Graph Based Similarity Measures for Synonym Extraction from Parsed Text

Einat Minkov
Dep. of Information Systems
University of Haifa
Haifa 31905, Israel
einatm@is.haifa.ac.il

William W. Cohen
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
wcohen@cs.cmu.edu

Abstract

We learn graph-based similarity measures for the task of extracting word synonyms from a corpus of parsed text. A constrained graph walk variant that has been successfully applied in the past in similar settings is shown to outperform a state-of-the-art syntactic vector-based approach on this task. Further, we show that learning specialized similarity measures for different word types is advantageous.

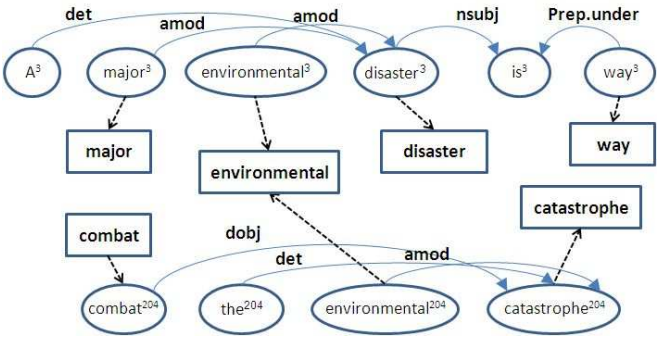

Figure 1: A joint graph of dependency structures

1 Introduction

Many applications of natural language processing require measures of lexico-semantic similarity. Examples include summarization (Barzilay and Elhadad, 1999), question answering (Lin and Pantel, 2001), and textual entailment (Mirkin et al., 2006). Graph-based methods have been successfully applied to evaluate word similarity using available ontologies, where the underlying graph included word senses and semantic relationships between them (Hughes and Ramage, 2007). Another line of research aims at eliciting semantic similarity measures directly from freely available corpora, based on the *distributional similarity* assumption (Harria, 1968). In this domain, vector-space methods give state-of-the-art performance (Padó and Lapata, 2007).

Previously, a graph based framework has been proposed that models word semantic similarity from parsed text (Minkov and Cohen, 2008). The underlying graph in this case describes a text corpus as connected dependency structures, according to the schema shown in Figure 1. The toy graph shown includes the dependency analysis of two sentences: "a major environmental disaster is under way", and "combat the environmental catastrophe". In the graph, word mentions (in circles) and word types (in squares) are both represented as nodes. Each word mention is linked to its corresponding word type; for example, the nodes "environmental^3" and "environmental^{204}" represent distinct word mentions and both nodes are linked to the word type "environmental".[1] For every edge in the graph, there exists an edge in the opposite direction (not shown in the figure). In this graph, the terms *disaster* and *catastrophe* are related due to the connecting path disaster $\longrightarrow$ disaster^3 $\overset{amod-inverse}{\longrightarrow}$ environmental^3 $\longrightarrow$ environmental $\longrightarrow$ environmental^{204} $\overset{amod}{\longrightarrow}$ catastrophe^{204} $\longrightarrow$ catastrophe .

Given *a query*, which consists of a word of interest (e.g., 'disaster'), various graph-based similarity metrics can be used to assess inter-node relatedness, so that a list of nodes ranked by their similarity to the query is returned to the user. An advantage of graph-based similarity approaches is that they produce similarity scores that reflect structural infor-

[1] We will sometimes refer to *word types* as *terms*.

mation in the graph (Liben-Nowell and Kleinberg, 2003). Semantically similar terms are expected to share connectivity patterns with the query term in the graph, and thus appear at the top of the list.

Notably, different edge types, as well as the paths traversed, may have varying importance for different types of similarity sought. For example, in the parsed text domain, noun similarity and verb similarity are associated with different syntactic phenomena (Resnik and Diab, 2000). To this end, we consider a *path constrained graph walk* (PCW) algorithm, which allows one to learn meaningful paths given a small number of labeled examples and incorporates this information in assessing node relatedness in the graph (Minkov and Cohen, 2008). PCW have been successfully applied to the extraction of named entity coordinate terms, including city and person names, from graphs representing newswire text (Minkov and Cohen, 2008), where the specialized measures learned outperformed the state-of-the-art *dependency vectors* method (Padó and Lapata, 2007) for small- and medium-sized corpora.

