Exploring an ontology via text similarity: an experimental study

Daniil Mirylenka[1,3], Marco Rospocher[1], Ivan Donadello[1,3], Elena Cardillo[2], and Luciano Serafini[1]

[1] Fondazione Bruno Kessler—IRST,
Via Sommarive 18, Trento, I-38123, Italy
{mirylenka,rospocher,donadello,serafini}@fbk.eu
[2] Institute for Informatics and Telematics—IIT-CNR,
Via P. Bucci 17B, Rende, I-87036, Italy
{elena.cardillo@iit.cnr.it}
[3] Department of Information and Communication Technology, University of Trento,
Via Sommarive 14, 38050, Trento, Italy

Abstract. In this paper we consider the problem of retrieving the concepts of an ontology that are most relevant to a given textual query. In our setting the concepts are associated with textual fragments, such as labels, descriptions, and links to other relevant concepts. The main task to be solved is the definition of a similarity measure between the single text of the query and the set of texts associated with an ontology concept. We experimentally study this problem on a particular scenario with a socio-pedagogic domain ontology and Italian language texts. We investigate how the basic cosine similarity measure on the bag-of-words text representations can be improved in three distinct ways by (i) taking into account the context of the ontology nodes, (ii) using the linear combination of various measures, and (iii) exploiting semantic resources. The experimental evaluation confirms the improvement of the presented methods upon the baseline. Beside discussing some issues to consider in applying these methods, we point out some directions for further improvement.

1 Introduction

Ontology-based intelligent systems support user tasks by exploiting the formalized information. Typically, these systems query the ontology—potentially using the content inferred via reasoning—to retrieve specific ontology elements that match the criteria for the given task. In this paper, we consider a peculiar example of interaction between the system and the underlying ontology: given a natural language text provided as input to the system, we want to retrieve the ontology entities whose textual content is most similar to the input text.

A concrete example of an intelligent system tackling the aforementioned problem is the ePlanning IEP system—a tool, currently under development—which aims to support the composition of Individualized Educational Plans (IEP) for pupils with special educational needs. An IEP has to be prepared for each pupil[1]

by the teachers, in collaboration with experts, such as neuropsychiatrists and psychologists, to ensure the appropriate education for pupils with disabilities, and has to comprise a collection of objectives and activities that a pupil should exercise to achieve them.

The task of composing an IEP is quite demanding, due to the multitude of aspects to be considered and the variety of expertise involved. The ePlanning IEP system aims to automate this process by suggesting adequate objectives and activities based on the profile of the pupils and the textual description of their inabilities. To achieve this goal, the system exploits a formal ontology describing pupils' functional abilities (for example, "Comprehending written language"), linked to objectives, and activities to be performed to achieve them. Each functional ability is encoded in the ontology as a concept and enriched with textual information, such as the label, the description of the ability, and the questions that help teachers identify the ability in a given pupil. The task of the ePlanning IEP system is to take a textual description of the pupil, find the *inabilities* that are "implicitly" mentioned in it, and suggest the corresponding objectives and activities, taking into account the pupil's profile.

In this paper we investigate several text similarity based strategies, mainly variants of the cosine similarity[11], for comparing the text provided as input to the system with the textual content associated to the concepts in the ontology. We show that taking into account the structure of the ontology as well as using the linear combination of different similarity measures improve the performance of the baseline method. Exploiting the semantic resources for Italian language turned out to be nontrivial in our experiments, and only improved the performance when combined with other similarity measures.

The paper is organized as follows. In Section 2 we introduce the IEP *Ontology* at the background of the ePlanning IEP system. In Section 3 we describe the general similarity framework, detailing all the different similarity strategies implemented. In Section 4 we describe the experimental setup, while in Section 5 we present and comment on the obtained results. Finally, in Section 6 we discuss some issues related to the task, and point out some directions for future work.

2 The IEP ontology

The ePlanning IEP system grounds on a formal ontology *IEPOntology* describing the main functional abilities of the pupils from nursery to the high school. The functional abilities are arranged into a taxonomy with three to four levels of depth. The more general abilities (e.g. "language abilities") are represented by the nodes close to the root, while the most specific abilities (e.g. "basic production of text") are represented by the leaves. Each leaf is associated to specific educational objectives and activities that the pupil should exercise to improve the ability in question. The activities vary according to the pupil's profile, and depend, for instance, on the pupil's school year and severity level of the disability.

