Ranking the Importance of Ontology Concepts Using Document Summarization Techniques

Youngho Kim, Petros Zerfos, Vadim Sheinin, Nancy Greco
IBM T.J. Watson Research Center, Yorktown Heights, NY
Email: younghokim927@gmail.com, {pzerfos, vadims, grecon}@us.ibm.com

Abstract

Automated Ontology Learning systems are nowadays practical and used in a variety of domains. By using these systems, subject matter experts (SMEs) and ontology designers can readily construct very large ontologies consisting of tens of thousands of concepts and their relations based on a corpus. However, ontologies of this size make it extremely challenging for such SMEs to understand and further tune these ontologies. Prior studies have proposed techniques for concept ranking based solely on the analysis of the structure of the ontology graphs. In this paper, we propose a novel approach, which further exploits a word-level summarization technique applied to the source documents used to generate the ontology. Using the document summarization technique, we devise features that measure concept importance based on source documents where concepts are extracted. We demonstrate the effectiveness of our approach by comparing with existing ranking methods and by devising a scalable evaluation process inspired from the document retrieval domain.

1 Introduction[1][2]

Automated Ontology Learning (AOL) from text is the task of creating ontologies without human involvement, based on input documents related to the domain of interest (Cimiano et al., 2006; Wong et al., 2012). The process for AOL typically includes the steps of domain terminology extraction, word sense disambiguation, concept discovery, relation learning, etc. (Cimiano et al., 2006). Enabled by recent advances in related research areas such as Natural Language Processing, Information Retrieval (IR), and Machine Learning, practical systems for automated ontology learning (e.g., ASIUM (Faure and Poibeau, 2000), OntoLearn (Navigli and Velardi, 2004), Text-to-Onto (Cimiano et al., 2005)) have finally become a reality.

Using AOL systems, subject matter experts and ontology designers can trivially create large-scale ontologies, consisting of tens of thousands of concepts and their relations. This, in turn, makes it very challenging for (human) users to explore and understand these ontologies. Traditional information visualization tools (e.g. IsaViz, Stanford Protege) facilitate navigation in complex ontology graphs, however the time and effort required to understand a very large ontology through visualization alone is still prohibitively high. *Ontology Understanding* by human users can be assisted through computational intelligence (Wu et al., 2008).

To help with ontology understanding, methods that rank concepts according to their importance in an ontology have previously been studied (Alani et al., 2006; Graves et al., 2008; Wu et al., 2008). Most of the existing techniques draw inspiration from the PageRank algorithm (Brin and Page, 1998) (e.g., Patel et al., 2003; Ding et al., 2004; Wu et al., 2008), since they typically consider an ontology as a directed graph and propagate the relative importance of concepts (i.e., vertices) via their associated relations (i.e., edges). Following this approach, high authoritative concepts are deemed more important.

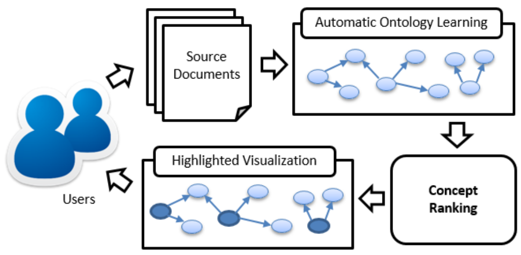

Figure 1: Concept Ranking Scenario

Prior work has considered ranking of concepts derived from ontologies that have been *manually* created by subject matter experts. In contrast, in

[1] Research performed while author Youngho Kim was with IBM T.J. Watson Research Center. He is now with Google, Inc.
[2] Contact author is: Petros Zerfos, pzerfos@us.ibm.com

our scenario (Fig. 1), we assume that the user has collected some (source) documents of interest, and the corresponding ontology is created based on these documents, through the use of an AOL system. Our focus is on investigating the problem of ranking the importance of concepts defined in this *automatically* generated ontology, which helps in understanding both the ontology as well as the source documents. Note that we use the terms “source documents” and “target documents” interchangeably to refer to the documents that the user provides as input to the AOL-based ontology generation process.

We hypothesize that a concept is potentially “important” if it is effective in representing most of the other concepts in an ontology and our goal is to find a relatively small set of concepts, which includes most of the key ones in the generated ontology. Our key insight is that the aforementioned task is, in fact, the main goal of *multi-document summarization* (e.g., Goldstein et al., 1999; Lawrie, 2001); for given target documents, summarization techniques identify key sentences (or words) to represent the documents. Thus, we adapt a document summarization technique to our concept-ranking task and, through experiments (Section 5), show that this technique is effective in identifying important concepts, compared to how they would have been manually judged by humans.

To apply this key insight, we devise novel metrics to measure the importance of each concept. While most previous work measure “importance” based solely on ontology structures (e.g., authority in PageRank and centrality within ontology graphs (Graves et al., 2008))), we propose new features that make use of the source documents where the ranked concepts are extracted from. Specifically, we assume that “important” concepts involve high topicality and are widely covering the others in the source documents (see Section 4 for more details). These document-based features are not considered in previous work, due to the basic assumption that they follow, which postulates that the ontology has been generated and curated by humans. Once the importance of each concept is estimated using our metrics, we apply a simple greedy algorithm to generate a concept-level summary (i.e., a ranked list of concepts). This algorithm iteratively selects the most representative (i.e. potentially important) concept so that the overall importance value of the selected concepts is maximized.

