J Data Semant
DOI 10.1007/s13740-014-0045-5

ORIGINAL ARTICLE

Word-Sense Disambiguation for Ontology Mapping: Concept Disambiguation using Virtual Documents and Information Retrieval Techniques

Frederik C. Schadd · Nico Roos

Received: 24 April 2013 / Revised: 1 August 2014 / Accepted: 14 September 2014

Abstract Ontology mapping is a crucial task for the facilitation of information exchange and data integration. A mapping system can use a variety of similarity measures to determine concept correspondences. This paper proposes the integration of word-sense disambiguation techniques into lexical similarity measures. We propose a disambiguation methodology which entails the creation of virtual documents from concept and sense definitions, including their neighbourhoods. The specific terms are weighted according to their origin within their respective ontology. The document similarities between the concept document and sense documents are used to disambiguate the concept meanings. First, we evaluate to what extent the proposed disambiguation method can improve the performance of a lexical similarity metric. We observe that the disambiguation method improves the performance of each tested lexical similarity metric. Next, we demonstrate the potential of a mapping system utilizing the proposed approach through the comparison with contemporary ontology mapping systems. We observe a high performance on a real-world data set. Finally, we evaluate how the application of several term-weighting techniques on the virtual documents can affect the quality of the generated alignments. Here, we observe that weighting terms according to their ontology origin leads to the highest performance.

F. C. Schadd (✉)
Maastricht University, St. Servaasklooster 39,
6211 TE Maastricht, The Netherlands
e-mail: frederik.schadd@maastrichtuniversity.nl

N. Roos
Maastricht University, Bouillonstraat 8-10,
6211 LN Maastricht, The Netherlands
e-mail: roos@maastrichtuniversity.nl

Published online: 30 September 2014

Keywords Semantic web · Ontology mapping · Lexical similarity · Word-sense disambiguation · Virtual document

1 Introduction

Ontology mapping is an integral process for the facilitation on information exchange. Originating from the field of schema matching, this task is essential in numerous database-related applications, such as data integration, schema evolution and migration, data warehousing and web site creation and management [33,63]. This process has recently seen a rise in importance for ontology-based information systems, allowing functionalities like search [34] or querying [7,25] over heterogeneous data sources, or the integration of several ontology-based knowledge systems [8,77]. In this domain, given two specifications of knowledge domains consisting of a list of interrelated concepts and their meta-information, the main task entails the identification of concepts which denote the same meaning, and thus are used to model the same kind of information.

The need for robust mapping systems and possible shortcomings has been established [71,72]; ongoing research aims to overcome the current limitations of mapping systems to realize the automatic and accurate mapping of ontology concept under a variety of conditions [70,80].

This paper is an extended and updated version of an invited workshop paper [67]. The main contributions of this paper can be summarized as follows:

– We identify a lack in contemporary implementations of lexical similarity metrics, where senses which do not denote the correct meaning of an ontology concept are used for the similarity computation.

– We propose that lexical similarity metrics should perform the task of concept disambiguation prior to the similarity calculation.
– We present a method of concept disambiguation based on the similarity of concept and sense definitions.
– We propose the application of virtual documents and information retrieval techniques to determine the similarity between concepts and senses.
– We evaluate experimentally to what extent concept disambiguation can improve the performance of a lexical similarity metric.
– We establish to what extent a system using the proposed technique can compete with state-of-the-art mapping systems through a comparison evaluation.
– We determine which terms of concept descriptions should receive preferential weighting through the use of parameter optimization.
– We evaluate the impact that different virtual document weighting techniques can have on the mapping process. Furthermore, this impact is contrasted against the observed performance of a profile similarity utilizing the same document model.

The remainder of this paper is organized as follows. Section 2 will introduce the reader into the domain of ontology mapping and detail the necessary background knowledge. Section 3 discusses related research. Section 4 introduces possible applications of virtual documents in ontology mapping, introduces a document model and discusses possible weighting approaches. Section 5 details the proposed method of concept disambiguation and how virtual documents can be exploited in this process. The experimental results are presented in Sect. 6. Section 7 presents the conclusions of this paper and suggestions for future work.

2 Background Information

2.1 Ontology Mapping

Ontology mapping is the essential process facilitating the exchange of information between heterogeneous data sources. Here, each source utilizes a different ontology to model its data, which can lead to differences with regard to the syntax of the ontology, concept naming and structuring and the granularity with which the knowledge domain is modelled. Euzenat et al. [12] identified three main heterogeneity categories as terminological, conceptual and semiotic heterogeneities. Given two ontologies, these heterogeneities need to be resolved, which in turn allows for the exchange of information between any knowledge system which uses any of the two given ontologies to model its data. This is achieved by mapping concepts which model the same data, which are compiled into a list of correspondences, referred to as an *alignment*.

Formally, we define ontology mapping as a process which takes as minimal input two ontologies O_1 and O_2 and produces an output alignment A'.

Further, this process can take as input an already existing alignment A, external resources r and a set of parameters p. The pre-existing alignment can originate from a different system, thus allowing the combination of two systems in a cascade arrangement, or from the same system, allowing the possibility of designing an iterative mapping process. The set of parameters p incorporates any parameter which influences the mapping process, such as settings, weights or thresholds. While r is broadly defined, in practise the most commonly used resources are linguistic or domain thesauri and auxiliary ontologies.

Following established work [13,18], we define a correspondence between entities of two ontologies O_1 and O_2 as a 5-tuple $< id, e_1, e_2, q, c >$ such that:

– id is a unique identifier allowing the referral to specific correspondences.
– e_1 is a reference to an entity originating from the first ontology. Commonly a URI is used as referral to a specific entity.
– e_2 is a reference to an entity originating from the second ontology.
– q denotes the semantic relation between e_1 and e_2. Several types of relations can be modelled, such as generalization ($\sqsupseteq$), disjointness ($\perp$), overlapping ($\sqcap$) and equivalence ($\equiv$).
– c is a confidence value in the interval [0, 1], which is used to express the certainty that the particular relation holds.

Given the definition of a correspondence, an alignment A between two ontologies O_1 and O_2 is defined as a set of correspondences where each correspondence contains a reference to one entity of O_1 and one entity of O_2.

Figure 1 visualizes the basic architecture of an ontology mapping framework. The first essential process is the preparation of the input ontologies. Here, the two input ontologies are parsed into a common format in case one of these is formulated using a non-standard syntax or modelling language. Moreover, the input ontologies are pre-processed by for instance applying natural language processing techniques to the concept names. The second essential process is the computation of the pairwise concept similarities. For each of n given similarity measures, a correspondence matrix is computed indicating all pairwise similarities according to that particular measure. These matrices are then assembled into a similarity cube. Using an aggregation method, the similarity

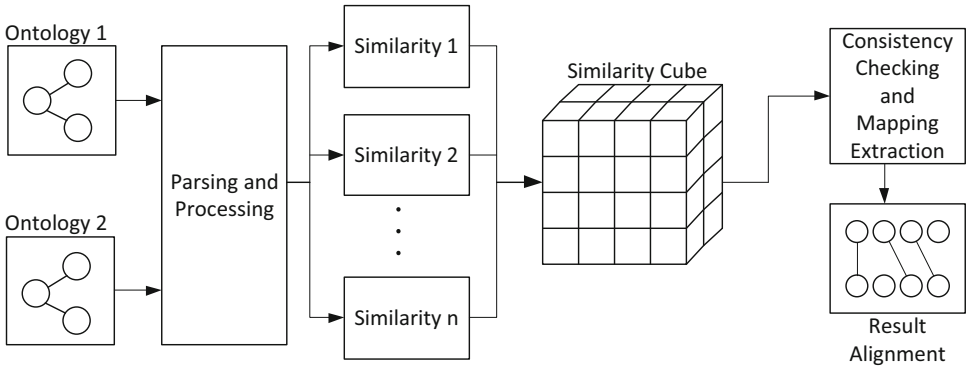

Fig. 1 Basic architecture of an ontology mapping framework

cube is then transformed into a two-dimensional similarity matrix. During the last process, a selection method is applied which extracts the output alignment between the two ontologies from the aggregated matrix. On this alignment one can apply reasoning techniques, most notably consistency checking, to improve the quality of the alignment.

An important design decision for the creation of a mapping system is the selection of similarity measures. There exists a wide variety of measures which can be described by the type of ontological information which they exploit and the kind of techniques which they apply [70].

The main focus of this paper lies on lexical similarities, which will be explained in more detail in the next subsection.

2.2 Lexical Similarity Measure

Lexical similarity measure (LSM) are commonly applied metrics in ontology mapping systems. These exploit externally available knowledge bases which can be modelled in ontological or non-ontological form, for instance by utilizing databases. Such a knowledge base contains a list of concepts describing the particular domain that is being modelled. Each concept description contains various kinds of information, such as synonyms and written explanations of that concept. If such a description does contain at least a list of synonyms, it is also often referred to as *synset* (synonym-set). Another important feature of a knowledge base is that each concept is also linked to other concepts using various semantic relations, thus creating a large relational structure. A LSM exploits these large structures by linking ontology concepts to the nodes in the external knowledge base, such that the proximity of concepts associated with source and target concepts provides an indication to their similarity. One can here distinguish between *semantic relatedness* and *semantic similarity* [75], where the semantic relatedness denotes the measured proximity by exploring all given relations, whereas the semantic similarity expresses the proximity using only *is-a* relations. Whether a LSM determines the *relatedness* or *similarity* depends on the utilized metric which expresses the proximity [5,19], since the definitions of these metrics typically also define which relations are exploited. For this research, as further detailed in Subsect. 5.3, the applied metric utilizes only *is-a* relations, rendering the base LSM which our approach intends to improve, as a measure of semantic similarity.

There exist several lexical knowledge bases which can be used as a resource for a LSM. These originate from different research efforts and were all developed with different capabilities, which can roughly be grouped as follows:

Global/Cross-Domain Knowledge Resources of this category intend model a multitude of domains, such that the similarity between concepts can be identified even if these are generally categorized in different domains. Examples of global resources are WordNet [50] and YAGO [76].

Domain Knowledge These resources intend to model common knowledge of a single specified domain. Typically, these domains are not very broadly defined, however, they are usually modelled in great detail. An example of such a resource is UMLS [4], which models the biomedical domain.

Abstract Upper Ontology This group of resources primarily focus on the creation of an abstract ontology using an upper-level list of concept descriptions. Such an ontology can then serve as base for domain-specific resources. SUMO [55], MILO [56], Cyc [44] and OpenCyc [73] can be categorized as abstract upper ontologies.

Multi-lingual When mapping ontologies, it can occur that some concept descriptions are formulated in a different language. In these situations, mono-lingual resources are insufficiently applicable, necessitating the usage of multi-lingual resources, e.g. UWN [11] or BabelNet [53].

LSMs are a powerful metric and are commonly used in contemporary state-of-the-art ontology mapping systems [32,66,70], with WordNet being the most widely used resource as basis. However, a common occurrence in concepts formulated using natural language is word-sense ambiguity. This entails that a word can have multiple and possibly vastly different meanings, such that one must eliminate all meanings which do not adequately confer the intended meaning of the word.

