UNSUPERVISED ONTOLOGY POPULATION USING LATENT SEMANTIC ANALYSIS

Theerayut Thongkrau and Pattarachai Lalitrojwong

Artificial Intelligence and Intelligent Systems Laboratory,
Faculty of Information Technology, King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand
Emails: s0066301@kmitl.ac.th, pattarachai@it.kmitl.ac.th

ABSTRACT

A large ontology such as lexical ontology is useful as the basic knowledge base in artificial intelligence and computational linguistics application. However, it is insufficient to recognize only existing instances for each concept. Adding new instances into the lexical ontology will expand knowledge in the system. In this paper, we propose an efficient unsupervised ontology population system that classifies new instances into a corresponding lexical ontology concept. Compared to previous related works, it does not require manual preprocessing to prepare training data. In terms of processing time, it does not need to search for many concepts in the lexical ontology. Our system employs latent semantic analysis together with context voting to find the appropriate concept of the instance. In sum, the system achieves higher accuracy when the lexical ontology contains a lot of concepts, which generally occurs in practical problems.

Index Terms-- *Ontology Population; Lexical Ontology; Latent Semantic Analysis*

1. INTRODUCTION

Ontology is the body of knowledge in a machine readable form. It is created to describe the extent of knowledge commonly used. Learning new instances to populate ontology, ontology population, can substantially enrich its content. The basic ontology population is also found in the area of named entity recognition (NER). This is an approach for classifying named entities or instances to concept names, such as people, places, companies, organizations, products, and so forth. A lexical ontology is used for several different purposes in artificial intelligence and computational linguistics applications, such as word sense disambiguation, information retrieval, automatic text classification, and automatic text summarization. The ontology-based question answering system (QA) in [1] using lexical ontology to locate domain ontology. For instance, the query "Where can I see Casino Royale?" is annotated as "Where can I see [MOVIE]?" The QA system can generate a query on the movie ontology, select related attributes, and acquire a result from the movie database. Many artificial intelligence problems, like natural language understanding, require extensive instance knowledge. Although the use of an existing ontology like lexical ontology is useful, it does not contain enough instances. New instances come into existence on a daily basis. Using the lexical ontology cannot help decide whether *Kings Island* refers to the island or the amusement park. So, classifying new instances in lexical ontology would significantly help those trying to apply lexical ontology in wider areas.

The general method used to classify instances is a similarity or memory-based approach in which the context of a phrase is used to disambiguate its sense or class or to discover other semantically related terms [2]. Its algorithm may work well with a certain number of concepts in the ontology. However, if the ontology contains too many concepts, this approach will take a lot of processing time. In this paper, we propose an approach to solve this problem. First, we attempt to acquire the meaning of the instance name using a search engine. Second, we apply the latent semantic analysis (LSA) to analyze possible concepts from words surrounded by the instance name. This technique yields the weight for each word. We will use this result to classify instances to concepts. By this method, we do not need to compare every concept in the lexical ontology. As a result, it dramatically reduces processing time. Besides, our method can be applied in any language and any domain.

The remainder of the paper is organized as follows. The next section describes related work. Section 3 introduces the LSA technique. Section 4 elaborates on the method of ontology population. Section 5 presents experiments for proving the validity of the approach. Finally, we address the conclusions.

2. RELATED WORK

Previous research work on ontology population can be categorized into two types: ontology-driven and document-driven. In ontology-driven systems, ontological data is used as input. Then they search for the corpus in web documents to identify instances and classify them into the corresponding ontology concept. For instance, OntoSyphon [3] identifies possible instances by using the ontology to specify web searches. It then verifies the candidate instances from redundancy in the web. On the other hand, document-driven systems start from a particular document. They try to identify instances or name entities from the document and classify them to the target ontology. Many of works require hard and tedious

manual preprocessing to prepare training data while some work like PANKOW [4] does not use machine learning. It annotates a specified document by extracting instances from the document and querying Google with ontology-based Heart patterns. Then it counts Heart patterns in every HTML document returning from the search engine. After that, PANKOW classifies the instances to a concept based on the results. C-PANKOW [5] presented an enhancement of PANKOW. It measures the similarity of each abstract content and the input document before accessing its relevant HTML document. Likewise, [6] does not prepare training data. It performs a top-down search along the ontology and stops at the concept that is most similar to the instance. Therefore, if the ontology contains too many concepts, this approach will take a lot of processing time.

3. LATENT SEMANTIC ANALYSIS

Latent Semantic Analysis (LSA) is a theory and method for extracting and representing the contextual-usage meaning of words by statistical computations applied to a large corpus of text [7]. The basic assumption of LSA is that the cognitive similarity between any two words is reflected in the way they co-occur in subsamples of the language. LSA is designed to uncover the latent semantic structure of a document collection (in Section 4 documents are called context) by building a semantic space. It therefore uses the word usage patterns that exist in the document collection, namely, the word co-occurrences.

