LiTeWi: A Combined Term Extraction and Entity Linking Method for Eliciting Educational Ontologies From Textbooks

Angel Conde, Mikel Larrañaga, Ana Arruarte, and Jon A. Elorriaga
Department of Computer Languages and Computer Systems, University of the Basque Country (UPV/EHU), Paseo de Manuel Lardizabal 1, Donostia-San Sebastián, Gipuzkoa 20018, Spain. E-mail: {angel.conde, mikel.larranaga, a.arruarte, jon.elorriaga}@ehu.es

Dan Roth
Department of Computer Science and The Beckman Institute, University of Illinois at Urbana-Champaign, 2700 Prairie Meadow Dr., Urbana, IL 61822, E.E.U.U. E-mail: danr@illinois.edu

Major efforts have been conducted on ontology learning, that is, semiautomatic processes for the construction of domain ontologies from diverse sources of information. In the past few years, a research trend has focused on the construction of educational ontologies, that is, ontologies to be used for educational purposes. The identification of the terminology is crucial to build ontologies. Term extraction techniques allow the identification of the domain-related terms from electronic resources. This paper presents LiTeWi, a novel method that combines current unsupervised term extraction approaches for creating educational ontologies for technology supported learning systems from electronic textbooks. LiTeWi uses Wikipedia as an additional information source. Wikipedia contains more than 30 million articles covering the terminology of nearly every domain in 288 languages, which makes it an appropriate generic corpus for term extraction. Furthermore, given that its content is available in several languages, it promotes both domain and language independence. LiTeWi is aimed at being used by teachers, who usually develop their didactic material from textbooks. To evaluate its performance, LiTeWi was tuned up using a textbook on object oriented programming and then tested with two textbooks of different domains—astronomy and molecular biology.

Received September 23, 2013; revised July 18, 2014; accepted July 20, 2014

 • Published online 7 April 2015 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/asi.23398

Introduction

Ontological engineering models anything computer science is interested in (Mizoguchi & Bourdeau, 2000). The interpretability of the ontologies at this level enables computers to answer questions about the model described by the ontologies. In particular, in recent years a great effort has been put into the development of educational ontologies, that is, ontologies that encapsulate the domain knowledge of a technology supported learning system and the related pedagogical knowledge (Fok & Ip, 2007). These ontologies describe the information about the topics to be mastered along with the pedagogical knowledge (e.g., pedagogical relationships among the topics) required by technology supported learning systems. The ontology for C-programming (Sosnovsky & Gavrilova, 2006), the ontology for Java programming (Ganapathi, Lourdusamy, & Rajaram, 2011), or ACM's Computer Ontology (Cassel, Davies, LeBlanc, Snyder, & Topi, 2008) are examples of developed educational ontologies. In a time when technology supported learning systems are being used more and more, providing aid tools for building such systems and, especially, tools for developing the learning content for those systems is essential.

Ontology learning (Buitelaar, Cimiano, & Magnini, 2005a; Maedche, 2002) refers to the application of a set of methods and techniques to enable the (semi-)automatic population of ontologies or the construction of ontologies from scratch from diverse information sources. Although ontology learning has mainly focused on the development of general-purpose ontologies, it can also remarkably contribute to lighten the workload in the construction of educational ontologies.

The work presented in paper focuses on the term extraction task and presents LiTeWi, a domain-independent tool for the elicitation of terms for educational ontologies from electronic textbooks that uses Wikipedia as an additional source of information. In this work, an evaluation of the tool for the English language is presented. LiTeWi was tested on the *Principles of Object Oriented Programming* textbook (Wong & Nguyen, 2010) to determine its optimum setup, and then two experiments with books from different domains are conducted to evaluate it: *Introduction to Astronomy* (Morison, 2008) and *Introduction to Molecular Biology* (Raineri, 2010).

The paper is structured as follows: First, ontology learning is briefly introduced. Next, the most broadly used term extraction techniques are presented. The third section describes how Wikipedia can be used as a source of information for ontology learning and for term extraction in particular. Next the proposed combined term extraction method is described. Then the experiment conducted to evaluate the performance of LiTeWi on English is presented. Finally, the conclusions and future work are described.

Ontology Learning

The term ontology was adopted from philosophy, where it is defined as the "theory of existence." There are many definitions for ontologies in the area of computer science. Neches et al. (1991, p. 40) proposed the following definition: "an ontology defines the basic terms and relations comprising the vocabulary of the topic area as well as the rules for combining terms and relations to define extensions to the vocabulary." However, Gruber (1991, p. 2) has provided the most popular definition of ontologies, which states that "an ontology is an explicit explanation of a conceptualization." This definition was slightly enhanced by Borst (1997, p. 12), who referred to ontologies as "formal specifications of a shared conceptualization."

According to Studer, Benjamins, and Fensel (1998, p. 25), "conceptualization refers to an abstract model of some phenomenon in the world by having identified the relevant concepts of that phenomenon. Explicit means that the type of concepts used, and the constraints on their use are explicitly defined. Formal refers to the fact that the ontology should be machine-readable. Shared reflects the notion that an ontology captures consensual knowledge, that is, it is not private to some individual, but accepted by a group." Ontologies capture and describe domain knowledge in a generic way and provide a commonly agreed understanding of a domain, which may be reused and shared across applications and groups (Chandrasekaran, Josephson, & Benjamins, 1999). They arose as a means to obtain shareable and reusable knowledge bases (Gruber, 1991) and are the core of the semantic web (Berners-Lee & Fischetti, 1999; Berners-Lee, Hendler, & Lassila, 2001).

Ontologies formalize the intentional aspects of a domain, whereas the extensional part is provided by a knowledge base that contains assertions about instances of concepts and relations as defined by the ontology (Buitelaar et al., 2005a, 2005b). Ontology learning entails the processes for the semiautomatic development of ontologies, whereas the process of defining and instantiating a knowledge base is referred to as knowledge markup or ontology population.

Ontology learning is inherently multidisciplinary and combines machine learning and natural language processing (NLP) techniques to elicit the components of the ontology from diverse sources of information, such as knowledge bases or text corpora. Ontology learning comprises, among other tasks, the identification of the domain concepts or topics and the definition of the semantic relationships among them (Buitelaar et al., 2005b; Maedche, 2002; Maedche & Staab, 2000; Pazienza & Stellato, 2012).

Ontology learning can also contribute to the development of educational ontologies, which describe information about the topics to be mastered along with the pedagogical knowledge (e.g., pedagogical relationships among the topics, difficulty level of the topics, or the relevance of the topics) required by technology supported learning systems. Gavrilova and colleagues followed a five-step procedure to build an educational ontology for C-programming (Gavrilova, Farzan, & Brusilovsky, 2005; Sosnovsky & Gavrilova, 2006):

- ***Glossary development:*** selecting all the essential topics in the domain.
- ***Laddering:*** structuring the topics of the ontology defining taxonomies, parthood relationships, and so on.
- ***Disintegration:*** breaking high-level concepts—big concepts—into a set of detailed ones—smaller concepts—where needed, using a top-down strategy.
- ***Categorization:*** grouping similar concepts and creating meta-concepts to generalize the groups via bottom-up structuring strategy.
- ***Refinement:*** updating the visual structure by excluding the excessiveness, synonymy, and contradictions.

The same approach was followed to build the Java Learning Object Ontology (JLOO) by Ganapathi et al. (2011), whereas Fok and Ip (2007) took a different approach, reusing existing domain ontologies and adapting them to build the Personalized education ontology (PEOnto).

The work presented here focuses on the term extraction process for Educational Ontologies. Therefore, relevant aspects on term extraction, along with some term extraction techniques, are outlined below.

Term Extraction

Term extraction identifies the most relevant terms in the analyzed source of information. This is one of the essential

DOI: 10.1002/asi

TABLE 1. Examples of syntactic patterns for term extraction.

Syntactic pattern	Examples
Noun$^+$ *Noun*	Computer science, solar system, hubble space telescope
(*Adj*|*Noun*)$^+$ *Noun*$^+$	Extra-solar planets, elliptical galaxies, giant tidal waves
((*Noun Prep*?) (*Adj*|*Noun*)***Noun*	Coloboma of retina, scotomas in low vision
((*Adj*|*Noun*)$^+$| (*Adj*|*Noun*)*(*Noun Prep*)?) (*Adj*|*Noun*)*) *Noun*	Acute exacerbation of chronic bronchitis

Note. ? represents that the element is optional, + that the element should appear at least once, and * that the element can appear 0 or more times.

tasks for ontology learning. Term extraction is widely used in text mining and information retrieval, for example, for indexing scientific literature according to keyphrases and main topics. Term extraction techniques are diverse, ranging from linguistic methods, which rely on the detection of the specific term patterns, to statistical methods that determine the *termhood* of a candidate term. Termhood refers to the degree that a linguistic unit is related to or represents domain-specific concepts (Kageura & Umino, 1996). Hybrid approaches combine both kinds of techniques for extracting the terms. In the following subsections those approaches are described.

Linguistic Approaches

The linguistic approaches for term extraction rely on the syntactic properties of the terms for their identification. These kinds of techniques work under the assumption that terms usually present characteristic syntactic structures (Benveniste, 1966; Bourigault, 1996). Moreover, Daille, Habert, Jacquemin, and Royauté (1996) confirmed in an empirical study that most terms appear in the form of short noun phrases.

Linguistic term extraction approaches apply the following procedure:

1. Perform a shallow linguistic analysis to enrich the analyzed text with *part-of-speech* information (e.g., nouns, verbs and adjectives). To fulfill such a task, a part-of-speech tagger such as the Stanford Log-Linear Part-Of-Speech Tagger (Toutanova, Klein, Manning, & Singer, 2003) or FreeLing (Padró & Stanilovsky, 2012) is required.
2. Identify and extract candidate terms through admissible surface forms or shallow parsing grammars (Buitelaar et al., 2005a). Table 1 provides some examples of the syntactic patterns that are frequently used for term extraction. Some works, for example, Daille et al. (1996), also deal with the identification and grouping of meaning-preserving term variants. For instance, the terms *mission of spacecraft* and *spacecraft mission* refer to the same topic. Therefore, both might be identified as meaning-preserving variants of the same term.
3. Apply linguistic filters, for example, a list of words (*stop-words*) that will be filtered out to refine the terminology.

Statistical Approaches

Statistical measures provide a means to distinguish among true and false terms given a set of candidate terms. These statistical measures determine whether a given candidate term might be a true term and how related it might be to the domain. Statistical measures can be classified in two groups considering their final goal: measures determining the *unithood*, that is, the degree of strength or stability of syntagmatic combinations and collocations to form a linguistic unit, and measures for the *termhood*, that is, the degree to which a linguistic unit is related to the domain (Pazienza, Pennacchiotti, & Zanzotto, 2005). Unithood measures, such as used previously (Dunning, 1993; Fano, 1961; Salton, Yang, & Yu, 1975), allow the recognition of complex linguistic units (called collocations) composed of words with a strong association, such as “day after” or “spacecraft mission.” On the other hand, termhood measures determine the relatedness of the candidate terms with the domain.

