Building Information Extraction System Based on Computing Domain Ontology

Chien D C Ta
Faculty of Computer Science and Engineering,
HoChiMinh City University of Technology
268 Ly Thuong Kiet HCM City, Vietnamese
tdcchien@gmail.com

Tuoi Phan Thi
Faculty of Computer Science and Engineering,
HoChiMinh City University of Technology
268 Ly Thuong Kiet HCM City, Vietnamese
tuoi@cse.hcmut.edu.vn

ABSTRACT
In this paper, we present an Information Extraction (IE) system, which is built from unstructured text based on Computing domain ontology. The IE system comprises four sequential processing steps: preprocessing, topic identifier, building domain specific ontology and extracting information from text corpus. The first two steps perform generic Natural Language Processing (NLP) and Machine Learning tasks, while the last two phases are application-specific and require a thorough understanding of the application domain. Furthermore, the paper focuses on evaluating the IE IEsystem by selected methods. One of these methods that we introduced here, is comparative. Comparative evaluation performed in this study use of Key Exchange Algorithm with the same corpus to contrast results. Results generated by such experiments show that this IE system outperforms Key Exchange Algorithm, respectably.

Categories and Subject Descriptors
H.3.3 [**Information Search and Retrieval**]: Information Filtering

General Terms
Theory, Algorithms

Keywords
Domain ontology, Knowledge based system, Semantic Relation, Information Extraction

1. Introduction
NLP is used to enhance the querying of information systems. In general, a query is first syntactically analyzed. The output then is converted to semantic structures, which are next translated into SQL queries. . This is done in two phases: first, a query reduction phase where useless words are dismissed, and second, a query expansion phase using synonyms or co-occurring terms, which are obtained from dictionaries.

In this paper's context, Information Extraction (IE) is the automatic extraction and meaningful interpretation of selected Information from natural language texts.

iiWAS '14, December 04 - 06 2014, Hanoi, Viet Nam

http://dx.doi.org/10.1145/2684200.2684329

Today, many IE systems exist that perform a wide variety of tasks and they are scattered in a large number of domains. It would be very useful to have an IE system that provides us the knowledge about a specific domain based on ontology, which is a goal that this work tries to achieve.

Our key contribution are as follows: (i) we propose an hybrid method combining NLP, Machine Learning and statistic method with more high precision on computing domain in order to extract instance data from Wikipedia, WordNet and ACM Digital Library; (ii) we build Computing Domain Ontology, which covers 170 distinct categories of computing domain; (iii) we build IE system providing knowledge of computing domain based on this ontology.

The rest of this paper is organized as follows: section 2 examines related work and overviews a sample of NLP applications and tools; section 3 introduces the proposed methodology; section 4 illustrates the methodology to evaluate system; section 5 discusses results and future works.

2. Related Work
As outlined from K.Church [1], NLP is used in many commercial applications that include word processing, translation, and information management. Information management applications use NLP for document retrieval and extracting structured data from natural language. Tru et al [2] introduced VN-KIM IE, the information extraction module of the semantic web system VN-KIM that his group developed. The function of VN-KIM IE was to automatically recognize named-entities in Vietnamese web pages, by identifying their classes, and addresses if existing, in the knowledge base of discourse. That information was then annotated to those web pages, providing a basis for NE-based searching on them, as compared to the current keyword based one.

In addition to the above, the use of NLP in the design of information systems was investigated by E.Metais [3]. E.Chieze et al. [4] built Information Extraction system that extract data automatically for text document summarization. T.Shintani et al. [5] introduced PAPITS system. This system is proposed the statistical method with Information Gain in order to category of text documents. S.Heni et al. [6] proposed an approach based on a Neural Principal Component Analysis that express the maximum variance of data and extract the principal component from it, by calculating the correlation between words of each document, to determine the keywords that give out the fields of interest of each document content. L.Zhang et al [7] have used ontology in the extraction of a partial building information model from the original complete model. F.Harrag [8] has proposed a methodology in order to detect and extract passages or sequences of words containing relevant to

information from the prophetic narrations texts. Generally, knowledge extraction is done by constructing a schema that models the contents of the data. S.Jonnalagadda et al [9] proposed an approach to minimize this limitation by combining supervised machine learning with empirical learning of semantic relatedness from the distribution of the relevant words in additional un-annotated text. A.Ittoo et al [10] presented a minimally-supervised approach for learning part–whole relations from texts. The novelty in their approach lies in the use of Wikipedia as a knowledge-base, from which they first acquire a set of reliable patterns that express part–whole relations.