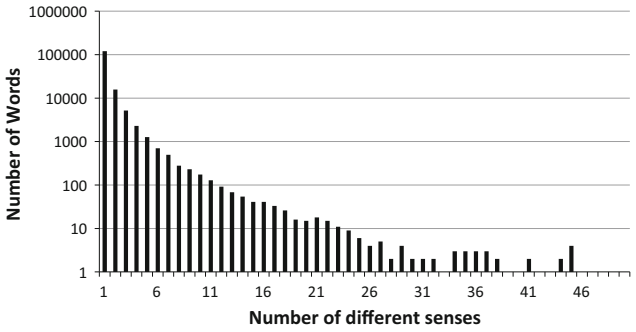

Fig. 2 Histogram showing the number of words in WordNet (*y*-axis) that have a specific number of senses (*x*-axis)

This task, while at a glance quite intuitive for a human can be deceptively difficult. Given the word *house* for instance, the intended meaning might be obvious to a human reader, however, this word has 14 different meanings listed in WordNet, such that an accurate identification of the correct sense is necessary to obtain accurate results. The histogram in Fig. 2 indicates the extent of such situations occurring within WordNet [50]. Here, all unique words that occur in WordNet have been gathered and binned according to how many different meanings each word describes.

One can see from Fig. 2 that while there is a large number of words with only one meaning, there is a significant proportion of words which do have more than one meaning and hence are ambiguous. The general working hypothesis, also adhered in this paper, is that a word in a given context has only a single correct sense. The rejection of this hypothesis, the acknowledgement of polysemous words, is an emerging field of research for which new approaches are emerging [10]. Ultimately a LSM has to calculate the similarity between two sets of senses, where the assumption whether these sets can contain multiple correct senses may influence the choice in specific employed techniques, including disambiguation methods.

LSMs can incorporate polysemous concepts by for instance calculating an aggregate similarity between these sets of senses [10,14,61]. However, if a domain expert determines that the concepts in the ontology are not polysemous, one can adapt the aggregation step by for instance only utilizing the maximum pairwise similarity [14] between sets of senses or by selecting the predominant sense as determined by a given corpus [45]. The inclusion of a word-sense disambiguation technique in a LSM, which this paper proposes, is likely to improve their accuracy.

2.3 Word-Sense Disambiguation

Word-Sense Disambiguation (WSD) can be described as the automatic identification of the correct sense(s) of a given word using the information in the proximity of that word as context. While in many works only one sense is associated with each word, we define WSD as a process which filters a set of possible candidate senses. The resulting sets may contain multiple senses if desired by the expert designing the system, for instance to accommodate polysemous words. In the classical problem of disambiguating words occurring in natural language, the available context information is a body of text co-occurring with the target word [52]. Depending on the input document or the applied approach, this body of context information can be limited to the sentence in which the target word appears or extended over the entire input document. The available context information originating from an ontology is different compared to a natural language document. In an ontology natural language is a rare occurrence and usually limited to brief concept descriptions in the form of annotations. Hence, context information must be extracted from the entire concept description, its associated properties and other related concepts.

Originally, WSD has been perceived as a fundamental task to perform machine translation [40,82]. Here, the establishment of accurate word senses is a requirement for the selection of correct word translations from a multi-lingual dictionary. While research into WSD halted for a decade after its acknowledged hardness [3], it has been re-instigated after [83] tackled this problem using formal semantics to achieve a computer understanding of natural language. For a more comprehensive overview of the history of WSD we suggest the reader consult the work of Ide and Véronis [28].

Many different approaches to WSD have been developed over the past decades. Due to the prevalence of applied machine-learning techniques, three general categories of approaches have emerged:

Supervised Disambiguation One can formulate WSD as a classification problem. Here, a training set is created by tagging sentences with the correct senses of its contained words. Once the training set has reached a sufficient size, one can use this as basis for a supervised classification method. Examples of such methods are decision lists, decision trees, Naive Bayes classifier, Neural-Networks, instance-based methods such as the kNN approach and ensemble methods which combine different classifiers [51,52].

Unsupervised Disambiguation These methods have the advantage that they do not rely on the presence of a manually annotated training set, a situation which is also referred to as the knowledge acquisition bottleneck [17]. However, unsupervised methods share the same intuition behind supervised methods, which is that words of the same sense co-occur alongside the same set of words [59]. These rely on clustering methods where each cluster denotes a different word sense.

Knowledge-based Disambiguation Instead of applying classification techniques, knowledge-based methods exploit

available knowledge resources, such as dictionaries, databases or ontologies, to determine the sense of a word [49]. These techniques are related to LSMs in that they often exploit the same knowledge resources. This group of techniques will be further discussed in Subsect 3.2.

For a more comprehensive survey of disambiguation techniques we suggest the reader consult the excellent survey by Navigli [52].

While originally conceived for the purpose of machine translation, WSD techniques have been applied in a variety of tasks [28]. In the field of information retrieval, one can apply WSD to eliminate search results in which at least some of the query keywords occur, but in a different sense than the given query [69]. This would lead to a reduction of false positives and hence increase the performance of the retrieval system.

WSD can also aid in the field of content and thematic analysis [39]. Here, the aim is to classify a given text into thematic categories, such as traditional (e.g. judicial, religious), practical (e.g. business), emotional (e.g. leisure, fiction) and analytical (e.g. science) texts. Given a corpus of training data, one can create a profile for each defined category consisting of the distributions of types of words over a text.

In the field of grammatical analysis WSD is required to correctly identify the grammatical type of ambiguous words [43]. WSD can also aid a speech synthesis system such that ambiguous words are phoneticised more accurately [74]. Yarowsky [84] applied WSD techniques for text processing purposes with the aim to automatically identify spelling errors.

2.4 Virtual Documents

The general definition of a virtual document (VD) [81] is any document for which no persistent state exists, such that some or all instances of the given document are generated at runtime. These stem from an emerging need for document to be more interactive and individualized, which is most prominently seen on the internet.

In the domain of lexical similarity metrics the basic data structure used for the creation of a virtual document is a linked-data model. It consists of different types of binary relations that relate concepts, i.e. a graph. RDF [37] is an example of a linked-data model, which can be used to denote an ontology according to the OWL specification [47]. A key feature of a linked-data model is that it not only allows the extraction of literal data for a given concept, but also enables the exploration of concepts that are related to that particular concept. From the linked-data resource information is gathered and stored in a document with the intention that the content of that document can be interpreted as a semantic representation of the meaning of a specific ontology concept. A specific model for the creation of such a virtual document will be presented in Subsect. 4.1.

3 Related Work

3.1 Ontology Mapping Approaches

Numerous approaches and systems have been developed, of which some are of particular interest due to the high-quality alignments they produce or specific techniques they employ. The framework AgreementMaker [9] matches ontologies using a layered approach. In the initial layer, similarity matrices are computed using syntactic and lexical similarities based on WordNet, among others, which are then used to create a set of mappings. Further iterations in subsequent layers refine the existing mappings using structural properties to create new mappings. After a sufficient amount of iterations, multiple computed mappings are selected and combined to form the final mapping.

The framework ASMOV [30] is capable of using general lexical ontologies, such as WordNet, as well as domain specific ontologies in its matching procedure. After creating a mapping using a set of similarity measures, a semantic verification process is performed to remove correspondences that lead to inferences which cannot be verified or are unlikely to be satisfied given the information present in the ontologies.

As evidenced by the results of the 2012 Ontology Alignment Evaluation Initiative [1], one of the current state-of-the-art ontology mapping system is YAM++, developed by Ngo et al. [54]. This system combines machine-learning and information retrieval techniques on the element level and similarity propagation techniques on the structure level to derive a mapping, to which consistency checking is applied to further increase the quality.

A notable feature of the Anchor-PROMPT system [57] is its exploitation of input alignments. Here, the concept pairs of the input alignment are interpreted as anchors, which serve as start and end point of a path-exploration approach. The taxonomies of both ontologies are explored in parallel, with each concept pair encountered during this traversal between anchors receiving an increased score. Concept pairs which have been encountered frequently during the many traversals are thus more likely to correspond with each other.

The S-Match approach [20] is a notable example for a semantic mapping approach. Here, concept sets are computed from the labels and nodes of the ontologies such that the comparison between the extensions of the nodes indicates which type of relation holds between the nodes. For a more in-depth survey of existing ontology mapping systems, we suggest the excellent surveys of Shvaiko and Euzenat [70] and Saruladha et al. [66].

Recently, several systems implemented approaches that directly compare concept profiles to linguistically determine concept similarities. The PRIOR and CroMatcher frameworks [23,42] achieve this by weighting the profiles using TF-IDF weights, whereas the Falcon-AO framework [27,62] applies a virtual document model with a parametrized weighting scheme. The web-based framework WeSeE [58] uses the concept profiles as input for web queries, where the resulting information snippets are compared according to their overlap. The techniques of these works share some similarities with parts of this work, specifically regarding the creation of profiles. However, these approaches either compare the profiles directly or use them in a different context.

3.2 Methods of Word-Sense Disambiguation

There exists a notable spectrum of word-sense disambiguation techniques, which have been used for varying purposes, however, certain techniques stand out due to their applicability to this domain. The method of context clustering [68] can be used to exploit large amounts of labelled training data. Here, co-occurrences with a target word are modelled as a vector in a word space and grouped into clusters according to their labelled word sense. Given a new occurrence of the given word, one can identify its sense by modelling a new context vector from its neighbourhood and classifying it using the created word-sense clusters. This can be done for instance by determining the centroid vector of each cluster and computing the vector distance for each centroid vector.

A more linguistic approach can be achieved through the application of selectional preferences [26]. By determining the grammatical types of words within a sentence, one can limit the amount of possible sense by imposing limitation according to the grammatical or semantic context of a particular word. Such a technique can be especially relevant for the mapping of ontology properties, since property names or labels can contain combination of grammatical types, e.g. nouns, verbs or adjectives, where its proper classification can improve their semantic annotations.

A very effective group of disambiguation methods is based on glossary overlap. The techniques of this group are essentially knowledge-based methods. These rely on the presence of a detailed corpus of word senses that include their descriptions in natural language. Determining the overlap between the set of words occurring in context of a target word and the different sense descriptions of that word within the given corpus can be used to determine its proper sense. This type of method has been pioneered by Lesk [38], which can be improved by incorporating the descriptions of words that are related to the different possible senses [2].

Cross-lingual word-sense disambiguation is another knowledge-based approach which exploits multi-lingual corpora [64]. A target word is translated into several distinct languages such that the intended sense is likely the one whose meaning has been preserved for the majority of the used languages.

Structural methods [60] exploit the concept structure of a given corpus. This is achieved by applying a similarity metric between word senses, such that the disambiguated sense of a word from a text is the particular sense which maximizes the aggregate similarities between all possible senses of the words occurring in the text and itself.

Budanitsky and Hirst [5] evaluated five different sense-similarity measures which serve as the basis for structural disambiguation methods; however, these are also applicable to lexical similarities between ontology concepts. For a more in-depth survey of word-sense disambiguation methods, especially the types which do not strongly relate to the techniques applied in this research, we suggest the reader consult the comprehensive survey by Navigli [52].

3.3 Word-Sense Disambiguation in Ontology Mapping

Given the large set of possible techniques originating from many different research areas that can be applied to the process of ontology mapping, only limited research has been performed into applying word-sense disambiguation techniques. Some of this research involves the creation of annotation frameworks, which can facilitate a standardized format of lexical concept annotations and can even provide a more fine-grained annotation. An example of such a framework is the work of Buitelaar et al. [6], who proposed a linguistic labelling system for the annotation of ontology concepts. While the primary intent of this system was the facilitation of ontology learning and natural language generation from ontologies, the linguistic meta-information of this system can also be used to disambiguate word senses, for instance by extracting selectional preferences generated from these annotations.

