Un Sistema de Extracción de Información Basado en Ontologías para Documentos en el Dominio de las Tecnologías de Información

An Ontology-Based Information Extractor for Data-Rich Documents in the Information Technology Domain

Sergio G. Jiménez V., Ing., Fabio A. González O., PhD.
National University of Colombia - Branch Bogota
{sgjimenezv,fagonzalezo}@unal.edu.co

Recibido para revisión 28 de Noviembre de 2007, aceptado 14 de Febrero de 2008, versión final 28 de Febrero de 2008

Resumen—**Este artículo se presenta un método de extracción de información adaptado a documentos ricos en datos, basado en el conocimiento representado en una ontología de dominio. El extractor combina un buscador aproximado de cadenas de caracteres y un algoritmo para desambiguación automática de sentidos de palabras (WSD). El buscador aproximado de cadenas de caracteres encuentra menciones a los términos combinando medidas de similitud a nivel de carácter y de palabra soportando acrónimos no estandarizados y estilos inconsistentes de abreviación. Proponemos una distancia de edición a nivel de caracteres con sensibilidad a prefijos llamada *root distance* y un nuevo algoritmo de similitud de cadenas a nivel de palabras para detección automática de acrónimos. Adicionalmente se utilizó una estrategia de WSD usando una medida de afinidad semántica basada en ontologías para resolver la ambigüedad inherente de algunos términos. El modulo de WSD encuentra combinaciones de sentidos para todo el documento optimizando la coherencia semántica del discurso. Nuestro enfoque resulta apropiado para la extracción de información en documentos ricos en datos que describen un solo objeto (i.e. producto) por documento. Los experimentos alcanzaron una precisión del 78,9% con una cobertura del 99.5% utilizando documentos y una ontología relacionada con el dominio de las computadoras portátiles.**

Palabras Claves—**Gestión del conocimiento, Extracción de Información, Ontologías, Búsqueda Aproximada de Cadenas, desambiguación automática de sentidos de palabras, Afinidad semántica**

Abstract—**This paper presents an information extraction method, suitable for data-rich documents, based on the knowledge represented in a domain ontology. The extractor combines a fuzzy string matcher and a word sense disambiguation (WSD) algorithm. The fuzzy string matcher finds mentions of terms combining character-level and token-level similarity measures dealing with non-standardized acronyms and inconsistent abbreviation styles. We propose a new character-level edit distance sensitive to prefixes called root distance and a token-level similarity algorithm for fuzzy acronym detection. Additionally, a WSD strategy using an ontology-based semantic relatedness measure is used to solve the inherent ambiguity of some entities. The WSD module finds a sense combination over all the document length optimizing the document semantic coherence. Our approach seems to be suitable to extract information from data-rich documents describing only one main object (i.e. product) by document. The results showed a precision of 78.9% with 99.5% recall using documents and an ontology related to laptop computers domain.**

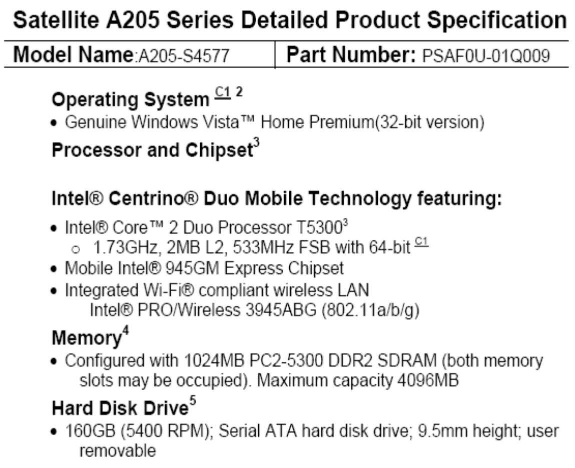
Keywords—**Knowledge Management, Information Extraction, Ontologies, Fuzzy String Searching, Word Sense Disambiguation, Semantic Relatedness**

I. INTRODUCTION

The amount of documents available electronically has increased dramatically with the vertiginous growth of the Web. Nevertheless, our capacity to "understand" automatically (i.e. machine reading) those documents is still considered an open problem [1]. Information Extraction (IE) is a reading process which aims to extract structured information from un-structured text documents. For instance, to obtain a list of authors, book titles and publishers from a set of literary critics. IE is considered a shallow reading process but it might be considered a step towards solving the machine-reading challenge.

