Voting Theory for Concept Detection

Amal Zouaq[1,2], Dragan Gasevic[2,3], and Marek Hatala[3]

[1] Department of Mathematics and Computer Science, Royal Military College of Canada, CP 17000, Succursale Forces, Kingston ON Canada K7K 7B4
[2] School of Computing and Information System, Athabasca University, 1 University Drive, Athabasca, AB, Canada T9S 3A3
[3] School of Interactive Arts and Technology, Simon Fraser University, 250-102nd Avenue, Surrey, BC Canada V3T 0A3
amal.zouaq@rmc.ca, {dgasevic,mhatala}@sfu.ca

Abstract. This paper explores the issue of detecting concepts for ontology learning from text. Using our tool OntoCmaps, we investigate various metrics from graph theory and propose voting schemes based on these metrics. The idea draws its root in social choice theory, and our objective is to mimic consensus in automatic learning methods and increase the confidence in concept extraction through the identification of the best performing metrics, the comparison of these metrics with standard information retrieval metrics (such as TF-IDF) and the evaluation of various voting schemes. Our results show that three graph-based metrics Degree, Reachability and HITS-hub were the most successful in identifying relevant concepts contained in two gold standard ontologies.

Keywords: Concept extraction, voting theory, social choice theory, ontology learning, graph-based metrics.

1 Introduction

Building domain ontologies is one of the pillars of the Semantic Web. However, it is now widely acknowledged within the research community that domain ontologies do not scale well when created manually due to the constantly increasing amount of data and the evolving nature of knowledge. (Semi) Automating the ontology building process (ontology learning) is thus unavoidable for the full-realization of the Semantic Web.

Ontology learning (from texts, xml, etc.) is generally decomposed in a number of steps or layers, which target the different components of an ontology: concepts, taxonomy, conceptual relationships, axioms and axioms schemata [3]. This paper is concerned with the first building block of ontologies which are concepts (classes). In fact, concept extraction is a very active research field, which is of interest to all knowledge engineering disciplines. Generally, research in ontology learning from texts considers that a lexical item (a term) becomes a concept once it reaches a certain value on a given metric (e.g. TFIDF). Numerous metrics such as TF-IDF, C/NC value or entropy [3, 4, 8, 15] have been proposed to identify the most relevant terms from corpora in

E. Simperl et al. (Eds.): ESWC 2012, LNCS 7295, pp. 315–329, 2012.

information retrieval and ontology learning. For example, some approaches such as Text2Onto [4] and OntoGen [7] rely on metrics such as TFIDF to evaluate term relevance. However, generally the presented solutions either adopt one metric or require that the user identifies the most suitable metric for the task at hand [3]. Following our previous work on graph theory based metrics for concept and relation extraction in ontology learning [19], we propose to enrich this perspective by:

- Testing various metrics from graph theory and
- Taking into account a number of metrics in suggesting suitable concepts based on the Social choice theory [5, 14].

1.1 Motivation

This work aims at exploring the following research questions:

- Do we obtain better results with graph-based metrics rather than with traditional information retrieval measures?

In our previous work [19], we showed that some graph-based metrics are a promising option to identify concepts in an ontology learning system. This paper continues exploring this aspect by enriching the set of studied measures and extending the experiment to another gold standard.

- Do we obtain better results with voting schemes rather than with base metrics?

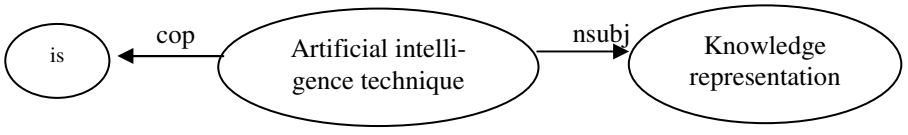
Social Choice Theory studies methods for the aggregation of various opinions in order to reach a consensus [5]. This theory is appealing in our case for two main reasons: firstly, at the practical level, it provides a mean to aggregate the results of various metrics in order to recommend concepts. Secondly, at the theoretical level, it gracefully integrates the idea of consensus, which is one of the main goals of ontologies. In fact, ontologies are meant to improve the communication between computers, between humans and computers and between humans [12]. At this level, another research question is: How can we mimic consensus with automatic ontology learning methods? Although consensus is generally concerned with human users, our hypothesis is that mimicking this characteristic at the level of automatic methods will provide more reliable results.

1.2 Contributions

This paper explores various metrics and voting schemes for the extraction of concepts from texts. Besides bringing a different perspective to this research avenue, the significance of our proposal is that it is applicable to a number of issues related to the Semantic Web, including (but not limited to) learning relationships, helping experts collaboratively build an ontology and reducing the noise that results from the automatic extraction methods.

