Measuring the Similarity between Implicit Semantic Relations from the Web

Danushka Bollegala*
The University of Tokyo
Hongo 7-3-1, Tokyo
113-8656, Japan
danushka@mi.ci.i.u-tokyo.ac.jp

Yutaka Matsuo
The University of Tokyo
Hongo 7-3-1, Tokyo
113-8656, Japan
matsuo@biz-model.t.u-tokyo.ac.jp

Mitsuru Ishizuka
The University of Tokyo
Hongo 7-3-1, Tokyo
113-8656, Japan
ishizuka@i.u-tokyo.ac.jp

ABSTRACT

Measuring the similarity between semantic relations that hold among entities is an important and necessary step in various Web related tasks such as relation extraction, information retrieval and analogy detection. For example, consider the case in which a person knows a pair of entities (e.g. *Google, YouTube*), between which a particular relation holds (e.g. acquisition). The person is interested in retrieving other such pairs with similar relations (e.g. *Microsoft, Powerset*). Existing keyword-based search engines cannot be applied directly in this case because, in keyword-based search, the goal is to retrieve documents that are relevant to the words used in a query – not necessarily to the relations implied by a pair of words. We propose a relational similarity measure, using a Web search engine, to compute the similarity between semantic relations implied by two pairs of words. Our method has three components: representing the various semantic relations that exist between a pair of words using automatically extracted lexical patterns, clustering the extracted lexical patterns to identify the different patterns that express a particular semantic relation, and measuring the similarity between semantic relations using a metric learning approach. We evaluate the proposed method in two tasks: classifying semantic relations between named entities, and solving word-analogy questions. The proposed method outperforms all baselines in a relation classification task with a statistically significant average precision score of 0.74. Moreover, it reduces the time taken by Latent Relational Analysis to process 374 word-analogy questions from 9 days to less than 6 hours, with an SAT score of 51%.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval

General Terms

Algorithms

Keywords

Relational Similarity, Web Mining, Natural Language Processing

*Research Fellow of the Japan Society for the Promotion of Science (JSPS)

WWW 2009, April 20–24, 2009, Madrid, Spain.
ACM 978-1-60558-487-4/09/04.

1. INTRODUCTION

Similarity measures can be categorized broadly into two types: *attributional* similarity measures and *relational* similarity measures. For attributional similarity measures, the objective is to compute the similarity between two given words by comparing the attributes of each word. For example, the two words *car* and *automobile* share many attributes (e.g. *has wheels*, *is used for transportation*). Consequently, they are considered as synonyms. On the other hand, relational similarity is the correspondence between semantic relations that exist between two *word pairs*. Word pairs that show a high degree of relational similarity are considered as *analogies*. For example, the two word pairs *(ostrich, bird)* and *(lion, cat)*. Ostrich *is a large* bird and lion *is a large* cat are illustrative of high relational similarity. The semantic relation, *is a large*, pertains between the two words in each word pair.

The information available on the Web can be considered as a vast, hidden network of classes of objects (e.g. named entities) that is interconnected by various semantic relations applying to those objects. Measuring the similarity between semantic relations is an important intermediate step in various tasks in information retrieval and natural language processing such as relation extraction [7, 8, 40], in which the goal is to retrieve instances of a given relation. For example, given the relation, ACQUIRER-ACQUIREE, a relation extraction system must extract the instance (*Google*, *YouTube*) from the sentence *Google completed the acquisition of YouTube*. Bootstrapping methods [25, 6, 14], which require a few seeds (ca. 10 pairs of instances per relation) have extracted numerous candidate instance pairs from a text corpus. Given a set of candidate instance pairs, a relational similarity measure can be used to compute the similarity between the relations in the seeds and in the candidates. Candidate instance pairs with high relational similarity with the seed pairs can then be selected as the correct instances of a relation.

Relational similarity measures have been used to find word analogies [10, 24, 31, 33, 38]. Word analogy questions have been used from the Scholastic Aptitude Test (SAT; Educational Testing Service) to benchmark relational similarity measures. An SAT word analogy question consists of a stem word pair that acts as the question and five choice word pairs, out of which only one is analogous to the stem. A relational similarity measure is used to compare the stem word pair with each choice word pair and to select the choice word pair with the highest relational similarity as the answer.