For example, the term frequency-inverse document frequency (TF-IDF) (Salton & Buckley, 1988) method combines the appearances of a term in a document with frequencies of the documents in which the term is found in a reference corpus to determine its termhood. On one hand, the term frequency measures the relevance of the term. The more frequently a term appears in a document, the more relevant it is. On the other hand, the inverse document frequency measures the specificity of the term. The more documents the term appears in, the less specific the term is.

Other methods, such as latent semantic analysis (LSA) (Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990), use more advanced statistical measures. LSA is a mathematical method for modeling the meaning of words and passages by analyzing representative text corpora. The Dirichlet Process Segmentation, which is a Bayesian method for nonparametric modeling, has also been recently applied for term extraction in Koilada, Newman, Lau, and Baldwin (2012).

Hybrid Approaches

The syntactic patterns used for the identification of the terminology are language-dependent. Furthermore, they might recognize candidate terms that are not representative of the domain being described. Therefore, means to determine the termhood should also be used. Statistical approaches such as those described above are valid to address this objective. Hybrid approaches combine linguistic and statistical techniques for term extraction. They rely on syntactic patterns for detecting candidate terms and use statistical measures to determine their domain-relatedness and relevance.

DOI: 10.1002/asi

In Earl (1970), one of the first hybrid systems, noun-phrases are extracted first as candidate terms and then ranked according to the frequency of their noun elements. In Daille (1994) the candidates terms are obtained using syntactic patterns and filtered using different statistical measures. Another similar approach is described in Justeson and Katz (1995), where expressions are used to extract the candidates, which in turn are ranked by frequency.

In Enguehard and Pantera (1995), a more complex approach is described where, in a first step, the terms are extracted according to their frequency. Then, in a second step, new terms are derived through linguistic heuristics applied to the terms retrieved in the first step.

A step further is to improve the linguistic analysis using semantic and contextual information. In Maynard and Ananiadou (1999), semantic information derived from thesauri, linguistic hints and statistical evidence are mixed for ranking candidate terms. For example, the NC-value, a complex heuristic measure based on C-value, adds context factor information considering the semantic, syntactic, and statistical properties of the context where the terms appear. This use of context information is also common in other approaches such as Velardi, Missikoff, and Basili (2001), where a shallow syntactic parser is used to select candidate term patterns and, then, two measures—domain relevance and domain consensus—are used to rank the candidate terms, that is, determine their termhood. The domain relevance measures the specificity of the candidate term with respect to the target domain, that is, whether the term is exclusive of the domain or is broadly used in other knowledge areas, whereas the domain consensus refers to the homogeneous use of the candidate term in the domain. To compute both measures, collections of documents on each covered domain must be provided.

KEA (Frank, Paynter, Witten, Gutwin, & Nevill-Manning, 1999; Medelyan & Witten, 2006) also relies on a hybrid process for the automatic extraction of keyphrases from documents. It first identifies a set of candidate terms (n-grams entailed by 1 to 3 words) and uses a machine learning algorithm to determine which candidates are good keyphrases. The machine learning algorithm uses four features, the TF-IDF score, the number of words of the candidate keyphrase, the first occurrence of the candidate (computed as the percentage of the document preceding the first appearance of the term in the document), and the number of phrases the candidate set is related to.

GenEx (Turney, 2000) approaches keyphrase extraction from text as a supervised learning task. GenEx has two componentes, Genitor (Whitley, 1989), which relies on a genetic algorithm, and Extractor (Turney, 1999), which implements the keyphrase extraction algorithm. To fulfill its purpose, first the Genitor model has to be trained with procedural domain knowledge. Then, Extractor is used with those learned parameters for keyphrase elicitation. The experimental results showed that the custom-designed algorithm designed for this task can generate better keyphrases than a general-purpose algorithm.

Hulth (2003) proposed a supervised machine learning approach in which linguistic knowledge (e.g., syntactic features) are used along with statistics (such as term frequency) for automatic keyword extraction. Hulth claimed that extracting noun phrases instead of n-grams increased the precision and including part-of-speech tags as features dramatically improved the keyword extraction performance.

HaCohen-Kerner, Gross, and Masa (2005) present an approach for eliciting keyphrases from scientific articles written in English. This paper combines different baseline methods similar to those used in summarization—for example, Kupiec, Pedersen, and Chen (1995)—and then applies common supervised machine learning methods in order to achieve the best combination of those baseline methods. In all methods, words and terms that have a grammatical role for the language are excluded from the key words list according to a ready-made stop list.

Wikipedia as a Source of Information

Wikipedia is a collaborative online encyclopedia containing more than 30 million articles in 287 languages (as of February 2014).[1] It has become one of the most popular reference works on the internet. Wikipedia has a vast, constantly evolving tapestry of richly interlinked textual information (Milne & Witten, 2013). Therefore, it is a really powerful resource for NLP research or data mining, as it provides an ever-growing source of manually defined concepts and relations.

The article is the basic element of Wikipedia. An article, in Wikipedia, is identified by a unique name and contains information about a concept, an event, or a relevant personality. Besides the content, the articles might also contain internal links to other articles and also external links. Articles are classified according to categories. A Wikipedia category provides a way to group related articles and add semantic knowledge to articles. A category has a unique name, and may have parent categories, child categories, and articles belonging to the category.

Experts who want to use Wikipedia as a source of machine-readable knowledge have three options to choose from. They can rely on third-party secondary structures, such as Freebase (Bollacker, Evans, Paritosh, Sturge, & Taylor, 2008), Yago2 (Hoffart et al., 2011), and DBPedia (Bizer et al., 2009). A second option would be to start from scratch and build their own algorithms for extracting the Wikipedia knowledge. Finally, a third option would be to develop and share the algorithms, rather than secondary resources.

The first approach is the easiest one, as the information obtained using the structures is already in a machine-readable format. Nevertheless, new innovations and mining

[1]http://en.wikipedia.org/wiki/Wikipedia

DOI: 10.1002/asi

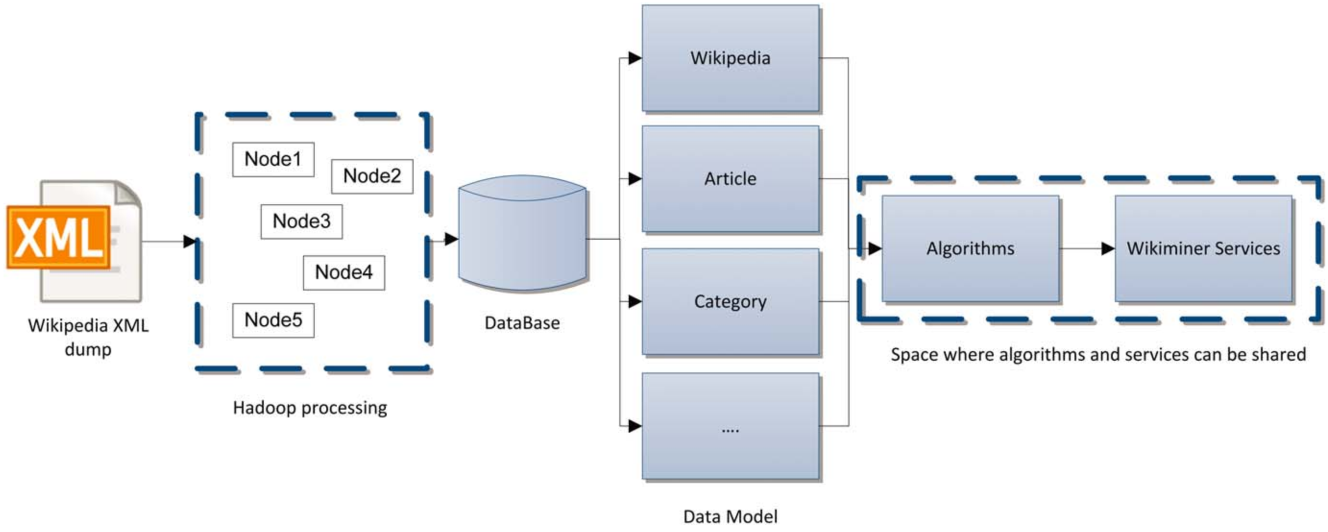

FIG. 1. General architecture of Wikiminer. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

techniques are introduced regularly, potentially rendering obsolete prebuilt resources. Moreover, if these structures are not maintained periodically, such resources forego one of the greatest strengths of Wikipedia: its propensity to grow rapidly and adapt itself to the world's changes. The second option, working directly from the raw source, allows researchers to innovate and find new ways to mine knowledge from Wikipedia. The content of Wikipedia is released in the form of large XML dumps with cryptic markup that requires substantial efforts to build usable machine-readable data. The third option, which involves using a toolkit that helps to process the contents of Wikipedia to form machine-readable data, also provides an easy way to apply and share different techniques for gathering the knowledge contained in Wikipedia. This allows researchers to focus on the algorithms for knowledge extraction instead of dealing with the Wikipedia dumps. The work presented throughout this paper falls into the second category, as it relies on Wikiminer for mining the knowledge from Wikipedia.

Wikiminer

Working with Wikipedia involves processing the database dumps to form machine-readable data. Wikiminer (Milne & Witten, 2013) was developed to fulfill such a purpose. Wikiminer is a platform where mining techniques take advantage of the Weka (Hall et al., 2009) machine-learning workbench and the power of distributed computing using Hadoop (White, 2010).

Wikiminer processes Wikipedia dumps to build a database with information about articles, categories, links, labels, and so on. As processing the Wikipedia dumps requires high computer resources, Hadoop can be used to take advantage of distributed computing. Wikiminer also provides a set of algorithms that allow data searches and comparisons to be performed. The functionality of Wikiminer is achievable through a set of web services. In Figure 1, the general architecture of Wikiminer is shown.