To evaluate our approach, we conduct manually judged experiments whose results are compared with prior concept ranking algorithms (e.g., PageRank and CARRank), as well as with our approach. In addition, to provide a more scalable and robust evaluation, we further design large-scale experiments relying on document retrieval techniques. Given a set of target (source) documents, we generate a concept ranking result, and examine whether the top ranked concepts are effective in retrieving the target documents out of a million documents. In other words, the target documents are successfully retrieved if the ranking method selects really “important” concepts.

In summary, the main contributions of our work are as follows.

1) We cast new light to the problem of ontology concept ranking for ontologies that were *automatically* generated from a corpus of source documents and suggest the use of a document summarization technique on these source documents to solve this problem.
2) We estimate the importance of concepts by using features extracted from the source documents where the concepts are discovered. Such source document-based features have seldom been investigated in prior work, which instead focused on the analysis of ontology graph structure (e.g., PageRank).
3) We perform human-labeled evaluation as well as more scalable experiments aided by document retrieval techniques. While almost every previous study relies on a manually labeled data set covering a few ontologies, we design a scalable experiment that can examine many different ontologies without requiring human effort; in the experiments, we examine 50 ontologies.

The remainder of this paper is organized as follows. Section 2 describes relevant related work. In Section 3, we provide background of our automated ontology learning system. Section 4 proposes our method to rank concepts, and Section 5 provides the evaluation of our approach. Section 6 concludes this paper.

2 Related Work

2.1 Ranking Concepts in Ontologies

On the Semantic Web, ranking ontologies relevant to user queries is important to retrieve relevant information (e.g., Alani et al., 2007; Ding et al., 2005). Without user queries, ranking ontology concepts has been researched to improve *Ontology Understanding* (e.g., Wu et al.,2008).

In the literature, PageRank (Brin and Page, 1998) has much influenced on many methods (e.g., OntoKhj (Patel et al., 2003) and HITS (Kleinberg, 1999)). Typically, these approaches weight on high authoritative nodes in ontology graphs. As an extension of PageRank, Wu et al., (2008) have proposed CARRank that performed the reverse mechanism of PageRank. In addition, Zhang et al., (2006) have used PageRank with textual scores of words by finding (virtual) documents related to a target ontology. Moreover, Graves et al., (2008) have considered more central concepts as "important" because they can relatively easily access the others in ontology graphs. More recently, Chen et al. (2010) have propagated the importance of concepts by only semantically correct relations.

2.2 Multi-document Summarization

Multi-document summarization can be viewed as the problem of selecting sentences (or words) to represent target documents (Goldstein et al., 1999). While sentence-level summaries are considered more reliable and easily readable, word-level summarization can include intuitive keywords and provide a succinct representation of target documents (Lawrie et al., 2003). Lawrie et al. (2001) has proposed a hierarchical summarization method for identifying topic words. According to that, they iteratively choose the words which are highly probable in language models derived from the target documents and frequently co-occurring with other vocabulary. In this paper, we exploit this technique for our concept ranking problem, and incorporate with the features that can capture the importance of the target concepts.

3 Background: Automated Ontology Learning

In this section, we provide a brief summary of our Automated Ontology Learning system for generating a target ontology. As described in Section 1, users provide the (source) documents of interest, and the target ontology is automatically created from these documents. We use an automated ontology learning system that we built in-house, which can handle a large-scale corpus and systematically discover concepts and their relations. Note that the selection of that system was based solely on practical considerations; the ontology concept ranking techniques that we describe in this work are *general* and *independent* of the AOL system that is chosen, and can be readily applied using any other publicly available AOL system (e.g., Velardi et al., 2013)).

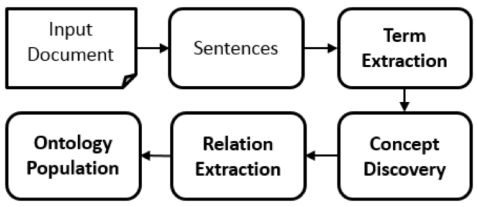

Figure 2: Ontology Learning Workflow

Given a set of documents, the main steps to generate an ontology are as follows:

- **Step-1:** Term Extraction & Word Sense Disambiguation
- **Step-2:** Concept Discovery
- **Step-3:** Relation Extraction
- **Step-4:** Ontology Population

Figure 2 depicts this process. For each document, we first recognize sentence boundaries, and then process each sentence to extract concepts and relations. The first step is identifying terms that appear in each sentence. We use a tokenizer and lemmatizer to recognize tokens and their most probable lemma (i.e., normalized) forms. Then we apply the Part-Of-Speech tagging and Word Sense Disambiguation (WSD). For WSD, we map each word to the most probable synset in WordNet (Miller, 1995). We assume that a word in a sentence includes only one sense ("one sense per word hypothesis"), and the similarity between sentence terms and each synset's gloss is measured for the mapping. Then, dependency parsing is performed to identify a syntactic structure for each sentence.

