comparing process. The algorithm would consider that two elements are equal if their POS tags belong to the same category. We propose three categories: nouns (NN, NNS, NNP, NNPS), verbs (VBD, VBN, VBP) and symbols (. , ! ? … “ “ ‘ - ( )). The new edit-distance algorithm is shown in the following functions. The function *fillDistanceMatrix* is used to calculate the distance matrix M and the function *fillDirectionMatrix* is used to calculate the direction matrix D. The function *findPath* is used to find the change route.

Algorithm *fillDistanceMatrix( )*

```
1   M [0][0] := 0;
2   for i := 1 to |A| do M [i][0] := M [i − 1][0] + 1;
3   for i := 1 to |B| do M [0][i] := M [0][i − 1] + 1;
4   for i := 1 to |A| do
5       for j := 1 to |B| do begin
```

```
6            if Pos( A[i] ) = Pos( B[ j] )
7                then cos t := 0;
8                else cos t := 1;
9            m1 := M [i − 1][ j − 1] + cos t;
10           m2 := M [i − 1][ j] + 1;
11           m3 := M [i][ j − 1] + 1;
12           M [i][ j] := min( m1, m2, m3);
13       end
```

Algorithm *returnDistance ( )*

```
1    return M [| A | −1][| B | −1];
```

Algorithm *Pos (word)*

```
1    return word's part-of-tag;
```

Algorithm *fillDirectionMatrix( )*

```
1    for i := 1 to | A | do D[i][ 0] := I;
2    for i := 1 to | B | do D[0][i] := R;
3    for i := 1 to | A | do
4        for j := 1 to | B | do begin
5            if Pos( A[i] ) = Pos( B[ j] )
```

```
6                then cos t := 0;
7                else cos t := 1;
8            m1 := M[i − 1][ j − 1] + cos t;
9            m2 := M[i − 1][ j] + 1;
10           m3 := M[i][ j − 1] + 1;
11           if M[i][ j] = m1 then
12               if cost = 0 then D[i][ j] := E;
13               else D[i][ j] := U;
14           if M[i][ j] = m2 then D[i][ j] := I;
15           if M[i][ j] = m3 then D[i][ j] := R;
16       end
```

```
Algorithm fillDirectionMatrix( )
1   i := | A |;   j := | B |;
2   while (i ≠ 0 && j ≠ 0) do begin
3       if M[i][ j] = M[i − 1][ j] + 1 then i := i − 1;
4       else if M[i][ j] = M[i][ j − 1] + 1 then j := j − 1;
5           else begin
6               i := i − 1; j := j − 1;
7           end;
8       print (D[i][ j]);
```

9 end;

Now the proposed algorithm will prefer a replacement operation to a deletion or an insertion operation between two elements that have a same POS tag. Applying the new algorithm to pattern (h) and (i), only one change route will be found:

<hook>/<hook> wrote/VBD the/DT very/IN old/JJ <target>/<target>

E E E R U E

<hook>/<hook> wrote/VBD the/DT - classic/JJ <target>/<target>

After the change route is created by *fillDirectionMatrix*, we can obtain a generalized pattern by maintaining the common elements shared by two patterns, and use wildcards or disjunctions to represent different elements. Every time there is an insertion or a deletion in the change route, the generalized pattern will have a wildcard, indicating that there can be any element in that position. Every time there is a replacement, the generalized pattern will have a disjunction of both elements. Every time there is an equal, the element is copied into the generalized pattern. Pattern (j) is the generalized pattern for (h) and (i).

(j) <hook>/<hook> wrote/VBD the/DT */* classic|old/JJ <target>/<target>

3.3.4 Generalization Pseudocode

When humans use natural languages to express semantic relationships, many lexical patterns of words are used. Therefore, it is impossible to capture all existing patterns for a given relation. In our approach, we choose similar patterns as one group, and then several new patterns will be

created, which are general enough to match all patterns in that group. We already explained how to use the modified edit-distance algorithm to create a generalized pattern. Here, we explain how to generalize patterns from a group of patterns.

We use the edit-distance value to measure the similarity between two patterns. The applied number is used to measure how many sentences can be matched by a pattern. If the edit-distance value between two patterns is small, the similarity between them is high. If a pattern's applied number is big, that means this pattern can match a lot of sentences.

Our generalization process uses the edit-distance value and the total applied number to select two patterns, which have the smallest edit-distance value and the biggest total applied number, to create a generalized pattern. Once a new pattern is generalized successfully, we use that pattern to replace the original ones. This process is repeated until there is no pattern that can be generalized. The generalization pseudocode is shown below:

1. Store all patterns in a set P
2. While true
    a) For each pair of patterns, calculate their edit-distance value and total applied number
    b) Take pattern p_i and p_j, who have the smallest edit-distance value and the biggest total applied number
    c) Obtain the generalized pattern p_g for p_i and p_j
    d) If p_g is a valid pattern, add it to P, and remove p_i and p_j from P
    e) If no pattern can be generalized correctly for each possible pattern pairs, return P

CHAPTER 4

PATTERN APPLICATION

Once a set of patterns are generalized from the training corpus, the pattern application procedure is straightforward. We introduce using an ontology to solve the ambiguity problem. The new process can improve precision rate dramatically. In this chapter, we describe the pattern application procedure. After that, we show how to create and use the ontology to solve the ambiguity problem.

4.1 Pattern Application Procedure

NLP tools, POS and NER, should be applied to each sentence in the testing corpus. The outputs should be combined together (described in Section 3.2). The matching process is shown as follows:

1. For each pattern, for example ($A1 hook A2 t \arg et A3$), in the set
2. For each sentence in the testing corpus
    a) Look for the left-hand-side content $A1$ in the sentence.
    b) Look for the middle content $A2$ in the sentence.
    c) Look for the right-hand-side content $A3$ in the sentence.
    d) The words between $A1$ and $A2$ are considered as *hook*, the words between $A2$ and $A3$ are considered as *target*.

e) For each extracted *hook* and *target*, use the ontology to query their relation. If the returned relation equals the pattern's relation, output *hook*, *target* and the relation.

4.2 Ontology Inference

Although the pattern application procedure is not complex, lexical patterns themselves can cause ambiguities. Ravichandran and Hovy (2002) have reported that wildcards might be harmful because they could unrestrictedly match incorrect context [14]. This problem hurts the precision rate dramatically. To consider a simple example, apply pattern (j) to sentence (13),

(j) <hook>/<hook> was/VBZ born/VBN */* in/IN <target>/<target> in|and/IN

(13) Janet Evanovich was born in 1943 in New Jersey and didn't begin writing until she was already married with children and in her thirties.

There is only one value, Janet Evanovich, that can be extracted for the *hook*, but the *target* has two possible values: 1943 and New Jersey.

A common solution used by existing approaches is to discard those patterns that cause ambiguities. For example, Alfonseca, Castells, Okumura and Ruiz-Casado (2006) use a validation textual corpus to calculate the precision rate in the pattern generalization process. If a pattern causes ambiguities more than three times, it will be discarded.