An interesting application of relational similarity in information retrieval is to search using implicitly stated analogies [21, 37]. For example, the query "Muslim Church" is expected to return "mosque", and the query "Hindu bible" is expected to return "the Vedas". These queries can be formalized as word pairs: (Christian, Church) vs. (Muslim,X), and (Christian, Bible) vs. (Hindu,Y). We can then find the words X and Y that maximize the relational similarity in each case.

Despite the wide applications of relational similarity measures, accurately measuring the similarity between implicitly stated relations remains a challenging task for several reasons. First, relational similarity is a dynamic phenomenon: it varies with time. For example, two companies can be competitors initially; subsequently one company might acquire the other. Second, there can be more than one relation between a given word pair. For example, between the two words *ostrich* and *bird*, aside from the relation *is a large*, there is also the relation *is a flightless*. A relational similarity measure must first extract all relations between the two words in each word pair before it can compute the similarity between the word pairs. Third, there can be more than one way to express a particular semantic relation in a text. For example, the three patterns – ***X** was acquired by **Y***, ***Y** completed the acquisition of **X***, and ***Y** buys **X*** – all indicate an acquisition relation between **X** and **Y**. In addition to the problems described above, measuring relational similarity between pairs in which one or both words are named entities (e.g., company names, personal names, locations, etc.) is even more difficult because such words are not well covered by manually created dictionaries such as WordNet[1] [23].

As described herein, we propose a relational similarity measure that uses a Web search engine to measure the similarity between implicitly stated semantic relations in two word pairs. Formally, given two word pairs, *(a,b)* and *(c,d)*, we design a function, *relsim*$((a,b),(c,d))$, that returns a similarity score in the range $[0,1]$. The proposed relational similarity measure first extracts implicitly stated relations that exist between the two words in each word pair. The measure then compares the extracted relations between word pairs.

Our contributions are summarized as follows:

- We propose a shallow, lexical-patterns-based approach to represent the various semantic relations that pertain between the two words in a given word pair. The proposed pattern extraction algorithm requires no language dependent preprocessing steps such as part-of-speech tagging or dependency parsing, which can be time consuming or even infeasible at the Web scale. We extract numerous lexical patterns that describe various semantic relations.

- We present an efficient sequential clustering algorithm to cluster lexical patterns, to identify the different patterns that describe a particular semantic relation. The proposed clustering algorithm requires only one pass through the set of extracted patterns. For that reason, it scales linearly with the number of patterns. We then use the clusters to define features for a supervised metric learning algorithm.

- We evaluate the proposed method in two tasks: classifying semantic relations between named entities, and solving SAT word-analogy questions. In the relation classification task, the proposed method significantly outperforms all baselines, including the state-of-the art Latent Relational Analysis (LRA) [33]. Moreover, the proposed method achieves an SAT score of 51.1 and reduces the time taken to answer 374 questions by LRA from 9 days to less than 6 hours.

[1] http://wordnet.princeton.edu/

2. RELATED WORK

The Structure Mapping Theory (SMT) [15] is based on the premise that an analogy is a mapping of knowledge from one domain (base) into another (target), which conveys that a system of relations known to hold in the base also holds in the target. The target objects need not resemble their corresponding base objects. This structural view of analogy is based on the intuition that analogies are about relations, rather than simple features. Although this approach works best when the base and the target are rich in higher-order causal structures, it can fail when structures are missing or flat [39].