Based on the above, the next step is to discover concepts that appear in each sentence. Typically, concepts can be any abstract or concrete objects. We syntactically define a concept as any possible noun phrase along with dependency relations (recognized by the parser). For example, from a dependency tree like:

(((JJ *large*) (CC *and*) (JJ *heavy*)) → (NN *dog*))

we can discover three concepts: "dog", "large dog" and "heavy dog". For each extracted concept, we use its lemmatized form or synset id as a key, and consolidate two concepts into one if they share the same key. Once concepts are discovered, we continue processing the sentence to identify taxonomic (i.e., hierarchical) relations within the identified concepts. We identify three main types of taxonomic relations:

i) **SubclassOf** is a dependency-based relation which directly connects a parent concept to its child. As an example, a partial dependency tree is given: (JJ *heavy*) → (NN *dog*), we discover two concepts, "heavy dog" and "dog", and "heavy dog" is a sub-class of "dog".

ii) **InstanceOf** denotes a "is-a" relation between two concepts, e.g., "John" is a "student". To identify this relation, we use lexical syntactic patterns like (subject [concept] and object [concept] depend on a verb ["call"]).

iii) **Hypernym** indicates a hierarchical relation between a more general concept (i.e., hypernym, e.g., "color") and a more specific concept (i.e., hyponym, e.g., "red"). We create the relation if two concepts are connected via "hypernym-hyponym" in WordNet.

The AOL system can extract non-taxonomical relations (e.g., "association", "casuality", etc.) as well based on the predicate of the sentence. However, in this paper, we explore using only taxonomic relations because most extracted relations are taxonomic, and utilizing non-taxonomic relations is left for future work.

After identifying concepts and their relations, we populate a target ontology by generating triples. We define a triple as a directional association of two concepts by a taxonomic relation, i.e., $\langle c_i \underset{r}{\rightarrow} c_j \rangle$ where c_i is a child ("dog") concept, c_j is a parent ("animal") concept, and r ∈ {SubclassOf, InstanceOf, Hypernym}. To build an ontology, we create a root concept (i.e., namely "thing"). Then, every triple whose parent concept does not appear in a child concept of any other triple is attached to the root concept as a child. Furthermore, we connect other triples following the hierarchical relations, and finally a large single concept hierarchy is generated. Note that the root concept is not used in concept ranking.

4 Ontology Concept Ranking

In this section, given an ontology generated from Section 3, we first generate a directed graph representing the ontology. Then, we describe our ranking model based on this graph.

4.1 Ontology Graph

In the literature, graph-based ontology representation is commonly used for identifying important concepts (e.g.,Wu et al., 2008; Graves et al., 2008). Since our ontology consists of triples, graph representation is straightforward. We define this ontology graph as follows.

Definition 1. Given an ontology O, the ontology graph $G = (V, E)$ of O is defined as a directed graph where V is the set of vertices representing all discovered concepts in O and E is a set of directed edges representing all relations in O.

4.2 Concept Ranking Model

4.2.1 Problem Formulation

Multi-document summarization aims at extracting key text fragments that effectively represent the whole target text. Given target documents, this technique focuses on selecting key sentences (or words) to represent (i.e., summarize) the target documents. Inspired from this, we exploit such keyword selection process for identifying (potentially) important concepts in the ontology. We formulate this ranking problem as follows.

Given an ontology graph G containing a set of concepts (as vertices) V, our task is to find a subset of V, which ideally contains all important concepts to summarize G. We view this problem as a variant of the Dominating Set Problem as:

Let E be a set of edges (i.e., relations) in G, and we generate a sub-set of V so that for every vertex not in S, an edge from this vertex to any one in V exists and the weight of vertices (i.e., the importance of concepts) in S is maximized. In a formal notation, find $S \subseteq V$ such that, $\exists\{u, v\} \in E$ for $\forall u \in V - S$ and $\exists v \in S$ where $u \neq v$, and $\underset{S}{\operatorname{argmax}} \sum_{v \in S} w_v$ where w_v is an importance value of v.

Note that the original problem in (Garey and Johnson, 1979) is minimizing the number of the vertices in S, and (Lawrie et al., 2001) also applied this dominating set problem to identifying keywords of an input text by generating word-to-word graphs.

Solving this problem, we obtain the dominating set S, which can be interpreted as an efficient concept-level summary for G. In the next section, we describe how to solve this problem.

4.2.2 Concept Ranking Algorithm

The Dominating Set Problem is NP-hard, so it is not easily solvable. However, a simple but effective greedy solution is proposed in (Lawrie et al., 2001). Based on this, we provide a greedy concept ranking algorithm to find S.

The concept ranking algorithm is depicted in Figure 3. This algorithm takes the ontology graph $G = (V, E)$ and the source documents D (used for generating G) as inputs. For each concept ($v \in V$), we first identify its children concepts (line 2). Since we only use hierarchical relations, every edge in G indicates a parent-child relationship between two concepts. Then, we estimate the importance of v in two different dimensions: *topicality* and *coverage*. The topicality (TP) means some topical importance of v based on D, i.e., how much important is v with respect to D (line 3). The coverage (CV) denotes how strongly children are related to v, i.e., a weight of the edge incoming to v (line 4). We provide more details of these measures in Section 4.2.3.