In this work, we apply the path constrained graph walk method to the task of eliciting general word relatedness from parsed text, conducting a set of experiments on the task of synonym extraction. While the tasks of named entity extraction and synonym extraction from text have been treated separately in the literature, this work shows that both tasks can be addressed using the same general framework. Our results are encouraging: the PCW model yields superior results to the dependency vectors approach. Further, we show that learning specialized similarity measures per word type (nouns, verbs and adjectives) is preferable to applying a uniform model for all word types.

2 Path Constrained Graph Walks

PCW is a graph walk variant proposed recently that is intended to bias the random walk process to follow meaningful edge sequences (paths) (Minkov and Cohen, 2008). In this approach, rather than assume fixed (possibly, uniform) edge weight parameters Θ for the various edge types in the graph, the probability of following an edge of type ℓ from node x is evaluated dynamically, based on the *history* of the walk up to x.

The PCW algorithm includes two components. First, it should provide estimates of edge weights conditioned on the history of a walk, based on training examples. Second, the random walk algorithm has to be modified to maintain historical information about the walk compactly.

In learning, a dataset of N labelled example queries is provided. The labeling schema is binary, where a set of nodes considered as relevant answers to an example query e_i, denoted as R_i, is specified, and graph nodes that are not explicitly included in R_i are assumed irrelevant to e_i. As a starting point, an initial graph walk is applied to generate a ranked list of graph nodes l_i for every example query e_i. A path-tree T is then constructed that includes all of the acyclic paths up to length k leading to the top M^+ correct and M^- incorrect nodes in each of the retrieved lists l_i. Every path p is associated with a maximum likelihood probability estimate $Pr(p)$ of reaching a correct node based on the number of times the path was observed in the set of correct and incorrect target nodes. These path probabilities are propagated backwards in the path tree to reflect the probability of reaching a correct node, given an outgoing edge type and partial history of the walk.

Given a new query, a constrained graph walk variant is applied that adheres both to the topology of the graph G and the path tree T. In addition to tracking the graph node that the random walker is at, PCW maintains pointers to the nodes of the path tree that represent the walk histories in reaching that graph node. In order to reduce working memory requirements, one may prune paths that are associated with low probability of reaching a correct node. This often leads to gains in accuracy.

3 Synonym Extraction

We learn general word semantic similarity measures from a graph that represents a corpus of parsed text (Figure 1). In particular, we will focus on evaluating word synonymy, learning specialized models for different word types. In the experiments, we mainly compare PCW against the dependency vectors model (DV), due to Padó and Lapata (2007). In the latter approach, a word w_i is represented as a vector of weighted scores, which reflect co-occurrence frequency with words w_j, as well as

properties of the dependency paths that connect the word w_i to word w_j. In particular, higher weight is assigned to connecting paths that include grammatically salient relations, based on the *obliqueness* weighting hierarchy (Keenan and Comrie, 1977). For example, co-occurrence of word w_i with word w_j over a path that includes the salient *subject* relation receives higher credit than co-occurrences over a non-salient relation such as preposition. In addition, Padó and Lapata suggest to consider only a subset of the paths observed that are linguistically meaningful. While the two methods incorporate similar intuitions, PCW learns meaningful paths that connect the query and target terms from examples, whereas DV involves manual choices that are task-independent.

3.1 Dataset

To allow effective learning, we constructed a dataset that represents strict word synonymy relations for multiple word types. The dataset consists of 68 examples, where each example query consists of a single term of interest, with its synonym defined as a single correct answer. The dataset includes noun synonym pairs (22 examples), adjectives (24) and verbs (22). Example synonym pairs are shown in Table 1. A corpus of parsed text was constructed using the British National Corpus (Burnard, 1995). The full BNC corpus is a 100-million word collection of samples of written and spoken contemporary British English texts. We extracted relevant sentences, which contained the synonymous words, from the BNC corpus. (The number of extracted sentences was limited to 2,000 per word.) For infrequent words, we extracted additional example sentences from Associated Press (AP) articles included in the AQUAINT corpus (Bilotti et al., 2007). (Sentence count was complemented to 300 per word, where applicable.) The constructed corpus, BNC+AP, includes 1.3 million words overall. This corpus was parsed using the Stanford dependency parser (de Marneffe et al., 2006).[2]. The parsed corpus corresponds to a graph that includes about 0.5M nodes and 1.7M edges.