This solution could solve some ambiguities, but it also causes another problem. Some patterns representing several different relations, especially those short patterns, can be discarded. For example, the pattern *<hook>/<hook> 's/POS <target>/<target>*, which can be used to represent writer-book, singer-song, location-building and other relations, will be discarded.

In order to solve this problem, if the *hook* or the *target* is a named entity, this information can be used to make a decision [21]. For example, *New York's Empire State Building*, the *hook* is a location entity. Therefore, the pattern <*hook*>/<*hook*> *'s/POS* <*target*>/<*target*> only indicates the location-building relation.

We take this idea a step further: an ontology is used to solve pattern ambiguities. Once a possible pair, *hook* and *target*, is extracted, we query the relationship between them in the ontology. When the returned relationship is the same as the pattern's relationship, we consider that the pattern has been applied successfully.

4.2.1 Ontology Creation

An ontology is a set of concepts within a domain and the relationships among those concepts. It provides the type of existing concepts, properties associated with concepts and relations among those concepts. In our experiment, we test five relations: birth-year, death-year, country-capital, writer-book and singer-song. Therefore, our ontology contains the person class, country class, book class, song class and relationships among them.

We randomly pick out ten writers and seven singers from Wikipedia. Then we extract the singers' album and song information from FreeDB[11], which is a free CD and music database service to look up textual metadata about music, audio or data CDs. Because some songs can be recorded by several different singers, we also add those singers' information into our Ontology. For example, we add Teresa Brewer, Amina Claudine Myers and Dinah Washington into the

[11] http://www.freedb.org/

Ontology because they recorded the songs originally sang by Bessie Smith. We try to collect all books written by those ten writers and all songs recorded by the seven singers. After that, we extract their birth year from Wikipedia. In total, the resulting ontology has 27 people: 10 people are writers; the others are singers. Table 7 shows their names, their birth year and their death year. The ten writers and seven singers are highlighted.

Table 7: 27 people and their birth year

Name	Type	Birth Year	Death Year
Amina Claudine Myers	Singer	1942	null
Bessie Smith	Singer	null	1937
Beverly Lewis	Writer	null	null
Charlaine Harris	Writer	1951	null
Chick Corea	Singer	1941	null
Christian McBride	Singer	1972	null
Dan Brown	Writer	1964	null
Dinah Washington	Singer	1924	1963
Don McLean	Singer	1945	null
Donald A Norman	Writer	1935	null
Douglas Brinkley	Writer	1960	null

George Kawaguchi	Singer	null	null
Glenn Beck	Writer	1964	null
Herbie Hancock	Singer	1940	null
James Patterson	Writer	1947	null
Janet Evanovich	Writer	1843	null
Jim Rogers	Singer	1942	null
Keith Whitley	Singer	1945	1989
LaVern Baker	Singer	1929	1997
Marjane Satrapi	Writer	1969	null
Marty Robbins	Singer	null	null
Max Lucado	Writer	1995	null
Michael Jackson	Singer	1958	2009
Starsound Orchestra	Singer	null	null
Tanya Tucker	Singer	null	null
Teresa Brewer	Singer	1931	2007
Vic Dickenson	Singer	1906	1984

The books are classified into six categories: fiction, history, nonfiction, novel, religion spirituality and science. There are 356 books extracted from Wikipedia, as well as 86 albums and 815 songs extracted from FreeDB. Countries and their capitals are randomly picked out from

Wikipedia. Figure 10 shows the ontology Schema, and the content of the ontology is shown in Appendix A.

4.2.2 Information Query

Using the ontology to query the relation between two concepts is straightforward. First, submit the extracted *hook* and *target* to the ontology. Next, find the instances whose property has a value equal to *hook* or *target*. Then, check whether a relationship exists between those two instances. If a relationship is found, return that relationship.

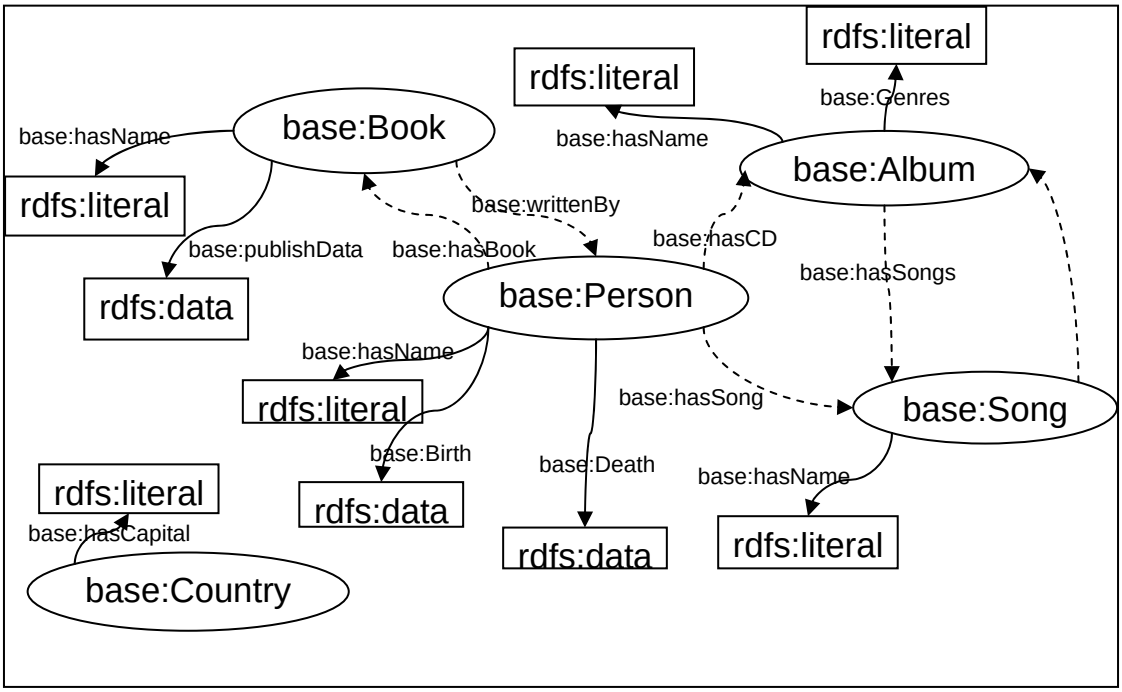

Figure 10: Ontology Schema

CHAPTER 5

RESULTS AND EVALUATION

We test our approach for five different relations: birth-year, death-year, country-capital, writer-book and singer-song. For each relation, we collect a training corpus and a testing corpus separately by using the Yahoo search engine. We extract patterns from the training corpus, and then use the generalization process to generalize extracted patterns. After that, we use the generalized patterns to extract concept pairs in the testing corpus. Based on the extracted concept pairs, we can calculate the recall and precision rate. In this chapter, we show the details of our experiments, the procedure used to calculate the recall and precision rate and the evaluation of our results.