Those research attempts were meant to build IE systems from web documents or text files based on ontology. They either used NLP techniques, the statistics, or Machine Learning approach for building ontology. Our research integrates NLP, Matching Learning and statistic method to build smart IE system based on domain specific ontology.

3. Information Extraction (IE) From Text Documents

The methodology described in this paper closely follows this procedure by identifying four stages in IE. The first and second stages relate to some methods of NLP. The third stage, domain specific ontology, describes the procedures for building ontology about Computing domain. The last stage, extract data from testing corpus, that stage considers determining keywords/key phrases of sentences in text documents, generates syntactic relation and measures them before displaying extracted results to users.

We propose a model of Information Extraction from text documents, as shown in Figure 1.

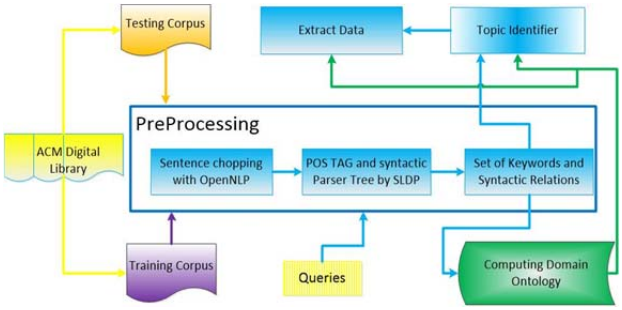

Figure 1. Model of IE based on Domain Ontology

There are some major components in this model: preprocessing, topic identifier, computing domain ontology (CDO) and extracting data.

3.1 Preprocessing

There are four steps in this stage, namely sentence chopping, part-of-speech tagging, syntactic parsing, and relation generation.

We separate text documents of ACM Digital Library into two sets, one for building CDO (training corpus), and another for testing purpose (testing corpus). However, both of them must take to preprocessing for implementing POS tag, syntactic parsing and generating syntactic relations. All keywords, key phrases, syntactic relations extracted from training corpus will be used for building CDO, while all keywords, key phrases, syntactic relations extracted from testing corpus will be used for displaying to end- user. In this stage, we combine Machine Learning and NLP tools to identify and extract keywords, key phrases and syntactic relations. Additionally, before inserting into CDO or displaying to users, they are evaluated by Information Gain factor (statistic method) in order to get rid of wrong words.

3.1.1 Sentence chopping

There may be one or more sentences in an input query or in a paragraph of a text document. These sentences are usually separated by symbols, such as dot (“.”), question mark (“?”), exclamation mark (“!”), etc. We use OpenNLP [11] to detect and extract sentences one by one before going through next stage.

3.1.2 Part-of-speech tagging (POG Tag)

Part-of-speech also called grammatical tagging or word-category disambiguation, is the process of marking up a word in a text (corpus) as corresponding to a particular part of speech, based on both its definition, as well as its context—i.e. relationship with adjacent and related words in a phrase, sentence, or paragraph. A simplified form of this is commonly taught to school-age children, in the identification of words as nouns, verbs, adjectives, adverbs [12], etc. A word can have many different parts-of-speech. The “correct” POS is defined using context. The question is how to accurately model context.

In order to make POS tagging, we use Stanford Lexical Dependent Tree [13] in case since we also get syntactic tree besides POS tag. Example: sentence “A quick brown fox jumped over the lazy dog”. The sentence’s syntactic tree is shown as Figure 2.