The input of LSA is a matrix *A* that is a term-by-document matrix. The rows of *A* represent terms, which mark up a document (or phrases). The columns of *A* represent documents (context information), which are of a predetermined size of text such as paragraphs, sentences, and so on. Initially, each column of the matrix *A* contains zero and non-zero elements. Each non-zero element of the matrix *A* is the frequency of term in the document.

The next step is factorization of matrix *A* using Singular Value Decomposition (SVD), as shown in Fig. 1. After that, it is desirable to reduce the number of dimensions in the matrix *A* by keeping its first *k* singular values. Since these are ordered in decreasing order along the diagonal of *S* and this ordering is preserved when constructing *U* and V^T, keeping the first *k* singular values is equivalent to keeping the first *k* rows of *S* and V^T and the first *k* columns of *U*. This process is termed dimensionality reduction, and A_k is referred to as the *Rank k Approximation of A* or the reduced SVD of *A*. This is exactly what is done in LSA. This latent semantics representation is a specific data structure in low-dimensional space in which documents and terms are embedded and compared. This hidden or latent data structure is masked by noisy dimensions and becomes evident after the SVD.

4. ONTOLOGY POPULATION IN OUR APPROACH

An ontology population process consists of four main tasks as shown in Fig. 2, instance detection, context collection, LSA processing, and context voting. The HTML document is an input of our system. It detects new instances from the HTML document and sends instance names to a search engine to acquire instance contexts. All instance contexts will be sent to produce a reduced term-by-context matrix (TCM_k) using LSA. Finally, it chooses the best candidates for the lexical ontology concepts of the instances and extends the ontology by adding the instances to their corresponding lexical ontology concepts.

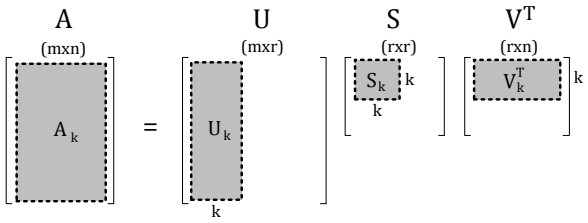

Figure 1. Factorization of Matrix *A* using SVD.

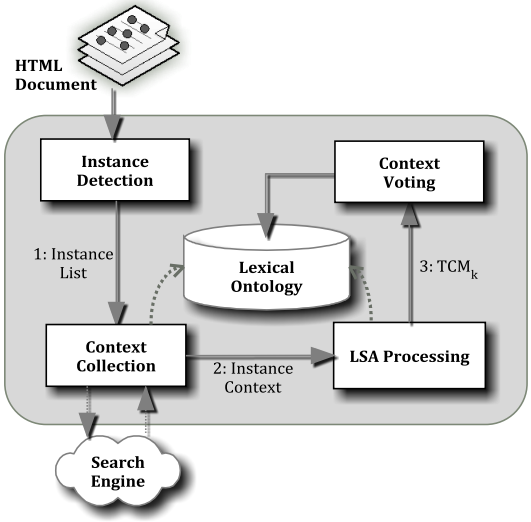

Figure 2. Steps of Ontology Population.

4.1. Instance Detection

The instance detection attempts to identify instances from the HTML document using the Stanford NER library. New instances are collected in the instance list for the next task to find their suitable concepts.

4.2. Context Collection

The context collection sends different forms of queries to a search engine. Then it gathers search results containing possible concepts of the instances. The detail of this task can be describes as follows:

4.2.1. Generating Queries

To obtain good instance contexts from the search result, the keyword to be sent to the search engine should be a phrase consisting of the instance name and other words resulting in the definition of the instance. For example, the query to be sent to the search engine may be in a form of "[instance] is a", "such as [instance]", or "What is [instance]".

4.2.2. Finding Possible Concepts

Because the search result is an XML document, we use XPath to access its content. To obtain possible concepts, there are three steps to perform, markup and format removal, tokenization, and filtration. The first step removes all markup tags and special formatting from the abstract content. The second step removes all punctuation and makes lowercase every word in the abstract. The last step removes all stop words and instance names. All nouns and noun phrases consisting of only nouns remaining in the document will be considered as possible concepts. The examples of possible concepts in each *Big Ben* context are underlined text in Fig. 3.

C1: Big Ben is a great *clock tower* at the *Palace* of Westminster.
C2: Big Ben is a nickname for the great *tower*.
C3: …such as Big Ben, the largest four-faced *clock tower* in the *world*.
C4: The *bell clock* at Westminster *Palace* is better known by the nickname Big Ben.
C5: Big Ben is a large *clock tower* that is located at the *Palace* of Westminster.
C6: Many *times* large *clocks* such as Big Ben are the *center* of a *town*.