2 Background

This paper is based on our ontology learning tool, OntoCmaps [19], which in turn is derived from our previous work [20, 21]. OntoCmaps is a "complete" ontology learning system in the sense that it extracts primitive and defined classes (concepts), conceptual relationships (i.e. relations with domain and range), taxonomical relationships (is-a links) and equivalence classes' axioms (e.g. AI = Artificial Intelligence).
OntoCmaps relies on dependency-based patterns to create a forest of multi-digraphs constituted of nodes (terms) and edges (hierarchical and conceptual relations). An example of pattern is:



Semantic Analysis
Is_a (knowledge representation, Artificial Intelligence technique)
By multi-digraphs, we mean that there can be multiple directed relationships from a given term X to a given term Y. For each term X, there can be various relationships to a set of terms S, which constitutes a term map. Some term maps might be isolated, others might be linked to other term maps through relationships, hence creating a forest. Figure 1 shows a term map around the term "intelligent agent", which can in turn be related to the term of agent, which has itself a term map (and so on). Once the extraction of term maps is performed, the tool filters the results based on various graph-based metrics by assigning several scores to the potential candidates. These scores serve to promote candidate terms as concepts in the ontology. In our previous work [19], we identified a number of graph-based metrics as potential useful measures for extracting terms and relationships. We found promising results by comparing these graph-based metrics (Degree, Betweenness, PageRank and HITS-Authority) to information retrieval metrics such as TF-IDF and TF. We showed that graph-based metrics outperformed these commonly used metrics to identify relevant candidate concepts. We also tested some voting schemes (intersection voting scheme and majority voting scheme) and discovered that they contributed in increasing the precision in our results.

This paper investigates further this previous study, by expanding the set of considered graph-based metrics and by using voting theory methods to consider the vote of each metric for the selection of domain concepts. In fact, voting theory can be used to consider the contribution of each metric and to decrease the noise that results from a NLP pipeline. Voting theory has been experimented in a number of works in artificial intelligence such as agent group-decision-making [6], information mashups [2], ontology merging [14] but to our knowledge, there is no ontology learning tool which proposed to identify concepts through graph-based measures and to increase the confidence of the extractions by aggregating the results of the various metrics through voting theory. This type of aggregation, resulting from the Social Choice Theory [14], seems similar in spirit to ensemble learning methods frequently used in machine learning [13]. However, as previously stated, experimenting voting theories has the

potential to mimic real-world vote aggregation and seems a suitable approach to establishing consensus in learning domain concepts.

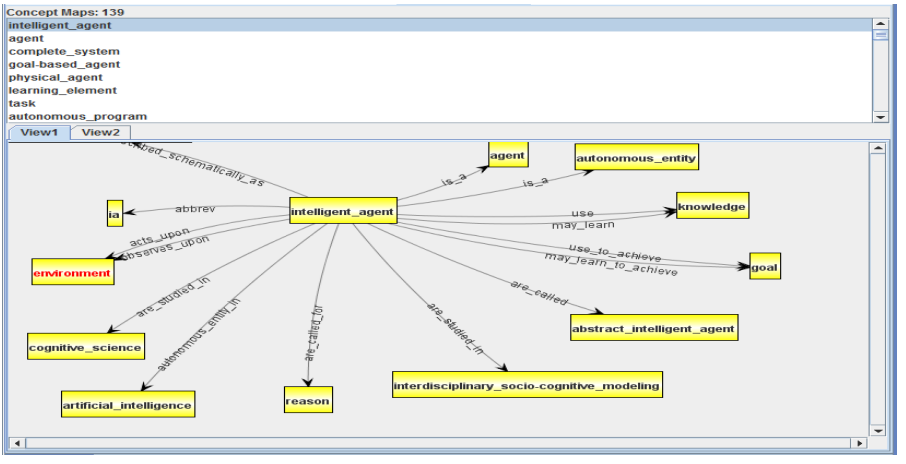


Fig. 1. An Example of OntoCmaps output

3 Voting Theory for Concept Detection

Concept detection through vote aggregation can closely be related to the problem of rank aggregation, which is a well-known problem in the context of Web search where there is a need of finding a consensus between the results of several search engines [5]. Vote aggregation can be defined as the process of reaching a consensus between various rankings of alternatives, given the individual ranking preferences of several voters [2]. In the context of a vote, each metric is considered as a voter.