Next, we initialize S, the ordered list where "important" concepts will be added; the insertion order denotes the rank of each concept (line 6). Besides, DC is the set that will hold the concepts dominated (i.e., covered) by the concepts in S (line 6). As we select dominated concepts, the weak relation between child and parent concepts is not sufficient proof that the child concept is "truly" covered by its parent concept. So, we validate by imposing a threshold; we define a relation-dependent threshold by taking a mean of every edge weight (line 8).

```
Algorithm 1. Concept Ranking
Input:
$D$ is the set of source documents to construct an ontology $O$.
$G = (V, E)$ is the ontology graph based on $O$ where $V$ is the set of concepts and $E$ is the set of hierarchical relations.
Output:
$S$ is the ranked list of important concepts.
Process:
1:  foreach $\forall v \in V$ do
2:      set $\mathrm{ch}(v) = \{u \in V | \exists \{u, v\} \in E\}$
3:      compute $\mathrm{TP}(v)$
4:      compute $\mathrm{CV}(v) = \sum_{u \in \mathrm{ch}(v)} w_{\{u,v\}}$
5:  endfor
6:  $S \leftarrow \emptyset$
7:  $DC \leftarrow \emptyset$
8:  $\theta = \mathrm{mean}\left(w_{\{v_i, v_j\}}\right)$ where $\forall \{v_i, v_j\} \in E$
9:  while $S \subset V$ and $|V| > 0$ do
10:     $\hat{v} \leftarrow \underset{v \in V}{\mathrm{argmax}}\, \mathrm{TP}(v) \times \mathrm{CV}(v)$
11:     Add $\hat{v}$ into the last position of $S$
12:     $V \leftarrow V \setminus \{\hat{v}\}$
13:     $T \leftarrow \emptyset$
14:     foreach $u \in \mathrm{ch}(\hat{v})$
15:         if $w_{\{u,\hat{v}\}} > \theta$ then
16:             $T \leftarrow T \cup u$
17:         endif
18:     endfor
19:     foreach $u \in T \setminus DC$ do
20:         foreach $v \in V$ do
21:             $\mathrm{CV}(v) \leftarrow \mathrm{CV}(v) - w_{\{u,v\}}$
22:         endfor
23:     endfor
24:     $\mathrm{DC} \leftarrow \mathrm{DC} \cup \mathrm{T}$
25: endwhile
26: return $S$
```

Figure 3: Concept Ranking Algorithm

Based on these, we now select important concepts until every other concept is dominated (covered) by the selected ones. We iteratively select the most important concept $\hat{v}$ by calculating the product of its topicality and coverage values (line 10). After adding $\hat{v}$ into S, we select dominated children for $\hat{v}$ based on the threshold (line 13-18). Then, we update the coverage values for the other concepts since the dominated children are not necessarily considered further (line 19-23). Finally, we store the dominated concepts for the next iteration, and return the list of dominating concepts after all iterations.

In terms of complexity, this algorithm performs in $O(|V|^3)$. We can cut-off some very rarely observed concepts (e.g., only once appeared in D) in V if too many concepts are given.

4.2.3 Concept Importance Measures

For our concept ranking algorithm (Fig. 3), we estimate the importance of each concept in two different dimensions: *topicality* and *coverage*. We describe each dimension as follows.

Topicality: the topicality of a concept measures the extent of how informative the concept is to describe the target ontology O. To measure this, we adopt a language modeling approach (Ponte and Croft, 1999) using the source documents D. In other words, topically important concepts in D may be also important in O. Specifically, we first generate concept-specific language models from D. We define one large pseudo-document L_D by concatenating every source document (i.e., $L_D = \bigcup_{d \in D} d$). For each concept cpt, we compute its generative probability based on L_D as:

$$\mathrm{P}(cpt|L_D) \approx \prod_{w \in cpt} \mathrm{P}(w|L_D)$$

where w is a word that appears in cpt.
Since a concept can contain multiple words, we use unigram language models to estimate the whole concept's probability. The unigram language model is given as:

$$\mathrm{P}(w|L_D) = \frac{\mathrm{freq}(w: L_D)}{|L_D|}$$

where $\mathrm{freq}(w: L_D)$ is the word frequency in L_D and $|L_D|$ is a word-level document length.

Then, the topicality $\mathrm{TP}(cpt)$ of a concept is estimated as its contribution to the Kullback-Leibler divergence between this concept-specific model and a general language model:

$$\mathrm{TP}(cpt) = \mathrm{P}(cpt|L_D) \cdot \log_2 \frac{\mathrm{P}(cpt|L_D)}{\mathrm{P}_c(cpt)}$$

where $\mathrm{P}_c(cpt)$ is a general probability of cpt. The general probability can be estimated from a general English corpus (e.g., Google n-gram) or some large document corpus in the same domain of the source documents (e.g., academic paper repository if the user inputs academic papers). By doing this, we can identify "important" concepts that are highly generative from D and relatively less common in general. This divergence-based measure is typically used in the area of Information Retrieval (Cronen-Townsend, 2002).

