Classifying Domain-Specific Terms Using a Dictionary

Su Nam Kim
Dept. of CSSE
University of Melbourne
Melbourne, Australia
snkim@csse.unimelb.edu.au

Lawrence Cavedon
School of CS
IT
RMIT University
Melbourne, Australia
lawrence.cavedon@rmit.edu.au

Abstract

Automatically building domain-specific ontologies is a highly challenging task as it requires extracting domain-specific terms from a corpus and assigning them relevant domain concept labels. In this paper, we focus on the second task: i.e., assigning domain concepts to domain-specific terms. Motivated by previous approaches in related research (such as word sense disambiguation (WSD) and named entity recognition (NER)) that use semantic similarity among domain concepts, we explore three types of features — contextual, domain concepts, topics — to measure the semantic similarity of terms; we then assign the domain concepts from the best matching terms. As evaluation, we collected domain-specific terms from FOLDOC, a freely available on-line dictionary for the the *Computing* domain, and defined 9 domain concepts for this domain. Our results show that beyond contextual features, using domain concepts and topics derived from domain-specific terms helps to improve assigning domain concepts to the terms.

1 Introduction

Domain-specific terms are terms that have significant meaning(s) in a specific domain. For example, terms such as *Gulf* and *Kuwait* are associated with the domain of oil due to their frequent appearances in contexts related to oil although they indicate geographical areas. In some resources, domain-specific terms are further categorized in terms of their domain concepts (i.e., semantic labels/classes). For example, *Firefox* belongs to the domain concept Software, while *Prolog* is associated with the domain concept Programming. In this paper, we use the term **domain concept** for consistency. Note that in previous work, the meaning of the domain-specificity is associated with either word senses (e.g. (Magnini et al., 2002; Rigutini et al., 2005)) or the statistical use of terms in context (e.g. (Drouin, 2004; Milne et al., 2006; Kida et al., 2007; Park et al., 2008; Kim et al., 2009; Vivaldi and Rodrguez, 2010)). In WordNet, the domain concept is assigned based on the word senses. Similarly, WordNet Domain has terms with domain concepts per sense. However, most work previously conducted work used domain-specificity is based on statistical use. In this paper, we follow the latter definition, i.e., domain-specificity associated with the statistical use of the term.

Domain-specificity of terms has been leveraged in various natural language processing (NLP) and related tasks, such as word sense disambiguation (WSD) (Magnini et al., 2002), named entity recognition (NER) (Kazama and Torisawa, 2007), and query expansion (Rigutini et al., 2005). Resources containing domain information fall into two groups: the list of domain-specific terms without domain concepts (e.g. Agrivoc, EUROVOC, ASFA Thesaurus); and with domain concepts (e.g. WordNet, WordNet Domain, Unified Medical Language System (UMLS)). Although there have been efforts developing such knowledge resources, the task has been generally carried out by hand, requiring high cost and time. Further, even hand-crafted resources are often limited in terms of quality and quantity.

Moreover, the content needs to be constantly updated/maintained as new words are added. As a result, there has been recent work on automatic ontology builders (e.g. `OntoLearn, Text2Onto`) that work by extracting domain-specific terms and tagging domain-concepts to build such resources.

Building a domain-specific ontology requires two main tasks — extracting domain-specific terms and assigning the domain concept(s). There have been several methods proposed for each task and also as a complete ontology builder—we describe such work in Section 2. In this paper, our interest lies on assigning domain concepts to the existing domain-specific resources. In one sense our task can be viewed as building a taxonomy from dictionaries (Rigau et al., 1998) and/or a semantic class labelling task (Punuru and Chen, 2007). Since some resources are already publicly available (despite shortcomings), utilizing these resources reduces the time for manually developing training data, and should lead to robust systems due to consistent labeling. In addition, such resources are reusable for enlarging the existing resources or creating new semantic resources.