3.1 Metrics

After the extraction of term maps, OntoCmaps assigns rankings to the extracted terms based on scores from various measures from graph theory (see below). In fact, since OntoCmaps generates a network of terms and relationships, computational network analysis methods are thus applicable and in this case. As outlined by [11], text mining in general and concept extraction in particular can be considered as a process of network traversal and weighting. In this paper, in addition to Degree, PageRank, HITS-Authority and Betweenness presented in [19], we computed three additional metrics HITS-Hubs, Clustering coefficient and Reachability centrality. As explained below, these metrics are generally good indicators of the connectedness and the accessibility of a node, which are two properties that might indicate the importance of a node (here a term) [11, 19].

The following metrics were calculated using the JUNG API [10]:

Degree (Deg) assigns a score to each term based on the number of its outgoing and incoming relationships;

PageRank (Prank) calculates the eigenvector probability of a term with a constant probability of the random walk restarting at a uniform-randomly chosen term [10];

HITS assigns hubs-and-authorities scores to terms based on complementary random walk processes. In our case, we considered that hubs scores and authority scores were two different metrics (Hits-hubs and Hits-Authority);

Betweenness (Bet) calculates a centrality measure where vertices that occur on many shortest paths between other vertices have higher betweenness than those that do not.

Clustering Coefficient (CC) is a measure of the connectedness between the neighbors of a node. It is given by the proportion of links between the terms within a term neighborhood divided by the number of relations that could possibly exist between them.

Reachability centrality (Reach) calculates the distance between each pair of terms using the Dijkstra algorithm.

Each of these metrics produces a ranked list of terms, i.e. a full-ordering of the extracted terms. Full-ordering is considered as the ideal scenario for rank aggregation [5].

3.2 Voting Theory Score-Based Methods

Here, we introduce voting theory methods, which can generally be divided in two main classes: score-based methods and rank-based methods.

In the score-based methods, each metric assigns a score to the elements of a list (here the extracted terms) and the resulting list must take into account the score assigned by each metric. Given the universe of Domain Terms DT, which is composed of all the nominal expressions extracted through our dependency patterns [19], the objective of the vote is to select the most popular terms $t \in DT$, given multiple metrics $m \in M$. Each metric m computes a score Stm for a term t. This score is used to create a fully ordered list TM for each metric.

Sum and maximum values are generally two functions that are used to assign an aggregated score to the terms [18]. We implemented two voting schemes based on scores: the intersection voting scheme and the majority voting scheme.

In the **Intersection Voting Scheme,** we select the terms for which there is a consensus among all the metrics and the score assigned is the sum of the scores of each individual metric normalized by their number.

In the **Majority Voting Scheme**, we select the terms for which there exists a vote from at least 50% of the metrics. The score is again the normalized sum of the score of each individual metric participating in the vote.

Each graph-based metric produced a full list of terms (DT) ordered in the decreasing order of scores. Top-k lists (partial ordering) may be created from full-ordered lists through setting up a threshold over the value of the metrics. In fact, such a threshold might be set to increase metrics' precision: in this case, only the portion of the list whose score is greater than or equal to a threshold is kept for each metric and the voting schemes operate on these partial lists.

3.3 Voting Theory Rank-Based Methods

In rank-based methods, also called positional methods, the elements are sorted based not on their score but on their positions in the lists. Besides the score, we consider a rank r_{tm}, which is the position of a term t within the ordered list produced by a metric m. The total number of ranks in metric m is R_m, which is defined as $\max_t(r_{tm})$, the lowest assigned rank (1 being the best rank). There might be more terms than ranks, because multiple terms might share a rank position.

Following [2], we implemented three positional voting schemes: Borda Count, Nauru and RunOff as these schemes (especially Nauru and Borda) are generally widely accepted in voting theory.

Borda Count Voting Scheme: This method assigns a "rank" r_{tm} to each candidate, with the lowest possible rank assigned to missing entries (usually 0). A candidate who is ranked first receive n points (n=size of the domain terms to be ranked), second *n-1*, third *n-2* and so on. The "score" of a term for all metrics is equal to the sum of the points obtained by the term in each metric.

Nauru Voting Scheme: The Nauru voting scheme is based on the sum of the inverted rank of each term in each metric ($\text{sum}(1/r_{tm})$). It is used to put more emphasis on higher ranks and to lessen the impact of one bad rank [2].

RunOff Voting Scheme: This voting scheme selects terms one at a time from each metric in a fixed order starting from the highest ranked terms. Once the same term has been selected by at least 50% of the metrics it is added to the voting list and further mentions of it are ignored. This operation is repeated until no remaining terms exist (full-ordering).