IEPOntology formalizes the functional abilities as the hierarchy of ontology classes (concepts), in which the classes along the same directed path are linked

with sub/superclass relation. Some textual information is associated with each ability in the form of ontology annotation, such as the label and the detailed description. Other pieces of information associated with abilities are represented as entities in the ontology, for instance, the possible ICD-10[4] code of the clinical diagnosis provided by the physician and the possible ICF-CY[5] code of the functional diagnosis provided by the neuropsychiatrist. Such entities cannot be directly linked with the ontology classes. Therefore, for each class representing the ability we also instantiate an entity, and link it to the related entities with ontological relations. Additional relations are introduced for the leaf nodes in order to link them to the corresponding recommended activities.

In this way, a concept representing a functional ability is associated with a number of text fragments that either directly annotate it (e.g. the label and the description), or annotate the entities to which it is linked, such as the ICD-10 and the ICF-CY codes. Given the textual description of a pupil's functional problems provided by the teacher, the task of the ePlanning IEP system is to find the concepts (nodes of the ontology), that are most similar to the given description according to their associated textual fragments.

3 The general framework

The formulated task can be viewed as a retrieval problem, to which we have taken the following general approach: we consider a scoring function $score(q, \cdot)$ that measures the similarity of the concept nodes to the query text. Depending on the specific approach, we return either the top k highest-scoring nodes or the nodes whose score exceeds a pre-defined threshold. The specific methods we investigated differ in the way the scoring function is defined.

In its general form, $score(q, p)$ is a function of the text associated with the node p, as well as its context (nodes related to p). For simplicity we ignored the contextual information in most of our experiments. Let $texts(p)$ be the set of text fragments associated with the node p. We consider a generic similarity function $sim(\cdot, \cdot)$ defined on the pairs of texts, and define the score of a node as the aggregated similarity between the texts of the node and the input query:

$$score_{sim}(q, n) := aggr(\{sim(q, t) \mid t \in texts(n)\}), \tag{1}$$

where *aggr* is some aggregation function, such as *average*. Note that the scoring function is parameterized by the function measuring the similarity between texts.

The three obvious choices for the aggregation function are *min*, *max*, and *average*. In our experiments, *max* has shown the best performance. Experimenting with the softmax function (generalization of *min*, *max*, and *average*) has not improved the performance significantly.

The scoring function (1) requires defining the similarity between texts, which can be done in numerous ways. The DKPro library[2] contains the implementations for the dozens of known similarity measures. Every such measure, as

[4] `http://www.who.int/classifications/icd/en/`
[5] `http://www.who.int/classifications/icf/en/`

well as the many possible combinations of them, can be used in definition (1). We designed a principled scheme for choosing the appropriate similarity measure, based on machine learning. This scheme requires a sample of "ground truth" data—textual queries along with the corresponding relevant concepts—provided by the experts (teachers). For the reasons beyond our control, we have not been able to obtain such a sample, and had to resort to inferior approximations of the designed approach. Nonetheless, we describe the original approach in the following section for the sake of completeness. The implemented approximate methods, and their limitations are discussed in Section 3.2.

3.1 The principled scheme: learning the combination of measures

The main intuition is that multiple text similarity measures capture different aspects of similarity, and the combination of various measures should perform better that any single measure in isolation. Let S_1, S_2, $\cdots$, S_m be the set of available similarity measures, such as those implemented in the DKPro library. We define the combined score of a node as the linear combination of scores (as in definition (1)) induced by these measures:

$$score(q, p) := \sum_{i=1}^{m} \alpha_i \cdot score_{S_i}(q, p). \tag{2}$$

The values of the weights α_i are learned from the training data by the following procedure. Suppose we have "ground truth" examples of the form $(q, P^r(q))$, where q is the input query, and $P^r(q) = \{P^r_j(q)\}_{j=1}^{N(q)}$ is the corresponding set of relevant concepts, as identified by an expert ($P \backslash P^r(q)$ thus being the set of irrelevant concepts). From each such example we construct N training data points, corresponding to the concepts in the ontology. For a given example $(q, P^r(q))$ and a node p, we build a data point of the form (x, y), where x is an m-dimensional vector of the node's scores according to different similarity measures:

$$x := (score_{S_1}(q, p), score_{S_2}(q, p), \cdots, score_{S_m}(q, p)), \tag{3}$$

and y encodes whether the concept p is relevant according to the ground truth:

$$y := 1 \text{ if } p \in P^r(q), \text{ else } 0. \tag{4}$$

The set of the constructed data points is used to train a linear classifier, such as Logistic Regression, or SVM[8]. The classifier learns to predict y, the relevance status of a concept, based on x, the values of the similarity functions.

In order to identify the relevant nodes for a new query, we transform the query into N vectors of the form (3) (one vector for each concept node in the ontology), and apply the trained classifier to each vector. The vectors for which the classifier predicted 1 correspond to the relevant concepts. Being linear, the classifier takes the decision based on the linear combination of the components of x, which has exactly the form (2). In this way, the classifier learns the optimal (linear) combination of various similarity functions.

3.2 The approximate scheme and the baseline method

In the absence of the training data, the described principled scheme of combining the similarity measures is not available. In this case, one option is to evaluate the performance of the similarity measures in isolation. Another option is to examine the linear combinations of the subsets of the measures by "manually" varying the weights α_i. As the number of subsets is exponential (with respect to the number of measures), the latter option is impractical for the subsets sizes greater than 2, which corresponds to the pairwise combinations. Furthermore, "manual" search for the weights α_i is prone to overfitting the evaluation criteria, especially given the relatively high dimensionality of the search space. Adding more parameters to the method, as in sections 5.1–5.5, increases the dimensionality of the parameter space, further increasing the mentioned problems. We thus experimented on individual similarity measures and, in some cases, their pairwise combinations.

The simplest use of our framework takes the scoring function (2) with a single similarity measure. Our default similarity measure is based on the vector space model with TF weighting[9], which is used widely in practice. This measure is defined as the cosine between the term vector representations of the texts:

$$sim(t_1, t_2) := cos(vec(t_1), vec(t_2)). \tag{5}$$

Every component of the vector representation of a text corresponds to a term, and is proportional to its frequency in the text. The vectors are normalized to have the unit length. As per the standard practice, the terms are lemmatized, and the stop words are removed from the texts in a pre-processing step. In our task it proved difficult to attain significant improvement over this baseline.

4 Experimental setup

We first applied the baseline method (see Section 3.2) to 60 short queries and had the results evaluated by teachers. The collected feedback was then used to build the dataset for evaluating the other methods. We will describe the initial experiment and the built dataset in more detail.

4.1 Evaluation ("ground truth") dataset

We collected about 60 sentences appearing in the actual IEPs (Table 1 shows some examples). For each sentence we ranked the nodes with the baseline method and presented the teachers with the three highest-scoring ones. The teachers could mark any of the presented nodes as incorrect (irrelevant), and optionally suggest additional relevant nodes. From this feedback we constructed an approximate validation dataset, in which for a given query q, the relevant nodes $P^r(q)$ consisted of the nodes confirmed as relevant plus the suggested nodes.

Because of the small size, as well as the limitations (discussed further), the constructed dataset could not enable the machine learning scheme described in Section 3.1. However, it allowed us to evaluate and compare the performance of

Federica is not able to take shower on her own.
Federica is not able to use public transport, neither to cross the street independently.
Paolo is not able to eat independently.
The pupil tries to attract the teachers' attention with exhibitionistic behavior.
Difficulties in the autonomous management of tasks and routines.