5.1 Experiment Settings

For a particular relation, a seed list is randomly created from our ontology. For each pair in the seed list, a random number of web pages are downloaded through the Yahoo search engine (Section 3.1). Table 8 shows the number of seed pairs for each relation, the number of downloaded pages, the number of unique patterns extracted from each sentence and the number of generalized patterns. For all relations, the number of downloaded pages is much greater than the number of unique patterns extracted from each sentence. That is because some sentences appear in different web pages a lot of times. Some generalized patterns are shown in APPENDIX

B. The applied number means the number of sentences can be matched by the pattern in the training corpus.

Using the same downloaded web pages, a testing corpus is created (Section 3.1) for each relation. We combine the birth-year, death-year, writer-book and singer-song testing corpus together, and then discard those sentences whose *hook* value is not in the following list: *Jim Rogers, Keith Whitley, Herbie Hancock, Marty Robbins, Michael Jackson, Tanya Tucker, Bessie Smith, Beverly Lewis, Charlaine Harris, Dan Brown, Donald A Norman, Douglas Brinkley, Glenn Beck, Marjane Satrapi, James Patterson, Janet Evanovich* and *Max Lucado*. The first seven people are singers, and the other ten people are writers. The country-capital testing corpus is added into the combined testing corpus directly. After that, there are 1788 sentences left in the testing corpus. Those sentences are classified into six categories: sentences with a birth-year relation, sentences with a death-year relation, sentences with a country-capital relation, sentences with a writer-book relation, sentences with a singer-song relation and sentences with no relation.

Table 8: Number of seed pairs for each relation, number of downloaded pages, number of unique patterns after the extraction and number of generalized patterns

Relation	Seeds	Pages	Unique Patterns	Gener. Patterns
Birth-year	21	1331	634	182
Death-year	5	423	130	24
Country-capital	11	203	144	29

Writer-book	279	4745	2033	441
Singer-song	157	5232	1390	373

Once the testing corpus is created, patterns generalized for a given relation should be applied to the testing corpus. Based on the output of the pattern application process, a confusion matrix [23] can be created for that relation. Table 9 shows an example of a confusion matrix. The entries in the matrix have the following meaning for our experiment:

For a particular relation *R*, apply its generalized patterns *P* to the testing corpus,

- a is the number of sentences that do not have relation *R* and are predicted by *P* correctly
- b is the number of sentences that do not have relation *R* and are predicted by *P* incorrectly
- c is the number of sentences that have relation *R* and are predicted by *P* incorrectly
- d is the number of sentences that have relation *R* and are predicted by *P* correctly

Table 9: A confusion matrix for a particular relationship R

		Predicted	
		Negative	Positive
Actual	Negative	a	b
	Positive	c	d

Several standard terms have been defined for the confusion matrix. In our experiment, we only use three of them: the recall (true positive rate), the precision and the F-measure. They are defined as the following:

- The recall is the proportion of sentences with relation *R* that are correctly predicted by *P*:

$$recall = \frac{d}{c+d}$$

- The precision is the proportion of sentences with relation *R* predicted by *P* that are correct:

$$precision = \frac{d}{b+d}$$

- The F-measure is a single measure that characterizes performance:

$$F-measure = \frac{2*recall*precision}{recall+precision}$$

Table 10 shows the recall rate, the precision rate and the F-measure rate for each relation. We run each experiment twice, the first time the experiment did not use the ontology for pattern disambiguation, and the second time the experiment used the ontology to solve the ambiguity problem. We could learn from those experiments that the ontology-based pattern disambiguation process can improve both the precision rate and the F-measure rate.

Table 10: Patterns' recall, precision and F-measure rate with/without Ontology

Relation	Without Ontology			With Ontology		
	recall	precision	F-measure	recall	precision	F-measure
Birth-year	62.8%	71.2%	66.7%	67.5%	100%	80.6%

Death-year	78.1%	73.2%	75.6%	70.2%	100%	82.5%
Country-capital	75.4%	69.7%	72.4%	82.1%	100%	90.2%
Writer-book	55.3%	63.3%	59%	52%	100%	68.4%
Singer-song	61%	59%	60%	58%	100%	73.4%

We have observed that POS would tag a word with an incorrect tag. For example, *born* can be tagged as NN (*common noun*) instead of VBD (perfect verb); *is* can be tagged as JJ (adjective) instead of VBD (perfect verb); and *in* can be tagged as JJ (adjective) instead of IN (subordinating conjunction). This problem can reduce the recall rate and increase the total number of generalized patterns because our approach treats a word with a correct tag and the same word with a wrong tag as two different elements.

We also found that invalid date formats can influence our approach, especially for the birth-year and death-year relation. Several invalid date formats, such as *1999 10 3* and *March 8 1928*, are found in the training corpus. Patterns extracted from those date formats are wrong because NER cannot identify MMDD and YYYY entities correctly. For example, NER identifies *1999 10 3* as cardinal numbers.

We have to choose Cross-validation, which is a technique for assessing how the results of a statistical analysis will generalize to an independent data set, to measure our approach's recall rate because none of the existing approaches provide the recall rate. For each relation, we use its testing corpus to run a four cross-validation to measure the recall rate. The testing corpus is

partitioned equally into four complementary subsets: a, b, c and d. We run four rounds of experiments on them. Each round will choose one subset as a testing set, and choose the other three subsets as a training set. Then the final recall rate is the average value of the four experiments. As a result, we get the following recall rates for each relation:

- Birth-year: 63.7%
- Death-year: 69.4%
- Country-capital: 84.1%
- Writer-book: 56.2%
- Singer-song: 59.6%

CHAPTER 6

RELATED WORK

Several automatic information extraction approaches, which extract structured information from free text, are proposed for different purposes. Concerning their implementations, we classify them into two groups: approaches based on distributional properties of words and approaches based on pattern extraction and matching.

Approaches based on distributional properties of words focus on studying co-occurrence distributions of words, which can be used to calculate a distance between the concepts represented by those words. The distance can be used to enrich the ontologies with new concepts [24-26] and to extract non-taxonomic relations [27, 28]. The previous approaches are usually used to identify new concepts for an existing ontology, and the other approaches are used to learn association rules between related concepts to extract other relations.

Approaches based on pattern extraction and matching rely on lexical patterns which are learned from relationships between concepts in free text. [29, 30] manually define regular expressions to extract concepts and relations. [31] automatically learns such patterns for company merge relations. [19] uses patterns to identify concepts from unannotated text. [11] uses patterns to help information extraction from semi-structured and free text.

Our approach benefits pattern representation and pattern generalization from the ideas and techniques presented in pattern based approaches. Compared with approaches based on

distributional properties of words, pattern based approaches consider not only word occurrence, but also grammar and word order. We believe that pattern based approaches are more suitable in the annotation process, such as adding tags to concepts in editing Wikipedia articles. That is because users would input their articles one sentence at a time. Therefore, any annotation tags should be added only based on one sentence, not the whole article. One kind of pattern based approaches, which is called the Rote method, achieves this purpose very well.

Brin (1999) introduced an approach to extract an *author-book* list from independent data sources automatically [13]. This approach starts with a small seed set of *author-book* pairs. The approach tries to find all occurrences of those pairs from the web. Based on those occurrences, patterns for the *author-book* relation are generalized. After that, those patterns are used to search the web and to extract new *author-book* pairs. This approach can use the new extracted pairs to generate more patterns, and so forth. As a result, a large list of *author-book* pairs and patterns for finding them will be created. This approach applies the Dual Iterative Pattern Relation Expansion algorithm[12] to generalize patterns. Its seed set only has five *author-book* pairs, and it extracted over 15,000 books with few human interventions.