Nouns	movie : film
	murderer : assassin
Verbs	answered : replied
	enquire : investigate
Adjectives	contemporary : modern
	infrequent : rare

Table 1: Example word synonym pairs: the left words are used as the query terms.

3.2 Experiments

Given a query like {*term*="*movie*"}, we would like to get synonymous words, such as *film*, to appear at the top of the retrieved list. In our experimental setting, we assume that the word type of the query term is known. Rather than rank all words (terms) in response to a query, we use available (noisy) part of speech information to narrow down the search to the terms of the same type as the query term, e.g. for the query "film" we retrieve nodes of type τ =*noun*.

We applied the PCW method to learn separate models for noun, verb and adjective queries. The path trees were constructed using the paths leading to the node known to be a correct answer, as well as to the otherwise irrelevant top-ranked 10 terms. We required the paths considered by PCW to include exactly 6 segments (edges). Such paths represent distributional similarity phenomena, allowing a direct comparison against the DV method. In conducting the constrained walk, we applied a threshold of 0.5 to truncate paths associated with lower probability of reaching a relevant response, following on previous work (Minkov and Cohen, 2008). We implemented DV using code made available by its authors,[3] where we converted the syntactic patterns specified to Stanford dependency parser conventions. The parameters of the DV method were set to *medium* context and *oblique* edge weighting scheme, which were found to perform best (Padó and Lapata, 2007). In applying a vector-space based method, a similarity score needs to be computed between *every* candidate from the corpus and the query term to construct a ranked list. In practice, we used the union of the top 300 words retrieved by PCW as candidate terms for DV.

We evaluate the following variants of DV: hav-

[2] http://nlp.stanford.edu/software/lex-parser.shtml

[3] http://www.coli.uni-saarland.de/~pado/dv.html

	Nouns	Verbs	Adjs	All
CO-Lin	0.34	0.37	0.37	0.37
DV-Cos	0.24	0.36	0.26	0.29
DV-Lin	0.45	0.49	0.54	0.50
PCW	0.47	0.55	0.47	0.49
PCW-P	**0.53**	**0.68**	**0.55**	**0.59**
PCW-P-U	0.49	0.65	0.50	0.54

Table 2: 5-fold cross validation results: MAP

ing inter-word similarity computed using Lin's measure (Lin, 1998) (DV-Lin), or using cosine similarity (DV-Cos). In addition, we consider a non-syntactic variant, where a word's vector consists of its co-occurrence counts with other terms (using a window of two words); that is, ignoring the dependency structure (CO-Lin).

Finally, in addition to the PCW model described above (PCW), we evaluate the PCW approach in settings where random, noisy, edges have been eliminated from the underlying graph. Specifically, dependency links in the graph may be associated with pointwise mutual information (PMI) scores of the linked word mention pairs (Manning and Schütze, 1999); edges with low scores are assumed to represent word co-occurrences of low significance, and so are removed. We empirically set the PMI score threshold to 2.0, using cross validation (PCW-P).[4] In addition to the specialized PCW models, we also learned a uniform model over all word types in these settings; that is, this model is trained using the union of all training examples, being learned and tested using a mixture of queries of all types (PCW-P-U).

3.3 Results

Table 2 gives the results of 5-fold cross-validation experiments in terms of mean average precision (MAP). Since there is a single correct answer per query, these results correspond to the mean reciprocal rank (MRR).[5] As shown, the dependency vectors model applied using Lin similarity (DV-Lin) performs best among the vector-based models. The improvement achieved due to edge weighting compared with the co-occurrence model (CO-Lin) is large, demonstrating that syntactic structure is very informative for modeling word semantics (Padó and Lapata, 2007). Interestingly, the impact of applying the Lin similarity measure versus cosine (DV-Cos) is even more profound. Unlike the cosine measure, Lin's metric was designed for the task of evaluating word similarity from corpus statistics; it is based on the mutual information measure, and allows one to downweight random word co-occurrences.