LiTeWi: A Term Extractor for Educational Ontologies That Uses Wikipedia

The term extraction techniques described previously are aimed at the identification of the most relevant terms in a document and have been broadly used for ontology learning. The work presented here facilitates the development of educational ontologies for technology supported learning systems, in particular the extraction of the topics to be mastered by the students. LiTeWi,[2] a term extractor that uses Wikipedia, combines diverse term extraction methods to fulfill such a goal. LiTeWi was developed for teachers who want to develop an educational ontology for technology supported learning systems. Those teachers usually rely on one or more textbooks to determine the topics to be mastered by learners and extract the didactic resources required for such a learning process. LiTeWi is intended to be usable on documents of any domain. Therefore, any domain-dependent technique was discarded. In addition, LiTeWi was designed to be easily extended to support new languages. Nevertheless, the experiments described throughout this paper were conducted on documents written in the English language. To cope with the term extraction in a new language, some minor steps must be taken: First, a

[2]A demo featuring some of the abilities of LiTeWi can be accessed at http://galan.ehu.es/lidom/

DOI: 10.1002/asi

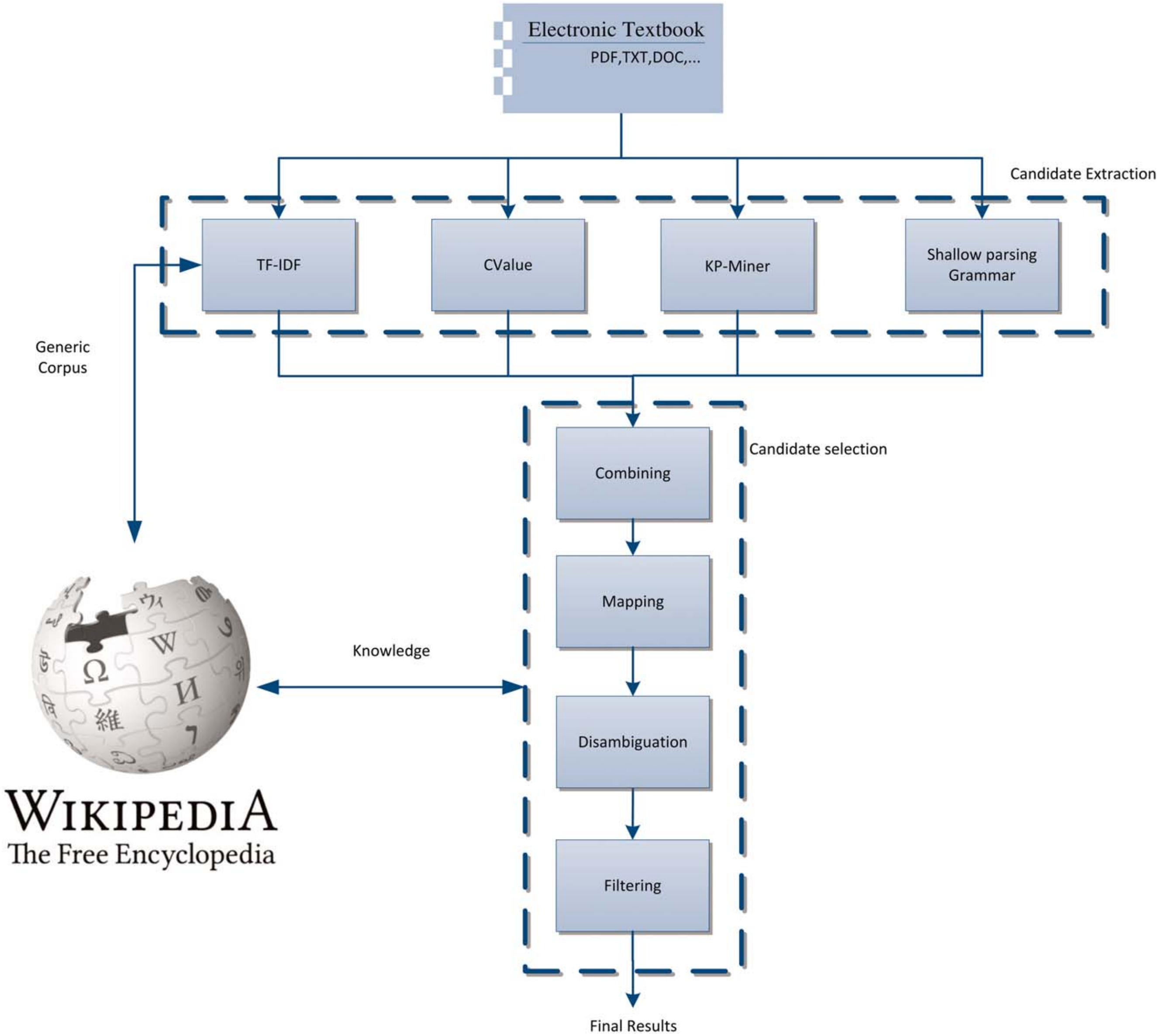

FIG. 2. Overview of LiTeWi. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

part-of-speech analyzer has to be provided for that language in case the currently used parsers do not recognize it. In addition, some resources (e.g., patterns for the term elicitation) must be defined for the new language. Wikiminer also has to be trained for the new language to compute relatedness measures.

Figure 2 illustrates the combined term extraction approach carried out by LiTeWi, which entails the identification of term candidates using TF-IDF, CValue, KP-Miner, and Shallow Parsing, and the combination and the refinement of the results to obtain the final set of terms. To tune it up, LiTeWi was first tested on the *Principles of Object Oriented Programming* textbook (Wong & Nguyen, 2010), which consists of 67 pages and more than 30,000 words. For this text, the document index was used as the reference list of terms to be extracted. The performance of LiTeWi was measured in terms of *precision*, that is, the proportion of extracted terms that are in the reference list, and *recall*, that is, the percentage of the terms in the reference list extracted by the system. In addition, F1-score, the harmonic mean of precision and recall, was used. Next, each step of the process and its setup are described in more detail.

Candidate Extraction

In the approach here described, the extraction of the candidate terms entails running several term extraction

DOI: 10.1002/asi

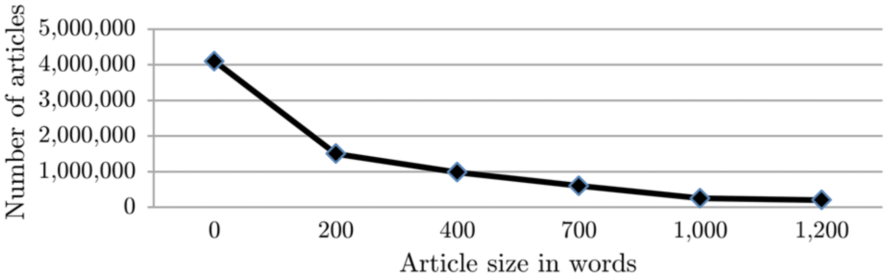

FIG. 3. Number of Wikipedia articles versus article size in words.

techniques in parallel aimed at obtaining the pursued terms. In a subsequent process the unwanted terms will be filtered from the candidate list. The extraction of candidate terms entails running the algorithms with low thresholds where possible in order to identify as many terms as possible and to prevent discarding "real" terms. In the following sections, the used term extraction techniques are described together with the experiments conducted to empirically determine their initial configuration.

TF-IDF. The TF-IDF (Salton & Buckley, 1988) technique for term extraction allows the identification of terms in a document. Besides term frequency, this technique also considers the relevance of the terms in the document. TF-IDF requires a corpus to distinguish common terms from the relevant ones, those which appear in the analyzed document but are not frequently used in any other document. As this work aims at being domain-independent, Wikipedia was chosen as the generic corpus to be used. Wikipedia has more than 4,400,000 English articles (as of February 2014) covering the terminology of a huge number of domains. Therefore, it can be used as a generic corpus for many domains.

For processing Wikipedia, first an XML raw dump on Wikipedia needs to be downloaded and then the content and the titles of the articles are extracted using the *Wikimedia Extractor*.[3] After extracting all the articles along with their content, the term frequencies for each article in Wikipedia is calculated. This process is time-consuming, due to the huge amount of data to analyze. To accelerate the process, an *Apache Lucene*[4] index, a high-performance search engine, was used to process, store, and elicit all the required information for processing the algorithm.

As can be observed in Figure 3, Wikipedia entails articles of different granularity or length. Given that short articles might refer to a limited set of topics, they might considerably affect the performance of the TF-IDF method. Therefore, the TF-IDF extractor was first tested on the *Principles of Object Oriented Programming* to determine whether small articles, the use of stopwords (words that are common and, therefore, in term extraction, are usually filtered out prior to the process), or the stemming (reducing inflected words to their stem) might affect its performance. Concerning the size of the article, the best results were obtained when filtering articles with a length of fewer than 700 words.

TABLE 2. Top terms for English in Wikipedia.

Term	Term frequency	Document frequency
He	8,241,073	1,222,034
From	6,994,944	1,941,597
His	6,731,768	1,183,499
Were	4,407,804	1,081,245
Which	4,150,383	1,416,728
Also	3,503,109	1,414,638
Has	3,417,285	1,289,822

The use of stopwords did not affect the performance, as those words tend to appear in almost every document and, therefore, have low scores (see Table 2). For instance, words such as "he," "from," or "his" usually have a high term frequency but are used in most documents; therefore, they are not considered representative and obtain a low TF-IDF score, avoiding the need to build and test an appropriate stopword list. However, the default stopword list for English used by *Apache Lucene*, which entails 33 words (see Appendix A), is applied to reduce the size of the Lucene index.

Stemming was discarded, as it negatively affected the performance of the method. Given that some important word variants are converted to the stemmed word, they are lost. For example, "Abstraction," which is a relevant topic in programming, was converted to "Abstract" using stemming. As "Abstract" is a common word in the Wikipedia corpus, it was discarded because of its score. In addition,

[3]http://medialab.di.unipi.it/wiki/Wikipedia_Extractor

[4]http://lucene.apache.org/

DOI: 10.1002/asi

TABLE 3. CValue results using different filters.

Filter	Precision(%)	Recall(%)
Noun+	10.48	25.17
$(Adj|Noun)^{+}(Adj|Noun)^{*}$	7.67	30.39
$(Noun\ Prep)^{?}(Adj|Noun)^{*}\ Noun$	6.84	33.5

Note. ? represents that the element is optional, + that the element should appear at least once, and * that the element can appear 0 or more times.

a filter that removes the possessive genitive ("s") is applied.

CValue. The CValue (Frantzi, Ananiadou, & Mima, 2000) is a domain-independent technique for extracting nested terms. It relies on statistical (frequency) and linguistic information and takes into account the occurrence of term candidates as a part of longer terms. In the work described here, a Java version of the algorithm, which uses the Illinois Part of Speech Tagger,[5] was developed and employed.

CValue requires a linguistic filter to choose the terms to be weighted from the processed texts. Different linguistic filters lead to different results, affecting the precision and recall of the output list. Linguistic filters can be classified into two types (Mima & Ananiadou, 2000):

- *Close filters*, which are strict about the text fragments they permit. For example, Dagan and Church (1994) used a filter that only allows sequences of nouns (e.g., *Noun*+).
- *Open filters*, which are more flexible and accept several kinds of strings. This kind of filter may have a negative effect on precision but will be positive in terms of recall. Justeson and Katz (1995) used open filters such as $(Adj|Noun)^{+}(Adj|Noun)^{*}$ and $(Noun|Prep)^{?}(Adj|Noun)^{*})$ *Noun* for term extraction.

To determine which filter should be used, different filters were tested (see Table 3) on the *Principles of Object Oriented Programming* textbook. Given that, at this stage of the process, the goal is to maximize the recall, the $(Noun|Prep)^{?}(Adj|Noun)^{*}Noun$ open filter, which yields the best results, was chosen. Some examples of the term candidates identified by the CValue method are shown in Table 4.

KP-Miner. KP-Miner (El-Beltagy & Rafea, 2009), which stands for keyphrase miner, is a system for the extraction of Arabic and English keyphrases from both text or html documents. Unlike other existing keyphrase extraction systems, KP-Miner does not require any prior training to fulfill its task. El-Beltagy and Rafea (2009) reported that KP-Miner produced comparable results for English for both KEA (Frank et al., 1999) and Extractor (Turney, 2000), two of the most broadly used keyphrase extraction systems. Examples of extracted terms using this system are illustrated in Table 5.

TABLE 4. Example of extracted terms with their CValue.

Term	CValue
Design pattern	28.41
List structure	23.5
Variant behaviour	17
Invariant behaviour	16
Concrete subclass	15.75
Gui component	15
Concrete implementation	13

TABLE 5. Example of terms extracted by KP-Miner.