Coverage: the coverage of a concept denotes the extent to which the concept can represent its children concepts. Note that the ontology contains only hierarchical relations (i.e., {Hypernym, InstanceOf, SubclassOf}). We assume that a parent concept strongly related to its children concepts would effectively represent the children. In other words, more widely covering concepts are more "important". To capture this, given a concept, we collect its every child, and estimate the weight of the edge (i.e., relation) from the child to its parent. This relation weight denotes how strongly a parent concept can subsume its children. To estimate this, we use the source documents. Given a relation from a child concept u to a parent concept v, its weight $w_{\{u,v\}}$ is calculated as:

$$w_{\{u,v\}} \approx \mathrm{P_z}(v|u) = \frac{\mathrm{freq_z}(v,u)}{\mathrm{freq_z}(u)}$$

where z is the size of a word window and $\mathrm{freq_z}(u)$ is the number of windows containing u in the source documents.

We use the conditional probability within $\{u, v\}$ as the weight, derived from D. So, the child u highly co-occurring with its parent v in D is considered as "covered" by v. Note that, alternatively, point-wise mutual information can be used, but to capture the "subsumption" of children concepts, the conditional probability looks intuitively more effective. In experiments, we empirically set z as 20, which can vary up to the number of source documents.

5 Evaluation

For evaluation, we choose the patent domain as our target domain, and automatically generate a patent ontology by applying the AOL system (Section 3) to patent documents. After generating the ontology, we conduct two different experiments: (1) Ranking Quality Experiment and (2) Retrieval Experiment.

5.1 Ranking Quality Experiment

5.1.1 Experimental Setup

Ontology Generation. To generate a target ontology, we first collect a set of source documents. For this, we use "coffee process" as a query, and retrieve relevant patents from a patent search engine. Then, we manually select 15 "relevant" patents used for building a "coffee process" patent ontology. As a result, we extract 1,934 concepts with 2,011 (hierarchical) relations. Based on these, we perform concept ranking algorithms, and evaluate final ranked results.

Evaluation Metrics. To measure the quality of concept ranking results, we can employ a number of IR metrics used for evaluating ad-hoc retrieval methods (e.g., document ranking algorithms). Among them, we use Precision, NDCG (Normalized Discounted Cumulative Gain (Jarvelin and Kekalainen, 2002), and Recall, which are typically used in ad-hoc retrieval tasks. To use these metrics, we basically need a rank result (i.e., a list of ranked concepts) and a set of labeled instances (i.e., a list of "important" concepts). We describe how to obtain these labeled instances later in this section.

Baseline Ranking Methods. As baselines, we employ two existing concept ranking algorithms: 1) PageRank (Brin and Page, 1999) and 2) CARRank (Wu et al., 2008). We choose PageRank since it has been widely used in the ontology ranking problem domain and inspired other ranking methods (e.g., OntoKhoj (Patel et al., 2003) and Swoogle (Ding et al., 2004)). In addition to this, we employ CARRank that has been shown as outperforming several ranking methods including PageRank. Thus, we compare our ranking approach with these baseline methods.

Importance Label Judgment. To obtain labeled concepts, we employ two human annotators who are graduate students and can understand the content of "coffee process" patents. We first ask each annotator to read all 15 patents. Then, we show the list of all concepts for determining which concepts are more "important" than the others, i.e., binary judgment whether each concept is "important" or not. For evaluation, we obtain 54 "important" concepts where both annotators agree on the judgment.

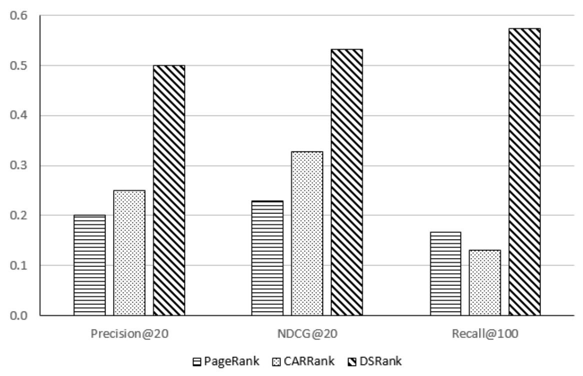

Figure 4: Ranking Quality Comparison

5.1.2 Ranking Quality Results

We verify the quality of the ranked result obtained by each method. Figure 4 shows the evaluation result by Precision at top 20, NDCG at top 20, and Recall at top 100. In that, PageRank and CARRank denote the baseline methods, and DSRank means our dominating-set-based ranking algorithm (Algorithm 1 in Fig. 3).

Rank	PageRank	CARRank	DSRank
1	group	**product**	**coffee**
2	meat	collection	berry
3	**beverage**	**coffee**	aroma
4	**coffee**	aroma	**coffee bean**
5	**powder**	coffee formulation	**coffee beverage**
6	nutrient	organism	**process**
7	liquid	distribution	acid
8	acid	calibration solution	Solution
9	part	**grinding**	**coffee concentrate**
10	**product**	dry ingredient	**extraction**

Table 1: Ranked Concept Examples

First, our ranking algorithm can outperform the baseline algorithms in every metric. That is, DSRank can effectively select "important" concepts. While the baseline methods only use relations to propagate relative importance, we use topical importance directly estimated from the source documents. This difference looks significant to find "important" concepts that the users manually identified. Second, DSRank dramatically performs better in terms of Recall. Comparing to early performance metrics (i.e., Precision @20 and NDCG@20), the improvement in Recall@100 means that the users can obtain more important concepts as they observe more ranks. Third, CARRank performs slightly better than PageRank within top 20 ranks. However, observing top 100 ranks, PageRank can find slightly more important concepts than CARRank.