Figure 3. The Context of *Big Ben*

4.3. LSA Processing

4.3.1. Creating TCM

The first step is to represent the possible concepts as a matrix. Each row comprised of unique words and each column comprised of contexts. From Fig. 4 there are 8 concepts so the TCM has 8 rows as shown in Fig. 4(a). A cell value is the number of times the concepts occurs in each context. For example, in C1, clock, palace, and tower occur once while the other concepts do not occur.

4.3.2. Normalizing Frequencies

The number of times the concept occurs in each context is the raw frequency. In fact, some context may repeat few times or has keyword spamming. We made them to less susceptible by normalizing frequencies. A normalization function is generally applied to each element of TCM as shown in Formula (1), where $W_{i,j}$ is a term weight, $f_{i,j}$ is frequency of possible concept i in context j, and $max\ f_j$ is maximum frequency in context j.

$$W_{i,j} = \frac{f_{i,j}}{\max\ f_j} \tag{1}$$

4.3.3. Creating TCM_k

This step is the factorization of TCM using SVD, as shown in Fig. 4(b). After that, it is desirable to reduce the number of dimensions in the matrix TCM by keeping its first k dimensions. For example, when $k=2$, we keep the first two columns of U, the first two rows and columns of S, and the first two rows of V^T. The parts of the matrices retained are depicted in Fig. 4(c). Their product of U_k, S_k, and V_k^T is simply computed to construct TCM_k as shown in Fig 4(d). This process has readjusted term weights which are now either incremented or lowered in the truncated matrix TCM_k. Let us underscore that the redistribution is based on co-occurrence terms. Look at shaded cells of *bell* and *tower* in Fig. 4(a) and Fig. 4(d). The word *tower* does not appear at C4. Since both C1 and C4 contains *palace*, the zero entry for *tower* has been replaced with 0.42 in TCM_k. In contrast, the value 1 for *bell* at C4 in TCM has been replaced with 0.09 in TCM_k. The reflecting fact is unexpected in this context and should be counted as unimportant in characterizing the instance context.

TCM

	C1	C2	C3	C4	C5	C6
bell	0	0	0	1	0	0
center	0	0	0	0	0	1
clock	1	0	1	0	1	1
palace	1	0	0	1	1	0
time	0	0	0	0	0	1
tower	1	1	1	0	1	0
town	0	0	0	0	0	1
world	0	0	1	0	0	0

(a)

U

0.05	-0.13	0.50	0.59	-0.39	0.42
0.11	0.49	0.13	-0.02	-0.21	-0.52
0.64	0.27	-0.09	0.04	0.53	0.42
0.43	-0.33	0.64	-0.12	0.21	-0.42
0.11	0.49	0.13	-0.02	-0.21	0.11
0.59	-0.30	-0.38	-0.20	-0.61	0.00
0.11	0.49	0.13	-0.02	-0.21	-0.01
0.16	-0.01	-0.37	0.77	0.14	-0.42

S

2.98	0.00	0.00	0.00	0.00	0.00
0.00	1.89	0.00	0.00	0.00	0.00
0.00	0.00	1.52	0.00	0.00	0.00
0.00	0.00	0.00	0.89	0.00	0.00
0.00	0.00	0.00	0.00	0.67	0.00
0.00	0.00	0.00	0.00	0.00	0.00

V^T

0.55	-0.19	0.11	-0.32	0.20	-0.71
0.20	-0.16	-0.25	-0.22	-0.91	0.00
0.46	-0.02	-0.55	0.68	0.09	0.00
0.16	-0.25	0.75	0.53	-0.26	0.00
0.55	-0.19	0.11	-0.32	0.20	0.71
0.32	0.92	0.19	-0.01	-0.14	0.00

(b)

U_k

0.05	-0.13
0.11	0.49
0.64	0.27
0.43	-0.33
0.11	0.49
0.59	-0.30
0.11	0.49
0.16	-0.01

S_k

2.98	0.00
0.00	1.89

V_k^T

0.55	-0.19	0.11	-0.32	0.20	-0.71
0.20	-0.16	-0.25	-0.22	-0.91	0.00

(c)

TCM_k

	C1	C2	C3	C4	C5	C6
bell	0.14	0.07	0.08	0.09	0.14	-0.17
center	0.00	-0.08	0.13	-0.17	0.00	0.94
clock	0.95	0.30	0.87	0.18	0.95	1.08
palace	0.83	0.35	0.60	0.36	0.83	-0.17
time	0.00	-0.08	0.13	-0.17	0.00	0.94
tower	1.09	0.44	0.83	0.42	1.09	0.05
town	0.00	-0.08	0.13	-0.17	0.00	0.94
world	0.26	0.10	0.22	0.08	0.26	0.13

(d)

Figure 4. LSA Processing Steps

We can draw vectors and conduct a visual inspection from U_k and V_k^T as shown in Fig. 5. Dark arrows represent the vectors of C1 to C6. Notice the vector direction, related contexts will be grouped in the same direction. C1 to C5 are in the same direction because their terms co-occur in the same context. Light arrows represent the term vectors. *Tower* and *palace* are grouped in the same direction. On the other hand, *time*, *town*, and *center* are in another direction.