"object," "java," "drjava," "abstraction," "invariant,""computer," "variant"

Shallow Parsing. The final goal of the approach presented throughout this paper is the extraction of terms for educational ontologies, learning topics, for which educational material can be found in the analyzed document. In a previous work, Conde, Larrañaga, Calvo, Arruarte, and Elorriaga (2012) defined the didactic resource grammar, a grammar that represents the most common syntactic structures used in didactic resources (e.g., definitions, examples, or theorems). The didactic resource grammar, which is implemented using the constraint grammar formalism (Karlsson, Voutilainen, Heikkila, & Anttila, 1995), was developed to enable the automatic extraction of learning objects from electronic documents. An adapted version of the didactic resource grammar is used by the Shallow Parsing method to identify fragments of the document that might contain didactic resources. The grammar consists of 250 rules: 145 rules that try to extract terms from definitions, 59 rules that try to extract terms from examples, and 46 rules to extract terms from principle statements. The Shallow Parsing is carried out in two main steps: First, the text fragments containing potential didactic resources are filtered using the grammar. Then, noun phrases are extracted from those text fragments (see Algorithm 1) using the Illinois Chunker.[6] Those noun phrases entail the candidate terms.

[5]http://cogcomp.cs.illinois.edu/page/software_view/POS

[6]http://cogcomp.cs.illinois.edu/page/software_view/Chunker

DOI: 10.1002/asi

ALGORITHM 1. Shallow Parsing algorithm for term extraction.

```
Input: Tokenized (POS-tags) Sentence List β, A grammar λ to apply
Output: A list of terms δ extracted applying the grammar to the sentences.

termList = new List;
for each tokenized sentence α in sentence list β do
    candidateSentence = applyGrammarToSentence(α, λ);
    if (candidateSentence.hasRuleMapped) then
        nounPhrasesSentence = extractN P(candidateSentence);
        term = extractTermRule (nounPhrasesSentence,
        candidateSentence.ruleApplied);termList.addTerm(term);
    end
end
return termList;
```

TABLE 6. Example of constraint grammar rules.

Pattern	Example
Concept + (NOT Prep.) + (is|are) + [Det.]	**Java** *is a* programming language.
	Objects are the primary units used to create abstract models.
Concept + (refer|refers) + (to)	**Abstraction** *refers to* object oriented programming.
(This|That) + (is|are) + (called) + **Concept**	*That is called* the **Green House Effect**.
	This is called an **Array List**.
(what) + (is|are) + [Det.] + **Concept**	*what are those* **astronomical observatories**.
(is|are) [adverb] + (called|known as|defined as) + **Concept**	This phenomenon *is known as* **dynamic reclassification**.
	We use what *is called* the **assignment operation**.
Concept + (is|are) + (used) + [Det.] + .	A **list** is used to store objects. Abstract notion of a container structure.
	A **stack** is used to model systems that exhibit LIFO insert/removal behavior.

Table 6 shows some examples of terms, highlighted in bold, that were identified in a sentence selected by the grammar along with the rules that allowed their elicitation.

Candidate Selection

After running all the term extraction methods mentioned previously, the results are combined as follows. First, all the results are combined in a large term list. Then each term is mapped to one or more Wikipedia articles. Next, the terms with more than one sense/meaning, that is, more than one mapped article, are disambiguated. Finally, those terms not related to the desired domain are filtered. These steps are depicted in more detail in the following subsections.

Combining term candidates. After all the techniques for term extraction have finished, the returned results (all the terms obtained by the techniques described previously) are combined in a large list of terms. Then the elicited terms are normalized to ignore case and number differences.

After all the terms are combined, the stopword list shown in Appendix A is applied to filter out unwanted terms. This stopword list was constructed combining the proposals of Salton (1971) and Fox (1990).

Mapping terms to Wikipedia articles. In this step, the terms obtained in the previous step are related to their corresponding Wikipedia articles. This entails searching in Wikipedia to determine whether each selected term can be related to one or more Wikipedia articles, each one representing a possible sense/meaning of the term.

In Wikipedia, each article, besides the title, has a set of labels that represent different variants for the title name of the article. To map the candidate terms with the Wikipedia articles, Wikiminer is used. To fulfill this task, Wikiminer uses both the article title and the set of labels. Some of this set of labels with their respective Wikipedia articles can be seen in Table 7. Wikiminer requires some prior configuration to carry out its work. Some tests were conducted to determine the best method for mapping terms considering both recall and precision. Three different configurations were tested. The first one ignores case differences. The second one uses the Porter Stemmer (Porter, 1997). Besides ignoring case differences, it also removes possessive genitive cases. Finally, the third one ignores case differences, removes the possessive genitive, and uses the Pling Stemmer from Yago2s Java Tools[7] in order to remove plural cases. In the setup experiments, the first achieved the best performance in terms of precision (29.34%). Surprisingly, its recall (55.26%) was quite satisfactory. The second one achieved 3.26% recall with 10.1% precision. The method using Pling Stemmer performed best, as it mapped 62.5% of the candidate terms with 25.35% precision. Therefore, this last alternative was selected.

After this process, the terms are related to one or more Wikipedia articles. Those which are not related to any article will be deleted. In the tests carried out, the size of the list is

[7]http://www.mpi-inf.mpg.de/yago-naga/javatools/

TABLE 7. Example of labels for different Wikipedia articles.

Wikipedia title	ID	Labels
Java (island)	69336	Java, Javanese, Java Island, Island of Java, Jawa Dipa . . .
Java (programming language)	15881	Java Programming Language, Java, JAVA, java . . .
Earth	9228	Earth, earth, earth's, the Earth, global, planet Earth . . .
Solar System	26903	solar system, Sol system, Sol, star systems, the solar system . . .
Search for extraterrestrial intelligence	28153	SETI, S.E.T.I., Search for Extra-Terrestrial Intelligence . . .
List (computing)	208382	list, lists, Lists, list type, vector, sequence containers . . .

TABLE 8. Example of extracted terms with their possible meanings.

Term	Meanings
Java	Island, programming language, software platform
GUI	Graphical user interface, type of bowl-shaped Chinese vessel
Container	Intermodal container (transport), abstract data type
Light years	Light-year, light years (Kylie Minogue album)
Einstein	Albert Einstein, einstein (crater)
Keyboard	Keyboard instrument, electronic keyboard, computer keyboard

reduced by half. Furthermore, articles with the same sense/meaning or with the same list of senses/meanings are combined. To speed up the mapping step, a database that contains the normalized article names and labels was built to use with Wikiminer. This database is used to compare the title names and labels with the candidate terms.

Disambiguating the terms. In the previous step, each term was related to one or more Wikipedia articles, each one representing a different sense/meaning. Therefore, a disambiguation process of the terms with more than one possible meaning is necessary. In the tests, a quarter of the terms needed disambiguation. Some examples of terms with their associated senses are shown in Table 8.

A method employing the Milne and Witten Global disambiguation (Milne & Witten, 2008) approach is used to fullfill this task, to which end the Wikiminer Compare Service is used. This service provides a way for disambiguating term pairs using a classifier that takes as features:

- The data provided by Wikipedia. Wikipedia provides statistics about how an article label is associated to a sense/meaning. For example, 55% of "Java" labels refer to the programming language, whereas 15% of them refer to the Indonesian island. These statistics yield three features for the classifier: the average, maximum, and minimum prior probabilities of the two concepts.
- The semantic relatedness between the concepts. The relatedness score can be computed using the links of the articles as features. Milne and Witten (2013, p. 228) claim that "Wikipedia articles reference each other extensively, and at first glance the links between them appear to be promising semantic relations. Unfortunately, the article also contains links to many irrelevant concepts (e.g. terms not related to the domain of the analyzed book). Therefore, an individual link between two Wikipedia articles cannot be trusted." There are different possibilities for computing the relatedness measure; for instance, using the article in-links (those links that refer to the article) and the article out-links (those links that are inside the article and refer to other articles). Both measures use different sets of links. The normalized distance measure is based on an approach that looks for documents that mention the terms of interest, and was adapted to use the links made to articles. The vector similarity measure is based on an approach that looks for terms mentioned within two documents of interest, and was adapted to use the links contained within articles. However, there is no reason why each measure should not be applied to the other link direction. Thus, each of the measures described above yields two features, one for in-links and the other for out-links. Finally, another measure taking into account the link counts for each article could be used. Different configurations were tested. As pointed out by Milne and Witten (2013), the more features used, the higher the performance is. Therefore, the measure that combined the links-in, links-out, and link-counts was selected for computing the relatedness score.

Each element of the candidate term list is disambiguated, following the approach summarized in Algorithm 2, to obtain its most plausible sense. The Wikiminer Comparing Service is used to fulfill such a task, to which end it requires a list of "gold terms" (terms with a unique meaning and that are relevant to the domain). But how does one choose terms that are relevant to the domain and which have a unique meaning? Longer terms might be expected to be related to fewer articles. An analysis was conducted on the test results to determine whether or not the hypothesis was correct. As can be observed in Figure 4, the number of senses/meanings decreases as the term size in n-grams (number of words composing the term) increases.

Therefore, the more n-grams a term has, the more specific it is. Nevertheless, domain-relevant terms are required. Hence, the monosemic terms with highest CValue score are chosen for the "gold terms" list. This decision was made after making some tests with the CValue and observing that the top-scored terms are almost always relevant in the domain.

Once the disambiguation finishes, an additional process is carried out to identify and combine terms that have been mapped to the same Wikipedia article.

Filtering domain-related terms. In the final step, those terms that are not related to the domain are deleted. For this task the "gold term" list built in the disambiguation step is used. This task attempts to relate each elicited term with the terms in the "gold term" list, to which end the Wikiminer Comparing Service was employed. Again, this service was configured to rely on the in-links, out-links, and link-counts to determine the domain-relatedness.

ALGORITHM 2. Disambiguation algorithm given a gold terms list.

```
Input: Gold term list β, term λ to relate to the domain
Output: If the term_λ is related to the domain returns the term_λ, if not returns null
for each term α in the gold term list β do
     sense, probability = wikiminerCompare(term_α, term_λ);
     term.addProbableSense(sense, probability);
end
List probableSense_δ = term_α.getSenseList();
for each probableSense α in the probable sense list δ do
     average = calculateAverage(probableSense_α);
end
return term_α.maxAugSense();
```

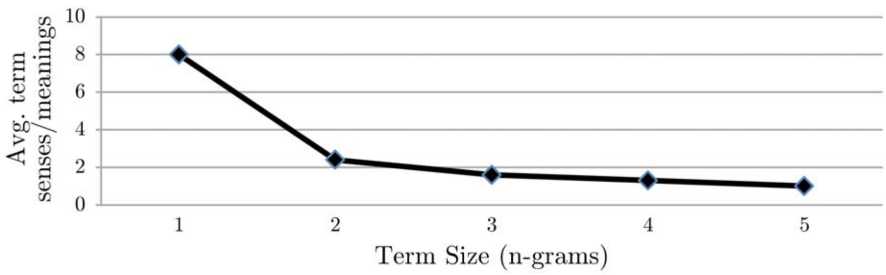

FIG. 4. Term size (n-grams) versus average number of senses/meanings.