Table 1. Examples of sentences appearing in IEPs (translated from Italian).

the different methods in a sufficiently reasonable way. The constructed dataset has the following shortcomings:

Short sentences: In the original task, the queries are assumed to be entire paragraphs of text describing a pupil. In contrast, the queries in the constructed dataset are single sentences. First, this makes the task more difficult, as the short queries contain less relevant information to inform the similarity functions. Therefore, the performance on the original task may be underestimated. Second, a single sentence of a IEP usually mentions a single problem, while in a paragraph several problems may be touched upon. Therefore, it is important to identify nodes relevant to different mentioned problems, while this aspect is not reflected in our short-query dataset. Last, the short sentences pose problems for the similarity measures that normalize by the text length, such as the cosine.

Bias towards the baseline: The "ground truth" nodes were not provided by the teachers independently, but in the form of corrections to the results produced by the baseline method. As people are generally inclined to accept the default rather than provide corrections, the built dataset is potentially biased towards the baseline method. In other words, the baseline method, and the methods similar to it are likely to show relatively better performance on this dataset, than if the ground truth nodes had been provided independently by the teachers.

Three returned nodes: When building the dataset, the baseline method was constrained to return the top three nodes. After the teachers' corrections, the number of "ground truth" nodes per query ranged from one to six, averaging about 2.6. In this way, the "ground truth" for every given query was biased towards containing three nodes, while the true number of relevant nodes per query may vary differently.

4.2 Evaluation metric

In order to evaluate a method on a single query, one needs to compare the results (nodes) produced by the method to the ground truth results for this query, and measure the discrepancy between these sets with some sort of similarity score. The score aggregated across all the queries in the "ground truth" dataset give an estimate of the overall performance of the method.

In our case the ground truth results and the results of our methods are of different nature: for a given query, the former is a small set of nodes with unspecified order, while the latter is essentially a ranking of all the nodes in the ontology. We have designed an evaluation metric for comparing such results. At

the high level, the metric assures that the ground truth nodes are high in the ranking produced by the method.

More precisely, we compute the sum of the reciprocal ranks of the ground truth results, normalized by the maximum possible reciprocal sum (which corresponds to the case when the ground truth nodes are ranked the highest possible by the method). To give an example, suppose that the method returns the results in the following order: 1, 2, 3, 4, and so on, while the ground truth consists of the documents {2, 3, 5}. In this case, our metric evaluates to $((1/2+1/3+1/5)/(1+1/2+1/3)) \approx 0.56$. In general, the value of the metric is greater than zero and less or equal to one, with equality reached if and only if the ground truth nodes occupy the topmost positions in the ranking.

5 Methods and results

The methods we describe here are modifications of the baseline method described in Section 3.2. In our experiments, we applied and evaluated these modifications independently from each other. Although applying the modifications in combination is possible (and sensible), evaluating the exponential number of combinations is impractical and may lead to over-fitting of our small evaluation dataset (as discussed in Section 3.2). An appropriate ground truth dataset could allow for automatic selection of the combinations through cross-validation, and possibly also through the supervised machine learning. We describe the individual methods in the following sections.

The baseline method measured **0.57**, as evaluated as described in Section 4.

5.1 IDF scores

The baseline approach relies on the cosine similarity between the term vector representations of the text fragments. The values of the vector components are computed with the term frequency (TF) weighting scheme.

It is typical in information retrieval to include the inverse document frequencies (IDF) into the term weights[9]. IDF measures the specificity of a term, that is, the number of documents in which the term appears. In its general form, IDF is a function that gives lower values for the terms that appear in more documents. One typical formula is $IDF(t) = log(N/N_t + 1)$, where $\mathrm{N_t}$ is the number of documents in which the term appears, and N is the total number of documents. The weight of a term t in a document d is computed as $TF(t,d) * IDF(t)$.

The IDF weights are most informative when they are computed on a large corpus of text documents. Having no such corpus at our disposal, we utilized the ontology itself for the computation of the IDF weights. This required turning ontology into a document corpus, which we did in two different ways. Remember that every node in the ontology is associated with a number of short texts. In the first way, we considered every such text as a separate document in the corpus, while in the second way we joint the texts belonging to the same node into a single document. Both ways scored around **0.55**, thus not improving the performance

on our test data collection. We expect that the IDF weights computed with respect to a large corpus of relevant texts should show better results.