Specifically, IE aims to find some selected entities and simple relations in text documents. In our book publishing example, three target fields can be extracted (i.e. author, title and publisher) and two possible relations are *is-written-by* and *is-published-by*. Generally, IE seeks only for a small part of document information in comparison with the entire document information. Many other entities and complex relations expressed using natural language are disregarded. For instance, in our literary critics example, feelings, opinions, favorability of the critic and final critic's recommendation are out of the reach of IE.

However, other types of documents could be considered with higher extracted/included information ratio. For instance, laptop computer data-sheets describe products in detail, including features, composing parts and performance capabilities (see Fig. 1). In the remainder of this paper we refer that type of documents as data-rich [2]. Those documents are plenty of entities to be extracted and relationships between them are clearly defined. Consequently, it is possible to think that IE performed over *data-rich* documents is a machine-reading process not as shallow as IE over natural language text documents.


Satellite A205 Series Detailed Product Specification

Model Name:A205-S4577	**Part Number:** PSAF0U-01Q009

Operating System [C1] [2]
- Genuine Windows Vista™ Home Premium(32-bit version)

Processor and Chipset[3]

Intel® Centrino® Duo Mobile Technology featuring:
- Intel® Core™ 2 Duo Processor T5300[3]
  - 1.73GHz, 2MB L2, 533MHz FSB with 64-bit [C1]
- Mobile Intel® 945GM Express Chipset
- Integrated Wi-Fi® compliant wireless LAN
  Intel® PRO/Wireless 3945ABG (802.11a/b/g)

Memory[4]
- Configured with 1024MB PC2-5300 DDR2 SDRAM (both memory slots may be occupied). Maximum capacity 4096MB

Hard Disk Drive[5]
- 160GB (5400 RPM); Serial ATA hard disk drive; 9.5mm height; user removable

Fig. 1 Sample of a laptop data-sheet.

Our IE approach for data-rich docs is to borrow from Natural Language Processing field a word sense disambiguation (WSD) [3] strategy as labeling technique. WSD aims to label all open class words in a text sequence [4]. Moreover, WSD has coverage broader than other un-supervised labeling tasks commonly used in IE such as named entity recognition. Our method is full un-supervised and uses an ontology to represent all specific domain knowledge required for extracting product features from Information Technology (IT) domain data-sheets.

In addition, we enrich our model improving the simplistic string matching using a fuzzy string matcher [6]. That matcher combines edit-distance algorithms at character and word level, supporting entity identification and dealing with *non*-standardized acronyms and abbreviations.

This paper is organized as follows. Problem definition is presented in Section II. In Section III, we give the required background. Sections IV and V present our proposed approach to the problem. Results from some experiments carried out using our model are presented and discussed in Section VI. Section VII review related work. Finally we present some concluding remarks and discuss future work in Section VIII.

II. PROBLEM DEFINITION

Data-rich documents are frequently used to describe products with technical and commercial purposes on the IT domain (see Fig. 1). Those commercial data-sheets have great importance in e-commerce environments, because, based in the information contained in them, people make buying decisions. For instance an individual buyer in the Internet using a price engine (e.g. Pricegrabber[1]) looking for a laptop computer, can find hundreds of buying options with tens of features in its data-sheets.

To extract product features from data-sheets in an automatic way is the first step to assist decision making processes. Additional use scenarios for that information are data-mining, information retrieval refinement, automatic document translation, question answering and information integration from texts, among others.