Our basic approach is to use semantic similarity between domain-specific terms. Contextual features have often been employed for semantic similarity in various tasks, such as text categorization (Joachims, 1998) and dialogue act classification (Ivanovic, 2005). Thus, we also explore using context as base features. Furthermore, we explore the use of rich semantic features. That is, we employ the domain concepts and topics derived from known domain-specific terms over the same resource as additional features. We detail our rich semantic features in Section 4.2 and 4.3. In evaluation, we applied our approaches to the domain Computing, as the interest in this domain is growing due to the large volume of web corpora, including social media such as web forums and blogs.

In the following sections, we describe related work and the existing resources in Section 2 and 3. We then describe our features in Section 4, and evaluate our methods in Section 5. We summarize our work in Section 6.

2 Related Work

There are two individual sub-tasks that deal with domain-specific terms — extraction/identification and labeling domains/concepts. Further, extracting domain-specific terms is combined with technical term extraction in order to extract candidates, while identification of domain-specific terms is a binary decision (i.e., with a given term, determining whether it is domain-specific to the target domain).

A number of extraction methods have been proposed (Drouin, 2004; Rigutini et al., 2005; Milne et al., 2006; Kida et al., 2007; Park et al., 2008; Kim et al., 2009; Vivaldi and Rodrguez, 2010). Most used supervised approaches while (Park et al., 2008; Kim et al., 2009; Vivaldi and Rodrguez, 2010) undertook the task in an unsupervised manner. In addition, (Rigutini et al., 2005) used sense-based domain-specificity, while others used statistical use-based measures to determine domain-specificity of the candidate terms. (Rigutini et al., 2005) is motivated by the intuition that, similar to word sense disambiguation, domain-specificity can be identified using contextual semantic similarity in which those terms occur, since domain-specificity is associated with the word senses. (Milne et al., 2006) studied Wikipedia entries as domain-specific terms and crosschecked the terms in `Agrivoc` with Wikipedia entries to verify the domain-specificity of Wikipedia entries. The basic idea in (Kida et al., 2007) is that a domain can be identified via a list of known technical domain terms. As unsupervised approaches, (Park et al., 2008) introduced a probability based weighting in order to measure the domain-specificity of the term over a large corpus. Similarly, (Kim et al., 2009) used term frequencies across the documents using modified TF·IDF, which replace a document with a domain. (Vivaldi and Rodrguez, 2010) made use of Wikipedia categories and page structures. The intuition is that the Wikipedia categories are domain-specific, thus, by retrieving the Wikipedia entries through the category trees starting with a target domain, the domain-specific terms under the target domain can be automatically retrieved.

The task of domain assignment has some relationship to the word sense disambiguation (WSD) and named entity recognition (NER) tasks. While WSD attempts to assign the correct sense of terms

from some given repository of senses (typically, WordNet), assigning domains to domain-specific terms in our work first requires that we construct the repository. NER is a subtask of information extraction that involves finding named entities and assigning each a tag from a predefined set of categories, such as LOCATION or PERSON. The difference with our task in this paper is that in both WSD and NER, the target terms are generally in some use context (i.e., the correct word sense of target term depends on that context), while our targets are isolated, i.e., appear out of context. In this paper, our approach is closer to corpus-based WSD which normally uses co-occurrence of terms between two corpora.

In recent years, there have been systems proposed to extract terms and to assign semantic labels to them (Navigli and Velard, 2004; Cimiano and Vlker, 2005; Nicola et al., 2009). OntoLearn (Navigli and Velard, 2004) has three components. First, it extracts a domain terminology from Web sites. It then assigns the domain concepts in order to build a hierarchical structure of ontologies. The system uses semantic similarity between WordNet concepts for component words in a candidate and conceptual relations among the concept components based on word senses. Finally, ontologies in WordNet are trimmed and enriched with the extracted domain concepts. Text2Onto (Cimiano and Vlker, 2005) is another ontology builder and includes three components. First, the system represents the knowledge as metadata in the form of instantiated modeling primitives called Probabilistic Ontology Model, which is language-independent. Second, the system uses the user interaction in order to measure the domain-specificity for candidates. Finally, it accumulates the ontologies based on previously added ontologies to overcome computational redundancy over time when corpus/documents are changed. More recently, (Nicola et al., 2009) developed an automatic ontology builder by combining Unified Software Development Processing (UP) and UML. It bases its characteristics on UP and uses UML to support the preparation of the blueprints of the ontology development.

