# An Ontology-driven Rote Extractor For Pattern Disambiguation

Sheng Yin
University of Georgia
Department of Computer Science
Athens, GA 30602-7404
yin@cs.uga.edu

Ismailcem Budak Arpinar
University of Georgia
Department of Computer Science
Athens, GA 30602-7404
budak@cs.uga.edu

## ABSTRACT
In this paper, we describe an ontology-driven pattern disambiguation process for Rote Extractors. 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 test our experiments with/without the ontology. The results show that our approach can dramatically improve the performance of existing pattern-based Rote Extractors.

## Keywords
Edit-Distance, Lexical Pattern, OWL, Ontology Query, Pattern disambiguation, Pattern Generalization, POS tagging, NER and Semantic Web.

## 1. INTRODUCTION
Due to the increasing amount of information contained in the web, an automatic process is needed to fulfill different tasks, such as to search, retrieve and process web content. However, most of the 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.

Several pattern-based annotation procedures [5-9] 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 other approaches, pattern-based annotation approaches consider not only tokens, but also document structure and they can actively learn patterns for any relation. Therefore, they can be used in a widespread domain.

We redefine a new pattern representation for pattern-based annotation approaches. We address the problem of 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 (eg., 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 even 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. In addition, our approach can use the found relation to solve ambiguity problems.

## 2. Rote Extractors
We focus on one pattern based approach, the Rote method [10], 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 [11].

According to the definition of rote extractors [6], 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 $(p,q)$, 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. In the following sections, $x$ is called the *hook*, and $y$ is called the *target*. Therefore, two elements *(hook,target)* have the relation $r$.



ACMSE '10, April 15-17, 2010, Oxford, MS, USA.

$$P(r(p,q)|A1pA2qA3) = \frac{\sum_{x,y \in r} c(A1xA2yA3)}{\sum_{x,z} c(A1xA2zA3)} \quad (1)$$

In order to apply the Rote method, a process is used widely:

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 $A1hookA2targetA3$ 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.

# 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.

## 3.1 Textual Corpus Extraction

A list of concept pairs for the relation should be created for textual corpus extraction. 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 [7]. Mini-biographies are used by Mann and Yarowsky (2005) as a pair list [6]. 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.

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.

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. We only allow ASCII symbols to be used in our approach. In order to do that, we convert Unicode symbols into their corresponding ASCII symbols, and discard those sentences that are converted incorrectly.

## 3.2 Natural Language Processing

When the training corpus and the testing corpus are created, Part-of-speech Tagging (POS) tool and NER tool 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 (1) 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 (1) 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. We define a new tag, Entity, to label those named entities extracted by NER. Sentence (2) 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. Our approach could recognize four different data formarts: YYYY-MM-DD, MM/DD/YYYY, U.S.A. style and European style.

(1) Janet/NNP Evanovich/NNP is/VBZ an/DT American/JJ writer/NN ,/, born /VBN in/IN 1943/CD ,/, in /IN New /NNP Jersey /NNP ./.

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

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 /NNP will be changed into Location/entity. The content for each named entity is also recorded.

## 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.

In our approach, for each sentence in the training corpus, two related concepts are the *hook* and the *target*. The surrounding content is extracted from the output created by NLP tools, which combines the POS tag and named entities. $A1$ is a small content window before *hook*, and $A3$ is a small content window after *target*. The length of $A1$ and $A3$ is also called the content window size (cWin). If the length of the content between *hook* and *target* is greater than 2*cWin, $A2$ is the content after *hook*, whose length equals cWin and the content before *target* (length equals cWin) connected by a star wildcard. Otherwise, $A2$ is the content between *hook* and *target*.

We use patterns to represent each surrounding context extracted from the training corpus. Wildcards and disjunctions are allowed in patterns. 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: BOS (beginning of sentence, with POS tag BOS), EOS (end of sentence, with POS tag EOS), <hook> (means this element is the

hook, with POS tag <hook>) and <target> (means this element is the target, with POS tag <target>).

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 (3). Therefore, pattern (a) cannot match this sentence.

