Implementing A Reverse Dictionary using  
Explicit Semantic Analysis

Omar Khazamov, Remy Oukaour, and Ganesa Thandavam

CSE 507, Stony Brook University

May 23, 2013

# Abstract

We present an implementation of Explicit Semantic Analysis using a recent Wikipedia database as a corpus. This technique evaluates similarity of two texts as the similarity of their concept vectors, where concepts correspond to Wikipedia articles that include the text. Compared with baseline ESA shows improvement in correlating its relatedness scores with human judgments. Extrinsic evaluation is also promising for reverse dictionary lookup and word sense disambiguation.

# Introduction

Our initial goal was to improve on existing WordNet-based visualizers, which work by displaying connections between an input word’s synsets and their related synsets (synonyms, hyponyms, hypernyms, and other relations). Tools such as Visuwords and Visual Thesaurus, while effective at the specific task of looking up words in the WordNet hierarchy, were ineffective at providing natural semantic information about the input: they display all matching synsets, even obscure ones, and have no way to evaluate semantic similarity between two input words.

After evaluating several techniques for measuring semantic relatedness, we settled on one that used Wikipedia, not WordNet, as a source of data. Explicit semantic analysis of the Wikipedia database seemed promising to us for its massive amount of real-world knowledge, despite lacking the structure of WordNet’s relations.

We wrote an implementation of Explicit Semantic Analysis (ESA) in PHP, running on an Apache server. During its development we discovered that in addition to measuring semantic similarity, ESA could be more efficiently used for other applications, such as reverse dictionary lookup and word association.

# Semantic Relatedness

Standard dictionary-based measures of word pair similarity are based only on a single path linking those words in the knowledge graph. This captures the similarity of the words, but not their relatedness. Relatedness measures incorporate the notion of similarity, as well as enhancing it with other relations such as antonymy and meronymy. Therefore, a relatedness model is required that incorporates information from every explicit or implicit path connecting the two words in the graph. We considered measures such as those discussed in [5].

A method described in [2] uses a random walk over nodes and edges derived from WordNet links and corpus statistics. The authors use a novel divergence measure, ZKL, that outperforms existing measures for computing semantic relatedness of pairs. In their experiments, they were able to achieve a relatedness measure highly correlated with human similarity judgments by rank ordering.

In another approach, [3], the authors use a random walk over a graph derived from Wikipedia. Using both Google PageRank and HITS algorithm implementations in their experiments, they show that Wikipedia does not perform as well on the smaller datasets, but outperforms WordNet on large datasets by a wide margin.

Using the overlap of words’ WordNet glosses as semantic relatedness measure might seem to be an overly simple idea; however, in the authors of [4] show through extrinsic evaluation that extended gloss overlap improves word sense disambiguation (WSD) by as much as 89% in some cases.

# Explicit Semantic Analysis

After a survey of existing literature, we settled on using a large dataset of text from which to derive semantic world knowledge. Recognizing individual words in the data would not be a problem, since dictionaries provide word lists in any language; however, abstracting semantic concepts from raw text would be more of a challenge. We would need to somehow use predetermined natural concepts defined by humans, without requiring annotation of a huge corpus. As proposed in [1], for that we used concepts defined by Wikipedia articles.

Explicit semantic analysis treats each Wikipedia article as a semantic concept. An article is represented as a vector or weights for each unique word, where each weight is the word's TF-IDF score [2]. We build an inverted index mapping words to concept vectors, as it is more useful for looking up user-input text.

Finding the concepts associated with a given word amounts to looking up the concepts with the highest TF-IDF scores for that word. For example, here are the top 10 concepts associated with the words “dog” and “cat”:

|  |  |  |  |
| --- | --- | --- | --- |
| **dog** | | **cat** | |
| **Concept** | **TF-IDF sum** | **Concept** | **TF-IDF sum** |
| Breed-specific legislation | 21.413992572402 | *Cat* | 21.900153014209 |
| *Dog* | 21.288740778337 | Plácido Domingo discography | 21.292385580959 |
| Dingo | 20.071396561365 | Iriomote cat | 19.677004277512 |
| Dog meat | 19.276514333065 | Rephlex Records discography | 19.353106214489 |
| Australian Cattle Dog | 18.996457036782 | Cats and humans | 18.556459603467 |
| Dog training | 18.823445019563 | Winged cat | 18.32730933366 |
| Dog health | 18.80728402844 | Black cat | 18.300950400036 |
| Dog breed | 18.502844414966 | Black Cat (comics) | 18.193597534497 |
| Dogs in warfare | 18.358541557236 | Vietnam Combat Artists Program | 18.138731130387 |
| Greater Swiss Mountain Dog | 18.149876307073 | Catalogue of Women | 17.997899920639 |

Table 1: ESA results for two words.

…

# Implementation

The workflow for our application was fairly straightforward:

* Download and parse the Wikipedia database
* Compute TF-IDF scores for every (word, concept) pair
* Build an inverted index of words to concept vectors
* Given user input, compute its ESA vector, sorted by descending TF-IDF score
* Compute the cosine similarity between vectors for two words (useful as a measure of semantic relatedness)

The main challenge was to parse the Wikipedia dump and handle the resulting inverted index. The raw XML file, downloaded on April 4, 2013, was 40.4 GB and had over 13 million articles. We initially tried splitting it into one file per article, but the overhead of many small files ended up being greater than just handling a single large one. We abandoned the idea of using the Wikiprep tool (which would strip out MediaWiki markup) for the same reason. Ultimately we wrote Python scripts that used a streaming XML parser to access the markup, and regular expressions to almost completely isolate the plain text. We then output the inverted index as a SQLite database file, using stemmed forms of the words.

To query this database with user input, we tried using Amazon’s RDB service for high-performance query processing. However, the bottleneck turned out to be fetching the data from disk, so we ended up using a standard laptop to process the data and concentrated on minimizing disk access. Building an index on the word ID column of the database sped up queries, but increased the size of the database file such that it would not entirely fit in RAM. As such, queries vary in time depending on whether the particular relevant entries are in RAM or on disk; it can range from less than a second to nearly a minute to look up a word.

# Evaluation

…

# Future Work

…

# Conclusion

…

# References

[1] Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*.

[2] Thad Hughes and Daniel Ramage. 2007. Lexical Semantic Relatedness with Random Graph Walks. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.

[3] Michael Strube and Simone Paolo Ponzetto. 2006. WikiRelate! Computing Semantic Relatedness Using Wikipedia. In *Proceedings of the 21st National Conference on Artificial intelligence*, Volume 2.

[4] Satanjeev Banerjee and Ted Pedersen. 2003. Extended Gloss Overlaps as a Measure of Semantic Relatedness. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*.

[5] Alexander Budanitsky and Graeme Hirst. 2006. Evaluating WordNet-based measures of lexical semantic relatedness. In *Computational Linguistics*, 32(1):13–47.