We further analyze the quality of ranking results using some examples. Table 1 shows the top 10 concepts ranked by each method. In that, we indicate manually-judged "important" concepts by bold typeface. A first observation is that PageRank is effective in finding very general concepts, which would contain more children (e.g., "beverage" and "product"). This is because PageRank generally promotes high "authority" (i.e., containing more children) concepts. However, this algorithm fails to place topically important concepts (e.g., "coffee bean" and "coffee concentrate") at higher ranks. A second observation is that CARRank can find not only general concepts, but also some specific concepts (e.g., "coffee formulation" and "calibration solution"). This is because CARRank weights the relations started from "hub" concepts (containing more outgoing edges). However, this algorithm is also missing topically important concepts. Third, different from these, our DSRank method looks effective in finding topically important concepts (e.g., "coffee bean", "coffee beverage", and "coffee concentrate" as well as generally important concepts (e.g., "process" and "extraction"). These concepts seem to be useful to express key information about "coffee process".

5.2 Retrieval Experiment

We now conduct a more robust and scalable experiment. Although the previous evaluation directly verifies the ranking quality by human-labeled examples, due to the manual judgment, this experiment is limited in testing with many more ontologies generated from different documents. To overcome this limitation, we design a document retrieval-based evaluation. For that, we hypothesize that "important" concepts can represent not only a target ontology but also its source documents, and a search query containing such concepts should be effective in retrieving all the source documents. In other words, we set source documents as "relevant" documents, and generate a concept-based query for retrieving the source documents. Figure 5 depicts this workflow.

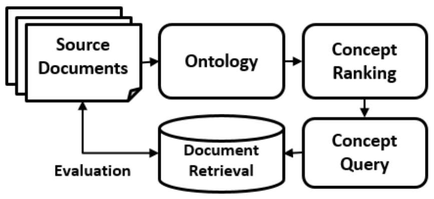

Figure 5: Retrieval Experiment Workflow

5.2.1 Experimental Setup

Test Collection: As a test collection, we use the USPTO (United States Patent and Trademark Office) patents provided by NTCIR-6 (Fujii et al., 2006). This collection includes 981,948 patents published from 1993 to 2000, and provides a pool of topics (as initial search queries) and the list of "relevant" patents for each topic. To develop the sets of source documents, we randomly select 50 topics, and use their relevant patents as the source documents. Accordingly, we automatically construct 50 different topic-specific ontologies. The average number of the source documents over the 50 topics is 14.4.

Baseline Methods: As done in the previous experiment, we use PageRank and CARRank as baselines in this evaluation. In addition to this, we adopt MEAD, a text summarization method described in (Radev et al., 2004), to validate the retrieval performance gained by concept queries. MEAD directly generates a summary from the source documents without the ontology, and the words appeared in the summary are used in a search query. The reason to employ MEAD is that we compare our concept-based summary with a traditional sentence-based summary in terms of retrieval performance.

Retrieval Model: For retrieval, we use the Indri search engine (Strohman et al., 2005) and the query likelihood model (Ponte and Croft, 1999) is employed. Since we measure the retrieval performance purely affected by the selection of query terms (i.e., by concept ranking algorithms), more advanced query models that influence on the retrieval by query term weighting (e.g., Relevance Models (Lavrenko and Croft, 2001) and Dependence Models (Metzler and Croft, 2006)) are not considered.

Search Query Generation: For each topic, we generate a concept-based search query by selecting the top-k concepts among all ranked concepts. In the experiment, we consider k values ranging from 20 to 200 with an increment of 20, and select an optimal k value (see Fig. 6). Then, we use MEAD to generate a sentence-level summary with the same size to the concept query, and perform a summary-based retrieval by the same retrieval model. Note that we remove basic stop-words (e.g., prepositions) in each query.

Evaluation metrics: Since we attempt to examine the effectiveness of ranked concepts by retrieving the source documents for each ontology, the rank of each retrieved source document is less important. In other words, the queries would be effective if all source documents are retrieved within some reasonable rank point (e.g., top 20). Thus, instead of rank-sensitive metrics (e.g., NDCG), we largely use Recall at top 20 and 100 for evaluation. Besides, MAP (Mean Average Precision) is employed to identify overall performance of retrieval results.