Agichtein (2001) used a similar approach to extract relations from large text collections [15]. This approach accepts a manually created instance set in a given relation. Then the approach searches for occurrences of the example instances in the local documents. This approach learns patterns from those occurrences. After that, patterns can be used to discover new instances in the documents, but only the most reliable instances can be added into the start instance set. The

[12] http://www.alexmayers.com/projects/DIPRE/

expanded instance set is used as the seed set for the next iteration, and the same process is repeated. This approach generates a term vector for each instance occurrence, and then it clusters those vectors by using a simple single-pass bucket clustering algorithm[13]. Generalized patterns are represented as cluster centers.

Ravichandran and Hovy (2002) explored a pattern based question answering system for open-domain questions [14]. It also uses a few hand-crafted examples as the seed for each question. They have developed a method for learning text patterns automatically from a tagged corpus which is built from the Internet. Patterns are then extracted and standardized. Their standardization process uses the suffix trees algorithm, which is primarily used for detecting DNA sequences. Although the suffix trees algorithm allows this approach to learn optimal length patterns, this algorithm still causes two problems. One is that the patterns have no external knowledge. That means their patterns do not consider POS and named entities. The other problem is that their patterns do not allow unrestricted wildcard matching. Therefore, the patterns cannot handle long-distance dependencies.

Alfonseca, Castells, Okumura, and Ruiz-Casado (2006) proposed a new approach to learn patterns for finding semantic relationships in unrestricted text [20]. It uses POS tags and named entities to guide the pattern generalization process. A modified edit-distance algorithm can be used to create a more general pattern which allows wildcards. In order to improve the precision rate, this approach would discard those patterns that match concepts incorrectly more than three times. Also the authors tested their approach in a way that the patterns obtained for a given

[13] http://maya.cs.depaul.edu/~classes/ds575/clustering/CL-alg-details.html

relation are evaluated on different relations. The results of 9 relations show that 5 relations attain a precision rate higher than 50%. Ruiz-Casado, Alfonseca, and Castelss (2006) used a similar process to recognize relations in Wikipedia [21]. In order to improve the precision rate, the authors proposed two solutions. The first one is to consider the named entity tag of the extracted concept to guide the pattern application process. The second one is to force one special pattern to be used to indicate one relation.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

We propose a new lexical pattern-based annotation approach, which use an ontology to solve the ambiguity problem. We also defined a new pattern format to improve the recall rate of some relations. We examined our approach for five relationships: birth-year, death-year, country-capital, writer-book and singer-song. An ontology is created, which contains a country class, person class, book class, song class and relationships among them. The training corpus and testing corpus are collected randomly by using the Yahoo search engine. Compared with other approaches, our testing corpus contains six relationships: birth-year, death-year, country-capital, writer-book, singer-song and none relation. By using the ontology, we achieve a very good result measured in terms of precision rate (100%) and recall rate (greater than 52%).

Concerning future work, there is room for improving our method. Because our approach uses the ontology to solve most ambiguities, the patterns learned for a relation could be more general. For example, Stemming, which is the process to obtain the canonical form of all the words, can reduce the words "fishing", "fished", "fish" and "fisher" to the canonical word, "fish".

Currently, the ontology is created for several particular relations. During our approach, the ontology is not changed. When we analyze our results, we found that patterns did extract other related concepts that are not contained in the ontology. Those concepts should be added into the ontology and used to solve other ambiguities. Therefore, our approach can automatically

improve its precision rate by expanding the ontology knowledge. We investigate these in future work, with hopes of providing a platform which can be used for any relation.

REFERENCES

1. Daelemans, W. and V. Hoste, *Evaluation of Machine Learning Methods for Natural Language Processing Tasks*, in *Proceedings of the Third International Conference on Language Resources and Evaluation*. 2002. p. 755-760.
2. Daelemans, W. and A.v.d. Bosch, *Memory-Based Language Processing (Studies in Natural Language Processing)*. 2005: Cambridge University Press.
3. Yarowsky, D., *Unsupervised word sense disambiguation rivaling supervised methods.* Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics. 1995.
4. Manning, C.D. and H. Schiitze, *Foundations of Statistical Natural Language Processing*. 1 ed. 1999: The MIT Press.
5. Berners-Lee, T., J. Hendler, and O. Lassila, *The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities.* Scientific American, 2001: p. 34-43.
6. Ding, L., et al., *Swoogle: a search and metadata engine for the semantic web*, in *Proceedings of the thirteenth ACM international conference on Information and knowledge management*. 2004, ACM: Washington, D.C., USA.
7. Mayfield, J. and T. Finin, *Information retrieval on the semantic web: Integrating inference and retrieval*, in *Proceedings of the SIGIR Workshop on the Semantic Web*. 2003.
8. McGuinness, D.L. and P.P.d. Silva, *Explaining answers from the Semantic Web: the*

Inference Web approach. Web Semantics: Science, Services and Agents on the World Wide Web, 2004. **1**: p. 397-413.

9. Gruber, T.R., *A translation approach to portable ontology specifications.* Knowl. Acquis., 1993. **5**(2): p. 199-220.
10. Popov, B., et al., *KIM:a semantic platform for information extraction and retrieval.* Nat. Lang. Eng., 2004. **10**(3-4): p. 375-392.
11. Soderland, S., *Learning Information Extraction Rules for Semi-Structured and Free Text.* Mach. Learn., 1999. **34**(1-3): p. 233-272.
12. Mann, G.S. and D. Yarowsky, *Multi-field information extraction and cross-document fusion*, in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. 2005, Association for Computational Linguistics: Ann Arbor, Michigan.
13. Brin, S., *Extracting Patterns and Relations from the World Wide Web*, in *Selected papers from the International Workshop on The World Wide Web and Databases*. 1998, Springer-Verlag.
14. Ravichandran, D. and E. Hovy, *Learning surface text patterns for a Question Answering system*, in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. 2002, Association for Computational Linguistics: Philadelphia, Pennsylvania.
15. Agichtein, E., et al., *Snowball: a prototype system for extracting relations from large text collections.* SIGMOD Rec., 2001. **30**(2): p. 612.
16. Fortuna, B., D. Mladenič, and M. Grobelnik, *Semi-automatic Construction of Topic Ontologies* in *Semantics, Web and Mining*. 2006, Springer Berlin / Heidelberg. p. 121-131.
17. Khan, L. and F. Luo, *Ontology Construction for Information Selection*, in *Proc. of the*

14th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2002). 2002. p. 122-127.