5.2 Wikipedia-based semantic similarity

It is often useful to compare the texts via semantic similarity measures, which go beyond the surface form of the words. One example of such similarity measure is based on the Explicit Semantic Analysis (ESA)[5]. ESA represents texts as vectors, not in the space of terms, but in the space of documents, for some large and comprehensive document collection, typically, Wikipedia articles. The similarity between texts is then computed as the cosine between these document-based vector representations. We indexed the articles of the whole Italian Wikipedia with Apache Lucene (`http://lucene.apache.org/`), and used it in conjunction with the ESA-based similarity measure implemented in the DKPro Similarity library. However, using the implementation turned out to be prohibitive in terms of speed, as it was not performance-optimized.

As a proxy for the ESA, we implemented a simpler similarity measure based on the indexed Wikipedia articles. In this measure, the text is submitted as a query to Lucene, and the top k returned articles are taken as the representation of the text. The similarity between the texts is computed as the Jaccard similarity[7] between the sets of these top k articles. In our experiments we tried different values of k ranging from 1 to 1000, but the performance was always inferior compared to the baseline. The reasons for that should further be investigated, but we hypothesize that they may be related to the specificity of our document corpus (ontology texts, as well as the queries), the low coverage of Wikipedia in this domain, and the noise (large amount of irrelevant Wikipedia articles, such as those about TV series, which cover many terms).

5.3 Linear combination of measures

In the task of retrieving the relevant ontology nodes, various text similarity measures can be naturally combined. A simple way to perform this combination is to compute the weighted sum of the similarity scores returned by the different measures, and use this score to determine the relevant documents:

$$score_{S_1,S_2}(q,n) := \alpha \cdot score_{S_1}(q,n) + (1-\alpha) \cdot score_{S_2}(q,n) \tag{6}$$

The optimal value of α can easily be found through the grid search.

In one experiment we linearly combined the baseline method with the Lucene-based similarity measure (Section 5.2). The parameter of the Lucene-based method—the number of top retrieved documents taken as the representation of the text—was set to 100. By ranging the parameter α we were able to achieve the performance of **0.62** (with α=0.3), improving over the baseline by 5 per cent. Figure 1 shows the performance of the linear combination depending on the parameter α. In this case, $\alpha = 0$ corresponds to the Lucene-based method,

while $\alpha = 1$ corresponds to the base line. From the figure we can see that, although the Lucene-based method alone shows inferior performance, it improves the baseline when applied in combination.

As discussed in Section 3.2, combining multiple similarity measures with grid search is impractical. A more appropriate way is apply supervised machine learning, for which a better ground truth dataset is needed.

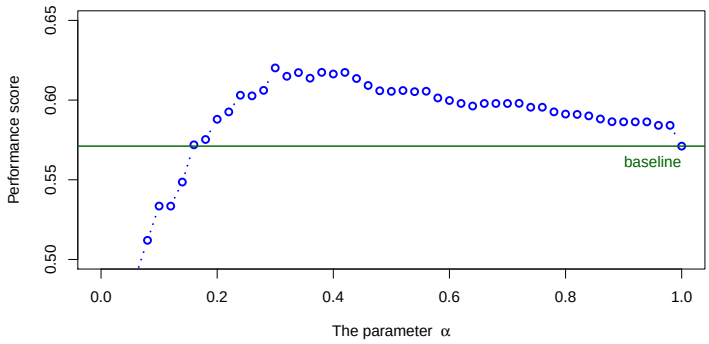

Fig. 1. Performance of the linear combination with the Lucene-based measure.

5.4 Taking the structure of the ontology into account

In the previous approaches, the nodes of the ontology were taken independently from each other. The context of the node, however, carries additional information that can be taken into account to improve the performance in our retrieval task. A simple but effective approach that we implemented is combining the score of a node with the scores of all its ancestors up to the root. We applied geometrically decreasing weights $(1, \alpha, \alpha^2, \alpha^3, \dots)$ to the node, its parent, grand parent, and so on, respectively. The value of the parameter α controls the relative contribution of the node's ancestors into the final score. Thus, the value of $\alpha = 1$ is equivalent to computing the average of the scores along the path from the node to the root, while the value of $\alpha = 0$ corresponds to taking into account only the node's own score. The best value of α (around 0.34), found with the grid search, improved the performance of the baseline by 3 per cent (from **0.57** to **0.60**). Figure 2 shows the performance of this method depending on the value of α.