McCrae et al. [46] proposed a common model for linking different lexical resources to ontology concepts. This model not only includes constructs modelling terms and their senses, but also models the morphosyntactic properties of term which would allow for a more fine-grained annotation of ontology concepts.

Some ontology mapping systems apply WSD to aid their lexical similarity measures. The AUTOMS system [35], which is designed for the task of ontology merging, employs a technique called HCONE-Merge [36]. Part of this technique involves the process of *latent semantic indexing* (LSI), which is used to associate senses with the given ontology concepts. The approach assumes that concepts are monosemous. Ontology concepts are associated with the sense which resulted in the highest score when querying a latent semantic space using a binary query. This space is created by performing singular value decomposition on the sense descriptions.

Po and Sorrentino [61] introduced a probabilistic WSD method which has been included in the *AgreementMaker* system [10]. Here, each ontology concept is annotated with a set of possible senses, where each sense is annotated with a probability value. This probability value is determined by combining the results of several WSD techniques, i.e. structural disambiguation, domain disambiguation and first-sense heuristic, using the Dempster-Shafer Theory. This method is related to our work due to its application of the basic *Lesk* method as one of the different WSD techniques. The approach of our paper also relies on the principle behind the *Lesk* method, such that substituting our approach with the basic *Lesk* method could improve the WSD accuracy of the *AgreementMaker* system.

4 Virtual Documents in Ontology Mapping

As stated in Subsect. 2.2, a feature lacking in contemporary lexical similarity measures is the disambiguation of concept senses. A knowledge-based approach is the most logical choice since the result of the disambiguation process is required to be in the form of identified senses from the given lexical resource. If one were to choose a different type of approach, it would be necessary to map the disambiguation result, for instance an identified context cluster, to a lexical sense definition which adds more degrees of uncertainty to the disambiguation process. From the knowledge-based approaches, a method based on glossary overlap is the most suited candidate. This is because these methods generally outperform selectional preference-based approaches [52].

In glossary-overlap methods, the context of a target word is gathered and matched against the context of different sense definitions. Hence, to realize the gathering and mapping of context information one can utilize virtual documents. Given the domain of ontology mapping, it is therefore possible to utilize virtual documents in two different ways, one of which being the primary contribution of this paper:

Profile Similarity Given two ontology concepts, one can determine their similarity by creating their respective virtual documents, in the literature also referred to as profiles [24,41], and applying a document similarity metric [62]. Hence, the applied metric can be treated as a concept similarity and directly integrated into a ontology mapping system.
Concept Sense Disambiguation This paper proposes the application of virtual documents within a lexical similarity metric of an ontology mapping system. Here, virtual documents can be used to perform WSD on the ontology concepts, more specifically the concept names, to determine their intended meaning. The intention of this is that using only senses resulting from WSD in a LSM will cause the LSM to produce more accurate similarity measures.

The virtual document model serves as context gathering method. Once created, it is necessary to process the virtual documents to facilitate the computation of document similarities. This is done by transforming the documents into a vector-space model [65], where each dimension in this high-dimensional space represents the occurrence of a specific term. A given document is formulated as a vector of this space, such that the values for each dimension represent its respective term-frequency (TF) within that document. By comparing these document vectors using a specific metric, it becomes possible to asses to what extent two arbitrary documents address the same topic. In information retrieval, this document metric is used to retrieve relevant documents.

In Subsect. 4.1, we will present the used document model for this research, including a parametrized weighting scheme. Subsection 4.2 will discuss an alternative weighting scheme.

4.1 Ontology Document Model

We will provide a generalized description of the creation of a virtual document based on established research [62]. The generalization has the purpose of providing a description that is not only applicable to an OWL/RDF ontology like the description given in the work by Qu et al. [62], but also to non-ontological knowledge sources. While a variety of external resources can be utilized, for this research we will use the most widely utilized resource, which is WordNet [50]. To provide the functions that are used to create a virtual document, the following terminology is used:

Synset: Basic element within a knowledge source, used to denote a specific sense using a list of synonyms. Synsets are related to other synsets by different semantic relations, such as hyponymy and holonymy.
Concept: A named entity in the linked-data model. A concept denotes a named class or property given an ontology, and a synset when referring to WordNet.
Link: A basic component of a linked-data model for relating elements. A link is directed, originating from a source and pointing towards a target, such that the type of the link indicates what relation holds between the two elements. An example of a link is a triplet in an RDF graph.
sou(s), type(s), tar(s): The source element, type and target element of a link s, respectively. Within the RDF model, these three elements of a link are also known as the subject, predicate and object of a triplet.
Collection of words: A list of unique words where each word has a corresponding weight in the form of a rational number.
+: Operator denoting the merging of two collections of words.
$\times$: Operator denoting multiplication.

A concept definition within a linked-data model contains different types of literal data, such as a name, different labels, annotations and comments. The RDF model expresses some of these values using the *rdfs:label*, *rdfs:comment* relations. Concept descriptions in WordNet have similar capacities, but the labels of a concepts are referred to as its synonyms and the comments of a concept are linked via the glossary relation.

Definition 1 Let ω be a concept of a linked-data model, the description of ω is a collection of words defined by (1):

$$\begin{aligned} Des(e) = {} & \alpha_1 \times \text{collection of words in the name of } \omega \\ & + \alpha_2 \times \text{collection of words in the labels of } \omega \\ & + \alpha_3 \times \text{collection of words in the comments of } \omega \\ & + \alpha_4 \times \text{collection of words in the annotations of } \omega \end{aligned} \tag{1}$$

where each α_1, α_2, α_3 and α_4 is a rational number in [0, 1], such that words can be weighed according to their origin.

Next to accumulating information that is directly related to a specific concept, one can also include the descriptions of neighbouring concepts that are associated with that concept via a link. Such a link can be a standard relation that is defined in the linked-data model, for instance the specialization relation and also an ontology-defined property if the used syntax allows the property to occur as a predicate. While theoretically the presented model would also allow instances to be included if these are present in the ontology, it is very unlikely that a given knowledge resource contains similar specific instance information for which an overlap can be determined. Hence, given instances are filtered from the ontologies before the creation of the documents.

The OWL language supports the inclusion of blank-node concepts which allow complex logical expressions to be included in concept definitions. However, since not all knowledge resources support the blank-node functionality, meaning anonymous concepts defined using a property restriction, among which WordNet, these are omitted in our generalization. For more information on how to include blank nodes in the description, consult the work by Qu et al. [62].

To explore neighbouring concepts, three neighbour operations are defined. $SON(\omega)$ denotes the set of concepts that occur in any link for which ω is the source of that link. Likewise $TYN(\omega)$ denotes the set of concepts that occur in any link for which ω is the type, or predicate, of that link and $TAN(\omega)$ denotes the set of concepts that occur in any link for which ω is the target. WordNet contains inverse relations, such as hypernym being the inverse of the hyponym relation. When faced with two relations with one being the inverse of the other, only one of the two should be used such that descriptions of neighbours are not included twice in the virtual document. The formal definition of the neighbour operators is given below.

Definition 2 Let ω be a named concept and s be a variable representing an arbitrary link. The set of source neighbours $SON(\omega)$ is defined by (2), the set of type neighbours $TYN(\omega)$of ω is defined by (3) and the set of target neighbours $TAN(\omega)$ of ω is defined by (4).

$$SON(\omega) = \bigcup_{sou(s)=\omega} \{type(s), tar(s)\} \tag{2}$$

$$TYN(\omega) = \bigcup_{type(s)=\omega} \{sou(s), tar(s)\} \tag{3}$$

$$TAN(\omega) = \bigcup_{tar(s)=\omega} \{sou(s), type(s)\} \tag{4}$$

Given the previous definitions, the definition of a virtual document of a specific concept can be formulated as follows.

Definition 3 Let ω be a concept of a linked-data model. The virtual document of ω, denoted as $VD(\omega)$, is defined by (5):

$$\begin{aligned} VD(\omega) = Des(\omega) + \beta_1 \times & \sum_{\omega' \in SON(\omega)} Des(\omega') \\ + \beta_2 \times & \sum_{\omega' \in TYN(\omega)} Des(\omega') + \beta_3 \\ \times & \sum_{\omega' \in TAN(\omega)} Des(\omega') \end{aligned} \tag{5}$$

Here, β_1, β_2 and β_3 are rational numbers in [0, 1]. This makes it possible to allocate a different weight to the descriptions of neighbouring concepts of ω compared to the description of the concept ω itself.

We will provide a brief example for the resulting term weights in a virtual document that is created using this model. For this, we will use an example ontology provided in Fig. 3.

Suppose one would want to construct a virtual document representing the concept *Car*. The term weights of this document are determined through the merger of the description

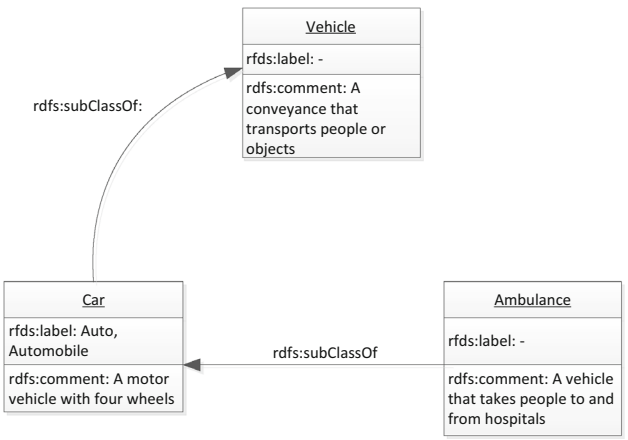

Fig. 3 Example ontology for the construction of a virtual document

Table 1 Term weights for the document representing the concept *Car*, according to the example ontology displayed in Fig. 3

Term	Weight	Term	Weight
a	$\alpha_3 + \beta_1 \times \alpha_3 + \beta_3 \times \alpha_3$	Motor	α_3
Ambulance	$\beta_3 \times \alpha_1$	Objects	$\beta_1 \times \alpha_3$
Auto	α_2	People	$\beta_1 \times \alpha_3 + \beta_3 \times \alpha_3$
Automobile	α_2	Takes	$\beta_3 \times \alpha_3$
Car	α_1	That	$\beta_1 \times \alpha_3 + \beta_3 \times \alpha_3$
Conveyance	$\beta_1 \times \alpha_3$	Transports	$\beta_1 \times \alpha_3$
Four	α_3	To	$\beta_3 \times \alpha_3$
From	$\beta_3 \times \alpha_3$	Vehicle	$\alpha_3 + \beta_1 \times \alpha_1 + \beta_3 \times \alpha_3$
Hospitals	$\beta_3 \times \alpha_3$	Wheels	α_3

of the concept *Car* and the weighted descriptions of the concepts *Vehicle* and *Ambulance*. The term weight of the word *car* would be α_1, since the term only occurs in the name of the concept *Car*. The term *vehicle* would receive the weight $\alpha_3+\beta_1\times\alpha_1+\beta_3\times\alpha_3$. This is because the term occurs in three locations in the neighbourhood of the concept *Car*, once in a comment of the given concept, once in the name of a source neighbour and once in a comment of a target neighbour. The sum of these particular occurrences hence forms the final term weight for this word. The full list of term weights of the document representing the example concept *Car* can be viewed in Table 1. For the sake of demonstration the list also includes the weights of stop-words.