18. Blaschke, C. and A. Valencia, *Automatic Ontology Construction from the Literature*, in *Genome Inform Ser Workshop Genome Inform*. 2002.
19. Riloff, E. and M. Schmelzenbach, *An empirical approach to conceptual case frame acquisition*, in *Proceedings of the Sixth Workshop on Very Large Corpora*. 1998. p. 39-45.
20. Alfonseca, E., et al., *A rote extractor with edit distance-based generalisation and multi-corpora precision calculation*, in *Proceedings of the COLING/ACL on Main conference poster sessions*. 2006, Association for Computational Linguistics: Sydney, Australia.
21. Ruiz-Casado, M., E. Alfonseca, and P. Castells, *From Wikipedia to Semantic Relation-ships: a semi-automated Annotation Approach*. 2006.
22. Wagner, R.A. and M.J. Fischer, *The String-to-String Correction Problem.* J. ACM, 1974. **21**(1): p. 168-173.
23. Kohavi, R. and F. Provost, *Glossary of terms.* Machine Learning, 1998. **2**(3): p. 271-274.
24. Lee, L.J., *Similarity-based approaches to natural language processing*. 1997, Harvard University Press.
25. Faure, D. and C. Nedellec, *A Corpus-based Conceptual ClusteringMethod for Verb Frames and Ontology Acquisition*, in *LREC workshop on adapting lexical and corpus resources to sublanguages and applications*. 1998.
26. Cimiano, P., A. Hotho, and S. Staab, *Clustering Concept Hierarchies from Text*, in *Proceedings of LREC*. 2004.

27. Abecker, A., A. Bernardi, and M. Sintek, *Proactive knowledge delivery for enterprise knowledge management*, in *Proceedings of the 11th Conference on Software Engineering and Knowledge Engineering*. 1999.

28. Maedche, A. and S. Staab, *Discovering conceptual relations from text*, in *Proceedings of the 14th European Conference on Artifial Intelligence*. 2000.

29. Hearst, M.A., *Automated Discovery of WordNet Relations, An Electronic Lexical Database and Some of its Applications*. 1998, MIT Press.

30. Berland, M. and E. Charniak, *Finding parts in very large corpora*, in *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*. 1999, Association for Computational Linguistics: College Park, Maryland.

31. Kietz, J.-U., A. Maedche, and R. Volz, *A method for semi-automatic ontology acquisition from a corporate intranet*, in *Proceedings of the EKAW'00 Workshop on Ontologies and Text*. 2000.

APPENDIX A

ONTOLOGY CONTENT

Writer-book Relation

Person Instance	Book Instance
Douglas Brinkley	Driven Patriot: The Life and Times of James Forrestal, Fear and loathing in America: the brutal odyssey of an outlaw journalist, 1968-1976, FDR and the creation of the U.N., Windblown world: the journals of Jack Kerouac, 1947-1954, The Boys of Pointe du Hoc: Ronald Reagan, D-Day, and the U.S. Army 2nd Ranger Battalion, Wheels for the world : Henry Ford, his company, and a century of progress, 1903-2003, The Majic Bus: An American Odyssey, Witness to America, Tour of Duty: John Kerry and the Vietnam War, Rosa Parks, The Great Deluge: Hurricane Katrina, New Orleans and the Mississippi Gulf Coast, The Reagan Diaries, Parish Priest: Father Michael McGivney and American Catholicism, The Wilderness Warrior: Theodore Roosevelt and the Crusade for America, Gerald R. Ford, The Boys of Pointe Du Hoc, Dean Acheson: The Cold War Years, 1953-71, Jimmy Carter: The Unfinished Presidency, The Mississippi: and the Making of a Nation, The World War II Memorial: a grateful nation remembers, Voices of Valor : D-Day, June 6, 1944, Rise to Globalism: American Foreign Policy Since 1938, Road novels 1957-1960, The proud highway: saga of a desperate southern gentleman, 1955-1967, American Heritage History
Beverly Lewis	The Redemption of Sarah Cain, Reach for the Stars, Only the Best, Second-Best Friend, Just Like Mama, California Crazy, Echoes in the Wind, Photo Perfect, The Missing, Dreams on Ice, The Mystery of Case D. Luc, Shadows Beyond the Gate, Good-Bye, Dressel Hills, Sanctuary with David Lewis, The Crossroad, The Granny Game, The Secret, Freshman Frenzy, Mystery Letters, What is God Like, Tarantula Toes, Catch a Falling Star, The Longing, Mailbox Mania, The Preacher's daughter, The Brethren, Better Than Best, The Upside-Down Day, Backyard Bandit Mystery, Little White Lies, Green Gravy, What is Heaven Like, The Midnight Mystery, Cows in the House, The Double Dabble Surprise, The Englisher, The Sunroom, The Mudhole Mystery, The Telling, Whispers Down the Lane, House of Secrets, The Covenant, Secret Summer Dreams, The Great TV Turn-Off, Secret in

	the Willows, In Jesse's Shoes, Frog Power, A Cry in the Dark, The Shunning, The Chicken Pox Panic, Star Status, Night of the Fireflies, Windows on the Hill, The Creepy Sleep-Over, It's a Girl Thing, The Crabby Cat Caper, Piggy Party, The Betrayal, The Beverly Lewis Amish Heritage Cookbook, Best Friend, Worst Enemy, The Revelation, The Crazy Christmas Angel Mystery, Big Bad Beans, Fiddlesticks, The Postcard, The Reckoning, Hide Behind the Moon, Follow the Dream, October Song, Mystery Mutt, The Stinky Sneakers Mystery, The Parting, The Confession, Straight-A Teacher, The Forbidden, No Guys Pact, Tree House Trouble, The Prodigal, Pickle Pizza, The Sacrifice, Sealed With a Kiss, No Grown-ups Allowed, Eight is Enough, A Perfect Match, The Trouble With Weddings, Annika's Secret Wish with CD
…	
Max Lucado	A Heart Like Jesus, No Wonder They Call Him the Savior, On the Anvil (Shaped by God), Just Like Jesus (Devotional), The Great House of God, God Came Near, Let the Journey Begin, Grace for the Moment, God's Promises For You, Traveling Light, Next Door Savior, You Are Special, Hermie, A Common Caterpillar, The Heart of Christmas, A Gentle Thunder, Turn, And the Angels Were Silent, One Incredible Moment, Your Special Gift, Cosmic Christmas (An Angel's Story), In the Footsteps of Jesus, Grace For The Moment II, Tell Me the Secrets, The Inspirational Study Bible, Six Hours One Friday, Cure for the Common Life, Walking with the Savior, Every Day Deserves a Chance: Wake Up to the Gift of 24 Hours, A Love Worth Giving, The Cross, Just In Case You Ever Wonder, Opening Windows, The Christmas Cross, Topical Bible Study Series, Traveling Light for Mothers, Safe in the Shepherd's Arms, Facing Your Giants, The Crippled Lamb, God's Inspirational Promise Book, Just Like Jesus, The Applause of Heaven, Everyday Blessings, He Still Moves Stones, It's Not About Me, Fearless, God's Mirror, America Looks Up, Life Lessons Bible Study Guides, Come Thirsty, 3:16: The Numbers of Hope, Cast Of Characters, The Gift for All People, What the Cross Means to Me, When Christ Comes, When God Whispers Your Name, The Final Week of Jesus, Just For You, Experiencing the Heart of Jesus, He Chose the Nails, The Glory of Christmas, In the Eye of the Storm, He Did This Just For You, In the Grip of Grace, Because I Love You, God Thinks You're Wonderful