3 Data

Multiple taxonomy resources such as WordNet and Wikipedia are available for identifying terms in the Computing domain. However, not all of these systematically assign semantic labels to terms. To determine suitability of popularly used resources for our task, we first investigated their utility.

WordNet3.0 (Fellbaum, 1998) includes domain information but the number of terms with domain information is very limited. Moreover, it does not include many terms related to the Computing domain, and those terms it does include are often proper nouns (e.g. *Firefox*) or compound nouns (e.g. *wireless connection*). For WordNet Domain, (Magnini et al., 2002) developed a domain system and semi-manually assigned domains to WordNet terms in terms of their word senses. Although the size of the resource with domain information is larger than WordNet3.0 (i.e., 6,050 nouns with one sense), it is still a relatively small resource. Moreover, the domain called *factotum* (i.e., "undecided") is used for many terms, which makes it less usable.

Wikipedia[1] is the largest folksonomy taxonomy. It contains terms (hereafter, *entries*) and categories per entry. It also provides hyperlinks between entries and entry pages. Despite its vast size, Wikipedia is not designed to be a dictionary, thus it does not contain definitions entries. Moreover, the categories are not systematically organized and often contain noise (which are not relevant to the target domains).

Wiktionary[2], on the other hand, is a growing online dictionary for all domains. It has characteristics similar to WordNet, such as definitions and relations (e.g. hypernym, synonym). It also partially contains domain information per word sense. In addition, it is linked to Wikipedia. However, since one term could have multiple senses and not all senses are tagged with a specific domain, it requires a preprocessing step to discover terms related to the specific domain (for this paper, Computing domain only). The number of unique terms in Wiktionary is 10,586 without counting individual word senses.

[1] http://en.wikipedia.org/wiki/Main_Page
[2] http://en.wiktionary.org/wiki/dictionary

The final resource we consider is FOLDOC[3]. FOLDOC is the (so far) largest handcrafted on-line dictionary for the Computing domain. It also contains definitions and sub-domains such as *hardware* and *operating system*, and provides hyperlinks to other dictionary terms in the definition and links to Wikipedia and OneLook dictionary search[4]. Unlike Wiktionary, only word senses of terms related to Computing are listed, thus, all terms in the dictionary are relevant to the Computing domain. For these reasons, as well as the size of the resource being sufficient to evaluate our method, we decided to use FOLDOC for our purposes. To understand FOLDOC better, we checked the overlaps between it and other resources in Table 1. Note that all resources were retrieved in March 2011.

Source	WikiDic	Wikipedia	WordNet
Instance	10,586	10,863,326	117,798
Overlap	1,682	9,917	2,756

Table 1: Overlap between the FOLDOC dictionary and other resources.

FOLDOC contains 14,826 unique terms with multiple senses, resulting in 16,450 terms in total. 13,072 and 3,378 terms are *direct* and *redirect*, respectively (the concept of *direct* and *redirect* is the same as that in Wikipedia). Among direct terms, 8,621 terms have manually assigned domain concept(s). The total number of domain concepts in the dictionary is 188. Finally, we manually mapped 188 onto 9 domain concepts which are super-labels. For example, labels in FOLDOC, security, specification, Unix are mapped on to Networking, Documentation, OS, respectively. Note that, based on our observations, we found many of the labels defined in FOLDOC are too fine-grained, and some are used only for 1 or 2 terms. Furthermore, the labels are not hierarchically structured. In addition, similar to the trade-off between fine-grained vs. coarse-grained word senses, we believe coarse-grained labels would be more usable (e.g. document classification using coarse-grained labels of terms), thus, we used 9 super-labels in this work.

Table 2 shows the final domain concepts we used and the number of instances in each domain concept. Note that since one term can have multiple semantic labels, the total number of instances with one label is 10,147. Table 3 shows the number of terms with multiple senses and multiple domain concepts.

Domain (Terms)	Domain (Terms)
CS (755)	Documentation (1,906)
HW (1,490)	Jargon (298)
Networking (1,220)	OS (363)
Programming (3,042)	SW (144)
Other (929)	
Total	10,147(8,621)

Table 2: Data Size per Domain Concept. *8,621* is the number of word types, where *CS, HW, OS, SW* indicate *computer science*, *hardware*, *operating system*, and *software*, respectively.