4 Methodology

4.1 Dataset

We used a corpus of 30,000 words on the SCORM standard which was extracted from the SCORM manuals [16] and which was used in our previous experiments [19]. This corpus was exploited to generate a gold standard ontology that was validated by a domain expert. To counterbalance the bias that may be introduced by relying on a unique domain expert, we performed user tests to evaluate the correctness of the gold standard. We randomly extracted concepts and their corresponding conceptual and taxonomical relationships from the gold standard and exported them in Excel worksheets. The worksheets were then sent together with the domain corpus and the obtained gold standard ontology to 11 users from Athabasca University, Simon Fraser University, the University of Belgrade, and the University of Lugano. The users were university professors (3), postdoctoral researchers (2), and PhD (5) and master's (1) students. The users were instructed to evaluate their ontology subset by reading the domain corpus and/or having a look to the global ontology. Each user had a distinct set of items (no duplicated items) composed of 20 concepts and all their conceptual and taxonomical relationships. Almost

29% of the entire gold standard was evaluated by users and overall more than 93% of the concepts were accepted as valid and understandable by these users. This size of the sample and the fact that the sample evaluated by the users was selected randomly can provide us with solid evidence that the results of the user evaluation of the sample can be generalized to the entire gold standard.

To improve its quality, there have been slight modifications to the previous gold standard: class labels were changed by using lemmatization techniques instead of stemming, which introduced some changes in the GS classes. Additionally, some defined classes were also created, and new relationships were discovered due to new patterns added to OntoCmaps. The following table shows the statistics associated to the classes in our current GS[1].

Table 1. GS1 statistics (SCORM)

Primitive classes	Defined Classes	Conceptual Relationships	Taxonomical Relationships
1384	81	895	1121

Once the GS ontology was created, we ran the OntoCmaps tool on the same corpus. The aim was to compare the expert GS concepts with the concepts learned by the tool. We ran our ontology learning tool on the SCORM corpus and generated a ranking of the extractions based on all the above-mentioned metrics: Degree, Betweenness, PageRank, Hits, Clustering Coefficient and Reachability. The tool extracted 2423 terms among which the metrics had to choose the concepts of the ontology.

We also tested our metrics and voting schemes on another smaller corpus (10574 words) on Artificial Intelligence (AI) extracted from Wikipedia pages about the topic. The tool extracted 1508 terms among which the metrics had to choose the concepts of the ontology. Table 2 shows the statistics of the extracted AI gold standard.

Table 2. GS2 statistics (Artificial Intelligence)

Primitive classes	Defined Classes	Conceptual Relationships	Taxonomical Relationships
773	65	287	644

As previously explained, OntoCmaps produced a ranking of terms based on the various metrics introduced in this study, and we divided our results in Top-N lists, gradually increasing the number of considered terms. Recall that metrics order terms from the highest rank to the lowest one. The point was to determine how quickly the accuracy of the results would degrade as we expand the set of considered terms.

Since the SCORM GS contained 1384 primitive classes (concepts), we limited the evaluation to the first 1500 terms in our experiments on SCORM. In the AI GS, we stopped at Top-600 with 773 primitive classes in the GS. We then divided each dataset in small Top-k lists versus large Top-k lists. Small lists are expected to have the higher precision as they include the best rated terms. In the SCORM GS, small lists

[1] http://azouaq.athabascau.ca/Corpus/SCORM/Corpus.zip

included Top-k, k=50, 100, 200 (up to ~14.5% of the expected terms) and large lists had k>200. In the AI GS, small lists were Top-50 and Top-100 (up to ~13% of the expected terms).

4.2 Evaluation Criteria

Our experimental evaluation of the different ranking methods tests each of the individual metrics and each of the aforementioned voting systems. There are a number of methods that are used to evaluate similar types of research: in information retrieval and ontology learning, the results are generally evaluated using precision/recall and F-measure [3]. In our case, we chose to concentrate on the precision measure as ontology learning methods obtain difficultly good precision results (see for example [Brewster et al., 2009] and the results of Text2Onto in [4] and in our experiments [19]). Moreover, it is better to offer correct results to the user rather than a more complete but rather a noisier list of concepts [9]. In voting theory and rank aggregation studies [2], the results are often evaluated through Social Welfare Function (SWF). A SWF is a mathematical function that measures the increased social welfare of the voting system. SWF employed in similar research include Precision Optimal Aggregation and the Spearman Footrule distance [2, 17]. Given that Precision Optimal Aggregation is similar in spirit to the precision metric employed in information retrieval, we employed standard precision (Precision Function) against our GS:

Precision = items the metric identified correctly / total number of items generated by the metric

This precision metric was computed for a number of Top-N lists.