Among the PCW variants, the specialized PCW models achieve performance that is comparable to the state-of-the-art DV measure (DV-Lin). Further, removing noisy word co-occurrences from the graph (PCW-P) leads to further improvements, yielding the best results over all word types. Finally, the graph walk model that was trained uniformly for all word types (PCW-P-U) outperforms DV-Lin, showing the advantage of *learning* meaningful paths. Notably, the uniformly trained model is inferior to PCW trained separately per word type in the same settings (PCW-P). This suggests that learning *specialized* word similarity metrics is beneficial.

4 Discussion

We applied a path constrained graph walk variant to the task of extracting word synonyms from parsed text. In the past, this graph walk method has been shown to perform well on a related task, of extracting named entity coordinate terms from text. While the two tasks are typically treated distinctly, we have shown that they can be addressed using the same framework. Our results on a medium-sized corpus were shown to exceed the performance of *dependency vectors*, a syntactic state-of-the-art vector-space method. Compared to DV, the graph walk approach considers higher-level information about the connecting paths between word pairs, and are adaptive to the task at hand. In particular, we showed that learning specialized graph walk models for different word types is advantageous. The described framework can be applied towards learning other flavors of specialized word relatedness models (e.g., hypernymy). Future research directions include learning word similarity measures from graphs that integrate corpus statistics with word ontologies, as well as improved scalability (Lao and Cohen, 2010).

[4] Eliminating low PMI co-occurrences has been shown to be beneficial in modeling lexical selectional preferences recently, using a similar threshold value (Thater et al., 2010).

[5] The query's word inflections and words that are semantically related but not synonymous were discarded from the ranked list manually for evaluation purposes.

References

Regina Barzilay and Michael Elhadad. 1999. *Text summarizations with lexical chains, in Inderjeet Mani and Mark Maybury, editors, Advances in Automatic Text Summarization*. MIT.

Matthew W. Bilotti, Paul Ogilvie, Jamie Callan, and Eric Nyberg. 2007. Structured retrieval for question answering. In *SIGIR*.

Lou Burnard. 1995. *Users Guide for the British National Corpus*. British National Corpus Consortium, Oxford University Computing Service, Oxford, UK.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *LREC*.

Zellig Harria. 1968. *Mathematical Structures of Language*. Wiley, New York.

Thad Hughes and Daniel Ramage. 2007. Lexical semantic relatedness with random graph walks. In *EMNLP*.

Edward Keenan and Bernard Comrie. 1977. Noun phrase accessibility and universal grammar. *Linguistic Inquiry*, 8.

Ni Lao and William W. Cohen. 2010. Fast query execution for retrieval models based on path constrained random walks. In *KDD*.

Liben-Nowell and J. Kleinberg. 2003. The link prediction problem for social networks. In *CIKM*.

Dekang Lin and Patrick Pantel. 2001. Discovery of inference rules for question answering. *Natural Language Engineering*, 7(4).

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *COLING-ACL*.

Chris Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press.

Einat Minkov and William W. Cohen. 2008. Learning graph walk based similarity measures for parsed text. In *EMNLP*.

Shachar Mirkin, Ido Dagan, and Maayan Geffet. 2006. Integrating pattern-based and distributional similarity methods for lexical entailment acquisition. In *ACL*.

Sebastian Padó and Mirella Lapata. 2007. Dependency-based construction of semantic space models. *Computational Linguistics*, 33(2).

Philip Resnik and Mona Diab. 2000. Measuring verb similarity. In *Proceedings of the Annual Meeting of the Cognitive Science Society*.

Stefan Thater, Hagen F$'$urstenau, and Manfred Pinkal. 2010. Contextualizing semantic representations using syntactically enriched vector models. In *ACL*.