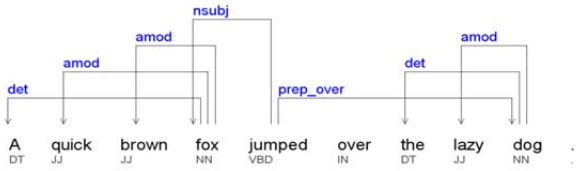

Figure 2. Dependent tree of the sentence “A quick brown fox jumped over the lazy dog”

The table 1 is shown the POS tags used in the tagging and parsing phases using SLDP

Table 1. POS tags used in tagging and parsing

No	Symbol	Part of speech
1	*CC*	Coordinating conjunction
2	CD	Cardinal number
3	EX	Existential there
4	FW	Foreign word
5	IN	Preposition or subordinating conjunction
6	JJ	Adjective
7	*JJR*	Adjective, comparative
8	JJS	Adjective, superlative
9	LS	List item marker
10	MD	Modal
11	NN	Noun, singular or mass
12	NNS	Noun, plural
13	NNP	Proper noun, singular
14	NNPS	Proper noun, plural
15	PDT	Predetermine
16	POS	Possessive ending

17	PRP	Personal pronoun
18	RB	Adverb
19	RBR	Adverb, comparative
20	VP	Verb, base form
21	VBD	Verb, past tense
22	VBG	Verb, gerund or present participle
23	VBZ	Verb, 3rd person singular present
24	VBP	Verb, non-3rd person singular present
25	DT	Determiner

3.1.3 Syntactic parsing

Syntactic parsing is the procedure of assigning a syntactic parse tree to a sentence. This is a fundamental problem in language processing for both machines and humans in order to understand a sentence. Parse trees like the one produced in Figure 2 represent several levels of information. The non-terminal directly above each word in the sentence is the POS for that word. For example, the tree indicates that "fox" is an NN (non-proper noun), "jumped" is a VBD (past-tense verb). A parse tree also describes the hierarchical grouping of words into phrases: "A quick brown fox" is an NP (noun phrase), "jumped over the lazy dot" is a VP (verb phrase), and so on. Finally, a parse tree reveals grammatical relations between phrases or words. In a rule S →NP VP, the NP is almost always the subject of the verb within the VP. It is noted that a rule in a parse tree is given as (parent → $child_1$... $child_n$), where $child_1$... $child_n$ are the children nodes of the node parent. Similarly, in a rule VP → VB NP, NP usually represents an Object–Verb relationship (lazy dog is the object of jumped). The step of relation generation uses such rules to generate useful relationships.

3.1.4 Relation generation

In this stage, we want to find all syntactic relations that are relevant to Computing domain. These relations can be of any kind: Subject–Verb (SV), Subject–Verb–Object (SVO), Adjective–Noun (AN), etc. The complement-adjunct distinction generated by the syntactic parsing phase greatly simplifies the task of this step. We read dozens of computing reviews and online news documents, and tried to identify the principal relations that passed on important information. Our statistics showed that roughly 75% of those were based on SVO relations, 20% were SV relations, and the other 5% were AN, Verb-Adverb (VAdv) and other relations. Therefore, it was a natural choice to extract SVO and SV relations from the parsed text. We then insert these syntactic relations into domain specific ontology.

3.1.5 Topic Identifier

According to statistics, topic of a query/sentence is usually represented by nouns, noun phrases, and verb phrases. We use Natural Language Processing tools in this stage, including OpenNLP and SLDP in order to implement Part-of-Speech Tagging and parsing. We then have a set of keywords, which be used to match with instance data of domain ontology to determine topic of a sentence. We propose the topic identifying algorithm, as follows.

Procedure Identify_Topic(Set of dependencies)

Subject = Find_Subject(Set of dependencies)

Predicate =Find_Predicate(Set of dependencies)

Direct_InDirectObj= Find_DIObj((Set of dependencies)

Set of Keywords = [Subject, Predicate, Direct_InDirectObj Merge(Subject, Predicate, Direct_InDirectObj)]

Topic = Find_topic(Set of Keywords, CDO)

Return topic

End Pro

3.1.6 Computing Domain Ontology (CDO)

There are four layers in CDO:

- The first layer is known as the Topic layer. In order to build it, we extract items from ACM Categories [14]. We obtain over 170 different categories from this site and rearrange them in this layer.
- Next layer is known as the Ingredient layer. In this layer, there are many different instance data, which are defined as nouns or compound nouns from vocabulary on Computing domain, e.g., "robot", "Super vector machine", "Local Area network", "wireless", "UML", etc. In order to setup this layer, we use Wikipedia focus on English language and Computing domain.
- The third layer of CDO is known as the Synset layer. To setup this layer, we use WordNet ontology. Similar to Wikipedia, we only focus on Computing domain. This layer encloses a set of synsets. A synset includes synonyms, hyponyms, and hypernyms of instance data of the Ingredient layer. As we use two ontologies, namely Wikipedia and WordNet, we have over 800,000 instances that belong to many different categories of Computing domain. That is an advantage of our IE system.
- The last layer of CDO is known as the sentence layer. Instances of this layer are sentences that represent syntactic relations extracted from preprocessing stage (Figure 3). Hence, these sentences are linked to one or many terms of the Ingredient layer. This layer also includes sentences that represent semantic relations between terms of Ingredient layer, such as, IS-A, PART-OF, MADE-OF, RESULT-OF, etc. The overall picture of CDO is shown as Figure 3.

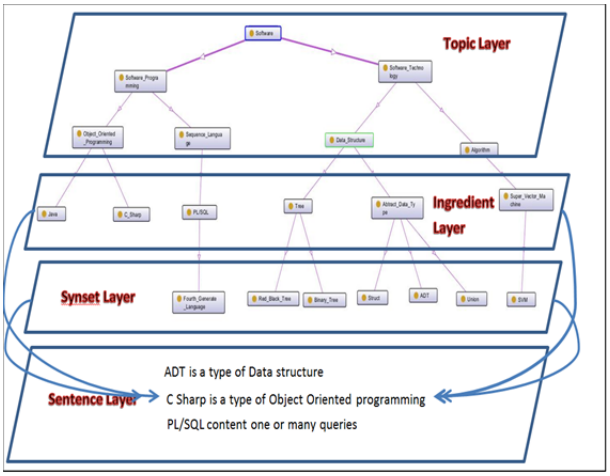

Figure 3. CDO hierarchy is presented by Protégé

3.1.7 Extracting data based on CDO

After identifying topics of an input query, we extract information that is related to these topics based on CDO.

Our target is to extract the main points from testing corpus that have the same query topic delivered from the Topic Identifier. Firstly, we cluster all text documents of testing corpus based on their categories. As testing corpus is received from ACM Digital Library, each text document is assigned a category, which is in the same the ACM category hierarchy. We then chop all sentences in text documents, and implement POS tagging and parse these sentences. In this stage, we re-use OpenNLP and SLDP. Based on syntactic relations from SLDP,

we have a set of candidate keywords, which are evaluated by Information Gain. Any keyword that has Information Gain factor greater than the threshold value T (T=0.6 in case) becomes a main keyword. As a finally result, we have set of main keywords.

Moreover, the result of data extraction also presents the semantic relations between keywords. That is another advantage of our proposal. The semantic relations may be IS-A, PART-OF, HAVING-OF, etc., that are extracted directly from WordNet or CDO.

We will illustrate our approach using a simple example. Instead of file processing, we input only one sentence, e.g., "Java is Object Oriented Programming language" for demo of purpose. We show how to extract data based on this sentence. First step is to chop sentence with OpenNLP. In our case, we have only one sentence. The next step is to map the sentence into a corresponding POS tag sequence and parsing as shown in Figure 4.

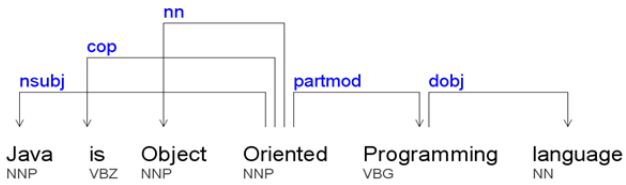

Figure 4. Dependent Tree for "Java is object oriented programming language"

The above tree reflects the syntactic structure of the sentence. We also get syntactic relations (dependency) from SLDP as follows:

Nsubj (Oriented-4, Java-1)
Cop (Oriented-4, is-2)
Nn (Oriented-4, Object-3)
Root (ROOT-0, Oriented-4)
Partmod (Oriented-4, Programming-5)
Dobj (Programming-5, language-6)

Where:

- Nsubj, Cop, Nn, Root, Partmod, Dobj represent for nominal subject, copular, noun compound modifier, root of tree, participial modifier, direct object.
- The numbers after the words indicate the word position in the sentence
- All the dependencies that are generated from SLDP form a triple: a grammatical relation holding between two words (head and dependent).