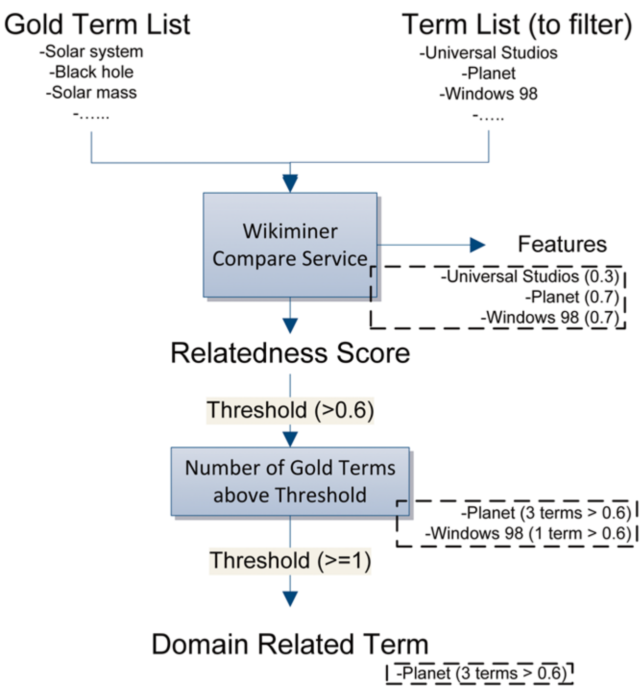

FIG. 5. Overview of filtering algorithm. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

The candidate term list is filtered following the process described in Figure 5. First, the Wikiminer Comparing Services computes each term domain-relatedness. Those topics whose score is below the threshold are dropped. Finally, those terms that are related with at least the minimum amount of "gold terms" are selected. Some experiments were conducted to determine the optimal thresholds and the number of "gold terms" that the candidates should be related to (at least one term, two terms or three terms).

As can be observed in Figure 6, the best results were obtained when requiring the candidate term to be related with at least one of the "gold term" list entries, with a relatedness score over 0.6. Therefore, this is the setup that achieves the best compromise between recall and precision.

From the filtering step, a list of terms related to the domain is obtained. Moreover, as the terms are mapped to Wikipedia articles, the information contained by the articles can be used, for example, to provide the translations of the term to each supported language in Wikipedia and to find related terms or term variants.

Evaluation

In this section, the experiment conducted to evaluate LiTeWi is described. Before presenting the conducted experiment, an overview of the different existing evaluation techniques for term extraction is provided; the evaluation of LiTeWi was carried out based on those term extraction evaluation techniques. Given that LiTeWi was designed for teachers who may want to elicit the terminology from the textbook they use, it was tested with two textbooks of different domains. LiTeWi was evaluated in two steps: evaluation of the candidate terms extraction process and evaluation of the candidate selection process.

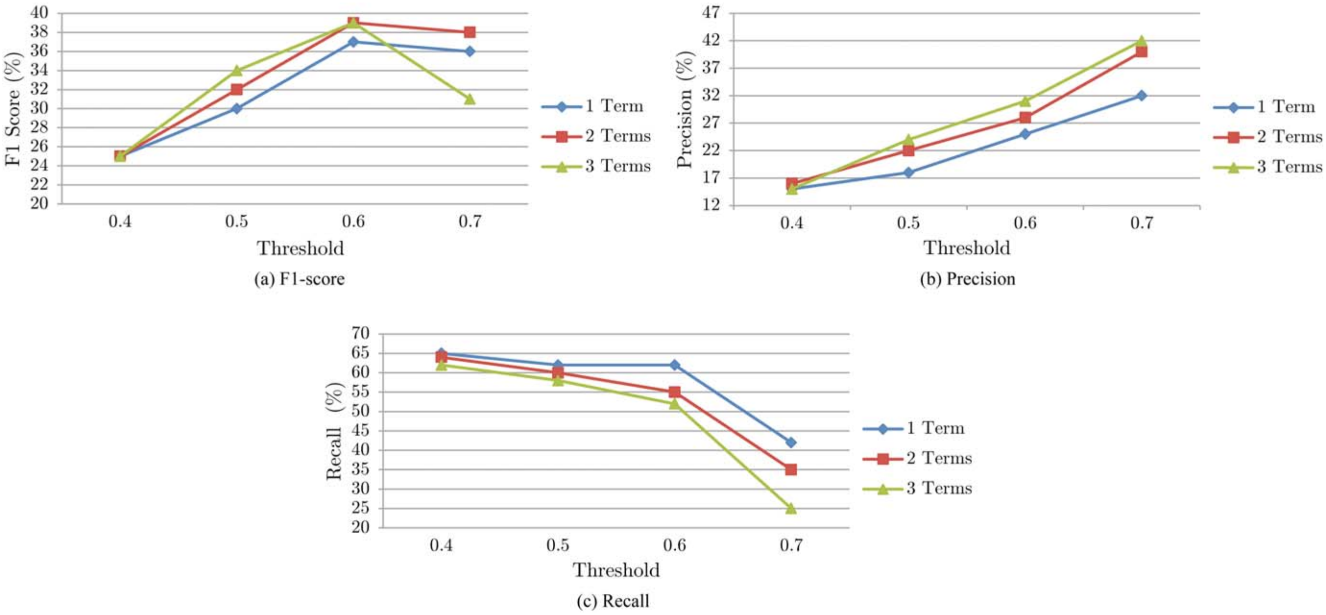

FIG. 6. Performance regarding threshold values. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Overview of the Term Extraction Evaluation Approaches

The evaluation of a term extraction technique is not an easy task, mainly because of the absence of a precise linguistic definition of what a term is (Pazienza et al., 2005). This leads to two main different approaches for evaluating the performance of the term extraction:

- ***Reference list**:* a list of terms is used as gold standard. In most cases this list is an already existing terminology for the specific domain. If the list does not exist, it can also be defined by experts who examine the same corpus used for the term extraction process. Using this approach, the performance of the term extraction is measured in terms of precision (the proportion of extracted terms that are also in the reference list) and recall (the percentage of the terms in the reference list extracted by the system). An example of a system using this method can be found in Daille (1994).
- ***Validation**:* this method is applied usually when a gold standard is not available or when the characteristics of the process have to be made explicit. In this method, the performance is evaluated by human experts who validate the extracted terms. However, this kind of manual evaluation tends to be a time-consuming activity. In Zanzotto (2002) the procedures and difficulties of carrying out this kind of process are described. The validation should be performed by more than one expert to have a more reliable evaluation. In addition, each expert should be aware of the notion of what a term is, as different experts are likely to produce different evaluations based on their own term intuition. Using this method, the performance is measured in terms of precision (the proportion of extracted terms validated by the experts from the total). The recall cannot be measured in this method as no gold standard exists.

Applied Evaluation Method

In the work presented here, the term extraction process was evaluated using both a reference list as a gold standard and an expert validation. For such a purpose, an evaluation was carried out with the following procedure: First the tool was tuned on the *Principles of Object Oriented Programming* textbook (Wong & Nguyen, 2010), then evaluated on two different books of different domains. The index of the analyzed textbooks was used as either a reference list or a gold standard. In addition, the elicited terms were manually analyzed by experts to determine whether the terms are related to the domain.

The first book used for the evaluation was the *Introduction to Astronomy* (Morison, 2008) textbook. This book consists of 150 pages of plain text and more than 110,000 words. The index is composed of 378 unique terms, of which 114 are single word terms (1-grams), 189 terms are 2-grams, 57 terms are 3-grams, and 18 terms are 4-grams. In all, 322 (of 378) of the index terms were related to one or more Wikipedia articles. That is to say, 85.18% of the terms refer to at least one Wikipedia article, such a proportion being the best recall achievable.

The second book used for the evaluation was the *Introduction to Molecular Biology* (Raineri, 2010). This book consists of 139 pages of plain text with more than 70,000 words. The index is composed of 274 unique terms, of which 116 are single word terms, 119 of them 2-grams, 35 3-grams, 3 4-grams, and 1 5-gram. For this textbook, 220 of 274 of the index terms were related to one or more Wikipedia articles. Hence, the best achievable recall is 81.30%.

DOI: 10.1002/asi

In each book, LiTeWi was tested in a two-step process. The candidate extraction process, described previously, was evaluated as detailed in the next section. In this experiment, each term extraction technique was tested on its own, measuring the recall according to the reference list. The results of the candidate selection process were also evaluated using the same procedure. Besides, the remaining terms were also evaluated using the expert validation method. The validation allows recognizing terms that the authors might not have considered relevant when organizing the textbook, but could be interesting for developing an educational ontology. The validation was carried out by three experts. To determine the domain-relatedness of candidate terms, only those terms that were considered valid by all the experts were selected.

Results of the Candidate Extraction

The performance of the selected techniques is summarized in Table 9. The TF-IDF process identified 2,533 terms achieving 18.9% recall with 2.9% precision for *Introduction to Astronomy*, while it achieved 17.15% recall with 4.26% precision for *Introduction to Molecular Biology*.

The CValue process extracted 2,058 candidate terms, 6.9% of them contained in the index and covering 37.5% of the index for the first textbook and 31.75% of the terms, with 2.48% precision, for the second textbook.

The KP-Miner identified 18.9% of the terms for *Introduction to Astronomy* textbook with 7.8% precision. However, it performs remarkably worse for *Introduction to Molecular Biology*, where it could only elicit 3.9% of the terms with a poor precision.

Finally, the Shallow Parsing Grammar identified terms in sentences that might be part of didactic resources such as definitions or examples. For the first textbook, it gathered 267 terms of the terms in the gold standard, which entails 13.42% recall. This method achieved 19.1% precision considering the gold standard. For the second textbook, it gathered 2.18% of the terms with 7.22% precision.

Some researchers have reported remarkable performances for the TF-IDF method, which achieved similar scores to those obtained by the CValue method on certain domains (Zhang, Brewster, & Ciravegna, 2008). They pointed out that the performance of the algorithm might be influenced by the importance of the single-word terms in the domain. In the analyzed documents, only 30 to 42% of the index topics were single-word terms, which explains the poor performance of the TF-IDF in this experiment.

Because most of the topics are multiword terms, the multiword term extraction methods might be expected to perform better in terms of recall than the TF-IDF. The results, which confirmed that intuition, are consistent with those obtained previously (Frantzi et al., 2000; Koilada et al., 2012; Zhang et al., 2008). As can be observed, the CValue performed much better and showed the advantage of combining termhood and unihood for term extraction methods.

Results of the Candidate Selection

In this section, the evaluation of each step in the candidate selection process is presented.

Combining term candidates. Once the candidate extraction has finished, the results obtained with every technique are combined and duplicates removed. After this step, the candidate term sets entailed 12,279 candidates for *Introduction to Astronomy* and 17,201 for *Introduction to Molecular Biology*. As expected, combining the results of the algorithms increased recall remarkably. However, precision is further reduced as domain-unrelated terms, or even wrong terms, affect the precision. The drop in the *precision* was an anticipated effect, but the following steps will improve its score.

Mapping terms to Wikipedia articles. As mentioned previously, the candidate terms are related to the Wikipedia articles to determine their domain-relatedness and to later filter unrelated terms. Mapping the terms to Wikipedia articles reduced the term list from 17,201 terms to 6,574 items in the astronomy textbook. Furthermore, 1,831 terms related to one Wikipedia article (meaning only one sense/meaning) were found in the astronomy textbook. In the biology textbook, the candidate term list shrank from 12,279 to 2,688 terms, 880 of them being related to only one Wikipedia article.