4.2 Term-Frequency Weighting

Instead of weighting terms in a virtual document according to their origin from within their respective ontology, it is also possible to treat a virtual document as a standard natural language document.

This allows for the application of well-known weighting techniques originating from the field of information retrieval.

The most prominent weighting technique is TF-IDF [31]. This method relates the term-frequency (TF) of a word within a document with the inverse document frequency (IDF), which expresses in how many of the registered documents a term occurs. Given a collection of documents D and an arbitrary term t and the term-frequency of the term t within document d_x as $tf(t, d_x)$, the TF-IDF weight of the term t within document d_x is then specified as follows:

$$tf\text{-}idf(t, d_x, D) = tf(t, d_x) \times \log \frac{|D|}{|\{d \in D : t \in d\}|} \tag{6}$$

However, given the availability of ontological background knowledge which can aid the document creation process, it is questionable whether the application of a weighting scheme which is designed to be applied on texts formulated in natural language outperforms the weighting functionality supplied by the virtual document model. We will compare the performance of TF-IDF with the virtual document model in Sect. 6.3.

5 Concept Sense Disambiguation

Our proposed approach aims at improving matchers applying lexical similarity metrics. For this resarch, the applied LSM will use WordNet as knowledge resource. The synsets of WordNet will be used to annotate the meanings of ontology concepts and express their semantic relatedness.

The goal of our approach is to automatically identify the correct senses for each concept of an ontology by applying information retrieval techniques on virtual documents that have been created using either ontology concepts or word-sense entries from the knowledge resource. Given two ontologies O_1 and O_2 that are to be matched, O_1 contains the sets of entities $E_x^1 = \{e_1^1, e_2^1, ..., e_m^1\}$, where x distinguishes between the set of classes, properties or instances, O_2 contains the sets of entities $E_x^2 = \{e_1^2, e_2^2, ..., e_n^2\}$, and $C(e)$ denotes a collection of synsets representing entity e. The main steps of our approach, performed separately for classes, properties and instances, can be described as follows:

1. For every entity e in E_x^i, compute its corresponding set $C(e)$ by performing the following procedure:
   (a) Assemble the set $C(e)$ with synsets that might denote the meaning of entity e.
   (b) Create a virtual document of e, and a virtual document for every synset in $C(e)$.
   (c) Calculate the document similarities between the virtual document denoting e and the different virtual documents originating from $C(e)$.
   (d) Discard all synsets from $C(e)$ that resulted in a low similarity score with the virtual document of e, using some selection procedure.
2. Compute the lexical similarity for all combinations of $e^1 \in E_x^1$ and $e^2 \in E_x^2$ using the processed collections $C(e^1)$ and $C(e^2)$.

The essential operation of the approach is the exclusion of synsets from the lexical similarity calculation. This is determined using the document similarities between the virtual documents originating from the synsets and the virtual document originating from the ontology concepts. Figure 4 illustrates steps 1.b–2 of our approach for two arbitrary ontology entities e^1 and e^2:

Once the similarity matrix, meaning all pairwise similarities between the entities of both ontologies, is computed, the

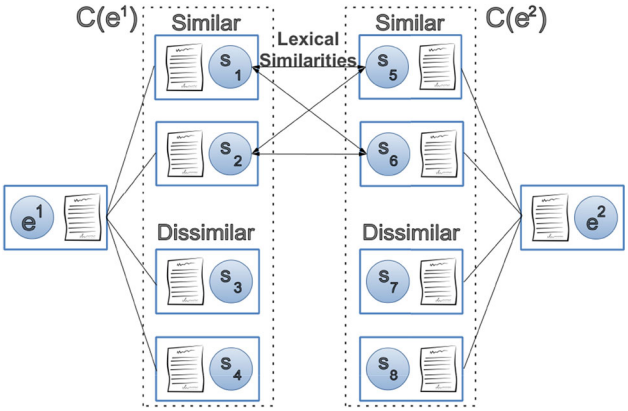

Fig. 4 Visualization of step 1.b–2 of the proposed approach for any entity e^1 from ontology O_1 and any entity e^2 from ontology 2

final alignment of the mapping process can be extracted or the matrix can be combined with other similarity matrices.

5.1 Synset Selection and Virtual Document Similarity

The initial step of the approach entails the allocation of synsets that might denote the meaning of a concept. The name of the concept, meaning the fragment of its URI, and alternate labels, when provided, are used for this purpose. While ideally one would prefer synsets which contain an exact match of the concept name or label, precautions must be made for the eventuality that no exact match can be found. For this research, several pre-processing methods have been applied such as the removal of special characters, stop-word removal and tokenization. It is possible to enhance these precautions further by for instance the application of advanced natural language techniques; however, the investigation of such techniques in this context is beyond the scope of this research. When faced with ontologies that do not contain concept names using natural language, for instance using numeric identifiers instead, and containing no labels, it is unlikely that any pre-processing technique will be able to reliably identify possible synsets, in which case a lexical similarity is ill-suited for that particular matching problem.

In the second step, the virtual document model as described in Sect. 4.1 is applied to each ontology concept and to each synset that has been gathered in the previous step. The resulting virtual document is represented using the well-known vector-space model [65]. To compute the similarities between the synset documents and the concept documents, the established cosine-similarity is applied [78].

5.2 Concept Disambiguation

Once the similarities between the entity document and the different synset documents are known, a selection method is applied to disambiguate the meaning of the given concept. Here, senses are only coupled to the concept if they resulted in a sufficiently high document similarity, while the remaining senses are discarded. To determine which similarity score can be considered sufficiently high, a selection policy needs to be applied. It is possible to tackle this problem from various angles, ranging from very lenient methods, discarding only the very worst synsets, to strict methods, associating only the highest scoring synset with the given concept. Several selection methods have been investigated for this research, such that both strict and lenient methods are tested:

G-MEAN The most lenient method aggregates the document similarities using the geometric mean and uses this as a threshold to discard senses with a lower similarity value.
A-MEAN Similar to the previous method, however, the arithmetic mean is used as a threshold instead.
M-STD This more strict method dynamically determines a threshold by subtracting the standard deviation of the document similarities from the highest obtained similarity. It has the interesting property that it is more strict when there is a subset of documents that is significantly more similar than the remaining documents, indicating a strong sense correspondence, and more lenient when it not as easy to identify the correct correspondences.
MAX The most strict method consists of dismissing all senses from C except for the one single sense that resulted in the highest document similarity.

Once all concepts of both input ontologies are disambiguated, one can compute the lexical similarity between concepts using the processed synset collections.

5.3 Lexical Similarity Metrics

After selecting the most appropriate synsets using the document similarities, the similarity between two entities can now be computed using their assigned synsets. This presents the problem of determining the similarity between two sets of synsets. To approach this task, we will evaluate three different methods of determining the lexical similarity between two collections of synsets.

A reasonable assumption is that each collection of synsets only contains one synset that represents the true meaning of its corresponding entity. Thus, if one were to compare two sets of synsets, assuming that the originating entities are semantically related, then one can assume that the resulting similarity between the two synsets that both represent the true meaning of their corresponding entities, should be a high value. Inspecting all pairwise similarities between all combinations of synsets between both sets should yield at least one high similarity value. When comparing two sets originating from semantically unrelated entities, one can assume that there should be no pairwise similarity of high value present.

Thus, in this scenario, a reasonable way of computing the similarity of two sets of synsets is to compute the maximum similarity over all pairwise combination between the two sets. This intuition is similar to the principle of Maximum Relatedness Disambiguation [60] in the sense that two concepts can be considered similar if a certain amount of their concept information can be considered similar by some measure.

Formally, given two concepts x and y, their corresponding collections of synsets $C(x)$ and $C(y)$, a measure of semantic similarity $sim(m, n) \in [0, 1]$ where m and n are two arbitrary synsets, we define the first lexical similarity lsm_1 between x and y as:

$$lsm_1(x, y) = \max_{m \in C(x); n \in C(y)} sim(m, n) \tag{7}$$

A potential weakness of lsm_1 is the eventuality where a concept has several appropriate senses. When comparing these senses to other collections, one might prefer a method which values the quantity of high similarities as well. For example, assume that the sense collections $C(x)$, $C(y)$ and $C(z)$ each contain two senses and that we wish to determine whether $C(y)$ or $C(z)$ are a more appropriate match for $C(x)$. Further, assume that each pairwise similarity between senses in $C(x)$ and $C(y)$ results in the value ψ, where as only one sense of $C(z)$ results in the similarity ψ with the remaining sense being unrelated and resulting in a similarity of 0. Computing $lsm_1(x, y)$ and $lsm_1(x, z)$ would both result in the value ψ. In this example, however, one would be more inclined to match x with y since the comparison with y resulted in more high similarity values. A way to adapt for this situation is to determine the best target sense for each sense in both collections and to aggregate these values, which we will denote as lsm_2. Given two concepts x and y, their corresponding collections of synsets $C(x)$ and $C(y)$, a measure of semantic similarity $sim(m, n) \in [0, 1]$ where m and n are two arbitrary synsets, we define lsm_2 as follows:

$$lsm_2(x, y) = \frac{\sum_{m \in C(x)} \left(\max_{n \in C(y)} sim(m, n)\right) + \sum_{n \in C(y)} \left(\max_{m \in C(x)} sim(n, m)\right)}{|C(x)| + |C(y)|} \tag{8}$$

A more general approach to determine the similarity between two collections of senses is to aggregate all pairwise similarities between the two collections. This has the potential benefit that similarity values which have no effect on the result of lsm_1 or lsm_2 are affecting the outcome of the lexical similarity measure. We will denote this measure as lsm_3. Formally, given two concepts x and y, their corresponding collections of synsets $C(x)$ and $C(y)$, a measure of semantic similarity $sim(m, n) \in [0, 1]$, we define lsm_3 as follows:

$$lsm_3(x, y) = \frac{\sum_{m \in C(x)} \sum_{n \in C(y)} sim(m, n)}{|C(x)| \times |C(y)|} \tag{9}$$

There exist various ways to compute the semantic similarity sim within WordNet [5] that can be applied, however, finding the optimal measure is beyond the scope of this paper since this is not a component of the disambiguation process. Here, a similarity measure with similar properties as the Leacock–Chodorow similarity [5] has been applied. The similarity $sim(s_1, s_2)$ of two synsets s_1 and s_2 is computed using the distance function $dist(s_1, s_2)$, which determines the distance of two synsets inside the taxonomy, and the over depth D of the taxonomy:

$$sim(s_1, s_2) = \begin{cases} \frac{D - \text{dist}(s_1, s_2)}{D} & \text{if dist}(s_1, s_2) \leq D \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

This measure is similar to the Leacock–Chodorow similarity in that it relates the taxonomic distance of two synsets to the depth of the taxonomy. To ensure that the resulting similarity values fall within the interval of [0, 1] and thus can be integrated into larger mapping systems, the log-scaling has been omitted in favour of a linear scale.

6 Experiments

In this section, the experiments that have been performed to test the effectiveness the approach of adding a concept disambiguation step to a lexical similarity will be presented. These experiments serve to evaluate different aspects of the proposed approach and to demonstrate the feasibility of word-sense disambiguation techniques for an ontology mapping system. The different experiments can be divided into the following categories:

- Subsect. 6.1 describes the performed experiments to evaluate the different concept disambiguation policies to determine whether lenient or strict policies should be preferred.
- The experiments described in Subsect. 6.2 demonstrate the potential performance a system can achieve when utilizing the proposed techniques.
- Subsection 6.3 presents the performed experiments which evaluate the considered virtual document weighting techniques.
- The runtime performance overhead and gains introduced by our approach will be analysed in Subsect. 6.4.