Many IE approaches use given or learned extraction rules, or machine learning models trained with labeled corpora (see [7] for a survey). Generally, availability of labeled corpora is limited and the obtained models are not suitable for real world applications. Additionally, most of those approaches use context and structural homogeneity as primary evidence, which is poor in the considered data-rich documents.

On the other hand, ontology-driven approaches [2][8][10] use domain and lexical evidence to perform the extraction. Particularly, Embley [8] argues that obtained models are more resilient and usable than obtained ones from labeled corpora or extraction rules. We are in agreement with this idea.

Considering data-rich documents, they are commonly summarized and have a hierarchical presentation structure that use itemized lists without complex natural language structures. In spite of the fact that IT data-rich documents are written in a natural language (i.e., English), documents follow the rules of another syntactical structure given by composing parts and attributes of the object being described. Our information extractor is based on the idea that when an object (i.e. product) is being described, its composing parts are listed and recursively, each part is decomposed again until a concrete linearization is achieved in a specific language. For instance a laptop computer is composed of *processor*, *memory*, *hard disk*, etc.; similarly a *processor* has its own composing parts and attributes. In brief, its data-sheet description follows that structure

Ontologies are artifacts useful to describe real word entities and ideas. In computer science, ontologies are graphs whose vertices are entities and edges correspond to semantic relations between entities [9]. Ontology definition languages such as OWL[2] offer a broad set of constructors allowing complex ontology modeling. The most common relation types used in ontologies are hypernyms (i.e. *is-a*) and meronyms (i.e. *is-part-*

1 http://computers.pricegrabber.com/laptop/,
2.http://www.w3.org/2004/OWL/

of). The former allows defining entity class hierarchies and the latter describes properties of each entity.

Clearly, an ontology can describe all possible composing parts and attributes of a laptop computer. Data-sheets such as the shown in Fig.1 make reference to a subset of all possible domain ontology entities. The main problem that concerns us here is to identify all references to ontology entities in a data-sheet, and label them hierarchically according to the ontology as is shown in Fig.2.

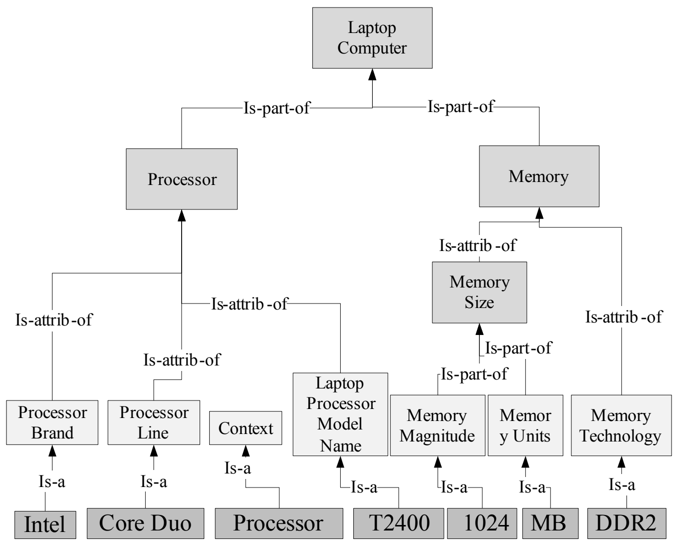


Fig. 2. Text document sequence sample hierarchically labeled with ontology entities.

From now, two sub problems arise. Firstly, entities are referred in documents in multiple and unexpected manners, making their enumeration difficult in the ontology. Secondly, there are many entities intrinsically ambiguous. For instance, *512MB* could be associated in a laptop computer with *installed main memory, maximum installable memory, video memory, operating system memory requirement, specific software hard disk space requirement, processor cache memory,* etc.

III. BACKGROUND

A. Fuzzy String Searching

The task of identify mentions of real-world entities in documents is know with different names in different communities. For instance the database community refers it as duplicate record detection, record linkage, database hardening and deduplication. Other names are used to describe the same concept, namely: entity disambiguation, entity resolution, reference reconciliation, identity uncertainly, co-reference resolution and others. This task is known with the term *fuzzy string searching* (FSS) and is based on string comparison techniques that allow errors. FSS helps to bring resilience some IE processes.