For the voting methods, we also calculated a social welfare function (SWF) by computing the proportion of the contribution of each metric to the overall ranking. In our case, the SWF is defined by the number of terms from the gold standard which were included in the promoted concepts of the overall ranking proposed by each voting method.

4.3 Experiments

Quality of Individual Metrics. In [1], the authors indicate that the performance of each individual ranker might have a strong influence over the overall impact of the aggregation. Therefore, we decided first to assess the performance of each metric in various partial lists: Top-50, Top-100, Top-600, Top-1000, Top-1500 and Top-2000. Table 3 show the performance of each metric in each of these lists.

For smaller N-Lists (N=50,100, 200), we can notice that Betweenness, PageRank and Degree are the best performing metrics, while the metrics Reachability, Degree and Hits-Hub become the best ones with larger lists (N=400..2000). Only the degree metrics seems to be constantly present in the best three results of each Top-N list.

Table 3. Precision results for each metric on the SCORM GS

	Bet	**Prank**	**Deg**	**HITS (Auth)**	**HITs (Hubs)**	**Reach**	**CC**
Top-50	96.00	96.00	94.00	86.00	88.00	92.00	70.00
Top-100	96.00	82.00	95.00	77.00	87.00	89.00	75.00
Top-200	88.00	81.00	87.00	79.50	84.50	85.50	78.50
Top-400	77.00	76.00	79.25	73.50	80.25	81.75	73.50
Top-600	75.00	69.67	75.00	71.67	80.50	82.33	69.00
Top-1000	66.30	63.80	71.90	66.10	77.30	77.60	63.40
Top-1500	63.47	61.07	66.67	61.07	71.20	70.07	62.27
Top-2000	61.35	60.90	64.00	60.90	63.95	63.95	61.45

In order to compare our results and make another experiment, we tested our metrics on the second gold standard (AI). The following table shows the results of this experiment. We notice that HITS-Hub and Reachability give the best performance overall.

Table 4. Precision results for each metric on the AI Gold Standard

	Bet	**Prank**	**Deg**	**HITS (Auth)**	**HITs (Hub)**	**Reach**	**CC**
Top-50	78.00	74.00	88.00	74.00	88.00	84.00	84.00
Top-100	71.00	69.00	86.00	62.00	88.00	81.00	67.00
Top-200	70.00	61.50	75.00	57.50	79.50	73.00	56.50
Top-400	60.75	50.75	64.00	57.00	74.50	73.00	54.50
Top-600	56.50	48.83	62.50	53.50	69.67	68.17	54.67
Top-1000	52.80	50.60	57.10	50.60	58.20	58.20	52.30

Choice of Metrics Combinations. Next, we computed the SWF Precision Optimal Aggregation (which in our case is equal to the Precision measure) for each voting method. In order to test various combinations and identify if some metrics were performing better than others, we ran the Weka tool on the SCORM GS, on the AI GS and on the merged set of AI and SCORM GSs (ALL). For each row in the GS, data contained a given term, the scores attributed by each metric and whether or not the term has been included in the GS (Yes/No). Using subset selection in Weka, we tried to identify a subset of features (metrics) which had a significant effect for predicting if a term should belong to the GS or not. Thus, we ran a wrapper-based subset selection algorithm which used a 10 fold cross-validation based on the CfsSubsetEval Attribute Evaluator and a BestFirst search in order to determine the most important attributes.

Table 5. Metrics selection using CfsSubsetEval Attribute Evaluator[2] and a BestFirst search

Attributes	Number of folds (%) SCORM	Number of folds (%) AI	Number of folds (%) ALL
1 Betw	**10(100 %)**	1(10 %)	0(0 %)
2 Prank	4(40 %)	**10(100 %)**	8(80 %)
3 Deg	**10(100 %)**	**10(100 %)**	2(20 %)
4 HITS(Auth)	0(0 %)	7(70 %)	**10(100 %)**
5 HITS(Hubs)	**10(100 %)**	0(0 %)	1(10 %)
6 Reach	**10(100 %)**	0(0 %)	**10(100 %)**
7 CC	8(80 %)	0(0 %)	0(0 %)

Table 5 shows us how many times each metric was selected during a 10-fold cross validation. We can see that some metrics are used more times than others during each cross validation. According to these results, only two metrics Degree and Reachability are present in all 10 folds of our cross-validation (10(100%)) over two datasets: Degree appears over the SCORM and AI datasets while reachability appears over the SCORM and combined (All) datasets. However, we can notice that each individual GS has other significant metrics.