The third step is the extraction of relevant relations and keywords from the syntactic parse tree. Based on these above relations and topic identifying algorithm, it can recognize that the Subject–Verb–Object relation "**is** (Java, programming language)" is extracted from the parse tree and "Java", "programming language" are main keywords. Based on these keywords and CDO, topic identifier will return query's topic, namely "Oriented programming language". We then extract keywords, key phrases, and semantic relations from testing corpus having category to be in the same this topic.

Moreover, the IE system also detect some semantic relations in sentence layer of CDO that are related to "java" and "programming language" keywords, such as, "Java is a programming language that enables the programmer to associate a set of procedures with each type of data structure" or "java is a platform-independent object-oriented programming language".

The final step is the evaluation of such relations and keywords before displaying result. The evaluation of these results is shown as Figure 5.

Figure 5. Evaluation keyword and syntactic relation

Finally, the result of data extraction is shown in table 2.

Table 2. Results of data extraction from testing corpus based on CDO

Display results of the data extraction
• Java is programming language • Java programming language tutorial • Java programming language for beginners • Features java programming language • Applications of java programming language • Java language has its own structure, syntax rules, and programming paradigm • The Java Development Kit • The Java Runtime Environment • Structure of a Java object • Java with multithread in program • Java and C Sharp are Object Oriented Programming • The Java programming language from its inception has been publicized as a web programming language. • Java is a programming language that enables the programmer to associate a set of procedures with each type of data structure. • Java is a platform-independent object-oriented programming language.

4. Evaluation of IE System

4.1 Evaluation based on three measures

The performance of information extraction (IE) system is measured using three factors: Precision, Recall and F-measure [2] . These factors are calculated by each category in ITO as below:

$$P(C_i) = \frac{Correct(Ci)}{Correct(Ci)+Wrong(Ci)} \quad (1)$$

$$R(C_i) = \frac{Correct(Ci)}{Correct(Ci)+ Missing(Ci)} \quad (2)$$

$$\text{F} - \text{Mesure} = 2\frac{\text{Precision}*\text{Recall}}{\text{Precision}+\text{Recall}} \quad (3)$$

Where C_i represents a category in ITO and correct, wrong, missing represent the number of results that are correct, wrong, missing while the system extracts data from the testing corpus, respectively. We pick a random five categories from ACM Digital Library as below:

- One corpus with 150 papers (from ACM Digital Library) in Software category.
- One corpus with 150 papers (from ACM Digital Library) in Logic Design.
- One corpus with 150 papers (from ACM Digital Library) in Operating System category.
- One corpus with 150 papers (from ACM Digital Library) in Artificial Intelligent category.
- One corpus with 150 papers (from ACM Digital Library) in Process Management category.

Table 3. Evaluation of IE system

Categories	Precision	Recall	F-Measure
Artificial Intelligent	97.03%	88.62%	93.00%
Operating System	84.47%	81.37%	83.00%
Logic Design	96.41%	54.72%	70.00%
Process Management	96.72%	76.02%	86.00%
Software	96.52%	92.19%	95.00%

According to the results of evaluation, the IE system yields a performance respectably. These results confirm that all our proposed algorithms and models are valid.

4.2 Comparative approach

We use Key Exchange Algorithm [15] for comparative approach. KEA is an algorithm for extracting key phrases from text documents. It can be either used for free indexing, where key phrases are selected from the document itself, or for indexing with a controlled vocabulary. KEA can also be used for automatic tagging. KEA is implemented in Java and is platform independent. It is open-source software distributed under the GNU General Public License. In order to compare, two following testing corpora are used

- One corpus with 150 papers (from ACM Digital Library) in Operating System category.
- One corpus with 150 papers (from ACM Digital Library) in Artificial Intelligent category.