Info.	1	2	3	4	5	6	7
Label	7172	1353	79	8	–	–	–
Sense	7957	475	124	41	11	2	2

Table 3: Terms with multiple labels and senses.

4 Methodology

4.1 Feature Set I: Bag-of-Words

n-gram-based bag-of-words (BoW) features are one of the most broadly applied features to measure the semantic similarity between two terms/texts. This has been used in various tasks such as document classification (Joachims, 1998), dialogue act classification (Ivanovic, 2005) and term classification (Lesk, 1986; Baldwin et al., 2008).

As shown in (Hulth and Megyesi, 2006), keywords along the contextual features (i.e., simple 1-grams) are useful in identifying semantic similarity. However, keywords are often multi-grams such as 2-grams (e.g. *Fast Ethernet, optical mouse*) and 3-grams (e.g. *0/1 knapsack problem, Accelerated Graphics Port*). Sharing the same intuition, some previous work (Ivanovic, 2005) employed not only 1-grams but also 2-grams for the classification task. Similarly, we also observed that terms are often multi-grams. Thus, in this work, we also explored various n-grams. In evaluation, we tested 1- and 2-grams individually as well as the combination of 1-

[3]http://foldoc.org/

[4]http://www.onelook.com/

and 2-grams together (i.e., 1+2-grams). Note that since previous work has shown that the use of lemmas performed better than raw words, we chose to use lemmas as features. We also tested BoWs from nouns and verbs only. As feature weights, we tested simple Boolean and TF·IDF. In evaluation, the features are filtered with respect to the frequency of indexing words. That is, we tested three different term frequencies (i.e., frequency $\geq$ 1, 2, and 3) in order to select the indexing terms as BoW features.

4.2 Feature (II): Domain Concepts of Domain-Specific Terms

BoW features are useful for measuring the semantic similarity between two targets. However, we observed that since the number of terms in the dictionary definition is small, there will be a lack of context (similar to shortcomings reported in (Lesk, 1986) for WSD using dictionary definition). On the other hand, we noticed that a term's definition often contains terms which belong to the same domain concepts. For example, the target term *Ethernet* belongs to the domain concept Networking, and its definition is "A local area network first described by ...". *Local area network* in the definition also belongs to the same domain concept Networking. Hence, we use the domain concept(s) of dictionary terms found in the definition of the target term as a feature.

We also extend the target's definition with its dictionary terms. To overcome the pitfall of the algorithm in (Lesk, 1986) due to lack of terms in the definition, (Baldwin et al., 2008) utilized the extended definition from the dictionary terms found in the target's definition. Similarly, instead of the definition of dictionary terms in the target's definition (i.e., extended definition), we used the domain concept(s) of dictionary terms in the extended definition. We hypothesised that using these domain concepts would provide more direct information about domain concepts of the target term.

In Table 4, we demonstrate how and what to extract as domain concepts for the target *database*. The definition of the target *database* contains dictionary terms *database, table, flat file, comma-separated values*. Since *database* is the target term itself and *comma-separated values* is not found in the dictionary, we use the domain concepts from *table* and *flat file* only, which include CS, and OS, HW. Note that some terms have multiple labels as described in Section 3. We also extend *table* and *flat file* to obtain the extended domain concepts from the extended definition. Finally, we accumulated domain concepts, CS, Programming, Documentation from *records*, CS from *relational database* and Documentation from *flat ASCII* from the extended definitions.