Based on these results, we decided to compute the following voting schemes:

- *Intersection Voting Schemes* (IVS_1, IVS_2 and IVS_3), where IVS_1 is based on all the metrics except the clustering coefficient (which appears to be significant only for SCORM): Hits_Hub, Hits_Authority, PageRank, Degree, Reachability and Betweenness. IVS_2 uses Reachability and Betweenness while IVS_3 is based on Betweenness, Reachability, Hits_Hub and Degree.
- *Majority Voting Schemes* (MVS_1 and MVS_2), where MVS_1 and MVS_2 uses the same metrics respectively as IVS_1 and IVS_3.
- *Borda, Nauru and Runoff* were all based on the metrics Betweenness, Reachability, Degree and HITS-Hubs which are the best metrics for the SCORM GS.

Precision Optimal Aggregation Results on the SCORM and AI GS. In the Top-50 list of the SCORM GS, we noticed that all the voting schemes, except Runoff (96% precision), were successful (100% precision) in identifying relevant concepts among the highest ranked 50 terms. However, as the number of considered terms increases (Table 6), we can notice that the Intersection voting schemes and the majority voting schemes (~82%) beat slightly the other voting scheme systems (Runoff: 77.5%, Nauru: 79.8%, and Borda: 80.5%). In our experiments on the AI GS (Table 6), the best performing voting schemes were:

- Nauru first (90%) and then Runoff, IVS_1 and MVS_2 with 88% in the Top-50 list
- Runoff first (81.5%), Nauru (80%) and then IVS_1 and MVS_2 with 79% in the Top-200 list;
- IVS_2 first (67.5%), Nauru and Runoff with 67%, Borda with 66% and then IVS_1 and MVS_2 with 65.5% in the Top-600 list.

[2] In Weka, CfsSubsetEval evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them.

Table 6. Performance of voting methods for Top-600 terms on the SCORM GS

	SCORM Top-600	AI Top-600
IVS_1	82.66	65.5
IVS_2	82.33	67.5
IVS_3	82.66	61.83
MVS_1	82.66	61.83
MVS_2	82.66	65.5
Borda	80.5	66
Nauru	79.83	67
RunOff	77.5	67

Comparison with other Metrics on the SCORM Gold Standard. In order to compare our results with some baselines, we computed standard measures used in information retrieval: Term frequency (TF) and TF-IDF as well as random term selection (see table 7). TF and TF-IDF were computed on two sets of terms: TF and TFIDF are computed on all the extracted terms from the corpus while TF(DT) and TFIDF(DT) are computed on domain terms only, i.e. terms that were selected by OntoCmaps as already potential domain terms through patterns and stop words filtering.

Table 7. Traditional Metrics results on the SCORM GS

	TFIDF	TF	TFIDF (DT)	TF (DT)	Random
Top-50	72.00	74.00	92.00	90.00	34.00
Top-100	70.00	66.00	89.00	88.00	33.00
Top-200	66.50	66.00	85.50	79.50	36.00
Top-400	56.75	52.50	72.25	69.50	39.00
Top-600	51.00	47.33	66.83	65.83	40.33
Top-1000	43.70	44.80	62.40	62.70	43.10
Top-1500	43.80	43.07	59.93	59.93	43.74
Top-2000	43.60	43.60	59.80	59.8.00	43.74

As we can see in table 7, the metrics TFIDF and TF are more successful when they are applied on the pre-filtered domain terms (TFIDF (DT) and TF (DT)). We can also notice that the graph-based metrics and their combination through voting schemes beat the traditional metrics (compare Table 3 and Table 7). Up to the Top-200 list, Betweenness is the best performing metrics, then Reachability (in Top-400, Top-600 and Top-1000), then HITS-Hub (Top-1500), and finally Degree (Top-2000).

Comparison with other Metrics on the AI Gold Standard. We repeated the same experiment on the AI GS. As shown in Table 8, among the traditional metrics, we can also notice that the best performing ones are TFIDF (DT) and TF (DT). If we compare these metrics from Table 8 with the graph-based ones (Table 4), we also see that again graph-based metrics have much better performance in all the Top-k lists (k=50, 100, 200, 400, 600 and 1000). For example, the best in the Top-50 list is the Degree and HITS-Hub with 88% versus (72% for TFIDF (DT)) and in the Top-600, the best is HITS-hub (69.67%) versus 50.83% for TF (DT) and TFIDF (DT).