The tested mapping system used for the performed experiments contains two similarity metrics: a lexical similarity using a configuration which is specified in the experimental setup and a syntactic similarity using the Jaro string similarity [29] applied on concept names and labels. The combined concept similarities are aggregated using the Naive descending extraction algorithm [48]. The tested system in Sects. 6.1 and 6.2.2 used the parameter schemes obtained from the

experiment presented in Sect. 6.3.2, while the system in Sect. 6.2.1 had a manually tuned parameter set.

When evaluating the performance of an ontology mapping procedure, the most common practise is to compare a generated alignment with a reference alignment of the same data set. Measures such as precision and recall [21], can then be computed to express the correctness and completeness of the computed alignment. Given a generated alignment A and reference alignment R, the precision $P(A, R)$ and recall $R(A, R)$ of the generated alignment A are defined as:

$$P(A, R) = \frac{R \cap A}{A} \tag{11}$$

$$R(A, R) = \frac{R \cap A}{R} \tag{12}$$

Given the precision and recall of an alignment, a common measure to express the overall quality of the alignment is the F-measure [21]. Given a generated alignment A and a reference alignment R, the F-measure can be computed al follows:

$$\text{F-Measure(A,R)} = \frac{2 * P(A, R) * R(A, R)}{P(A, R) + R(A, R)} \tag{13}$$

The F-measure is the harmonic mean between precision and recall. Given that these measurements require a reference alignment, they are often inconvenient for large-scale evaluations, since reference alignments require an exceeding amount of effort to create. The used data sets, however, do feature reference alignments, such that the performance of a mapping approach can easily be computed and compared.

6.1 Concept Disambiguation

To investigate to what extent disambiguation techniques can improve a framework using a lexical similarity, we evaluated our approach using different variations of our approach on the conference data set of the 2011 competition [15] from the Ontology Alignment Evaluation Initiative (OAEI) [16]. This data set consists of real-world ontologies describing the conference domain and contains a reference alignment for each possible combination of ontologies from this data set. We performed this evaluation using the three lexical similarity measures lsm_1, lsm_2 and lsm_3, evaluating each measure using the disambiguation policies *G-Mean*, *A-Mean*, *M-STD* and *MAX*. We denote *None* as the omission of the disambiguation step, such that its results denote the baseline performance of the respective lexical similarity measure. Figure 5 displays the different results when using lsm_1.

From Fig. 5 we can make several key observations. First, we can see that a stricter disambiguation policy clearly benefits the lsm_1 metric, evidenced by the steadily increasing F-measure. The low precision for lenient policies implies that there are numerous false positives which exhibit a higher semantic similarity than the true correspondences. When increasing the strictness of the filtering policy, the precision rises steadily, meaning an increasing amount of false positives is eliminated. We can also observe a slight drop in recall for stricter policies, particularly when comparing *M-STD* with *MAX*, which implies that in a few situations the wrong senses are filtered out.

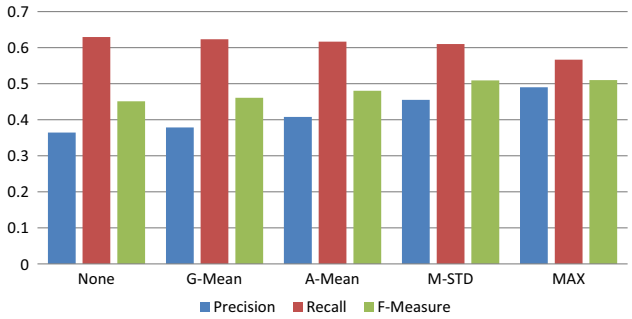

Fig. 5 Evaluation of disambiguation policies using the lexical similarity lsm_1 on the OAEI 2011 Conference data set

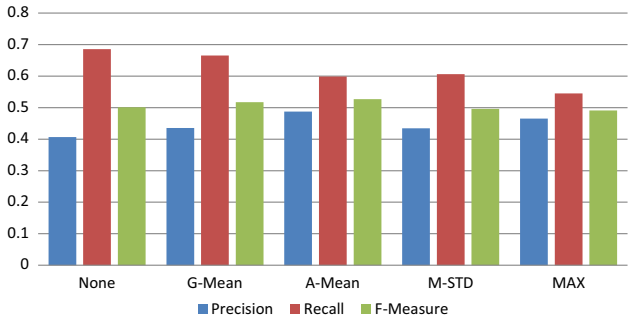

Fig. 6 Evaluation of disambiguation policies using the lexical similarity lsm_2 on the OAEI 2011 Conference data set

The same evaluation has been performed using the lsm_2 lexical similarity. The results of this evaluation can be seen in Fig. 6.

From Fig. 6 we can see that the disambiguation policies have a different effect on lsm_2, as opposed to lsm_1. We can observe an improvement in performance when applying *G-Mean* or *A-Mean* as policies, with F-measures of .517 and .526, respectively, compared to the baseline F-measure of .501. This improvement stems from an increase in precision, which more than compensates for the loss in recall. However, the F-measure decreases again when applying *M-STD* and *MAX* as policies. This implies that preferring to match concepts whose sense have multiple high pairwise similarities can be beneficial, since for *M-STD* and *MAX* it is unlikely at least that after the disambiguation step there are multiple senses left. Thus, main observation of this evaluation is that a disambiguation step is also beneficial for lsm_2, though not for all disambiguation policies.

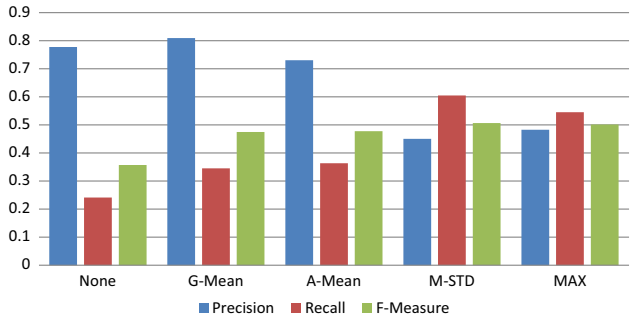

Fig. 7 Evaluation of disambiguation policies using the lexical similarity lsm_3 on the OAEI 2011 Conference data set

Lastly, the results of the evaluation when applying lsm_3 can be observed in Fig. 7.

From Fig. 7, we can see that the precision and recall values obtained by applying lsm_3 differ significantly when compared to the values obtained by applying lsm_1 or lsm_2. For the baseline and the policies *G-Mean* and *A-Mean* we can observe a very high precision and low recall. The high precision implies that a high average semantic similarity between collections of synsets is likely to represent a true correspondence. The low recall implies though that this does not occur very frequently. Upon applying the most lenient disambiguation policy *G-Mean*, we can see a drastic increase in both recall and F-measure. Applying the stricter policy *A-Mean* the recall and F-measure increase slightly, though at the cost of a reduced precision. The performance of *M-STD* is similar to its performance when applying lsm_1 or lsm_2, implying that it is not a regular occurrence that this policy retains more than one word sense.

Overall, we can conclude that the application of the proposed disambiguation method benefited the tested lexical similarity metrics. For lsm_1 and lsm_3 a strict disambiguation policy has produced the best results, while for lsm_2 the lenient policies have been shown to be most effective.

6.2 Framework Comparison

In this subsection, we will compare the performance of a mapping system utilizing our approach with the performance of established techniques. To do this, we have entered a configuration of our approach in the OAEI 2011 competition [15], of which the results are reported in Sect. 6.2.1. A comparison with the performances of additional and revised state-of-the-art systems will be presented in Sect. 6.2.2.

6.2.1 Preliminary OAEI 2011 evaluation

During the research phase of this approach, we entered the described system in the 2011 OAEI competition under the name *MaasMatch* to evaluate its performance. The configuration used the lexical similarity metric lsm_1 with disambiguation policy MAX, since at the time the performance of lsm_2 and lsm_3 was not evaluated, yet. The results of the competition on the conference data set can be seen in Fig. 8.

From Fig. 8 one can see that MaasMatch achieved a high precision and moderate recall over the conference data set, resulting in the fifth highest F-measure among the participants, which is above average. A noteworthy aspect of this result is that this result has been achieved by only applying lexical similarities, which are better suited at resolving naming conflicts as opposed to other conflicts. This in turn also explains the moderate recall value, since it would require a larger, and more importantly a more varied set of similarity values, to deal with the remaining types of heterogeneities as well. Hence, it is encouraging to see these good results when taking into account the moderate complexity of the framework.

A different data set of the OAEI competition is the *benchmark* data set. This is a synthetic data set, where a reference ontology is matched with many systematic variations of itself. These variations include many aspects, such as introducing errors or randomizing names, omitting certain types of information or altering the structure of the ontology. Since a base ontology is compared to variations of itself, this data set does not contain a large quantity of naming conflicts, which our approach is targeted at. However, it is interesting to see how our framework performs when faced with every kind of heterogeneity. Figure 9 displays the results of the OAEI 2011 evaluation [15] on the benchmark data set.

From Fig. 9 we can see that the overall performance MaasMatch resulted in a high precision score and relatively low recall score when compared to the competitors. The low recall score can be explained by the fact that the disambiguation method relies on collecting candidate synsets using information stored in the names of the ontology concepts. The data set regularly contains ontologies with altered or scrambled names, such that it becomes extremely difficult to allocate candidate senses which can be used for the disambiguation step. These alterations also have a negative impact on the quality of the constructed virtual documents, especially if names or annotations are scrambled or completely left out, resulting in MaasMatch performing poorly in benchmarks that contain such alterations. Despite these drawbacks, it was possible to achieve results similar to established matchers that address all types of heterogeneities. Given these results, the performance can be improved if measures are added which tackle other types of heterogeneities, especially if such measures increase the recall without impacting the precision.