5.5 Pairwise word relatedness measures

The drawback of the cosine similarity measure is that it treats the terms as independent, that is, ignores the relatedness between the terms. In order to address this drawback, one needs two ingredients: a pairwise term relatedness measure, and a way to aggregate the pairwise term relatedness scores into the final

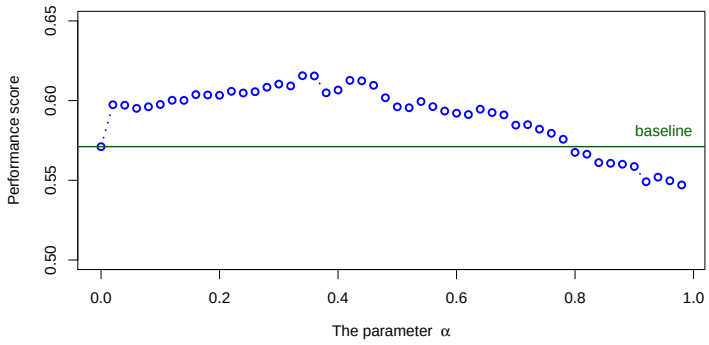

Fig. 2. Performance of the method taking the node's ancestors into account.

between-text similarity score. We implemented three pairwise term relatedness measures based on MultiWordNet[3], and applied two aggregation techniques.

The term relatedness measures were the following: 1) zero-one similarity measure—a trivial measure that returns 1 if the two terms are equal and 0 otherwise; 2) simple synonym-based measure—returns 1 if the two words appear in some synset, 0 otherwise; 3) synset overlap–based—Jaccard similarity between the sets of possible senses of the two terms. As the aggregation techniques we used: a) generalized vector space model (GVSM)[12], b) the technique described by Mihalcea et al.[10]. The combinations that showed reasonable performance in this experiment (though not better the baseline) where the following: zero-one similarity with both aggregation techniques (with GVSM it is equivalent to the raw cosine); and the synset overlap with both aggregation techniques. In these combinations GVSM performed slightly better than the technique of Mihalcea et al.. Table 2 summarizes the performance scores.

Table 2. Performance of the pairwise term relatedness measures.

term relatedness \aggregation technique	Mihalcea et al.	GVSM
zero-one	**0.56**	**0.57**
synonym	**0.44**	**0.40**
synset-overlap	**0.54**	**0.56**

The subsequent error analysis revealed that taking into account word relatedness introduced additional noise that counterbalanced the positive effect. For example, the query including the fragment "bisogno di mangiare" (roughly, "needing to eat") matched the concept node "Motivazione" ("motivation"), because "need" and "motivation" happen to be synonyms. In big texts this kind of problems could be averaged out, but in our case—with small queries and small textual fragments in the ontology—the impact of noise was high.

There are a number of more sophisticated WordNet-based term relatedness measures that could be investigated[4]. Applying these measures in case of Italian language texts requires integrating MultiWordNet with the DKPro similarity library, which we leave for future work.

6 Discussion and future work

We performed an initial investigation on the task of retrieving the relevant nodes from the IEP Ontology, given the input text query. The baseline approach based on the classical bag of words and the cosine similarity showed reasonable performance, but suffered from the classical problems, being limited by the verbatim word matches. We investigated several ways of enhancing the baseline method, and performed their initial evaluation.

Applying the IDF weighting did not improve the performance, when the weights were computed according to the ontology nodes. First, the textual fragments in the ontology are highly specific and focused, which leads to all the meaningful words having comparable weights. Second, viewed as the document corpus, the ontology is small (about 300 concepts as leaf nodes), and contains extremely short texts. Obtaining a (reasonably) large corpus of texts relevant to the domain of the ontology would help computing the useful IDF weights.

The contextual information about the nodes has proven to be promising in our experiments. We implemented a single, though intelligent, way of combining the scores of a node with the scores of its ancestors. It would be interesting to investigate other ways of taking into account the structure of the ontology.