Table 8. Traditional Metrics results on the AI GS

	TFIDF	**TF**	**TFIDF (DT)**	**TF (DT)**	**Random**
Top-50	38.00	50.00	72.00	70.00	28.00
Top-100	41.00	47.00	69.00	71.00	30.00
Top-200	40.00	39.00	62.50	61.50	25.50
Top-400	35.00	34.00	56.25	53.25	27.00
Top-600	32.83	31.50	50.83	50.83	28.00
Top-1000	28.00	28.10	56.30	56.40	27.56

Based on the results presented in Tables 3, 4, 7 and 8, we ran a paired sample t-test on each of these metrics combinations and the differences were statistically significant and in favor of graph-based metrics in general, and in favor of Degree, reachability and Hits-hubs in particular.

5 Discussion

In this section, we summarize our findings and the limitations of our work.

5.1 Findings

Our findings are related to our initial research questions:

Do we obtain better results with graph-based metrics rather than with traditional ones?
Obviously, it is possible to confirm this research hypothesis through our experiments with the best performing metrics being:

- SCORM – small lists : Betweenness, PageRank, and Degree
- SCORM- large lists: Hits-Hub, Degree, Reachability
- AI- small lists: Degree, Hits-Hubs, Reachability
- AI- large lists: Hits-Hub, Degree, Reachability

We can observe that Degree is constantly present and that Degree, Hits-Hub and Reachability seem to be the best performing graph-based metrics. This result is confirmed by our machine learning experiments (Table 5) for at least two metrics Degree and Reachability.

Do we obtain better results with voting schemes rather than with base metrics? As far as voting schemes are concerned, the first question is whether we were able to increase the precision of the results by using these voting schemes (see Table 9). In previous experiments [19], we noticed that some voting schemes were enabling us to get better performance but our ranked lists contained only those terms whose weight was greater than the mean value of the considered metric, which had already a strong impact on the precision of each metric.

Table 9. Comparison between voting schemes and base metrics

	SCORM	**AI**
Top-50	100% : All voting schemes except Runoff 96%: Bet and PageRank	90%: Nauru 88%: Deg and HITS-hub
Top-100	97%: IVS_3, MVS_1, MVS_2 96%: Bet	86%: Runoff 88%: HITS-hub
Top-200	87%: IVS_1 and MVS_2 88%: Bet	81.5%: Runoff 79.5%: HITS-hub
Top-400	83.75%: IVS_3 and MVS_1 81.75% : Reach	72.75%: Runoff 74.5%: HITS-hub
Top-600	82.67%: IVS_1, IVS_3, MVS_1, MVS_2 82.33%: Reach	67.5: IVS_2 69.67%: HITS-hub
Top-1000	77.7%: IVS_1 and MVS_2 77.6%: Reach	60.7%: IVS_1 and MVS_2 58.2%: HITS-hub and Reach
Top-1500	71.26%: IVS_1 and MVS_2 71.20%: HITS_hub	NA
Top-2000	65.15%: IVS_1 and MVS_2 64%: Degree	NA

Despite a small increase in almost all the cases in favor of voting schemes, the difference between voting schemes and base metrics such as Degree, Hits-Hub and Reachability was not really noteworthy. This asks the question whether such voting schemes are really necessary and whether the identified best graph-based metrics would not be enough, especially if we don't take the mean value as a threshold for the metrics. Having identified that the best base metrics were Degree, Reachability and HITS-hub, we tried some combinations of metrics on the SCORM GS. Despite an improvement of voting theory schemes (e.g. Borda) in some Top-n lists, we did not notice a major difference. Our future work will continue testing combinations of voting schemes and voting theory measures, based on these metrics, on various gold standards. We also plan to compare this voting-based approach with ensemble machine learning algorithms.

5.2 Limitations

One of the most difficult aspects in evaluating this type of work is the necessity to build a gold standard, which in general requires a lot of time and resources. Building a GS that represents a universal ground truth is not possible. Ideally, the experiments presented in this paper should be repeated over various domains to evaluate the generalizability of the approach. However, this is often impossible due to the cost of such a large scale evaluation. In this paper, we extended our previous evaluation on another corpus, and we also extended the set of tested metrics and voting schemes. Future work will have to continue the validation of our approach and to expand the set of "traditional" metrics (such as C/NC value) to be compared with graph-based metrics.

Another limitation is that the metrics that we propose for discovering concepts are graph-based metrics, which involves processing the corpus to obtain a graph while metrics commonly used in information retrieval such as TF-IDF only require the corpus. In our experiments, we always relied on OntoCmaps to generate this graph. However, we do not believe that this could represent a threat to the external validity of our findings, as these metrics are already applied successfully in other areas such social network analysis and information retrieval and are not dependent on anything else than a set of nodes (terms) and edges (relationships).