The results are shown in table 4.

Table 4. The result of evaluation using KEA algorithm on testing corpora

Categories	Precision	Recall	F-Measure
Operating System	76.14%	71.21%	74.00%
Artificial Intelligent	87.25%	85.61%	87.00%

The scores reported in table 4 and 5 reveal that our approach for extracting based on domain specific ontology outperforms the KEA algorithm.

5. CONCLUSION

This paper presents procedures of building an IE system based on domain specific ontology. The described general framework comprises four sequential processing steps, which are preprocessing, topic identifier, building domain specific ontology and extracting information from text corpus. In this case, domain ontology's focus is only on the Computing domain. The ontology is built based on text corpus, Wikipedia and WordNet. This paper also proposes a combined methodology including Machine Learning, NLP and statistic. Additionally, since data are collected from distinct sources, such as text files of ACM Digital Library, Wikipedia and WordNet, we have over 800,000 distinct instances that belong to information technology domain. That is an advantage of our IE system. Furthermore, we also ensure the semantic consistency of instances in this system. The overall evaluation of IE system implemented based on Precision and Recall measures. A comparative evaluation using KEA algorithm was also proposed. The experiment results show that our approach outperforms KEA respectably.

In the future work, we will focus on automated IE evaluations and improve the measures, such as precision and recall, in order to have an IE system perfectly.

References

[1] K. Church et al, "Commercial applications of natural language processing," *Communications of the ACM*, vol. 38, no. 11, pp. 71 - 79, Nov. 1995.

[2] Tru H. Cao et al, "VN-KIM IE: Automatic Extraction of Vietnamese Named Entities on the Web," *New Generation Computing*, vol. 25, no. 3, pp. 277-292, 2007.

[3] E.Metais et al, "Enhancing information systems management with natural language processing techniques," *Data & Knowledge Engineering*, vol. 41, no. 2-3, pp. 247-272, 2002.

[4] E.Chieze et al, "An Automatic System for Summarizationand Information Extraction of Legal Information," in *The 23rd Canadian Conference on Artificial Intelligence*, Ottawa, Canada, 2010.

[5] R.J.Kate et al, "Joint Entity and Relation Extraction using Card-Pyramid Parsing," in *The 14th Conference on Computational Natural Language Learning (CoNLL-2010)*, Uppsala, Sweden, 2010, pp. 203-212.

[6] Heni et al, "A Neural Principal Component Analysis for text based documents keywords extraction," in *IEEE 2011 3rd International Conference on Next Generation Networks and Services (NGNS)*, 2011.

[7] Zhang, Le et al, "Ontology Based Partial Building Information Model Extraction," *Journal of Computing in Civil Engineering*, vol. 27, no. 6, pp. 576-584, Dec. 2013.

[8] F.Harrag, "Text mining approach for knowledge extraction," *Computers in Human Behavior*, vol. 30, pp. 558-566, Jan. 2014.

[9] S.Jonnalagadda et al, "Enhancing clinical concept extraction with distributional semantics," *Journal of Biomedical Informatics*, vol. 45, pp. 129-140, 2012.

[10] A. Ittoo, G.Bouma, "Minimally-supervised extraction of domain-specific part–whole relations using Wikipedia as knowledge-base," *Data & Knowledge Engineering*, vol. 85, pp. 57-79, May 2013.

[11] The Apache Software Foundation. [Online]. https://opennlp.apache.org/

[12] [Online]. http://en.wikipedia.org/wiki/Part-of-speech_tagging

[13] [Online]. http://nlp.stanford.edu/software/lex-parser.shtml

[14] ACM. [Online]. http://www.acm.org/about/class/ccs98-html

[15] [Online]. https://code.google.com/p/kea-algorithm/

[16] J. Xu et al, "L1 Graph based on sparse coding for Feature Selection," *Lecture Notes in Computer Science*, vol. 7951, pp. 594-601, 2013.