Term disambiguation. In the next step, those terms related to more than one Wikipedia article were disambiguated. Moreover, those candidate terms that were mapped to the

TABLE 9. Results of the candidate extraction methods over the tested textbooks.

Measure	Textbook	Precision(%)	Recall(%)	F1 score(%)
TF-IDF	*Astronomy*	2.9	18.9	5.02
	Mol. Biology	4.26	17.15	6.82
CValue	*Astronomy*	6.9	37.5	11.65
	Mol. Biology	2.48	31.75	4.6
KP-Miner	*Astronomy*	7.8	18.9	11.04
	Mol. Biology	1.82	3.9	3.9
Shallow Parsing	*Astronomy*	19.1	13.42	15.76
	Mol. Biology	7.22	2.18	3.34

DOI: 10.1002/asi

same Wikipedia article, that is, they share the same meaning, were combined into one term. For the astronomy textbook, the candidate term list was reduced to 3,972 terms, 295 of them included in the gold standard. However, 1,803 were considered domain-related by the experts. On the other hand, the term list of the biology textbook was composed of 1,194 terms in total, 174 from the index and 455 related to the domain of the textbook, as the experts had stated. Table 10 presents the precision, recall, and F1-score (the harmonic mean) of this step for both textbooks, considering the gold standard and the expert validation. Given that the recall cannot be measured using the validation approach, the corresponding cells contain the nonapplicable (N.A.) value.

Filtering domain-related terms. In the final step, those terms that were not related to the domain were removed. A remarkably improved precision can be observed in both the index and the domain related terms while barely affecting the recall (see Table 11).

After this step, the resulting term list for the astronomy textbook was composed of 1,545 terms, 275 of them included in the gold standard and 1,217 of them related to the domain. On the other hand, the term list of the biology textbook was composed of 635 terms, 165 from the index and 455 related to the domain of the textbook. Table 11 summarizes the statistics of this process.

Results of the Overall Process

The results obtained by each technique along with the performance of LiTeWi are presented and compared in Figure 7.

Comparing LiTeWi with each term extraction technique it uses, LiTeWi outperforms the best chosen technique (CValue) by more than 30%, showing that this approach improves the results considerably.

As Hartmann, Szarvas, and Gurevych (2012) claim, there is a tendency to prefer hybrid term extraction methods that use the termhood and unithood measures as the CValue because of their superior performance. Nevertheless, there is no consensus on which is the optimal method. Some methods perform better on domains or type of corpus, while others are more successful on certain kinds of terms. LiTeWi provides an appropriate method that is valid for all these cases. It takes the advantages of the chosen techniques to get as high a recall as possible, and then using Wikipedia it tries to improve the precision of the results filtering unwanted terms. Some techniques perform better in certain domains than in others (see Figure 7). Given that LiTeWi combines several techniques, it collects all their results and has a more stable performance.

Furthermore, as the candidate terms have been mapped to Wikipedia articles, their translations to other languages can be elicited to build a multilingual domain ontology. For instance, in the astronomy textbook, 80% of the extracted terms have a Spanish translation, 84% have a French translation, and 39% have a Basque translation. In the biology textbook, 75% of the extracted terms have a Spanish translation, 74% have a French translation, and 32% have a Basque translation.

Comparison With Other Approaches

In this section, the approach presented here is compared with two statistical methods aimed at extracting multiword terms and Wikifier, a state-of-the-art entity linking tool. Entity linking refers to the task of determining the reference of entities mentioned in a text within a knowledge base. The statistical approaches tested on this comparison are the point wise mutual information (PMI) and chi-square (X^2). PMI evaluates the strength of the association between the words in a multiword term candidate. On the other hand, X^2 measures the significance of the association between the words in a multiword term candidate. Both methods originally aim to extract bigrams but are adapted to longer terms in da Silva and Lopes (1999).

TABLE 10. Results after disambiguation.

	Textbook	Precision(%)	Recall(%)	F1 score(%)
Gold Standard	*Astronomy*	7.51	77.83	13.69
	Mol. Biology	12.73	63.50	21.20
Expert validation	*Astronomy*	45.39	N.A.	N.A.
	Mol. Biology	38.10	N.A.	N.A.

TABLE 11. Results after domain termhood processing.

	Textbook	Precision(%)	Recall(%)	F1 score(%)
Gold Standard	*Astronomy*	17.96	72.55	28.79
	Mol. Biology	27.09	57.29	21.37
Expert validation	*Astronomy*	78.77	N.A.	N.A.
	Mol. Biology	71.65	N.A.	N.A.

DOI: 10.1002/asi

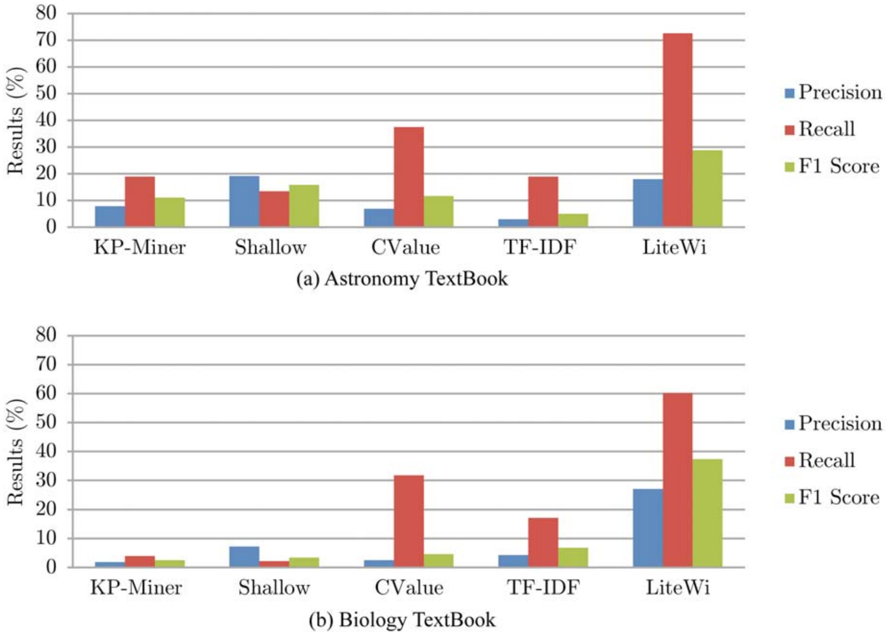

FIG. 7. Hybrid approach versus other algorithms. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

TABLE 12. PMI results.

	Textbook	Precision(%)	Recall(%)	F1 score(%)
Gold Standard	*Astronomy*	4.7	29.1	8.09
	Mol. Biology	3.71	24.53	6.44
Expert validation	*Astronomy*	13.11	N.A.	N.A.
	Mol. Biology	14.99	N.A.	N.A.

To run the PMI and the chi-square, the NLTK Python Toolkit (Bird, Klein, & Loper, 2009) was used. The procedure for setting up the algorithms was the same as that followed for LiTeWi. First, some empirical tests were carried out to tune up the method using the *Object Oriented Programming* textbook, and then the evaluation with the two textbooks mentioned above was performed.

As the algorithms are purely statistical, they return a lot of "noisy" terms, terms that do not make any sense. Then the stopword list that can be found in Appendix B (307 words) was applied to remove those terms. Besides, a minimum term frequency is required for multiword terms to be selected; those terms with a frequency less than 3 were filtered out.

Pointwise Mutual Information

The PMI (Fano, 1961) evaluates the strength of the association between the words in a multiword term candidate. It takes into account the probability of observing "*n*" variables together (the joint probability) with the probabilities of observing those "*n*" variables independently (chance).

The results presented in Table 12 were obtained applying the PMI method. From the astronomy textbook, a list of 2,340 terms was elicited, where 110 were part of the gold standard and 307 of them are related to the domain. In the case of the biology textbook, 1,587 terms were extracted, 59 of them being part of the index and 193 of them related to the domain.

The comparison to LiTeWi can be seen graphically in Figure 8, where remarkable performance differences, with a 200% increase in recall and more than a 200% increase in precision, can be observed between the PMI technique and LiTeWi.

Chi-Square

Chi-square (X^2) (Helmert, 1876; Plackett, 1983), measures the significance of the association between the words in a multiword candidate. It allows the identification of sequences of words that occur together more than they might by chance, and, hence, can be considered as terms.

Processing the X^2 technique resulted in a term list composed of 2,011 terms, where 304 terms were related to the domain and 94 form part of the index for the astronomy textbook. The term list for the biology textbook is composed

DOI: 10.1002/asi

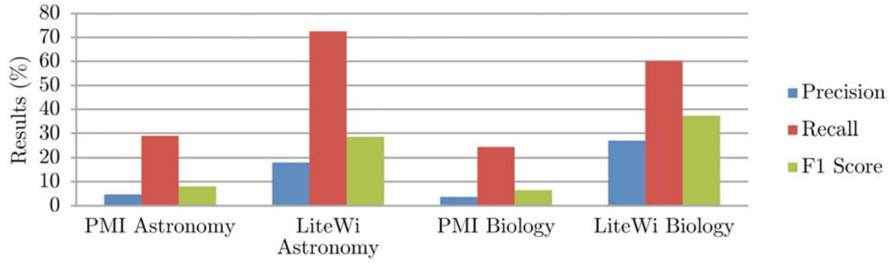

FIG. 8. LiTeWi versus PMI. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

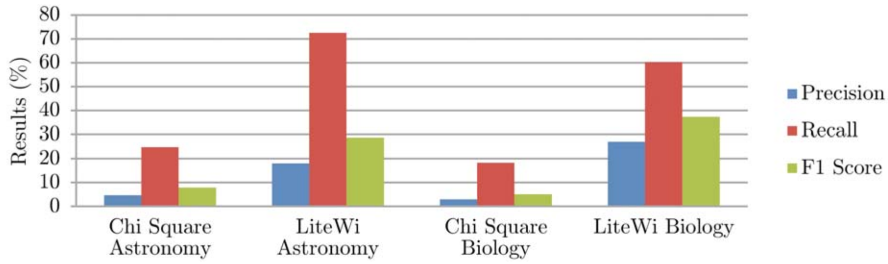

FIG. 9. LiTeWi versus X^2. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

TABLE 13. X^2 results.

	Textbook	Precision(%)	Recall(%)	F1 score(%)
Gold Standard	*Astronomy*	4.7	24.86	7.86
	Mol. Biology	2.97	18.24	5.10
Expert validation	*Astronomy*	15.11	N.A.	N.A.
	Mol. Biology	14.99	N.A.	N.A.

of 1,680 terms, of which 50 were included in the index and 193 were related to the domain. These results are described in Table 13.

The comparison with the proposed hybrid approach can be seen graphically in Figure 9. Again, LiTeWi outperforms X^2 in terms of recall and precision by more than 200%.

TABLE 14. Wikifier results.