Semantic similarity measures[6] were found to be nontrivial to be fruitfully exploited for our task. One difficulty is the scarce semantic resources for Italian language: for instance, in order to implement an approximation of Explicit Semantic Analysis (ESA)[5], we had to download, pre-process, and index the Italian Wikipedia. Similarly, to utilize a lexico-semantic knowledge base, we had to integrate MultiWordNet[3] with the DKPro similarity library[2]. Both semantic approaches did not surpass the baseline in our initial experiments, however the Wikipedia-based measure improved over the baseline when linearly combined with it. Additional work is thus needed in this direction to explore the full potential of semantic measures: more careful pre-processing of Wikipedia, efficient implementation of ESA, and the implementation of more advanced MultiWordNet-based term relatedness measures.

A distinct problem calling for the semantic methods is the **terminological mismatch** between the teachers' texts for describing the pupils and the textual fragments in the ontology: the former are expressed in the end-user vocabulary (teachers' language, that is a "lay language"), while the latter are expressed using the technical language (domain experts' language). A possible direction for improving the presented results would be the use of existing end-user vocabularies of the domain (including lay synonyms) as a gold standard for our system.

One aspect unexplored in this work is that *multi-sentence texts* can be used as queries. As described in Section 4.1, we experimented with one-sentence queries,

which typically describe a single problem. With a paragraph of text as an input, it will be important to identify the nodes relevant to multiple problems mentioned in the text. This may require a modification to our general approach. In particular, it needs to be investigated, whether the input text should be split into multiple queries, with the results combined at a later stage. Various ways of splitting the input, and combining the results should be evaluated.

Most importantly, further work on this problem requires an **appropriate ground-truth dataset** free of the limitations described in Section 4.1. Such a dataset would enable the principled **machine learning–based scheme** of combining the similarity measures in the optimal way (see Section 3.1). The potential of linearly combining the similarity measures has been confirmed in our experiments. The machine learning approach will allow for exploring the combinations of any number of measures in a sound way, as well as for tuning their parameters. Furthermore, the cross-validation on the ground truth dataset would provide for more reliable performance estimates.

Acknowledgements This work is supported by "ePlanning—Sistema esperto per la creazione di progetti educativo-didattici per alunni con Bisogni Educativi Speciali", funded by the Operational Programme "Fondo Europeo di Sviluppo Regionale (FESR) 2007-2013" of the Province of Trento, Italy.

References

1. Legge-quadro n. 104 per l'assistenza, l'integrazione sociale e i diritti delle persone handicappate, 5 February 1992.
2. Daniel Bär, Torsten Zesch, and Iryna Gurevych. Dkpro similarity: An open source framework for text similarity. In *ACL*, 2013.
3. L. Bentivogli and E. Pianta. Exploiting parallel texts in the creation of multilingual semantically annotated resources: The multisemcor corpus. *Nat. Lang. Eng.*, 11(3):247–261, September 2005.
4. Alexander Budanitsky and Graeme Hirst. Evaluating WordNet-based measures of lexical semantic relatedness. *Comput. Linguist.*, 32(1):13–47, March 2006.
5. Evgeniy Gabrilovich and Shaul Markovitch. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. IJCAI, 2007.
6. Sébastien Harispe, Sylvie Ranwez, Stefan Janaqi, and Jacky Montmain. Semantic measures for the comparison of units of language, concepts or entities from text and knowledge base analysis. *CoRR*, abs/1310.1285, 2013.
7. P. Jaccard. The distribution of the flora in the alpine zone. *New Phytologist*, 1912.
8. Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*. Springer, 2013.
9. Youngjoong Ko. A study of term weighting schemes using class information for text classification. SIGIR. ACM, 2012.
10. Rada Mihalcea, Courtney Corley, and Carlo Strapparava. Corpus-based and knowledge-based measures of text semantic similarity. AAAI. AAAI Press, 2006.
11. Amit Singhal. Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.*, 24(4):35–43, 2001.
12. S. K. M. Wong, Wojciech Ziarko, and Patrick C. N. Wong. Generalized vector spaces model in information retrieval. SIGIR '85. ACM, 1985.