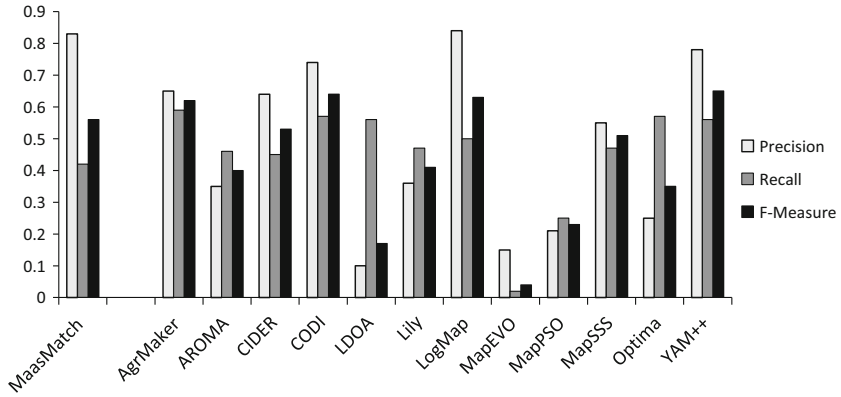

Fig. 8 Results of MaasMatch in the OAEI 2011 competition on the conference data set, compared against the results of the other participants

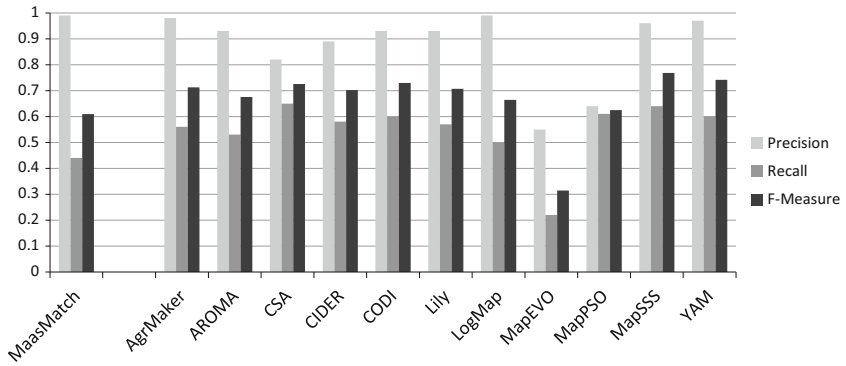

Fig. 9 Results of MaasMatch in the OAEI 2011 competition on the benchmark data set, compared against the results of the other participants

6.2.2 Comparison with OAEI 2013 frameworks

To give a more complete picture of the performance of our approach compared to other frameworks, we re-evaluated our approach using the 2013 conference data set [22] using the same evaluation methodology than the OAEI competition. This allows for the comparison with newer frameworks. Here, the frameworks *edna* and *StringEquiv* are purely string-based systems which serve as a baseline comparison. We limit the comparison to systems which performed above the lowest baseline, *StringEquiv*, for the sake of brevity. We test three variations of our approach, allowing each lexical similarity metric to be compared. As disambiguation policies we applied *MAX* for lsm_1 and *A-Mean* for lsm_2 and lsm_3. While *A-Mean* is sub-optimal for lsm_3 with regard to the F-measure, applying its best performing measure *MAX* would result in a performance similar to the configuration of lsm_1. The comparison of the OAEI 2013 performances with the three lexical similarity measures can be seen in Table 2.

One can observe from Table 2 that of the three tested lexical similarity measures, lsm_1 and lsm_2 scored above the two baseline matchers. The quality of the alignments produced by the two variants of the tested systems is very similar, especially with regard to the F-measure. Similar to its 2011 performance, the lsm_1 variant displayed a strong emphasis on precision, while the precision and recall of lsm_2 resembles the measures obtained by similarly performing systems, most notably *ODGOMS1_1* and *HerTUDA*. The performance of lsm_3 is more comparable to the *baseline* and the *OntoK* system.

Overall, we can conclude that a system using our approach can perform competitively with state-of-the-art systems, especially when taking into account the modest complexity of the tested system.

6.3 Weighting Schemes Experiments

In this section, we will demonstrate the effect of the parameter system of the used document model. We will demonstrate this effect when the model is used to calculate word-sense scores, as described in our approach, and the effect when the model is used in its original context as a profile similarity.

6.3.1 Preliminaries: Parameter Optimization

The applied VD model provides the possibility of parametrized weighting, which allows the emphasis of words depending on their origin.

Recall from Subsect. 4.1 that the model contains a set of parameters, being α_1, α_2, α_3, α_4, β_1, β_2 and β_3, which weight terms according to their place in the ontology. Next to evaluating the weighting approaches in the proposed WSD method, we will also test a profile similarity that uses the presented virtual document model for gathering the context

Table 2 Evaluation on the conference 2013 data set and comparison with OAEI 2013 frameworks

Framework	Precision	Recall	F-measure
YAM++	0.78	0.65	0.71
AML-bk	0.82	0.53	0.64
LogMap	0.76	0.54	0.63
AML	0.82	0.51	0.63
ODGOMS1_2	0.7	0.55	0.62
StringsAuto	0.74	0.5	0.6
ServOMap_v104	0.69	0.5	0.58
MapSSS	0.77	0.46	0.58
ODGOMS1_1	0.72	0.47	0.57
lsm$_1$	0.8631	0.4436	0.5685
lsm$_2$	0.7382	0.4797	0.5643
HerTUDA	0.7	0.46	0.56
WikiMatch	0.7	0.45	0.55
WeSeE-match	0.79	0.42	0.55
IAMA	0.74	0.44	0.55
HotMatch	0.67	0.47	0.55
CIDER_CL	0.72	0.44	0.55
edna	0.73	0.44	0.55
lsm$_3$	0.6327	0.5041	0.5466
OntoK	0.72	0.43	0.54
LogMapLite	0.68	0.45	0.54
XMapSiG1_3	0.68	0.44	0.53
XMapGen1_4	0.64	0.45	0.53
SYNTHESIS	0.73	0.41	0.53
StringEquiv	0.76	0.39	0.52

information of a concept, similar to the work by Qu et al. [62]. Here, given two concepts c and d, originating form different ontologies, and their respective virtual documents $VD(c)$ and $VD(d)$, a profile similarity can be created by computing the document similarity between $VD(c)$ and $VD(d)$. For each of the tested approaches the *conference* and *benchmark* data sets were used as separate training sets, resulting in four different parameter sets. We will use the terms Lex-B and Lex-C to refer to the parameter sets which have been generated by optimizing the LSM on the *benchmark* and *conference* data set, respectively. For the parameter sets which have been generated using the profile similarity we will use the terms Prof-B and Prof-C.

Tree-Learning Search (TLS) [79] was applied to optimize the different combinations of similarity metrics and training sets. TLS combines aspects of Monte-Carlo Tree Search and incremental regression tree induction in order to selectively discretize the parameter space. This discretized parameter space is then sampled using the Monte-Carlo method to approximate the optimal solution. The results of the performed optimization can be seen in Table 3.

From Table 3 some notable differences emerge. The parameter α_1 tends to have a higher value for profile similarities compared to the LSM parameters sets. This can be explained by the fact that the synset candidate collection step of the proposed disambiguation method selects candidate synsets using the processed ontology concept names as basis. Hence, all sysnet candidates will contain terms that are similar to the ontology concept name, diminishing their information value for the purpose of WSD. Conversely, values for α_2 tend to be higher for LSM parameter sets, indicating that matching alternative concept names are a strong indication of a concept's intended meaning.

6.3.2 Preliminaries: Test Setup

We will evaluate six different weighting schemes for virtual documents to investigate what impact these have on the mapping quality. The six weighting schemes were evaluated on the *conference* data set and can be described as follows:

TF As a reference point, we will evaluate the performance of standard term-frequency weights as a baseline, which is done by setting all VD parameters to 1.
Lex-C/Prof-C This scheme represents the VD model using optimized parameters that were obtained from the same data set. This scheme will be referred to by the name of its corresponding parameters set, which is Lex-C for the WSD evalaution and Prof-C for the profile similarity evaluation.
Lex-B/Prof-B Similar to the previous scheme, however, the parameter set was obtained through the optimization on a different training set.
TF-IDF This scheme entails the combination of term-frequency and inverse document frequency weights, as commonly seen in the field of information retrieval. Similar to TF weighting, all weights of the VD model will be set to 1.
*Lex-C/Prof-C * TF-IDF* It is possible to combine the VD model with a TF-IDF weighting scheme. This scheme represents such a combination using the parameter sets that have been obtained from the same data set. In the WSD experiment this scheme will be referred to as Lex-C * TF-IDF, while in the profile similarity experiment it will be referred to as Prof-C * TF-IDF.
*Lex-B/Prof-B * TF-IDF* Similar to the previous scheme, however, the parameter sets that were obtained from the benchmark data set are used instead.

The evaluation of the TF-IDF method and its combination with the VD model weighting is especially critical since previous work using this model has included TF-IDF weighting in its approach without evaluating the possible implications of this technique [62]. For each weighting method the computed alignments are ranked according to their similarity. For

Table 3 Optimized parameter sets for the VD model when applied to a LSM (Lex) and profile similarity (Prof) using the conference (C) and benchmark (B) data sets as training sets

Parameter set	α_1	α_2	α_3	α_4	β_1	β_2	β_3
Lex-C	0.51	0.68	0.58	0.42	0.32	0.07	0.06
Lex-B	0.52	0.99	0.08	0.65	0.01	0.09	0.16
Prof-C	0.71	0.02	0.01	0.58	0.09	0.04	0.01
Prof-B	0.85	0.13	0.54	0.32	0.90	0.32	0.99

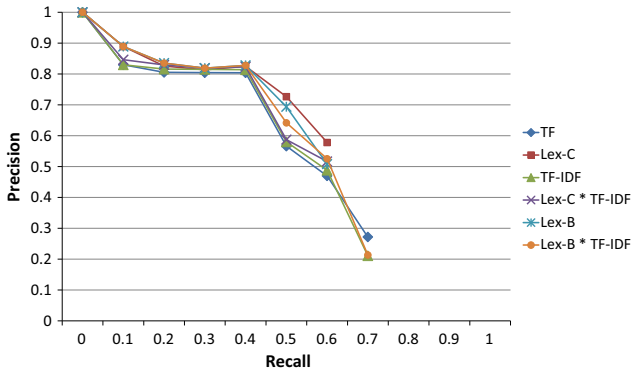

Fig. 10 Precision versus recall graph of the created alignments from the conference data set using the lexical similarities with the virtual document

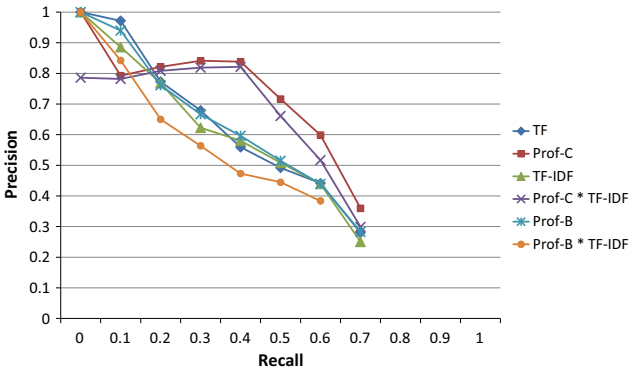

Fig. 11 Precision versus recall graph of the created alignments from the conference data set using the document similarities of the virtual documents

each ranking the interpolated precision values will be computed such that these can be compared.

6.3.3 Lexical Similarity with Applied WSD

The different weighting schemes have been separately applied to this approach and subsequently used to calculate mappings on the conference data set. The precision vs recall graph of the produced alignments can be seen in Fig. 10.

From Fig. 10 we can observe some key points. For lower recall values, Lex-C, Lex-B and Lex-B * TF-IDF weighting resulted in the highest precision values. When inspecting higher recall values, one can observe that the Lex-C and Lex-B weighting outperformed the remaining weighting schemes with differences in precision reaching values of 10 %. However, only the alignments generated with TF, TF-IDF and Lex-B * TF-IDF weighting achieved a possible recall value of 0.7 or higher, albeit at very low precision values. Another notable observation is the performance of TF-IDF based schemes. The standard TF-IDF scheme displayed performance similar to TF, thus being substantially lower than Lex-C or Lex-B. Also, the combination schemes Lex-C * TF-IDF and Lex-B * TF-IDF performed lower than their respective counterparts Lex-C and Lex-B. From this we can conclude that when applying VD-based disambiguation for a LSM, it is preferable to weight terms according to their origin and avoid the use of inverse document frequencies.