Turney et al. [35] combined 13 independent modules by considering the weighted sum of the outputs of each module to solve SAT analogy questions. The best performing individual module was based on the Vector Space Model (VSM). In the VSM approach [34], a vector is first created for a word pair *(X,Y)* by counting the frequencies of various lexical patterns containing *X* and *Y*. In their experiments, they used 128 manually created patterns such as "*X* of *Y*", "*Y* of *X*", "*X* to *Y*", and "*Y* to *X*". These patterns are then used as queries to a search engine. The numbers of hits for respective queries are used as elements in a vector to represent the word pair. Finally, the relational similarity is computed as the cosine of the angle between the two vectors that represent the two word pairs. Turney et al. [35] introduced a dataset containing 374 SAT analogy questions to evaluate relational similarity measures. An SAT analogy question consists of a stem word pair that acts as the question, and five choice word pairs. The choice word pair that has the highest relational similarity with the stem word pair is selected by the system as the correct answer. The average SAT score reported by high school students for word-analogy questions is 57%. The VSM approach achieves a score of 47% on this dataset.

Turney [31, 33] proposed Latent Relational Analysis (LRA) by extending the VSM approach in three ways: a) lexical patterns are automatically extracted from a corpus, b) the Singular Value Decomposition (SVD) is used to smooth the frequency data, and c) synonyms are used to explore variants of the word pairs. Similarly, in the VSM approach, LRA represents a word pair as a vector of lexical pattern frequencies. First, using a thesaurus, he finds related words for the two words in a word pair and create additional word pairs that are related to the original word pairs in the dataset. Second, n-grams of words are extracted from the contexts in which the two words in a word pair cooccur. The most frequent n-grams are selected as lexical patterns to represent a word pair. Then a matrix of word pairs vs. lexical patterns is created for all the word pairs in the original dataset and the additional word pairs. Elements of this matrix correspond to the frequency of a word pair in a lexical pattern. Singular value decomposition is performed on this matrix to reduce the number of columns (i.e. patterns). Finally, the relational similarity between two word pairs is computed as the average cosine similarity over the original word pairs and the additional word pairs derived from them. In fact, LRA achieves a score of 56.4% on SAT analogy questions.

Both VSM and LRA require numerous search engine queries to create a vector to represent a word pair. For example, with 128 patterns, the VSM approach requires at least 256 queries to create two pattern-frequency vectors for two word pairs before it can compute the relational similarity. In fact, LRA considers synonymous variants of the given word pairs. For that reason, it requires even more search engine queries. Methods that require numerous queries impose a heavy load on search engines. Despite efficient implementations, singular value decomposition of large matrices is time consuming. In fact, LRA takes over 9 days to process the 374 SAT analogy questions [33]. This is problematic when computing relational similarity on the scale of the Web. Moreover, in the case of named entities, thesauri of related words are not usually available or are not complete, which becomes a problem when creating the additional word pairs required by LRA.

Veale [38] proposed a relational similarity measure based on the taxonomic similarity in WordNet. The quality of a candidate analogy *A:B::C:D* (i.e. *A* to *B* as *C* to *D*) is evaluated through comparison of the paths in the WordNet, joining *A* to *B* and *C* to *D*. Relational similarity is defined as the similarity between the *A:B* paths and *C:D* paths. However, WordNet does not fully cover named entities such as personal names, organizations and locations, which becomes problematic when using this method to measure relational similarity between named entities.

Using a relational similarity measure, Turney [32] proposed an unsupervised learning algorithm to extract patterns that express implicit semantic relations from a corpus. His method produces a ranked set of lexical patterns that unambiguously describes the relation between the two words in a given word pair. Patterns are ranked according to their expected relational similarity (i.e. pertinence); they are computed using an algorithm similar to LRA. To answer an SAT analogy question, first, ranked lists of patterns are generated for each of the six word pairs (one stem word pair and five choice word pairs). Then each choice is evaluated by taking the intersection of its patterns with the stem's patterns. The shared patterns are scored by the average of their rank in the stem's list and the choice's lists. The algorithm picks the choice with the lowest scoring shared pattern as the correct answer. This method reports an SAT score of 54.6%.

Relational similarity measures have been applied in natural language processing tasks such as generating word analogies [10], and classifying noun-modifier compounds based on the relation between the head and the modifier [33, 24, 9, 24]. Davidov and Rappoport [10] proposed an unsupervised algorithm to discover general semantic relations that pertain between lexical items. They represent a semantic relation with a cluster of patterns. They use the pattern clusters to generate SAT-like word analogy questions for English and Russian languages. The generated questions are then solved by human subjects. They do not evaluate their method for relational similarity between named entities.