4.3 Feature (III): Topics of Domain-Specific Terms

Topic modeling (Blei et al., 2003) is an unsupervised method to cluster documents based on context information. As it is not a classification method, the topics produced by the topic modeling algorithm are abstract, thus not directly associated with predefined semantic labels. However, we observed that topic terms for each topic are generally associated with the domain concepts. From our observation, we hypothesized that (ideally) one topic is associated with one domain concept (although some may contain multiple domain concepts same as multi-sensed words). As such, we assigned topic ID(s) per dictionary terms using topic modeling software[5], then used this as an additional feature with BoWs. Likewise, we also obtained topic ID(s) for dictionary terms found in the definition of the target term. Note that depending on the features used to obtain topics, the association between topic IDs and our 9 domain concepts would change. Table 5 demonstrates the topics and how we extract the *Topic* and *extended Topic* features using the same example use above. *database* is tagged with topic ID = 1,2,4 while *table* has topic ID = 4. We represented features by accumulating the topic IDs over 9 topic IDs which are the same number of our domain concepts.

5 Experiments

For our evaluation, we first replaced the email addresses, numbers, urls with their category *EMAIL, NUMBER, URL*, respectively. We then performed POS tagging and lemmatization using `Lingua::EN::Tagger` and `morph` tools (Minnen et al., 2001), respectively. For learning, we

[5]The topic modeling tool we used can be downloaded from http://www.ics.uci.edu/ newman/code/

Term	Label	Definition
Target Term, "database" and its Domain Concept and Definition		
database	CS	A ~~*database*~~ containing a single *table*$_{\mathsf{CS}}$, stored in a single *flat file*$_{\mathsf{OS,HW}}$, often in a human-readable format such as ~~*comma-separated values*~~ or fixed-width columns.
Extending Definitions of Dictionary Terms found in Target's Definition		
table	CS	A collection of *records*$_{\mathsf{CS,Programming,Documentation}}$ in a *relational database*$_{\mathsf{CS}}$.
flat file	OS, HW	A single file containing *flat ASCII*$_{\mathsf{Documentation}}$ representing or encoding some structure, e.g. a ~~*database*~~, tree or network.

Table 4: Extracting domain concepts of dictionary terms in definitions.

Type	Value	Topics								
		T1	T2	T3	T4	T5	T6	T7	T8	T9
Target Term	*database*	1	1	0	1	0	0	0	0	0
Dictionary Term	*table*	0	0	0	1	0	0	0	0	0
in Target Definition	*flat file*	0	1	0	0	1	0	0	0	1
Feature	Direct	1	1	0	1	0	0	0	0	0
representation	Extended	1	2	0	2	1	0	0	0	1

Table 5: Extracting topics for target terms and dictionary terms.

simulated our method by both supervised and semi-supervised approaches. We used SVM (Joachims, 1998)[6] for supervised learning and SVMlin (Sindhwani and Keerthi, 2006)[7] for semi-supervised learning. For supervised learning, we performed 10-fold cross-validation over 8,621 terms which have manually assigned labels in FOLDOC. For semi-supervised learning, we used the same data for test and training and 4,451 unlabeled terms in FOLDOC as unlabeled data. As the baseline system, we used the feature TF·IDF valued *1-gram* from all terms with frequency ≥ 1. The performances are compared using micro-averaged F-score $\mathcal{F}_\mu$.

5.1 Supervised Learning

Table 6 shows performance by supervised learners with various BoWs. Note that we only report performance using TF·IDF since those using Boolean weights performed poorly. We also ran the experiments over different frequency which leads to various numbers of indexing terms. Finally, we tested noun- and verb-only features.

Overall, the best performance is produced by using both 1- and 2-grams with frequency ≥ 1, as this configuration contains the largest amount of features. However, the improvement by adding 2-grams is not significant (i.e., *52.01%* vs. *52.47%* in $\mathcal{F}_\mu$). Between using all terms vs. nouns and verbs only, using all terms performed slightly better. Despite dominant information derived from nouns and verbs, other POS tagged words also contributed to distinguishing the domain concepts. Likewise, the performances using nouns and verbs only are generally better when using features with frequency ≥ 1.