6.3.4 Profile Similarity

After having established the impact that different weighting techniques can have on the VD model when applied as context gathering method for a disambiguation approach, it would be interesting to see the impact of these techniques when the VD model is used for its original purpose [62]. Hence, in this subsection, we will detail the performed experiments with the six investigated weighting schemes when utilizing the virtual document model as the context gathering method for a profile similarity. All weighting schemes were used to calculate mappings on the conference data set. The measures of precision and recall were computed using the resulting alignments. The precision vs recall graph of these alignments can be seen in Fig. 11.

From Fig. 11 several key observations can be made. Initially, one can see that the overall two best performing schemes are Prof-C and Prof-C * TF-IDF weighting. The Prof-C * TF-IDF scheme displays a slightly worse performance than the Prof-C scheme. This indicates that the combination with TF-IDF weights not only failed to improve the term weights of the virtual documents, but rather it caused the representative strength of the VD to decrease, leading to alignments of lesser quality. The same contrast is visible when comparing Prof-B weighting with Prof-B * TF-IDF weighting.

Next, another observation can be made when contrasting the results of the TF-IDF weights with TF weights. Both

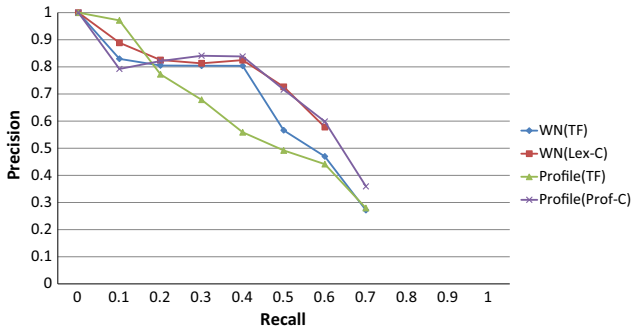

Fig. 12 Precision versus recall graph contrasting the LSM and Profile similarity performances using TF weighting with the VD model using the optimized parameter sets

schemes lead to alignments of similar quality, indicating that the combination of the inverse document frequencies to the term frequencies does not lead to the same improvements that one can observe when performing information retrieval on regular documents. Lastly, when comparing TF-IDF weighting to Prof-C and Prof-B weighting, one can see that TF-IDF weighting can at most match the performance of the other two schemes.

6.3.5 Improvement Comparison

To provide a better contrast with regard to how much the weighting scheme of the document model can impact the performance, we have compiled the baseline TF performances of the lexical and profile similarity along with the performances of the lexical and profile similarity when applying, respectively, optimized weights of the document model. The resulting precision versus recall graph can be seen in Figure.

From Fig. 12 we can observe that the application of the document model weighting scheme had a more drastic impact on the profile similarity than on the lexical similarity. This difference can be explained by the nature of the two similarity measures. In a profile similarity, the virtual documents of two concepts are compared directly, thus changing the weights associated with each term can severely alter the result when the documents are compared. This is evidenced by the stark differences between *Profile(TF)* and *Profile(Prof-C)*. For a lexical similarity, however, altering the weighting scheme is likely to cause little difference when comparing the concept document to a document representing an unrelated word sense, since the word-sense document is unlikely to contain many overlapping words. Thus, significant changes are only likely to occur in more ambiguous cases, where unrelated sense do have at least some term overlap. This explanation is substantiated in the difference in performance between *WN(TF)* and *WN(Lex-C)* over the different recall levels. For the lower recall levels, there is only little different between *WN(TF)* and *WN(Lex-C)*, meaning that the correspondences with higher confidence levels do not vary much and are hence associated with the same word senses regardless of weighting approach. Only for the higher recall levels is a large difference in performance observed, suggesting alternate associated word senses for the more difficult to disambiguate concepts.

6.4 Runtime Analysis

When designing an ontology mapping framework the issue of runtime can be an important factor. This becomes an increasingly important issue when attempting to create a mapping between large ontologies, with both ontologies containing several hundred up to thousands of concepts. Adding a disambiguation procedure to a lexical similarity might cause a decrease in runtime performance, which if sufficiently significant would make in infeasible to include this approach for a large mapping task. To establish how much runtime overhead our approach generates, we executed our system on the OAEI 2013 conference data set while recording the total runtimes for the three general steps of lexical similarity measure: the retrieval of candidate senses, the disambiguation procedure and the computation of the lexical similarity. The disambiguation procedure involves the process of creating the virtual documents, document similarity computations and application of the disambiguation policy. To accurately establish the overhead added to the runtime of a standard lexical similarity, no word senses are discarded in the disambiguation step. As lexical similarity metric, lsm_1 was applied, though in terms of runtime there is likely to be little difference since lsm_1, lsm_2 and lsm_3 all require the computation between all pairwise combinations of senses to obtain their results. The recorded runtimes are presented in Table 4.

From Table 4 we can see that the most time-consuming step of the entire similarity measure, consuming 74 % of the expended computation time, is the calculation of the actual similarity values after having disambiguated all the word senses. Determining all candidate word senses for the ontology concepts, which involves several string-processing techniques such as tokenization, word-stemming and stop-word removal, required 22 % of the spent computational time. The creation of virtual documents and disambiguation of senses only required 3 % of the computation time, meaning that the addition of this step increased the runtime by 3.65 %. Given the potential performance increases of our approach, one can conclude that the additional overhead introduced is negligible.

The previous comparison assumed a worse-case scenario where no senses are discarded. However, the filtering of senses can reduce the computational time for the lexical similarity by requiring fewer evaluations of the semantic similarity between senses. To see to what extent the different disambiguation policies reduce the runtime of this step, we

Table 4 Runtimes of the different elements of the lexical similarity on the conference data set

Computation	Sense retrieval	Disambiguation	Lexical similarity	Overhead (%)
Runtime (ms)	35,272	5,632	118,900	3.65

Table 5 Runtimes of the different elements of the lexical similarity for each disambiguation policy

Policy	Sense retrieval (ms)	Disambiguation (ms)	Lexical similarity (ms)	Runtime reduction (%)
None	35,272	5,632	118,900	0.0
G-mean	35,350	5,590	61,761	35.7
A-mean	35,780	5,828	24,847	58.4
M-STD	34,229	5,472	7,244	70.6
MAX	33,975	5,374	2,005	74.1

recorded the runtimes of each policy on the conference data set to establish possible performance gains.

From Table 5 we can observe that the application of the disambiguation policies can lead to significant improvements in terms of runtime. Applying the most lenient *G-Mean* policy leads to a reduction in runtime of 35 % where as the most strict policy reduces the overall runtime by 74.1 %.

Overall, we can conclude the the application of a disambiguation procedure can lead to significant improvements in runtime despite the addition of computational overhead of the disambiguation method.

7 Conclusion

Lexical similarity measures are a critical component in contemporary ontology mapping procedures. In this paper, we proposed the inclusion of word-sense disambiguation techniques into lexical similarity metrics to disambiguate ontology concepts, such that lexical similarities more accurately reflect the semantic relatedness between two ontology concepts. Further, we proposed a concept disambiguation method which identifies corresponding senses using a virtual document model. For each given concept and possible senses, a parametrized model is used to gather relevant information into separate virtual documents, such that their document similarities indicate which sense is the most likely to denote the meaning of the given concept.

We investigated to what extent the proposed disambiguation method can improve a mapping system. To do this several disambiguation policies, ranging from lenient to strict, have been evaluated using three different lexical similarity metrics. The experimental results show that the application of the disambiguation approach improves the performances of all the tests' similarity metrics, with two metrics favouring a strict and one metric favouring a lenient disambiguation policy.

To establish the potential of a mapping system utilized the proposed approach, we compared such a system with contemporary ontology mapping frameworks in the context of the OAEI 2011 and OAEI 2013 competition. The evaluation on the real-world data set shows very promising performance, exhibiting the fifth highest alignment quality among the tested frameworks. The evaluation on a synthetic benchmark revealed a dependency on properly formed concept names.

Furthermore, we investigated the effects of different weighting approaches for the terms in the virtual documents. These approaches were tested on the proposed disambiguation method and a profile similarity, which uses the assembled virtual documents as concept profiles. From this experiment, we can conclude that weighting terms according to their origin from their respective ontology is the superior weighting method. TF-IDF weighting did not result in any measurable benefits, suggesting that term-weighting methods that were developed with the assumption that these will be applied to texts written in natural language should be avoided.

Lastly, a performed runtime analysis revealed that the filtering of word senses reduces the runtime of the similarity metrics to such an extent that it more than off-sets the added overhead introduced by the disambiguation method.

We suggest that future work should focus on the robustness of the approach when faced with disturbed concept names or descriptions. This can be achieved for instance through the application of automatic spell checking or the usage of soft document similarities during the disambiguation process.

References

1. Aguirre J, Grau B, Eckert K, Euzenat J, Ferrara A, van Hague R, Hollink L, Jimenez-Ruiz E, Meilicke C, Nikolov A, Ritze D, Shvaiko P, Svab-Zamazal O, Trojahn C, Zapilko B (2012) Results of the ontology alignment evaluation initiative 2012. In: Proceedings of the 7th ISWC workshop on ontology matching, pp 73–115
2. Banerjee S, Pedersen T (2003) Extended gloss overlaps as a measure of semantic relatedness. In: Proceedings of the 18th international joint conference on artificial intelligence, San Francisco, CA, USA, IJCAI'03, pp 805–810