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

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

### *3.3.1 Edit-Distance based Generalization*

The edit-distance algorithm [13] 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.

In order to apply edit-distance algorithm to patterns, we allow the edit operation to change one element instead of one symbol each time. 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 (. , ! ? … " " ' - ( )). For example, pattern (b) and pattern (c) can be generalized into one pattern (d).

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

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

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

### *3.3.2 Generalization Pseudocode*

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 $P$

## 4. PATTERN APPLICATION

Once a set of patterns is 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.

### 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 $A1 hook A2 target A3$, 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 left-hand-side content $A2$ in the sentence.
    c. Look for the left-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 Creation and Inference

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, country, book, song classes 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 , which is a free CD and music database service to look up textual metadata about music, audio or data CDs. We try to collect all books written by those ten writers and all songs recorded by the seven singers. Because some songs can be recorded by several different singers, we also add those singers' information into our Ontology. 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. 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 1 shows the ontology schema.

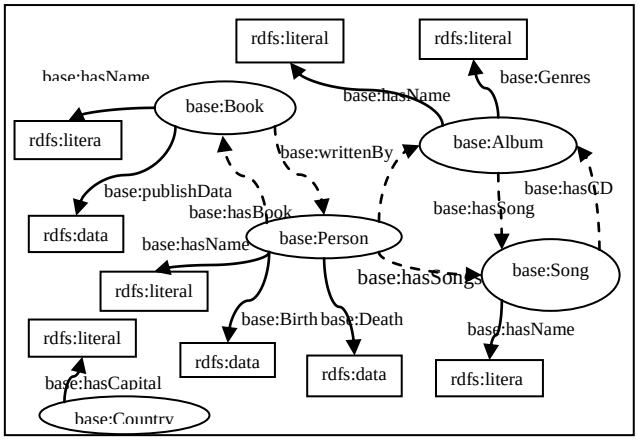


Figure 1: Ontology Schema

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

## 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, precision and F-measure rate.

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. Table 1 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. 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 for each relation. We combine the birth-year, death-year, writer-book and singer-song testing corpus together. 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 1: 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 |

Table 2 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 2: 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% | 62.8% | 100% | 77.1% |
| Death-year | 78.1% | 73.2% | 75.6% | 78.1% | 100% | 87.7% |
| Country-capital | 75.4% | 69.7% | 72.4% | 75.4% | 100% | 86% |
| Writer-book | 55.3% | 63.3% | 59% | 55.3% | 100% | 71.2% |
| Singer-song | 61% | 59% | 60% | 61% | 100% | 75.8% |

The precision rate reached 100% when the approach using the ontology. That is because our ontology contains enough knowledge to solve the ambiguity problem for our experiment. When the application process queries which relationship a concept pair belongs to, the ontology can always give a right answer. The experiment runs without the ontology, the precision rate is low. That is because the without the ontology, our approach cannot solve the ambiguity problem. Consider the case: apply the pattern *<hook>/<hook> 's/POS <target>/<target>* (writer-book relation) to Madonna's Hey You. Because our ontology has no information about Madonna, the application process cannot get the right relation between Madonna and Hey You. Therefore, the pattern will report the sentence has a writer-book relation incorrectly.

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.

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 the 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. For each relation, we use its testing corpus to run a four cross-validation to measure the recall rate.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%) and Singer-song (59.6%).

# 6. 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 55%).

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.

# 7. 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] Soderland, S., Learning Information Extraction Rules for Semi-Structured and Free Text. Mach. Learn., 1999. 34(1-3): p. 233-272.

[6] 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.

[7] 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.

[8] 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.

[9] Agichtein, E., et al., Snowball: a prototype system for extracting relations from large text collections. SIGMOD Rec., 2001. 30(2): p. 612.

[10] 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.

[11] Ruiz-Casado, M., E. Alfonseca, and P. Castells, From Wikipedia to Semantic Relation-ships: a semi-automated Annotation Approach. 2006.

[12] Wagner, R.A. and M.J. Fischer, The String-to-String Correction Problem. J. ACM, 1974. 21(1): p. 168-173.