Singer-song Relation

Person Instance	Song Instance
Michael Jackson	I Wanna Be Where You Are, Morning Glow, Got to Be There, Music and Me, Ben [From "Ben"], Ben, With a Child's Heart, We're Almost There, One Day in Your Life, Happy [From Lady Sings the Blues], Happy [Love Theme from Lady Sings the Blues], Rockin' Robin, Cinderella Stay Awhile, I'll Come Home to You, We've Got Forever, One Day in Your Life, Take Me Back, You Are There, Dapper Dan, Dear Michael, Just a Little Bit of You, We're Almost There, Just a Little Bit of You, Happy [From Lady Sings the Blues], Happy [Love Theme from Lady Sings the Blues], I Wanna Be Where You Are, Music and Me, Ben [From "Ben"], Ben, Rockin' Robin, With a Child's Heart, Got to Be There, People Make the World Go Round, We're Almost There, One Day in Your Life, Who Is It, Ben [From "Ben"], Ben, Black or White, Beat It, Rockin' Robin, Human Nature, Bad, The Man in the Mirror, The Way You Make Me Feel, She's Out of My Life, Smooth Criminal, I Want You Back, Another Part of Me, Shake Your Body (Down to the Ground), You Are Not Alone, Wanna Be Startin' Somethin', In the Closet, ABC, Can You Feel It, Got to Be There, P.Y.T. (Pretty Young Thing), The Girl Is Mine, Dirty Diana, Thriller, Leave Me Alone, Remember the Time, Billie Jean, Don't Stop 'Til You Get Enough, You Rock My World, Off the Wall, Heal the World, Will You Be There, The Love You Save, I Just Can't Stop Loving You, Rock With You, They Don't Care About Us, Earth Song, Blame It on the Boogie, I Wanna Be Where You Are, Got to Be There, I'll Be There, The Love You Save, Rockin' Robin, I Want You Back, Ben [From "Ben"], Ben, Dancing Machine, ABC, Melodie [1973 Mix], With a Child's Heart, Call on Me [1973 Mix], Call on Me, You Can Cry on My Shoulder, Never Can Say Goodbye, When I Come of Age, Cinderella Stay Awhile, Music and Me, Who's Looking for a Lover, Everybody's Somebody's Fool, Farewell My Summer Love, Dancing Machine, You Are There, Lonely Teardrops, Ben [From "Ben"], Ben, Don't Let It Get You Down [1973 Mix], Maybe Tomorrow, We've Got a Good Thing Going, To Make My Father Proud [1973 Mix], Take Me Back, All the Things You Are, Got to Be There, Love's Gone Bad [1972 Mix], Shoo-Be-Doo-Be-Doo-Da-Day, I'll Be There, One Day in Your Life, That's What Love Is Made Of, If'n I Was God [1973 Mix], I'll Come Home to You, Ain't No Sunshine, Dear Michael, Happy [From Lady Sings the Blues], Happy [Love Theme from Lady Sings the Blues], We're Almost There, Girl Don't Take Your Love from Me, In Our Small Way, Make Tonight All Mine [1973 Mix], Maria (You Were the Only One), Greatest

	Show on Earth, Love Is Here and Now You're Gone, People Make the World Go Round, Rockin' Robin, Just a Little Bit of You, I Wanna Be Where You Are, It's Too Late to Change the Time
Bessie Smith	Cold in Hand Blues, Baby Won't You Please Come Home, Reckless Blues, St. Louis Blues, Nobody in Town Can Bake a Sweet Jelly Roll Like Mine, Sobbin' Hearted Blues, I'm Wild About That Thing, I Ain't Got Nobody, You've Been a Good Old Wagon, Sing, Sing Prison Blues, 'Tain't Nobody's Bizness If I Do, Follow the Deal on Down, Frosty Mornin' Blues, Frankie Blues, Rockin' Chair Blues, Graveyard Dream Blues, Ticket Agent, Ease Your Window Down, Ticket Agent Ease Your Window Down, Far Away Blues, Mistreatin' Daddy, Eavesdropper's Blues, Boweavil Blues, Hateful Blues, I'm Going Back to My Used to Be, Moonshine Blues, Cemetary Blues, Chicago Bound Blues, Sorrowful Blues, Pinchbacks-Take 'Em Away!, Easy Come, Easy Go Blues, Any Woman's Blues, Whoa, Tillie, Take Your Time, My Sweetie Went Away, Haunted House Blues, Shipwreck Blues, Gimme a Pigfoot (And a Bottle of Beer), He's Got Me Goin', Careless Love [Master Take in Mono], Careless Love, Alexander's Ragtime Band, You've Been a Good Old Wagon, Jailhouse Blues, Cake Walkin' Babies (From Home), Any Woman's Blues, Hard Time Blues, My Sweetie Went Away, Nobody Knows You When You're Down and Out, Weeping Willow Blues, Downhearted Blues, Young Woman's Blues, Cold in Hand Blues, Dyin' by the Hour, Preachin' the Blues, Baby Doll, Trombone Cholly, The St. Louis Blues, Do Your Duty, Weeping Willow Blues, Take Me for a Buggy Ride, Nashville Woman's Blues, Sinful Blues, Nobody Knows You When You're Down and Out, Cold in Hand Blues, Jailhouse Blues, Do Your Duty, Careless Love [Master Take in Mono], Careless Love, St. Louis Blues, 'Tain't Nobody's Bizness If I Do, I Ain't Gonna Play No Second Fiddle, Sam Jones Blues, Gimme a Pigfoot (And a Bottle of Beer), Downhearted Blues, Backwater Blues, After You've Gone, Empty Bed Blues, Pt. 1, Reckless Blues, Empty Bed Blues, Pt. 2, Kitchen Man, J.C. Holmes Blues, I'm Wild About That Thing, St. Louis Blues, Young Woman's Blues, Nobody Knows You When You're Down and Out, Baby Won't You Please Come Home, Standin' in the Rain Blues, Jazzbo Brown from Memphis Town, 'Tain't Nobody's Bizness If I Do, Mama's Got the Blues, Empty Bed Blues, Black Mountain Blues, I Used to Be Your Sweet Mama, What's the Matter Now?, He's Got Me Goin', Careless Love [Master Take in Mono], Careless Love, Keep It to Yourself, Lock and Key, Do Your Duty, Baby Won't You Please Come Home, Jazzbo Brown from Memphis Town, Yes Indeed He Do!, Sweet Mistreater, There'll Be a Hot Time in the Old Town Tonight, You've Got to Give Me Some, Oh! Daddy Blues, Alexander's Ragtime Band, It Won't Be You, Baby Doll, I Ain't Got