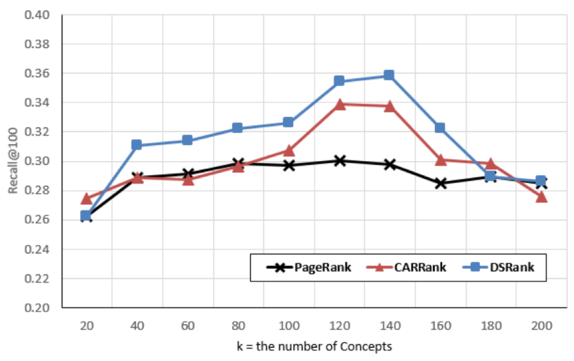

Figure 6: Recall@100 by top k concepts

5.2.2 Retrieval Results

We evaluate our concept ranking method (DSRank) by comparing with various baselines (PageRank, CARRank, and MEAD). We first identify how many concepts are effective in retrieving the source documents. Figure 6 shows the recall performance at the top 100 documents obtained by the concept queries using various numbers of concepts (i.e., top-k). First, we find that the queries using the top 120 or 140 concepts look more effective than the others. Second, all three concept ranking algorithms behave similarly in the retrieval task. Third, DSRank outperforms both baselines in most cases (from the top 40 to 160 concepts). We provide more detailed analysis of this result later of this section. Note that due to the limited space, the results with other rank points (e.g., top 20) are omitted, but they also showed the same trends.

Next, we evaluate with MEAD. Figure 7 shows the retrieval performance comparison using MEAD. We note that concept queries (i.e., PageRank, CARRank, and DSRank) perform better than MEAD queries (except for the case of PageRank and MEAD by Recall@100).

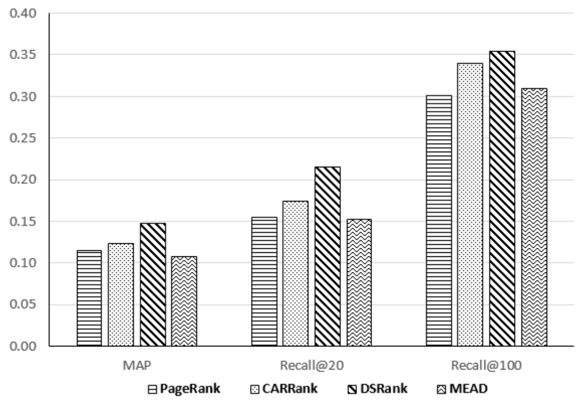

Figure 7: Retrieval Performance Comparison

To validate this, we perform Wilcoxon Rank-Sum test between two series of each metric values with 95% confidence (i.e, $p < 0.05$), and CARRank and DSRank can significantly outperform MEAD in the recall metrics. Thus, the ranked concepts are beneficial to retrieve the source documents. Second, DSRank is more effective in top ranks (i.e., Recall@20, c.f., Recall@100). The statistically significant improvement of DSRank over all baselines is obtained in Recall@20 (using the Wilcoxon Rank-Sum test). This means that more "relevant" (i.e., source) documents are more efficiently retrieved by our method in early ranks. Overall, from this scalable experiment, we identify the retrieval effectiveness of the concepts obtained by our approach over various baselines.

6 Conclusion

In this paper, we described the problem of ranking ontology concepts extracted automatically from a corpus of source documents through the use of an Automated Ontology Learning (AOL) system. To solve this, we generate concept-level summaries to identify more important concepts. Comparing with existing ranking approaches, we introduce a novel summarization-based technique and further exploit the features that measure *topicality* and *coverage* as estimated based on the source documents. These techniques have not been explored in previous studies. Through extensive experiments, we demonstrate the effectiveness of our method in terms of ranking quality and retrieval effectiveness.

Our approach is based on the source documents and an AOL-based ontology generated from these documents. However, this technique can be easily extensible to handle well-known ontologies manually generated for target domains (e.g., Travel ontology and MeSH[3]) by retrieving the source documents relevant to the descriptions of concepts that have been used as input by the subject matter experts. Moreover, we can generate virtual documents matched with ontologies as done in (Qu et al., 2006). As future work, we will examine our approach using non-taxonomical relations, and test with other domains (e.g., legal documents).

References

Harith Alani, Christopher Brewster, and Nigel Shadbolt. Ranking Ontologies with AKTiveRank. In *Proceedings of the International Semantic Web Conference (ISWC'06), LNCS,* 4273:1-15.

Kemafor Anyanwu, Angela Maduko, and Amit Sheth. (2005). In *Proceedings of the International Conference on World Wide Web (WWW'05)*, pages 117-127.

Sergey Brin and Lawrence Page. (1998). The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7): 107-117.

Paul Buitelaar, Thomas Eigner, and Thierry Declerck. (2004). OntoSelect: A Dynamic Ontology Library with Support for Ontology Selection. In *Proceedings of the Demo Session at the International Semantic Web Conference (ISWC'04).*

Paul Buitelaar and Philipp Cimiano. (2006). Tutorial Notes of the EACL Tutorial on Ontology Learning from Text. *Conference of the European Chapter of the Association for Computational Linguistics (EACL).*

Philipp Cimiano, Andreas Hotho, and Steffen Staab. (2005). Learning Concept Hierarchies from Text Corpora using Formal Concept Analysis. *Journal of Artificial Intelligence Research (JAIR)*, 24(1): 305-339.

Philipp Cimiano, Johanna Völker. (2005). Text2Onto - A Framework for Ontology Learning and Data-driven Change Discovery. In *Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems (NLDB), LNCS* 3513: 227-238.

Philipp Cimiano, Johanna Völker, and Rudi Studer. 2006. Onotlogies on Demand? A description of the state-of-the-art applications, challenges, and trends for ontology learning from text. *Information, Wissenschaft und Praxis*, 57: 315-320.