Relational similarity measures have been used to classify the relationships between the head and the modifier in noun-compounds [33, 24, 9]. For example, in the compound *viral flu*, the *flu* (head) is *caused by* a *virus* (modifier). The *Diverse* dataset of Barker and Szpakowicz [1], which consists of 600 head-modifier pairs (noun-noun, adjective-noun and adverb-noun) is used as a benchmark dataset to evaluate relation classification of noun-compounds. Each noun-modifier pair in this dataset is annotated with one of the following five relations: *causal*, *temporal*, *spatial*, *participant*, and *quality*. Nakov and Hearst [24] proposed a linguistically motivated method that utilizes verbs, prepositions, and coordinate conjunctions that can help make explicit the hidden relations between the target nouns. They report a classification accuracy of 40.5% on the Diverse dataset using a single nearest neighbor classifier.

3. METHOD

3.1 Outline

Given two pairs of words (or named entities), *(a,b)* and *(c,d)*, the problem of measuring the similarity of implicit semantic relations between the two pairs can be viewed as a two-stage process.

First, we must extract the semantic relations that pertain in each word pair. We use a web search engine to retrieve the various contexts in which the two words in a word-pair cooccur. We then extract lexical patterns from the retrieved contexts to represent the various semantic relations that hold between two words. However, not all patterns represent different semantic relations. A single semantic relation can be expressed using more than one lexical pattern. For example, both lexical patterns *X acquired Y* and *Y was bought by X* indicate an ACQUISITION relation between entities *X* and *Y*. We present an efficient clustering algorithm to identify the various lexical patterns that denote a particular semantic relation.

Second, we must compare the extracted semantic relations between the two word pairs to compute their relational similarity. We model this problem as one of learning a distance metric between relationally similar and dissimilar word pairs. Unlike previously proposed relational similarity measures, we do not assume semantic relations to be independent, and learn a non-Euclidean Mahalanobis distance metric.

> Google to acquire YouTube for $1.65 billion in stock. Combination will create new opportunities for users and content owners everywhere...

Figure 1: A snippet returned for the query *"Google * * * YouTube"*.

3.2 Retrieving Contexts

We must first identify the implicitly stated relations that hold between the two words in each word pair to compute the relational similarity between two given word pairs. The context in which two words cooccur provides useful clues about the semantic relations that pertain between those words. We propose the use of text snippets retrieved using a Web search engine as an approximation of the context of two words. Snippets (also known as *dynamic teasers*) are brief summaries provided by most Web search engines along with the search results. Typically, a snippet contains a window of text selected from a document that includes the queried words. Snippets are useful for search because, most of the time, a user can read the snippet and decide whether a particular search result is relevant, without even opening the url. Using snippets as contexts is also computationally efficient because it obviates the need to download the source documents from the Web, which can be time consuming if a document is large.

A snippet for a query containing two words captures the local context in which they cooccur. For example, consider the snippet shown in Figure 1, returned by *Yahoo*[2] for the query *"Google * * YouTube"*. Here, the wildcard operator "*" matches one word or none in a document. The snippet in Figure 1 is extracted from an online newspaper article about the acquisition of YouTube by Google.

To retrieve snippets for a word pair *(A,B)*, we use the following seven types of queries: *"A * B"*, *"B * A"*, *"A * * B"*, *"B * * A"*, *"A * * * B"*, *"B * * * A"*, and *A B*. The queries containing the wildcard operator "*" returns snippets in which the two words, *A* and *B* appear within a window of specified length. We designate such queries a *wildcard* queries. We search for snippets in which the query words cooccur within a maximum window of three words (tokens). This process is intended to approximate the local context of two words in a document. The quotation marks around a query will ensure that the two words appear in the specified order (e.g. *A* before *B* in snippets retrieved for the query *"A * B"*). As a fallback in the case that all wildcard queries fail to return any snippets, we use the query *A B* (without wildcards or quotations) to retrieve snippets where *A* and *B* appear in any order.

[2] http://developer.yahoo.com/search/boss/