Finally, despite our focus on concepts in this paper, such a graph-based approach is worth the effort only if the aim is to extract a whole ontology and not only concepts, as it involves discovering terms and relationships between terms. This requirement is also closely linked to another limitation: since we rely on deep NLP to produce such a graph, it requires time to process the corpus and calculate the graph-based metrics. However, we believe that this is not a major limitation, as ontologies are not supposed to be generated on the fly.

6 Conclusion

In this paper, we presented various experiments involving a) the comparison between graph-based metrics and traditional information retrieval metrics and b) the comparison between various voting schemes, including schemes relying on voting theory. Our finding indicates that graph-based metrics always outperform traditional metrics in our experiments. In particular, Degree, Reachability and HITS-Hub seem to be the best performing ones. Although voting schemes increased precision in our experiments, there was only a slight improvement on the precision as compared to the three best performing metrics.

Acknowledgments. This research was funded partially by the NSERC Discovery Grant Program and the Burrough Wellcome Fund.

References

1. Adali, S., Hill, B., Magdon-Ismail, M.: The Impact of Ranker Quality on Rank Aggregation Algorithms: Information vs. Robustness. In: Proc. of 22nd Int. Conf. on Data Engineering Workshops. IEEE (2006)

2. Alba, A., Bhagwan, V., Grace, J., Gruhl, D., Haas, K., Nagarajan, M., Pieper, J., Robson, C., Sahoo, N.: Applications of Voting Theory to Information Mashups. In: IEEE International Conference on Semantic Computing, pp. 10–17 (2008)
3. Cimiano, P.: Ontology Learning and Population from Text. Algorithms, Evaluation and Applications. Springer (2006)
4. Cimiano, P., Völker, J.: Text2Onto. In: Montoyo, A., Muńoz, R., Métais, E. (eds.) NLDB 2005. LNCS, vol. 3513, pp. 227–238. Springer, Heidelberg (2005)
5. Dwork, C., Kumar, R., Naor, M., Sivakumar, D.: Rank aggregation methods for the Web. In: Proc. of the 10th International Conference on WWW, pp. 613–622. ACM (2001)
6. Endriss, U.: Computational Social Choice: Prospects and Challenges. Procedia Computer Science 7, 68–72 (2011)
7. Fortuna, B., Grobelnik, M., Mladenic, D.: Semi-automatic Data-driven Ontology Construction System. In: Proc. of the 9th Int. Multi-Conference Information Society, pp. 309–318. Springer (2006)
8. Frantzi, K.T., Ananiadou, S.: The C/NC value domain independent method for multi-word term extraction. Journal of Natural Language Processing 3(6), 145–180 (1999)
9. Hatala, M., Gašević, D., Siadaty, M., Jovanović, J., Torniai, C.: Can Educators Develop Ontologies Using Ontology Extraction Tools: an End User Study. In: Proc. 4th Euro. Conf. Technology-Enhanced Learning, pp. 140–153 (2009)
10. JUNG, `http://jung.sourceforge.net/` (last retrieved on December 6, 2011)
11. Kozareva, Z., Hovy, E.: Insights from Network Structure for Text Mining. In: Proc. of the 49th Annual Meeting of the ACL Human Language Technologies, Portland (2011)
12. Maedche, A., Staab, S.: Ontology Learning for the Semantic Web. IEEE Intelligent Systems 16(2), 72–79 (2001)
13. Polikar, R.: Bootstrap inspired techniques in computational intelligence: ensemble of classifiers, incremental learning, data fusion and missing features. IEEE Signal Processing Magazine 24, 59–72 (2007)
14. Porello, D., Endriss, U.: Ontology Merging as Social Choice. In: Proceedings of the 12th International Workshop on Computational Logic in Multi-agent Systems (2011)
15. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information Processing & Management 24(5), 515–523 (1988)
16. SCORM (2011), `http://www.adlnet.gov` (last retrieved on December 10, 2011)
17. Sculley, D.: Rank Aggregation for Similar Items. In: Proc. of the 7th SIAM International on Data Mining (2007)
18. Shili, L.: Rank aggregation methods. In: WIREs Comp. Stat. 2010, vol. 2, pp. 555–570 (2010)
19. Zouaq, A., Gasevic, D., Hatala, M.: Towards Open Ontology Learning and Filtering. Information Systems 36(7), 1064–1081 (2011)
20. Zouaq, A., Nkambou, R.: Evaluating the Generation of Domain Ontologies in the Knowledge Puzzle Project. IEEE Trans. on Kdge and Data Eng. 21(11), 1559–1572 (2009)
21. Zouaq, A.: An Ontological Engineering Approach for the Acquisition and Exploitation of Knowledge in Texts. PhD Thesis, University of Montreal (2008) (in French)