	Nobody, Cake Walkin' Babies (From Home), 'Tain't Nobody's Bizness if I Do, I Want Every Bit of It, Squeeze Me, After You've Gone, I'm Down in the Dumps, Yellow Dog Blues, Downhearted Blues, I'm Wild About That Thing, Cold in Hand Blues, I Used to Be Your Sweet Mama, Weeping Willow Blues, Baby Doll, Worn Out Papa Blues, Empty Bed Blues, Pts. 1-2, Reckless Blues, Jailhouse Blues, St. Louis Blues, Shipwreck Blues, Dyin' by the Hour, Nashville Woman's Blues, Careless Love [Master Take in Mono], Careless Love, Gimme a Pigfoot (And a Bottle of Beer), Alexander's Ragtime Band, My Sweetie Went Away, On Revival Day (A Rhythmic Spiritual), Backwater Blues, Them "Has Been" Blues, Kitchen Man, Need a Little Sugar in My Bowl, Do Your Duty, You've Been a Good Ole Wagon, Nobody Knows You When You're Down and Out, Careless Love [Master Take in Mono], Careless Love, Moan, You Moaners, I Ain't Gonna Play No Second Fiddle, Weeping Willow Blues, I'm Down in the Dumps, Boweavil Blues, Cake Walkin' Babies (From Home), Take Me for a Buggy Ride, Dying Gambler's Blues, At the Christmas Ball, Baby Won't You Please Come Home, Black Mountain Blues, There'll Be a Hot Time in the Old Town Tonight, After You've Gone, A Good Man Is Hard to Find, Jazzbo Brown from Memphis Town, Gimme a Pigfoot (And a Bottle of Beer), Aggravatin' Papa, Trombone Cholly, Dyin' By the Hour, Backwater Blues, Me and My Gin, Graveyard Dream Blues, St. Louis Blues, Alexander's Ragtime Band, Jailhouse Blues, Send Me to the 'Lectric Chair, Ticket Agent, Ease Your Window Down, Ticket Agent Ease Your Window Down, 'Tain't Nobody's Bizness If I Do, On Revival Day (A Rhythmic Spiritual), Shipwreck Blues, I Ain't Gonna Play No Second Fiddle, Need a Little Sugar in My Bowl, Gimme a Pigfoot and a Bottle of Beer, On Revival Day (A Rhythmic Spiritual), A Good Man Is Hard to Find, Downhearted Blues, Backwater Blues, The Yellow Dog Blues, Nobody Knows You When You're Down and Out, St. Louis Blues, Careless Love [Master Take in Mono], Careless Love, Muddy Water, 'Tain't Nobody's Bizness If I Do, Me and My Gin, Send Me to the 'Lectric Chair
…	
Marty Robbins	O Little Town of Bethlehem, Many Christmases Ago, Christmas Is for Kids, Christmas Kisses, Hark! The Herald Angels Sing, Little Stranger (In a Manger), One of You (In Every Size), Merry Christmas for You from Me, Christmas Time Is Here Again, A Christmas Prayer, The Joy of Christmas, Melba from Melbourne, Southern Dixie Flyer, Only a Picture Stops Time, Things That I Don't Know, Everybody's Darlin' Plus Mine, You Won't Have Her Long, Change That Dial, Making Excuses, She Means Nothing to Me Now, Urgently Needed, Rainbows, I Lived a Lifetime in a Day, Begging to

	You, I Can't Quit (I've Gone Too Far), Cigarettes and Coffee Blues, She Was Only Seventeen (He Was One Year More), Faleena (From El Paso), The Hanging Tree, Ain't I the Lucky One, Devil Woman, Walking Piece of Heaven, Ballad of the Alamo, Love Me, I Don't Know Why (I Just Do), A White Sport Coat (And a Pink Carnation), Ribbon of Darkness, Among My Souvenirs, Stairway of Love, Ruby Ann, I Walk Alone, Return to Me, That's All Right, Honkytonk Man, Mister Teardrop, Tonight Carmen, My Woman, My Woman, My Wife, I Couldn't Keep from Crying, El Paso City, The Cowboy in the Continental Suit, I'll Go on Alone, The Story of My Life, Don't Let Me Touch You, The Shoe Goes on the Other Foot Tonight, Singing the Blues, El Paso, Some Memories Just Won't Die, Knee Deep in the Blues, Just Married, You Gave Me a Mountain, Don't Worry, Big Iron, This Much a Man, Don't You Think, Two Gun Daddy, It Takes Faith, Life, Twentieth Century Drifter, Love Me, A Man and a Train, Crawling on My Knees, Walking Piece of Heaven, Shotgun Rider, Have I Told You Lately That I Love You, I Never Let You Cross My Mind, All the World Is Lonely Now, I'm So Lonesome I Could Cry, I'll Step Aside, I Hang My Head and Cry, It's Too Late Now (To Worry Anymore), Lovesick Blues, You Only Want Me When You're Lonely, Rose of Ol' Pawnee, Moanin' the Blues, Bouquet of Roses, Tennessee Toddy, Grown Up Tears, Baby, I Need You (Like I Need You), Ruby Ann, Footprints in the Snow, Long Gone Lonesome Blues, Baby's Gone, Call Me Up (And I'll Come Calling on You), Pretty Mama, Pain and Misery, Respectfully Miss Brooks, I Can't Quit (I've Gone Too Far), Long Tall Sally, Singing the Blues, It's a Long, Long Ride, Knee Deep in the Blues, You Don't Owe Me a Thing, That's All Right, Maybellene, Teenager's Dad, It's Driving Me Crazy, Mister Teardrop, Mean Mama Blues, I Couldn't Keep from Crying, Knee Deep in the Blues, Begging to You, It's Your World, I Can't Quit (I've Gone Too Far), Ruby Ann, I'll Go on Alone, Devil Woman, Just Married, Don't Worry, The Cowboy in the Continental Suit, The Story of My Life, A White Sport Coat (And a Pink Carnation), El Paso, Stairway of Love, Big Iron, Singing the Blues, Ribbon of Darkness, A White Sport Coat (And a Pink Carnation), Singing the Blues, Sink the Bismarck, My Woman, My Woman, My Wife, All for the Love of a Girl, North to Alaska, El Paso, Devil Woman, When It's Springtime in Alaska (It's Forty Below), The Battle of New Orleans, Bouquet of Roses, I Never Let You Cross My Mind, Sittin' in a Tree House, Have I Told You Lately That I Love You, I'll Step Aside, The Blues Country Style, Singing the Blues, Ain't I the Lucky One, A White Sport Coat (And a Pink Carnation), Lovesick Blues, The Last Time I Saw My Heart, Knee Deep in the Blues, I'm So Lonesome I Could Cry, Moanin' the Blues, Aloha Oe (Farewell to Thee), All the World Is