FSS is based on similarity metrics at character and token level. Among the most popular character-level similarity measures are the well known edit-distance or Levenshtein distance [12]. The edit distance between two strings is the minimum number of edit operations (i.e. insert, delete, replace) to transform one in the other. Several modifications to the original edit distance have been proposed varying cost schemas and adding edit operations (see [13] for a survey). Those metrics are useful to deal with typos, misspellings and optical character recognition errors.

Another metrics such as Jaro and Jaro-Winkler distances [14] are particularly useful to match people names. However, our special interest in character-level measures is their ability to deal with inconsistent abbreviations, un-standardized acronyms and typographical variations.

Unlike character-based similarity measures, token-level measures uses tokens (i.e. words) as comparison unit instead of characters. Some token-level algorithms are analogous to character-based algorithms adding an additional similarity metric between tokens [15]. That inter-token similarity metric can again be other character-level metric such as edit distance. Finally, both token and character level metrics have to be combined to calculate a final metric between two multi-token strings.

B. Ontology-based Semantic Relatedness

Semantic relatedness (SR) is a general level of association between two concepts [16]. For instance, in laptops domain, there is an intuitive notion that *video adapter* and *display panel* are semantically closer than *main battery* and *speakers*. The simplest SR metric used in ontologies is *path length* (see Fig. 3), which means the number of edges in the path between two entities linked with semantic relationships.

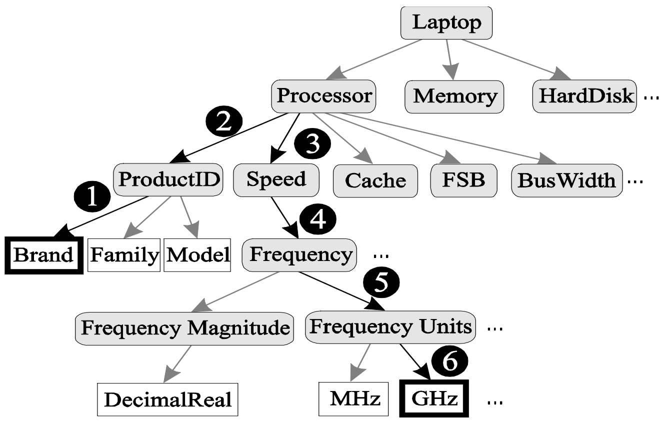


Fig. 3. Semantic Relatedness path length example.

Path-based measures used to assess semantic similarity (i.e. SR in *is-a* hierarchies) might generate misleading results because of pairs of entities closer to the root tend to be less related than entities closer to leaves. In order to correct this situation Leacock & Chodorow [17] proposed the expression (1) (*d* is the total ontology hierarchy height).

$$SIM(a,b) = -\log\left(\frac{pathLenght(a,b)}{2d}\right) \qquad (1)$$

Other approach to the same problem has been proposed by Wu & Palmer [18] using the concept of *lowest common subsumer* (LCS). LCS is the more distant concept from the root, common to two entities. For instance, in Fig. 3 *processor* is the LCS of *brand* and GHz. Wu & Palmer measure is given by (2) (the function *depth(x)* represents the number of edges from x to the root).

$$SIM(a,b) = \frac{2.depth(lcs(a,b))}{pathLenght(a,b) + 2.depth(lcs(a,b))} \quad (2)$$

C. *Word Sense Disambiguation*

Word sense disambiguation (WSD) is a Natural Language Processing task that aims to assign the correct sense to all open class (i.e. polysemes and homonyms) words in a text [3]. Firstly, in order to define the search space for the WSD problem, it is necessary a source of possible senses for each used word. That source of senses is usually a machine readable dictionary [19].