3. Bar-Hillel Y (1960) The present status of automatic translation of languages. Readings in machine translation, pp 45–77
4. Bodenreider O (2004) The unified medical language system (umls): integrating biomedical terminology. Nucl Acids Res 32(suppl 1):D267–D270
5. Budanitsky A, Hirst G (2001) Semantic distance in wordnet: an experimental, application-oriented evaluation of five measures. In: workshop on wordNet and other lexical resources, second meeting of the North American chapter of the association for computational linguistics, pp 29–34
6. Buitelaar P, Cimianop P, Haase P, Sintek M (2009) Towards linguistically grounded ontologies. The semantic web: research and applications, vol 5554., Lecture notes in computer scienceSpringer, Berlin, pp 111–125
7. Cruz I, Lucas W (1997) A visual approach to multimedia querying and presentation. In: Proceedings of the fifth ACM international conference on multimedia, ACM, pp 109–120
8. Cruz I, Xiao H (2009) Ontology driven data integration in heterogeneous networks. Theory, models and applications, Complex systems in knowledge-based environments, pp 75–98
9. Cruz I, Antonelli F, Stroe C (2009) Agreementmaker: efficient matching for large real-world schemas and ontologies. Proc VLDB Endow 2(2):1586–1589
10. Cruz IF, Palmonari M, Caimi F, Stroe C (2013) Building linked ontologies with high precision using subclass mapping discovery. Artif Intell Rev 40(2):127–145
11. De Melo G, Weikum G (2009) Towards a universal wordnet by learning from combined evidence. In: Proceedings of the 18th ACM conference on information and knowledge management, ACM, pp 513–522
12. Euzenat J (2001) Towards a principled approach to semantic interoperability. In: Proceedings of the IJCAI-01 workshop on ontologies and information sharing, pp 19–25
13. Euzenat J (2004) An api for ontology alignment. In: Proceedings of the international semantic web conference (ISWC), pp 698–712
14. Euzenat J, Shvaiko P (2007) Ontology matching, vol 18. Springer, Berlin
15. Euzenat J, Ferrara A, van Hague R, Hollink L, Meilicke C, Nikolov A, Scharffe F, Shvaiko P, Stuckenschmidt H, Svab-Zamazal O, Trojahn dos SC (2011a) Results of the ontology alignment evaluation initiative 2011. In: Proceedings 6th ISWC workshop on ontology matching (OM), Bonn (DE), pp 85–110
16. Euzenat J, Meilicke C, Stuckenschmidt H, Shvaiko P, Trojahn C (2011b) Ontology alignment evaluation initiative: six years of experience. J Data Semant XV, pp 158–192
17. Gale WA, Church KW, Yarowsky D (1992) A method for disambiguating word senses in a large corpus. Comput Humanit 26(5/6):415–439
18. Giunchiglia F, Shvaiko P (2003) Semantic matching. Knowl Eng Rev 18(3):265–280
19. Giunchiglia F, Yatskevich M (2004) Element level semantic matching. Meaning coordination and negotiation (MCN-04), p 37
20. Giunchiglia F, Shvaiko P, Yatskevich M (2004) S-match: an algorithm and an implementation of semantic matching. The semantic web: research and applications, pp 61–75
21. Giunchiglia F, Yatskevich M, Avesani P, Shvaiko P (2009) A large dataset for the evaluation of ontology matching. Knowl Eng Rev J 24:137–157
22. Grau BC, Dragisic Z, Eckert K, Euzenat J, Ferrara A, Granada R, Ivanova V, Jiménez-Ruiz E, Kempf AO, Lambrix P, et al. (2013) Results of the ontology alignment evaluation initiative 2013. In: Proceedings 8th ISWC workshop on ontology matching (OM), pp 61–100
23. Gulić M, Vrdoljak B (2013) Cromatcher-results for oaei 2013. In: Proceedings of the eighth ISWC international workshop on ontology matching, pp 117–122
24. Hau J, Lee W, Darlington J (2005) A semantic similarity measure for semantic web services. In: web service Semantics workshop 2005 at WWW2005
25. He B, Chang KCC (2006) Automatic complex schema matching across web query interfaces: a correlation mining approach. ACM Trans Database Syst (TODS) 31(1):346–395
26. Hindle D, Rooth M (1993) Structural ambiguity and lexical relations. Comput Linguist 19(1):103–120
27. Hu W, Qu Y (2008) Falcon-ao: a practical ontology matching system. Web Semant Sci Serv Agents World Wide Web 6(3):237–239
28. Ide N, Véronis J (1998) Introduction to the special issue on word sense disambiguation: the state of the art. Comput Linguist 24(1):2–40
29. Jaro M (1989) Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida. J Am Stat Assoc 84(406):414–420
30. Jean-Mary Y, Shironoshita E, Kabuka M (2009) Ontology matching with semantic verification. Web Semant 7:235–251
31. Jones KS (1972) A statistical interpretation of term specificity and its application in retrieval. J Doc 28(1):11–21
32. Kalfoglou Y, Schorlemmer M (2003) Ontology mapping: the state of the art. Knowl Eng Rev 18(1):1–31
33. Kim W, Seo J (1991) Classifying schematic and data heterogeneity in multidatabase systems. Computer 24(12):12–18
34. Kitamura Y, Segawa S, Sasajima M, Tarumi S, Mizoguchi R (2008) Deep semantic mapping between functional taxonomies for interoperable semantic search. The semantic web, pp 137–151
35. Kotis K, Valarakos A, Vouros G (2006a) Automs: automated ontology mapping through synthesis of methods. Ontol Matching, pp 96–106.
36. Kotis K, Vouros G, Stergiou K (2006b) Towards automatic merging of domain ontologies: the hcone-merge approach. Web Semant Sci Serv Agents World Wide Web 4(1):60–79
37. Lassila O, Swick R, W3C (1998) Resource description framework (rdf) model and syntax specification
38. Lesk M (1986) Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In: Proceedings of the 5th annual international conference on systems documentation, SIGDOC '86, pp 24–26
39. Litkowski K (1997) Desiderata for tagging with wordnet synsets or mcca categories. In: fourth meeting of the ACL special interest group on the Lexicon. Washington, DC: association for computational linguistics
40. Locke W, Booth A (1955) Machine translation of languages: fourteen essays. Published jointly by Technology Press of the Massachusetts Institute of Technology and Wiley, New York
41. Mao M (2007) Ontology mapping: an information retrieval and interactive activation network based approach. In: Proceedings of the 6th international the semantic web and 2nd Asian conference on Asian semantic web conference, Springer, Berlin, ISWC'07/ASWC'07, pp 931–935
42. Mao M, Peng Y, Spring M (2007) A profile propagation and information retrieval based ontology mapping approach. In: Proceedings of the third international conference on semantics. Knowledge and Grid, IEEE, pp 164–169
43. Marshall I (1983) Choice of grammatical word-class without global syntactic analysis: tagging words in the lob corpus. Comput Humanit 17(3):139–150
44. Matuszek C, Cabral J, Witbrock M, DeOliveira J (2006) An introduction to the syntax and content of cyc. AAAI Spring symposium
45. McCarthy D, Koeling R, Weeds J, Carroll J (2004) Finding predominant word senses in untagged text. In: Proceedings of the 42nd annual meeting on association for computational linguistics, association for computational linguistics, p 279
46. McCrae J, Spohr D, Cimiano P (2011) Linking lexical resources and ontologies on the semantic web with lemon. In: the semantic

web: research and applications, lecture notes in computer science, vol 6643, Springer, pp 245–259
47. McGuinness D, van Harmelen F (2004) OWL web ontology language overview. W3C recommendation, W3C
48. Meilicke C, Stuckenschmidt H (2007) Analyzing mapping extraction approaches. In: Proceedings of the ISWC 2007 workshop on ontology matching
49. Mihalcea R (2006) Knowledge-based methods for wsd. Word sense disambiguation, pp 107–131
50. Miller GA (1995) Wordnet: a lexical database for english. Commun ACM 38:39–41
51. Montoyo A, Suárez A, Rigau G, Palomar M (2005) Combining knowledge-and corpus-based word-sense-disambiguation methods. J Artif Intell Res 23(1):299–330
52. Navigli R (2009) Word sense disambiguation: a survey. ACM Comput Surv 41(2):10:1–10:69
53. Navigli R, Ponzetto S (2010) Babelnet: building a very large multilingual semantic network. In: Proceedings of the 48th annual meeting of the association for computational linguistics, association for computational linguistics, pp 216–225
54. Ngo D, Bellahsene Z, Coletta R (2012) Yam++-a combination of graph matching and machine learning approach to ontology alignment task. J Web Semant
55. Niles I, Pease A (2001) Towards a standard upper ontology. In: Proceedings of the international conference on formal ontology in information systems-volume 2001, ACM, pp 2–9
56. Niles I, Terry A (2004) The milo: a general-purpose, mid-level ontology. In: Proceedings of the international conference on information and knowledge engineering, pp 15–19
57. Noy N, Musen M (2001) Anchor-prompt: using non-local context for semantic matching. In: Proceedings of the workshop on ontologies and information sharing at the international joint conference on artificial intelligence (IJCAI), pp 63–70
58. Paulheim H, Hertling S (2013) Wesee-match results for oaei 2013. Proceedings of the eigth ISWC international workshop on ontology matching, pp 197–202
59. Pedersen T (2006) Unsupervised corpus-based methods for wsd. Word sense disambiguation, pp 133–166
60. Pedersen T, Banerjee S, Patwardhan S (2005) Maximizing semantic relatedness to perform word sense disambiguation. University of Minnesota supercomputing institute research report UMSI 25:2005
61. Po L, Sorrentino S (2011) Automatic generation of probabilistic relationships for improving schema matching. Inf Syst 36(2):192–208
62. Qu Y, Hu W, Cheng G (2006) Constructing virtual documents for ontology matching. In: Proceedings of the 15th international conference on World Wide Web, ACM, New York, NY, USA, WWW '06, pp 23–31
63. Rahm E, Bernstein PA (2001) A survey of approaches to automatic schema matching. VLDB J 10(4):334–350
64. Resnik P, Yarowsky D (1999) Distinguishing systems and distinguishing senses: new evaluation methods for word sense disambiguation. Nat Lang Eng 5(02):113–133
65. Salton G, Wong A, Yang C (1975) A vector space model for automatic indexing. Commun ACM 18:613–620
66. Saruladha K, Aghila G, Sathiya B (2011) A comparative analysis of ontology and schema matching systems. Int J Comput Appl 34(8):14–21, published by Foundation of computer science, New York
67. Schadd F, Roos N (2012) Coupling of wordnet entries for ontology mapping using virtual documents. In: Proceedings of the seventh international workshop on ontology matching (OM-2012) collocated with the 11th international semantic web conference (ISWC-2012), pp 25–36
68. Schütze H (1992) Dimensions of meaning. In: Proceedings of the 1992 ACM/IEEE conference on supercomputing, IEEE, pp 787–796
69. Schütze H, Pedersen JO (1995) Information retrieval based on word senses. In: Proceedings of the 4th annual symposium on document analysis and information retrieval
70. Shvaiko P, Euzenat J (2005) A survey of schema-based matching approaches. In: journal on data semantics IV, Springer, pp 146–171
71. Shvaiko P, Euzenat J (2008) Ten challenges for ontology matching. On the move to meaningful internet systems. OTM 5332:1164–1182
72. Shvaiko P, Euzenat J (2013) Ontology matching: state of the art and future challenges. Knowl Data Eng IEEE Trans 25(1):158–176
73. Sicilia M, Garcia E, Sanchez S, Rodriguez E (2004) On integrating learning object metadata inside the opencyc knowledge base. In: Proceedings of advanced learning technologies, 2004 IEEE international conference on, IEEE, pp 900–901
74. Sproat R, Hirschberg J, Yarowsky D (1992) A corpus-based synthesizer. Proc Int Conf Spok Lang Process 92:563–566
75. Strube M, Ponzetto SP (2006) Wikirelate! computing semantic relatedness using wikipedia. AAAI 6:1419–1424
76. Suchanek F, Kasneci G, Weikum G (2008) Yago: a large ontology from wikipedia and wordnet. Web Semant Sci Serv Agents World Wide Web 6(3):203–217
77. Talukdar PP, Ives ZG, Pereira F (2010) Automatically incorporating new sources in keyword search-based data integration. In: Proceedings of the 2010 ACM SIGMOD international conference on management of data, ACM, pp 387–398
78. Tan PN, Steinbach M, Kumar V (2005) Introduction to Data Mining, 1st edn Addison Wesley
79. Van Den Broeck G, Driessens K (2011) Automatic discretization of actions and states in monte-carlo tree search. In: Proceedings of the international workshop on machine learning and data mining in and around games (DMLG), pp 1–12
80. Wache H, Voegele T, Visser U, Stuckenschmidt H, Schuster G, Neumann H, Hübner S (2001) Ontology-based integration of information-a survey of existing approaches. In: IJCAI-01 workshop: ontologies and information sharing, vol 2001, pp 108–117
81. Watters C (1999) Information retrieval and the virtual document. J Am Soc Inf Sci 50:1028–1029
82. Weaver W (1955) Translation. Mach Transl Lang 14:15–23
83. Wilkes Y (1975) Preference semantics. In: Keenan E (ed) Formal semantics of natural language, Cambridge University Press, pp 329–348
84. Yarowsky D (1994) Decision lists for lexical ambiguity resolution: application to accent restoration in spanish and french. In: Proceedings of the 32nd annual meeting on association for computational linguistics, association for computational linguistics, pp 88–95