	Lonely Now, It's Too Late Now (To Worry Anymore), You Only Want Me When You're Lonely, Long Tall Sally, The Hanging Tree, I Hang My Head and Cry, The Story of My Life, She Was Only Seventeen (He Was One Year More), Rose of Ol' Pawnee, One Window, Four Walls, Mission in Guadalajara, Lovesick Blues, I'm So Lonesome I Could Cry, I Feel Another Heartbreak Coming On, Last Letter, Long Tall Sally, Wine Flowed Freely, I Hang My Head and Cry, In the Valley of the Rio Grande, Blues Country Style, Tall Handsome Stranger, Have I Told You Lately That I Love You, I'm Begining to Forget You, All the World Is Lonely Now, Singing the Blues, Ride Cowboy Ride, Story of My Life, Hawaiian Wedding Song, She Was Young and She Was Pretty, Once-A-Week Date, The Last Time I Saw My Heart, Foolish Decision, Sittin' in a Tree House, The Blues Country Style, Jeannie and Johnnie, Stairway of Love, The Hanging Tree, She Was Only Seventeen (He Was One Year More), Grown Up Tears, Just Married, Ain't I the Lucky One, The Story of My Life, A White Sport Coat (And a Pink Carnation), Please Don't Blame Me, Teenage Dream, I'm in the Mood for Love, If I Could Cry, September in the Rain, Don't Throw Me Away

APPENDIX B

LEXICAL PATTERNS

Birth-year Relation

Applied	Patterns
95	BOS/BOS <hook>/<hook> -LRB-|,|-|:/. */* -LRB-|,/. <target>/<target> -RRB-|-LRB-|.|,|-|?/.
58	BOS/BOS <hook>/<hook> "|-LRB-|,|-|:/. <target>/<target> -RRB-|.|-|\//.
34	BOS/BOS <hook>/<hook> ...|-LRB-|,|;|:/. */* ,/. <target>/<target> In|in/IN
22	-RRB-|"|-LRB-|,|-|\/|;|:/. <hook>/<hook> -LRB-|,/. <target>/<target> -RRB-|-/.
25	BOS/BOS <hook>/<hook> was/VBD */* ,/. <target>/<target> .|,|;/.
24	-RRB-|"|.|,|-|;|:/. <hook>/<hook> "|-LRB-|,|-|:/. */* -RRB-|-LRB-|,/. <target>/<target> -RRB-|.|,|-|;/.
20	in/IN <target>/<target> ,/. <hook>/<hook> was|worked|began|had|relocated|became|studied/VBD
20	BOS/BOS <hook>/<hook> was/VBD */* ,|\//. <target>/<target> in/IN
19	BOS/BOS <hook>/<hook> was|appeared/VBD */* until|in/IN <target>/<target> .|,|-/.
18	Runner|Jessie|novelist|biographer|NEW|Delores|facts|LECTURE|a|supposing|Songs|Singer|Beck/NN <hook>/<hook> -LRB-|.|,|:/. */* -LRB-|,|-/. <target>/<target> -RRB-|.|,|-/.
16	pianist\/composer|legends|Author|singer|novelist|chanteuse|Musican|RIP|pianist|Heart|Songs|Singer/NN <hook>/<hook> -LRB-|,/. <target>/<target> -RRB-|.|,|-|;/.
14	BOS/BOS <target>/<target> -|:/. <hook>/<hook> -LRB-|,/.
13	singer|Jessie|Biography|composer|singer\/songwriter|Jackie|writer/NN <hook>/<hook> born|was|Born/VBD */* -LRB-|,/. <target>/<target> -LRB-|.|,|-/.
12	*|,|\//. <target>/<target> -|:/. <hook>/<hook> ,/.
12	BOS/BOS <hook>/<hook> -LRB-|,/. */* in/IN <target>/<target> -RRB-|.|,/.
10	player|singer/NN <hook>/<hook> in/VBD <target>/<target> -LRB-|;/.
9	BOS/BOS <hook>/<hook> -LRB-|,/. */* born|oktober|de|novembre|June|Oct|April/NN <target>/<target> -RRB-|,/.
9	van|singer|musician|RIP/NN <hook>/<hook> op|Tribute|in/NN <target>/<target> -LRB-|-|;/.
9	of/IN <hook>/<hook> -LRB-|,/. */* born|in/VBD <target>/<target> -RRB-/.

9	BOS/BOS <hook>/<hook> was/VBD */* in/IN <target>/<target> while|in/IN
8	,/. <target>/<target> ,/. <hook>/<hook> was|grew|began|earned|won|had|moved/VBD
8	BOS/BOS <hook>/<hook> -LRB-|,/. */* MMDDEXPRESSION|PERSON/Entity <target>/<target> -RRB-|,|-/.
7	In|in/IN <target>/<target> ,/. */* bassist|composer|Jackie|keyboardist|vocalist/NN <hook>/<hook> was|burst|performed/VBD
7	BOS/BOS <hook>/<hook> Death|Memorial|Bandleader|Biography|Chick|Born|Glenn/NN */* -LRB-|,|-/. <target>/<target> -RRB-|,|-/.
7	,|:/. <hook>/<hook> was|opened/VBD */* ,/. <target>/<target> -RRB-|.|,/.
7	BOS/BOS <hook>/<hook> -LRB-/. */* born/VBD <target>/<target> -RRB-/.
6	of/IN <hook>/<hook> -LRB-|,|-|:/. <target>/<target> -/.
6	In|in/IN <target>/<target> ,/. <hook>/<hook> '|,/.
6	BOS/BOS <target>/<target> -|:/. */* singer|musician|Jackie|vocalist|Singer/NN <hook>/<hook> was/VBD
6	BOS/BOS <hook>/<hook> was/VBD */* ,/. <target>/<target> and/CC
5	Pianist\/vocalist|Bassist|Trombonist|Pianist/NN <hook>/<hook> born/VBD <target>/<target> in/IN
5	BOS/BOS <target>/<target> -/. <hook>/<hook> was/VBD
5	singer|musician/NN <hook>/<hook> in/IN <target>/<target> -LRB-|;/.
5	‘|-|:/. <hook>/<hook> -LRB-|,/. */* ,/. <target>/<target> in/IN
5	BOS/BOS <hook>/<hook> wurde/NN <target>/<target> in/IN
5	-/. <hook>/<hook> Death|Born|Tribute|Died/NN */* -LRB-|,/. <target>/<target> ,|-/.
4	Biography|Wiki|Jackie|Video/NN <hook>/<hook> -LRB-/. */* ,/. <target>/<target> in/IN
4	-LRB-/. <target>/<target> -/. */* late/JJ <hook>/<hook> ,/.
4	BOS/BOS <hook>/<hook> was/VBD */* MMDDEXPRESSION|LOCATION/Entity <target>/<target> -LRB-|.|,/.
4	by|on/IN <hook>/<hook> -LRB-|:/. */* IN|in/IN <target>/<target> ,/.
4	BOS/BOS <hook>/<hook> -LRB-/. */* ,|\//. <target>/<target> Birthplace|-17|Gary/NN
4	de|junio/NN <target>/<target> -RRB-/. */* como/NN <hook>/<hook> ,/.
4	-/. <hook>/<hook> YYYYEXPRESSION/Entity */* ,/. <target>/<target> -/.
4	BOS/BOS <target>/<target> -/. */* :/. <hook>/<hook> was/VBD
4	;/. <hook>/<hook> in/VBD <target>/<target> -LRB-/.
3	BOS/BOS <hook>/<hook> was/VBD */* MMDDEXPRESSION/Entity <target>/<target> in/IN
…	
1	When/WRB <hook>/<hook> -LRB-/. <target>/<target> -/.