Table 7 shows performance using both n-grams and semantic features. At first, adding rich semantic features (i.e., *Domain Concept, Topic*) significantly improved performance (*52.01%* vs. *60.62%*). In particular, *Topic* features helped to improve performance. As we hypothesised, topics are likely associated with domain concepts which resulted in performance improvements. *Domain concept* features also helped to gain higher performance, as they provide more direct semantic information than n-gram features. Between direct and extended (i.e., indirect) semantic features, we noticed that *extended*

[6] http://www.cs.cornell.edu/People/tj/svm_light/svm_hmm.html
[7] http://vikas.sindhwani.org/svmlin.html

Feature	Indexing	Frequency≥1 (F1)			Frequency≥2 (F2)			Frequency≥3 (F3)		
		$\mathcal{P}_\mu$	$\mathcal{R}_\mu$	$\mathcal{F}_\mu$	$\mathcal{P}_\mu$	$\mathcal{R}_\mu$	$\mathcal{F}_\mu$	$\mathcal{P}_\mu$	$\mathcal{R}_\mu$	$\mathcal{F}_\mu$
All	1	56.64	48.07	*52.01*†	56.64	48.07	52.01	56.56	48.00	51.93
Terms	2	54.42	46.19	49.97	54.42	46.19	49.97	51.68	43.87	47.45
	1+2	57.14	48.50	**52.47**	57.14	48.50	52.47	57.05	48.42	52.38
Noun	1	55.56	47.16	51.02	55.56	47.16	51.02	55.49	47.10	50.95
+	2	52.39	44.47	48.10	52.39	44.47	48.10	49.30	41.84	45.27
Verb	1+2	56.37	47.85	51.76	56.37	47.85	51.76	56.19	47.69	51.59

Table 6: Performances with BoW Features: Performance of the baseline system is marked with †. The best performance is bold-faced. *Indexing* means n-grams. Indexing value is TF·IDF.

		All Words			Noun+Verb		
Feature	Index	F1	F2	F3	F1	F2	F3
Domain	1	57.30	57.02	57.20	56.90	56.96	**57.51**
Concept	2	55.61	55.78	55.22	55.70	55.77	55.54
	1+2	56.84	57.16	57.07	56.09	56.57	56.41
Extended	1	**55.97**	55.86	55.74	55.63	55.13	55.88
Domain	2	53.44	53.98	53.31	53.79	53.82	53.43
Concept	1+2	55.91	55.82	55.50	55.54	55.49	55.30
Topic	1	60.58	**60.62**	60.50	59.65	59.99	59.75
	2	50.94	54.27	53.30	50.99	54.03	52.88
	1+2	59.75	60.11	59.82	58.37	59.48	59.20
Extended	1	59.18	59.09	59.10	59.47	**59.60**	59.40
Topic	2	52.77	53.54	52.11	50.44	51.51	50.03
	1+2	58.73	58.98	58.56	56.52	56.85	56.58

Table 7: Performances with Rich Semantic Features in $\mathcal{F}_\mu$: The best performances in each group are bold-faced.

features decreased performance as they tend to introduce more erroneous instances. Likewise, using all words as well as 1-grams performed better among all various n-grams with few exceptions.

Table 8 and Figure 1 show performance over individual classes and detail of predicted labels. This is the system using TF·IDF valued *1-grams* from all terms with frequency ≥ 2, as this was our best-performing system. Overall, we found that many domain concepts are mislabeled with Programming and Documentation since they are most often used concepts and could be a border concept for terms labeled with other domain concepts. For example, CS and Documentation are often labeled as Programming, while Networking is mislabeled as Documentation.

Finally, we used randomized estimation to calculate whether any performance differences between methods are statistically significant (Yeh, 2000) and found all systems exceeding the baseline system had p-value ≤ 0.05, which indicates significant improvement.

5.2 Semi-supervised Learning

For semi-supervised learning, we evaluated the impact of the size of training data. We observed that despite increasing training data, performance does not significantly improve. However, to compare the performance between supervised and semi-supervised systems, we simulated semi-supervised system with unused training data from FOLDOC. Table 9 shows the performance of semi-supervised learning using two groups of features: *1-gram with frequency $\geq$ 1*, and *1-gram with frequency $\geq$ 1 + Domain Concept*. Note that we did not test with *Topic* as topic IDs change according to the data.