Steve Cronen-Townsend, Yun Zhou, and W. Bruce Croft. (2002). In *Proceedings of the ACM SIGIR*

[3] https://www.nlm.nih.gov/mesh/

Conference on Research and Development in Information Retrieval (SIGIR'02), pages 299-306.

Li Ding, Rong Pan, Tim Finin, Anupam Joshi, Yun Peng, and Pranam Kolari. (2005). Finding and Ranking Knowledge on the Semantic Web. In *Proceedings of the International Semantic Web Conference (ISWC'05)*, *LNCS*, 3729: 156-170.

Radev, Dragomir, Timothy Allison, Sasha BlairGoldensohn, John Blitzer, Arda C¸ elebi, Stanko Dimitrov, Elliott Drabek, Ali Hakim, Wai Lam, Danyu Liu, Jahna Otterbacher, Hong Qi, Horacio Saggion, Simone Teufel, Michael Topper, Adam Winkel, and Zhu Zhang. (2004). MEAD – A Platform for Multi-document Multilingual Text Summarization. In *Proceedings of Conference on Language Resource and Evaluation (LREC'04).*

David Faure and Claire Nédellec. (1999). Knowledge Acquisition of Predicate Argument Structures from Technical Texts Using Machine Learning: The System ASIUM. In *Proceedings of the European Knowledge Acquisition Workshop (EKAW)*, pages 329-334.

Atsushi Fujii, Makoto Iwayama, and Noriko Kando. (2007). Overview of the patent retrieval task at the NTCIR-6 workshop. In *Proceedings of the sixth NTCIR Evaluation Workshop (NTCIR-6).*

Jade Godstein, Mark Kantrowitz, Vibhu Mittal, and Jaime Carbonell. (1999). Summarizing Text Documents: Sentence Selection and Evaluation Metrics. In *Proceedings of the ACM SIGIR '99*, pages 121-128.

Alvaro Graves, Sibel Adali, and Jim Hendler. (2008). A Method to Rank Nodes in an RDF Graph. In *Proceedings of the Poster and Demo Track of the 7th International Semantic Web Conference,* pages 82-87.

Kalervo Jarvelin and Jaana Kekalainen. (2002). Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4): 422-446.

Xing Jiang and Ah-Hwee Tan. (2006). OntoSearch: a full-text search engine for the semantic web. In *AAAI 2006 Proceedings of the 21 National Conference on Artificial Intelligence*, 2:1325-1330.

Jon M. Kleinberg. (1999). Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM*, 46(5): 604-632.

Victor Lavrenko and W. Bruce Croft. (2001). Relevance-based Language Model. In *Proceedings of the ACM SIGIR (SIGIR'99),* pages 120-127.

Dawn Lawrie, W. Bruce Croft, and Arnold Rosenberg. (2001). Finding topic words for hierarchical summarization. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'01)*, pages 349-357.

Dawn Lawrie. (2003). Language models for hierarchical summarization. *PhD Thesis, University of Massachusetts.*

Donald Metzler and W. Bruce Croft. (2005). In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'05)*, pages 472-479.

Roberto Navigli and Paola Velardi. (2004). Learning domain ontologies from document warehouses and dedicated websites. *Computational Linguistics*, 30(2):151-179.

Trevor Strohman, Donald Metzler, Howard Turtle, and W. Bruce Croft. (2005). Indri: a language-model based search engine for complex queries (extended version). *University of Massachusetts CIIR Technical Report.*

Jinoh Oh, Taehoon Kim, Sun Park, and Hwanjo Yu. (2012). PudMed search and exploration with real-time semantic network construction. In *Proceedings of the 18th ACM SIGKDD (KDD'12)*, pages 1572-1575.

Chintan Patel, Kaustubh Supekar, Yugyung Lee, E. K. Park. (2003). OntoKhoj: A Semantic Web Portal for Ontology Searching, Ranking and Classification. In *Proceedings of the 5th ACM International Workshop on Web Information and Data Management (WIDM'03)*, pages 58–61.

Jay M. Ponte and W. Bruce Croft. (1998). A language modeling approach to Information Retrieval. In *Proceedings of the ACM SIGIR (SIGIR'98)*

Yuzhong Qu, Wei Hu, and Gong Cheng. (2006). Constructing Virtual Documents for Ontology Matching. In *Proceedings of the 15th International Conference on World Wide Web (WWW'06)*, pages 23-31.

Paola Velardi, Stefano Faralli, and Roberto Navigli. 2013. OntoLearn Reloaded: A graph-based algorithmfor taxonomy induction. *Computational Linguistics*, 39(3): 665-707.

Wilson Wong, Wei Liu, and Mohammed Bennamoun. 2012. Ontology learning from text: a look back and into the future. *ACM Computing Surveys*, 44(4):20.

Gang Wu, Juanzi Li, Ling Feng, and Kehong Wang. (2008). Identifying Potentially Important Concepts and Relations in an Ontology. In *Proceedings of the International Semantic Web Conference (ISWC'08), LNCS*, 5318: 33–49.

Xiang Zhang, Hongda Li, and Yuzhong Qu. 2006. Finding Important Vocabulary Within Ontology. In *Proceedings of the 1st Asian Semantic Web Conference (ASWC)*, *LNCS*, 4185:106-112.