The following example has each open class word subscripted with the number of senses found in the on line version of The American Heritage Dictionary of the English Language[3].

"*Jackson found*$_{(4)}$ *good*$_{(42)}$ *medicine*$_{(9)}$ *in his summer*$_{(7)}$ *field*$_{(37)}$ *trips*$_{(23)}$ *with their unfolding*$_{(6)}$ *of new*$_{(15)}$ *horizons*$_{(8)}$"

The number of possible sense combinations in that sentence is 6,485'028,480.

SR metrics are useful to assess the semantic coherence of a specific sense combination. The final goal is to find a sense combination that maintains the better logical discourse coherence in agreement with a human judge. Due to the huge search space of the problem, techniques such as simulated annealing [20] and genetic algorithms [21] have been used to find near-optimal combinations. Other approaches aim to reduce the search space using "sliding" windows over the text sequence[22].

IV. FUZZY STRING MATCHER

Great part of entities used in the data-rich documents in question are referred using multi-token terms such as *wireless adapter* or *hard disk*. Those terms can be found in documents with many variations. For instance *serial-ATA* can be cited as *serial ATA*, *SATA*, *S-ATA*, *S.A.T.A.*, *Ser.ATA*, or *serial advanced technology attachment*. In order to match successfully *serial-ATA* with the proposed examples it is necessary in some cases to ignore periods and hyphens , and in some cases consider the *ATA* acronym as three one-character tokens. Consider the matching between *ser.-ATA* and *serial advanced technology attachment*. The pairs of tokens to be compared are: [*ser, serial*], [*A, advanced*], [*T, technology*] and [*A, attachment*]. Our fuzzy string matcher aims to deal with this and other similar cases. Figure 4 shows a block diagram with the architecture of the extraction system including the role of the fuzzy string matcher.

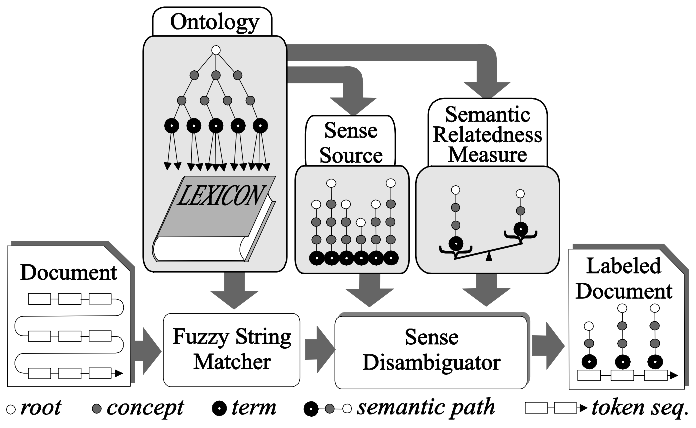


Fig. 4. Information extractor block diagram

A. *Root-Distance*

Considering the previous example (*ser.-ATA*), each pair of tokens could be compared using a character-based similarity metric such as the edit-distance [12]. We propose a modification to the edit-distance named *root-Distance*, which assigns higher cost to edit operations in heading character positions than in ending positions. Look at the following propositions:

i) EditDistance("A", "Serial") < *EditDistance*("A", "Advanced")
ii) *RootDistance*("A", "Serial") > *RootDistance*("A", "Advanced")

The first proposition is clear due to the fact that "Advanced" is the longest string and requires more edit operations to be transformed in "A". Though "A" and "Serial" are not semantically close, edit distance measure between them is closer. Consequently, it is clear that if heading character positions have higher edition costs, the distance measure can better capture a semantic relatedness as is shown in the second proposition.

Root distance edition cost (for the longest string to be compared) is 1 for the last character position and the string length for the first position for the linear case. Edit distance is a case of root distance when costs for character edition operations are 1 in all character positions. Fig. 5 shows a calculation example using the Wagner-Fischer dynamic programming technique [23] and linear edition costs.

3 http://www.thefreedictionary.com