G/P	CS	Document	HW	Jargon	Network	OS	Program	SW	Other
CS	435(40.6)	95(8.9)	140(13.1)	9(0.8)	29(2.7)	23(2.1)	297(27.7)	12(1.1)	31(2.9)
Document	74(3.8)	1101(56.3)	151(7.7)	43(2.2)	174(8.9)	33(1.7)	252(12.9)	17(0.9)	111(5.7)
HW	35(3.0)	86(7.4)	850(72.7)	10(0.9)	41(3.5)	18(1.5)	93(8.0)	2(0.2)	34(2.9)
Jargon	23(4.8)	77(16.1)	62(13.0)	145(30.4)	29(6.1)	16(3.4)	74(15.5)	7(1.5)	44(9.2)
Network	19(1.7)	205(18.0)	22(1.9)	12(1.1)	766(67.4)	12(1.1)	58(5.1)	4(0.4)	39(3.4)
OS	14(2.7)	42(8.2)	60(11.7)	11(2.2)	31(6.1)	198(38.7)	130(25.4)	6(1.2)	19(3.7)
Program	51(2.7)	86(4.5)	48(2.5)	24(1.3)	38(2.0)	21(1.1)	1576(83.3)	7(0.4)	42(2.2)
SW	32(4.6)	94(13.4)	48(6.8)	7(1.0)	52(7.4)	27(3.8)	302(43.0)	73(10.4)	67(9.5)
Other	72(5.8)	120(9.7)	109(8.9)	37(3.0)	60(4.9)	15(1.2)	260(21.1)	16(1.3)	542(44.0)

Table 8: Confusion Matrix with *Noun+Verb:1-gram+F2+Topic* where the proportion is presented in ()

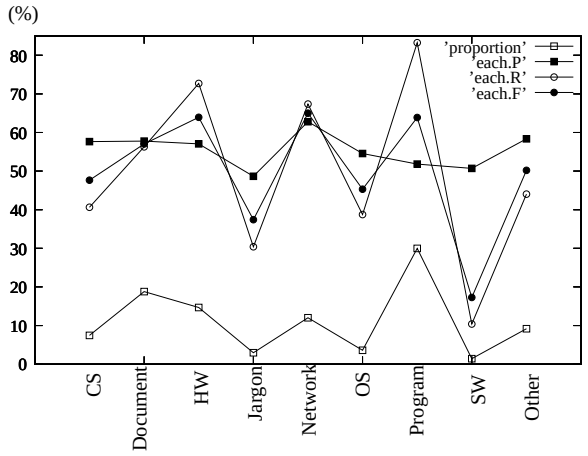

Figure 1: Performances over individual classes with *Noun+Verb:1-gram+F2+Topic*: *Proportion* means the amount of instances per domain concept in %.

	S	Semi-S (Unlabeled Term #)			
Fea.	0	1000	2000	3000	4415
1	52.01	56.94	58.05	57.97	58.86
1+D	57.30	**59.33**	58.02	57.83	57.17

Table 9: Performances by semi-supervised learning in %. *1* is unigram and *D* means domain concept.

The results show that the use of simple BoWs improves performance but does not exceed the best performance produced using *BoW+Domain Concept/Topic*. On the other hand, adding *Domain Concept* actually decreases performance when adding more unlabeled data, except when adding a small amount (i.e., 1000). We found that *Domain Concept* is sensitive, thus this decreased the overall performance when it includes more noise by semi-supervised learning. Previously, the outcomes of semi-supervised learning have shown that its effectiveness is somewhat dependent on the nature of the task and aspects of features. There have been mixed reports on improvement by semi-supervised learning; some work reported significant improvement while other showed little or no impact on the task. In this paper, we observed that semi-supervised learning would not help improve the performance on classifying domain concepts. We expect that since the best performance by the supervised system is not high enough, adding automatically assigned data as training data introduces further error.

6 Conclusion

We have proposed an automatic method to assign domain concepts to terms in FOLDOC using various contextual features as well as semantic features — *Domain Concept* and *Topic*. We demonstrated that the system performed best when using rich semantic features directly derived from dictionary terms. We also showed that for the target task, semi-supervised learning did not significantly improve performance, unlike for other tasks. As future work, we are interested in applying the proposed method to other existing resources in order to build a larger domain-specific ontology resource.