	Textbook	Precision(%)	Recall(%)	F1 score(%)
Domain related	*Astronomy*	18.55	N.A.	N.A.
	Mol. Biology	49.27	N.A.	N.A.
Index	*Astronomy*	3.55	62.96	6.72
	Mol. Biology	2.24	10.21	3.67

Wikifier

The Wikifier (Cheng & Roth, 2013; Ratinov, Roth, Downey, & Anderson, 2011) entity linking tool was developed to identify important entities and concepts in text, disambiguate them, and link them to Wikipedia. Wikifier follows these steps:

1. Identify which expressions should be linked to Wikipedia.
2. Disambiguate the ambiguous expressions and entities.

Both steps are similar to those made by LiTeWi for the same purposes, except that Wikifier requires a training corpus for both steps. Wikifier achieved 62.96% recall on astronomy, whereas this score dramatically dropped to 10.21% on molecular biology (see Table 14). In both textbooks, precision was very low. Regarding the domain-relatedness, it achieved 18.55% precision, whereas it performed much better on molecular biology (49.27%).

As can be observed in Figure 10, Wikifier obtained slightly worse results to those achieved by LiTeWi on astronomy. However, LiTeWi performed remarkably better on molecular biology. The poor results in biology may be related to the nature of how Wikifier was trained to detect which expressions should be linked to Wikipedia.

Conclusions and Future Work

At a time when technology supported learning systems are being used more and more, providing aids for

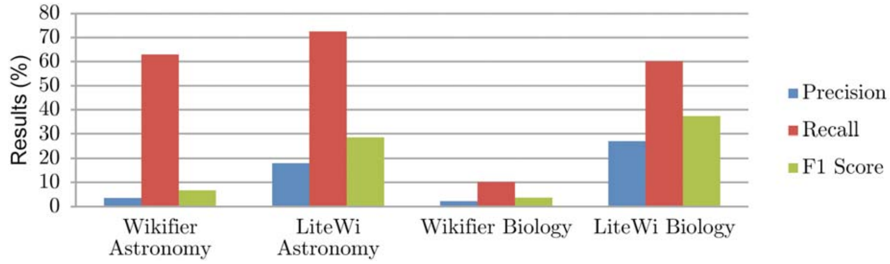

FIG. 10. LiTeWi versus Wikifier. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

building such systems and, especially, tools for developing the learning content for those systems, is essential. Ontology learning provides the means to semiautomatically build educational ontologies, that is, ontologies that encapsulate the domain knowledge for technology supported learning systems.

In this paper, LiTeWi, a tool that implements a domain-independent method for the elicitation of terms for educational ontologies from electronic textbooks was presented. It combines different approaches for the unsupervised term extraction using Wikipedia as a knowledge base. Term extraction with LiTeWi entails finding term candidates, using diverse term extraction techniques, and combining the results of those well-known algorithms to obtain the final set of elicited terms. In addition, LiTeWi maps the extracted terms to Wikipedia articles.

To determine its optimal setup, LiTeWi was tested on the *Principles of Object Oriented Programming* textbook. After the optimal setup was established, an evaluation was conducted on the *Introduction to Astronomy* and *Introduction to Molecular Biology* textbooks. The evaluation was carried out in two phases: First, the candidate term extraction process using LiTeWi was tested and compared to the performance of each term extraction technique. LiTeWi considerably outscored their performances. Next, the candidate selection process was also evaluated and positive results were obtained. Besides, the performance of the complete process of LiTeWi was compared with three other techniques—pointwise mutual information (PMI), chi-square (X^2), and Wikifier. Once again, LiTeWi outperformed them. In addition, as LiTeWi profits from Wikipedia for the term extraction process, it also elicits term variants for other languages. Therefore, it facilitates the acquisition of multilingual educational ontologies.

LiTeWi can be extended to allow the term extraction from documents written in other languages. LiTeWi combines three kinds of techniques to fulfill its purpose: statistical methods, NLP methods, and hybrid methods (CValue). Statistical methods are not restricted to a particular language. Regarding the NLP methods, this proposal only uses the DR Grammar derived previously (Conde et al., 2012). In fact, this grammar was adapted to English from the original grammar used for learning object extraction for the Basque language proposed previously (Larrañaga, Conde, Calvo, Arruarte, & Elorriaga, 2012). With respect to the hybrid methods, the authors have already developed CValue implementations for Spanish (https://github.com/Neuw84/). Both hybrid and NLP-based methods require part-of-speech information, to which end a POS parser for the new language must be provided if the current one does not support it. LiTeWi currently uses Freeling, which supports several languages (e.g., English, Spanish, French, etc.). In addition, LiTeWi requires Wikiminer to be able to deal with the new language, which in some cases requires Wikiminer to be trained to build the models that allow it to measure the relatedness. However, these models are publicly available for several languages (e.g., German) and the authors have already built it for Spanish. The inclusion of the Spanish language in LiTeWi is currently being addressed.

In the near future, the whole process of acquisition of educational ontologies (topics and pedagogical relationships between the topics) from electronic textbooks will be addressed. Wikipedia will be used as an additional source of information to fulfill such a task. In addition, more advanced techniques will be tested for the combination of the extracted candidate terms, for example, voting or supervised machine learning algorithms. Clustering techniques such as spectral clustering might also be useful to filter nonrelated words.

Acknowledgments

This work is supported by a fellowship from the Department of Science and Innovation of Spain (FPI-2010–035473), the Basque Government (GIC12/79), and the University of the Basque Country (UFI11/45). The authors thank both the external reviewers and the anonymous referees for their helpful comments and suggestions to improve the paper.

References

Benveniste, É. (1966). Problèmes de linguistique générale. Paris: Gallimard.

Berners-Lee, T., & Fischetti, M. (1999). Weaving the Web: The past, present and future of the World Wide Web by its inventor. New York: Harper Business.

Berners-Lee, T., Hendler, J.A., & Lassila, O. (2001, May). The semantic web. Scientific American, pp. 34–43.

Bird, S., Klein, E., & Loper, E. (2009). Natural language processing with python: Analyzing text with the natural language toolkit. Beijing, China: O'Reilly.

Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., & Hellmann, S. (2009). DBpedia—A crystallization point for the web of data. Journal of Web Semantics, 7(3), 154–165.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., & Taylor, J. (2008). Freebase: A collaboratively created graph database for structuring human knowledge. In SIGMOD '08: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data (pp. 1247–1250). New York: ACM.

Borst, W.N. (1997). Construction of engineering ontologies for knowledge sharing and reuse. Unpublished doctoral dissertation. University of Twente, Enschede.

Bourigault, D. (1996). Lexter, a natural language processing tool for terminology extraction. In Seventh EURALEX International Congress (pp. 771–779). Goteborg, Sweden: Göteborg University.

Buitelaar, P., Cimiano, P., & Magnini, B. (Eds.). (2005a). Ontology learning from text: Methods, applications and evaluation (Vol. 123). Amsterdam: IOS Press.

Buitelaar, P., Cimiano, P., Magnini, B., Buitelaar, P., Cimiano, P., & Magnini, B. (2005b). Ontology learning from text: An overview. In Ontology learning from text: Methods, applications and evaluation (Vol. 123, pp. 3–12). Amsterdam: IOS Press.

Cassel, L.N., Davies, G., LeBlanc, R., Snyder, L., & Topi, H. (2008). Using a computing ontology as a foundation for curriculum development. In SW-EL'08: Ontologies and Semantic Web for Intelligent Educational Systems at the 9th International Conference on Intelligent Tutoring Systems, ITS 2008 (pp. 21–29). Montreal: Springer.

Chandrasekaran, B., Josephson, J.R., & Benjamins, R.V. (1999). What are ontologies, and why do we need them? IEEE Intelligent Systems, 14(1), 20–26.

Cheng, X., & Roth, D. (2013). Relational inference for wikification. In Proocedings of the 2013 Conference on Empirical Methods on Natural Language Processing, EMNLP '13 (pp. 1787–1796). Seattle: Association for Computational Linguistics (ACL).

Conde, A., Larrañaga, M., Calvo, I., Arruarte, A., & Elorriaga, J. (2012). Automating the authoring of learning material in computer engineering education. In Proceedings of 2012 Frontiers in Education Conference (pp. 1376–1378). Seattle, WA: IEEE.

da Silva, J.F., & Lopes, G.P. (1999). A local maxima method and a fair dispersion normalization for extracting multi-word units from corpora. In Sixth Meeting on Mathematics of Language (pp. 369–381). Orlando.

Dagan, I., & Church, K. (1994). Termight: Identifying and translating technical terminology. In Proceedings of the Fourth Conference on Applied Natural Language Processing (pp. 34–40). Stroudsburg, PA: Association for Computational Linguistics.

Daille, B. (1994). Approach mixte pour extraction de termilogie: statistique lexicale et filters linguistiques. Unpublished doctoral dissertation. UniversitÔe Paris VII, Paris.

Daille, B., Habert, B., Jacquemin, C., & Royauté, J. (1996). Empirical observation of term variations and principles for their description. Terminology, 3(2), 197–257.

Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., & Harshman, R. (1990). Indexing by latent semantic analysis. Journal of the American Society for Information Science, 41(6), 391–407.

Dunning, T. (1993). Accurate methods for the statistics of surprise and coincidence. Computational Linguistics, 19(1), 61–74.

Earl, L.L. (1970). Experiments in automatic extracting and indexing. Information Storage and Retrieval, 6(4), 313–330.

El-Beltagy, S.R., & Rafea, A. (2009). KP-Miner: A keyphrase extraction system for English and Arabic documents. Information Systems, 34(1), 132–144.

Enguehard, C., & Pantera, L. (1995). Automatic natural acquisition of a terminology. Journal of Quantitative Lingufistics, 2(1), 27–32.

Fano, R. (1961). Transmission of information: A statistical theory of communications. Cambridge, MA: MIT Press.

Fok, A.W.P., & Ip, H.H.S. (2007). Educational ontologies construction for personalized learning on the web. In Evolution of teaching and learning paradigms in intelligent environment (Vol. 62, pp. 47–82). Berlin: Springer.

Fox, C. (1990). A stop list for general text. SIGIR Forum, 24, 19–35.

Frank, E., Paynter, G.W., Witten, I.H., Gutwin, C., & Nevill-Manning, C. (1999). Domain-specific keyphrase extraction. In Proceedings of 16th International Joint Conference on Artificial Intelligence (pp. 668–673). Burlington: Morgan Kaufmann Publishers.

Frantzi, K., Ananiadou, S., & Mima, H. (2000). Automatic recognition of multi-word terms: The c-value/NC-value method. International Journal on Digital Libraries, 3(2), 115–130.

Ganapathi, G., Lourdusamy, R., & Rajaram, V. (2011). Towards ontology development for teaching programming language. Lecture Notes in Engineering and Computer Science, I, 1845–1848.

Gavrilova, T., Farzan, R., & Brusilovsky, P. (2005). One practical algorithm of creating teaching ontologies. In Proceedings of Network Based Education (pp. 29–37). Rovaniemi: University of Lapland.

Gruber, T.R. (1991). The role of common ontology in achieving sharable, reusable knowledge bases. In Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning, KR'91 (pp. 601–602). Burlington: Morgan Kauffmann Publishers.

HaCohen-Kerner, Y., Gross, Z., & Masa, A. (2005). Automatic extraction and learning of keyphrases from scientific articles. In A. Gelbukh (Ed.), Proceedings of the Sixth International Conference on Computational Linguistics and Intelligent Text Processing (Vol. 3406, pp. 657–669). Berlin: Springer.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I.H. (2009). The WEKA data mining software: An update. ACM SIGKDD Explorations Newsletter, 11(1), 10–18.