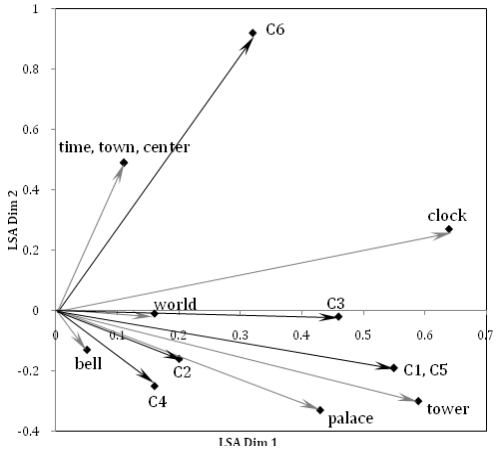

Figure 5. LSA Context Vector Space

4.4. Context Voting

Context voting is a task that assigns the concept(s) with the highest weight as the candidate(s) for each context. Then the concept candidate(s) happens the most in every context is selected to be the corresponding concept of the instance. In Fig. 6, C1, C2, C4, and C5 have the highest weight in *tower* concept. C3 and C6 have the highest weight in *clock* concept. The *tower* concept is the most happens in candidates then *Big Ben* assigns to the *tower* concept.

TCM_k

	C1	C2	C3	C4	C5	C6
bell	0.14	0.07	0.08	0.09	0.14	-0.17
center	0.00	-0.08	0.13	-0.17	0.00	0.94
clock	0.95	0.30	0.87	0.18	0.95	1.08
palace	0.83	0.35	0.60	0.36	0.83	-0.17
time	0.00	-0.08	0.13	-0.17	0.00	0.94
tower	1.09	0.44	0.83	0.42	1.09	0.05
town	0.00	-0.08	0.13	-0.17	0.00	0.94
world	0.26	0.10	0.22	0.08	0.26	0.13

Voting Table

Answer concept	Score
tower	**4**
clock	2

Figure 6. Example of Context Voting

5. EXPERIMENTS

In order to evaluate our approach, we prepared a standard data set of instances from existing lexical ontology, WordNet, by randomly selecting 450 instances. The result from our system was compared to the standard data set and measured in accuracy value. We conducted these experiments with two parameters. The first parameter was a reducing dimension k of U_k, S_k, and V_k^T. The second parameter was the number of context collection. With different values of k and the number of context collection between 10 and 100, we have found that our approach gives the accuracy in the range of 37.41% and 44.59% higher than the accuracy from PANKOW and C-PANKOW which are 19.69% and 15.24% respectively [5].

6. CONCLUSION

We propose an approach for ontology population using latent semantic analysis (LSA). It assumes that concept candidates are in the definition of the instances. Therefore, it attempts to acquire their meanings from a search engine. Then, it employs LSA to analyze possible concepts from words surrounded by the instance name. This technique computes the weights to be used for classifying the instances to their concepts. By this method, it does not need to compare every concept in the lexical ontology. The experiment demonstrates that our approach achieves higher accuracy than previous research.

REFERENCES

[1] F. Oscar, et al., "Addressing ontology-based question answering with collections of user queries," *Information Processing & Management*, vol. 45, no. 2, 2009, pp. 175-188.

[2] P. Cimiano, *Ontology Learning and Population from Text: Algorithms, Evaluation and Applications*, Springer, New York, 2006.

[3] L. K. McDowell and M. Cafarella, "Ontology-driven, unsupervised instance population," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 6, no. 3, 2008, pp. 218-236.

[4] P. Cimiano, et al, "Learning by Googling," *ACM SIGKDD Explorations Newsletter*, 2004, pp. 24-33.

[5] P. Cimiano, et al., "Gimme' the context: context-driven automatic semantic annotation with C-PANKOW," *Proceedings of the 14th international conference on World Wide Web* 2005, pp. 332-341.

[6] E. Alfonseca and S. Manandhar, "Extending a lexical ontology by a combination of distributional semantics signatures," *Proceedings of the* 13*th International Conference on Knowledge Engineering and Management*, 2002, pp. 1-7.

[7] T.K. Landauer and S.T. Dutnais, "A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge," *Psychological Review*, vol. 104, 1997, pp. 211-140.