References

Timothy Baldwin, Su Nam Kim, Francis Bond, Sanae Fujita, David Martinez, and Takaaki Tanaka. 2008. Mrd-based word sense disambiguation: Further extending lesk. In *Proceedings of 3rd International Joint Conference on Natual Language Processing*, pages 775–780, Hyderabad, India.

David Blei, Andrew Ng, and Michael Jordan. 2003. La-

tent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Philipp Cimiano and Johanna Vlker. 2005. Text2onto - a framework for ontology learning and data-driven change discovery. In *10th International Conference on Applications of Natural Language to Information Systems (NLDB)*, pages 227–238.

Patrick Drouin. 2004. Detection of domain specific terminology using corpora comparison. In *Proceedings of the fourth international Conference on Language Resources and Evaluation*, pages 79–82, Lisbon, Portugal.

Christiane Fellbaum, editor. 1998. *WordNet, An Electronic Lexical Database*. MIT Press, Cambridge, Massachusetts, USA.

Annette Hulth and Beata B. Megyesi. 2006. A study on automatically extracted keywords in text categorization. In *Proceedings of 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 537–544, Sydney, Australia.

Edward Ivanovic. 2005. Dialogue act tagging for instant messaging chat sessions. In *Proceedings of the ACL Student Research Workshop*, pages 79–84, Ann Arbor, USA.

Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of European Conference on Machine Learning*, pages 137–142.

Jun'ichi Kazama and Kentaro Torisawa. 2007. Exploiting Wikipedia as external knowledge for named entity recognition. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 698–707.

Mitsuhiro Kida, Masatsugu Tonoike, Takehito Utsuro, and Satoshi Sato. 2007. Domain classification of technical terms using the web. *Systems and Computers*, 38(14):2470–2482.

Su Nam Kim, Timothy Baldwin, and Min-Yen Kan. 2009. An unsupervised approach to domain-specific term extraction. In *Australasian Language Technology Association Workshop*, pages 94–98.

Michael Lesk. 1986. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the 1986 SIGDOC Conference*, pages 24–26, Ontario, Canada.

Bernardo Magnini, Carlo Strapparava, Giovanni Pezzulo, and Alfio Gliozzo. 2002. Using domain information for word sense disambiguation. *Natural Langauge Engineering*, 8(4):359–373.

David Milne, Olena Medelyan, and Ian H. Witten. 2006. Mining domain-specific thesauri from wikipedia : A case study. In *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 442–448, Washington, USA.

Guido Minnen, John Carroll, and Darren Pearce. 2001. Applied morphological processing of English. *Natural Language Engineering*, 7(3):207–223.

Robert Navigli and Paola Velard. 2004. Learning domain ontologies from document warehouses and dedicated web sites. *Computational Linguistics*, 30(2):151–179.

Antonio De Nicola, Michele Missikoff, and Roberto Navigli. 2009. A software engineering approach to ontology building. *Information Systems*, 34(2):258–275.

Youngja Park, Siddharth Patwardhan, Karhik Visweswariah, and Stephen C. Gates. 2008. An empirical analysis of word error rate and keyword error rate. In *Proceedings of International Conference on Spoken Language Processing*, Brisbane, Australia.

Janardhana Punuru and Jianhua Chen. 2007. Learning for semantic classification of conceptual terms. *Granular Computing, IEEE International Conference on*, 0:253.

German Rigau, Horacio Rodriguez, and Eneko Agirre. 1998. Building accurate semantic taxonomies from monolingual mrds. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 2*, pages 1103–1109.

Leonardo Rigutini, B. Liu, and Marco Magnini. 2005. An em based training algorithm for cross-language text categorization. In *Proceedings of the Web Intelligence Conference (WI)*, pages 529–535, Compiegne, France.

Vikas Sindhwani and S. Sathiya Keerthi. 2006. Large scale semi-supervised linear svms. In *Annual ACM SIGIR Conference on Research and Development in Information Retrieval(SIGIR)*.

Jorge Vivaldi and Horacio Rodrguez. 2010. Finding domain terms using wikipedia. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC2010)*.

Alexander Yeh. 2000. More accurate tests for the statistical significance of result differences. In *International Conference on Computational Linguistics (COLING)*, pages 947–953.