Hartmann, S., Szarvas, G., & Gurevych, I. (2012). Mining multiwords terms from Wikipedia. In Semiautomatic ontology development (pp. 226–258). Hershey: IGI Global.

Helmert, F. (1876). Ueber die wahrscheinlichkeit der potenzsummen der beobachtungsfehler und über einige damit im zusammenhange stehende fragen. Journal of Zeitschrift für Mathematik und Phisik, 21, 102–219.

Hoffart, J., Suchanek, F.M., Berberich, K., Lewis-Kelham, E., de Melo, G., & Weikum, G. (2011). YAGO2: Exploring and querying world knowledge in time, space, context, and many languages. In Proceedings of the 20th International Conference Companion on World Wide Web (pp. 229–232). New York: ACM.

Hulth, A. (2003). Improved automatic keyword extraction given more linguistic knowledge. In Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (pp. 216–223). Stroudsburg, PA: Association for Computational Linguistics.

Justeson, J.S., & Katz, S.M. (1995). Technical terminology: Some linguistic properties and an algorithm for identification in text. Natural Language Engineering, 1(1), 9–27.

Kageura, K., & Umino, B. (1996). Methods of automatic term recognition: A review. Terminology, 3(2), 259–289.

Karlsson, F., Voutilainen, A., Heikkila, J., & Anttila, A. (Eds.). (1995). Constraint grammar: Language-independent system for parsing unrestricted text. Berlin: Mouton de Gruyter.

Koilada, N., Newman, D., Lau, J.H., & Baldwin, T. (2012). Bayesian Text segmentation for index term identification and keyphrase extraction. Mumbai, India: Indian Institute of Technology Bombay.

Kupiec, J., Pedersen, J., & Chen, F. (1995). A Trainable document summarizer. In Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 68–73). New York: ACM.

Larrañaga, M., Conde, A., Calvo, I., Arruarte, A., & Elorriaga, J. (2012). Evaluating the automatic extraction of learning objects from electronic textbooks using ErauzOnt. In S. Cerri, W. Clancey, G. Papadourakis, & K. Panourgia (Eds.), Intelligent tutoring systems (Vol. 7315, pp. 655–656). Berlin/Heidelberg: Springer.

Maedche, A. (2002). Ontology learning for the semantic web. Alphen aan den Rijn: Kluwer Academic Publishers.

Maedche, A., & Staab, S. (2000). Mining ontologies from text. In Proceedings of the 12th International Conference on Knowledge Engineering and Management, EKAW 2000 (Vol. 1937, pp. 189–202). Berlin: Springer.

Maynard, D., & Ananiadou, S. (1999). Identifying contextual information for multi-word term extraction. In Proceedings of the 5th International Congress on Terminology and Knowledge Engineering (pp. 212–221). Innsbruck, Austria.

Medelyan, O., & Witten, I.H. (2006). Thesaurus based automatic keyphrase indexing. In Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries (pp. 296–297). New York: ACM.

Milne, D., & Witten, I.H. (2008). Learning to link with Wikipedia. In Proceedings of the 17th ACM Conference on Information and Knowledge Management (pp. 509–518). New York: ACM.

Milne, D., & Witten, I.H. (2013). An open-source toolkit for mining Wikipedia. Artificial Intelligence, 194, 222–239.

Mima, H., & Ananiadou, S. (2000). An application and evaluation of the C/NC-value approach for the automatic term recognition of multi-word units in Japanese. International Journal of Terminology, 6(2), 175–194.

Mizoguchi, R., & Bourdeau, J. (2000). Using ontological engineering to overcome AI-ED problems. International Journal on Artificial Intelligence in Education, 11(2), 107–121.

Morison, I. (2008). Introduction to astronomy and cosmology. Hoboken, NJ: Wiley.

Neches, R., Fikes, R., Finin, T., Gruber, T.R., Senator, T., & Swartout, W. (1991). Enabling technology for knowledge sharing. AI Magazine, 12(3), 36–56.

Padró, L., & Stanilovsky, E. (2012). FreeLing 3.0: Towards wider multilinguality. In Proceedings of the Language Resources and Evaluation Conference (LREC 2012) (pp. 2473–2479). Istanbul: European Language Resources Association (ELRA).

Pazienza, M.T., & Stellato, A. (Eds.). (2012). Semiautomatic ontology development: Processes and resources. Hershey, PA: Information Science Reference.

Pazienza, M.T., Pennacchiotti, M., & Zanzotto, F.M. (2005). Terminology extraction: An analysis of linguistic and statistical approaches. In D.S. Sirmakessis (Ed.), Knowledge mining (pp. 255–279). Berlin Heidelberg: Springer.

Plackett, R.L. (1983). Karl Pearson and the chi-squared test. International Statistical Review Revue Internationale de Statistique, 51(1), 59–72.

Porter, M.F. (1997). An algorithm for suffix stripping. In K. Sparck Jones & P. Willett (Eds.), Readings in information retrieval (pp. 313–316). San Francisco: Morgan Kaufmann Publishers.

Raineri, D. (2010). 11th hour: Introduction to molecular biology. Hoboken, NJ: Wiley.

Ratinov, L., Roth, D., Downey, D., & Anderson, M. (2011). Local and global algorithms for disambiguation to Wikipedia. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, HLT '11 (pp. 1375–1384). Stroudsburg: Association for Computational Linguistics.

Salton, G. (1971). The smart retrieval system; experiments in automatic document processing. Upper Saddle River, NJ: Prentice-Hall.

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. In Information processing and management (pp. 513–523). Philadelphia: Elsevier.

Salton, G., Yang, C.S., & Yu, C.T. (1975). A theory of term importance in automatic text analysis. Journal of the American Society for Information Science, 26(1), 33–44.

Sosnovsky, S., & Gavrilova, T. (2006). Development of educational ontology for c-programming. International Journal on Information Theories & Applications, 13(4), 303–308.

Studer, R., Benjamins, V.R., & Fensel, D. (1998). Knowledge engineering, principles and methods. Data and Knowledge Engineering, 25(1–2), 161–197.

Toutanova, K., Klein, D., Manning, C.D., & Singer, Y. (2003). Feature-rich part-of-speech tagging with a cyclic dependency network. In Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (Vol. 1, pp. 173–180). Stroudsburg, PA: Association for Computational Linguistics.

Turney, P. (1999). Learning to extract keyphrases from text. Ottawa: National Research Council Canada, Institute for Information Technology.

Turney, P. (2000). Learning algorithms for keyphrase extraction. Information Retrieval, 2, 303–336.

Velardi, P., Missikoff, M., & Basili, R. (2001). Identification of relevant terms to support the construction of domain ontologies. In Proceedings of the Workshop on Human Language Technology and Knowledge Management (Vol. 2001, pp. 5:1–5:8). Stroudsburg, PA: Association for Computational Linguistics.

White, T. (2010). Hadoop: The definitive guide (2nd ed.). Sebastopol: O'Reilly Media.

Whitley, D. (1989). The genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In Proceedings of the Third International Conference on Genetic Algorithms (ICGA-89) (pp. 116–121). San Francisco: Morgan Kaufmann Publishers.

Wong, S., & Nguyen, D. (2010). Principles of object-oriented programming. Retrieved from http://cnx.org/content/col10213/1.37/

Zanzotto, F.M. (2002). L'estrazione della terminologia come strumento per la model-lazione di domini conoscitivi. Unpublished doctoral dissertation. Università degli Studi di Roma Tor Vergata, Roma.

Zhang, Z., Brewster, C., & Ciravegna, F. (2008). Ciravegna f: A comparative evaluation of term recognition algorithms. In Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC08) (pp. 2018–2113). Marrakech, Morocco: European Language Resources Association (ELRA).

Appendix A. Stopword List Applied to TF-IDF Index

"a," "an," "and," "are," "as," "at," "be," "but," "by," "for," "if," "in," "into," "is," "it," "no," "not," "of," "on," "or," "such," "that," "the," "their," "then," "there," "these," "they," "this," "to," "was," "will," "with."

Appendix B. Stopword List Applied to the Term Extraction Techniques

"a," "about," "above," "after," "again," "against," "all," "am," "an," "and" "any," "are," "aren't," "as," "at," "be," "because," "been," "before," "being," "below," "between," "both," "but," "by," "can't," "cannot," "could," "couldn't," "did," "didn't," "do," "does," "doesn't," "doing," "don't," "down," "during," "each," "few," "for," "from," "further," "had," "hadn't," "has," "hasn't," "have," "haven't," "having," "he," "he'd," "he'll," "he's," "her," "here," "here's," "hers," "herself," "him," "himself," "his," "how," "how's," "i," "i'd," "i'll," "i'm," "i've," "if," "in," "into," "is," "isn't," "it," "it's," "its," "itself," "let's," "me," "more," "most," "mustn't," "my," "myself," "no," "nor," "not," "of," "off," "on," "once," "only," "or," "other," "ought," "our," "ours," "ourselves," "out," "over," "own," "same," "shan't," "she," "she'd," "she'll," "she's," "should," "shouldn't," "so," "some," "such," "than," "that," "that's," "the," "their," "theirs," "them," "themselves," "then," "there," "there's," "these," "they," "they'd," "they'll," "they're," "they've," "this," "those," "through," "to," "too," "under," "until," "up," "very," "was," "wasn't," "we," "we'd," "we'll," "we're," "we've," "were," "weren't," "what," "what's," "when,"

“when’s,” “where,” “where’s,” “which,” “while,” “who,” “who’s,” “whom,” “why,” “why’s,” “with,” “won’t,” “would,” “wouldn’t,” “you,” “you’d,” “you’ll,” “you’re,” “you’ve,” “your,” “yours,” “yourself,” “yourselves,” “*,” “/,” “!,” “?,” “a,” “able,” “about,” “across,” “after,” “all,” “almost,” “also,” “am, “among,” “an,” “and,” “any,” “are,” “as,” “at,” “be,” “because,” “been,” “but,” “by,” “can,” “cannot,” “could,” “dear,” “did,” “do,” “does,” “either,” “else,” “ever,” “every,” “for,” “from,” “get,” “got,” “had,” “has,” “have,” “he,” “her,” “hers,” “him,” “his,” “how,” “however,” “i,” “if,” “in,” “into,” “is,” “it,” “its,” “just,” “least,” “let,” “like,” “likely,” “may,” “me,” “might,” “most,” “must,” “my,” “neither,” “no,” “nor,” “not,” “of,” “off,” “often,” “on,” “only,” “or,” “other,” “our,” “own,” “rather,” “said,” “say,” “says,” “she,” “should,” “since,” “so,” “some,” “than,” “that,” “the,” “their,” “them,” “then,” “there,” “they,” “to,” “too,” “was,” “us,” “wants,” “was,” “we,” “were,” “what,” “when,” “where,” “which,” “while,” “who,” “whom,” “why,” “will,” “with,” “would,” “yet,” “+,” “–,” “[,””],” “ ,” “ ,” .”,” ,”,” “(,””),” “whose,” “[,” “>,” “etc.”

DOI: 10.1002/asi