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.

These results mostly fit with our intuitions of what concepts are related to the words "dog" and cat"; compared with WordNet, there are many more available concepts than synsets, and they avoid obscure but similar words ("andiron" for "dog") in favor of common and closely-related ones (dog breeds and dog-related topics).

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)
- Use singular vector decomposition to further refine the concept search

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; even this ended up being 12.5 GB.

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 and TF-IDF columns 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.

We built a web interface for users to easily query the database. Sample outputs are presented below.

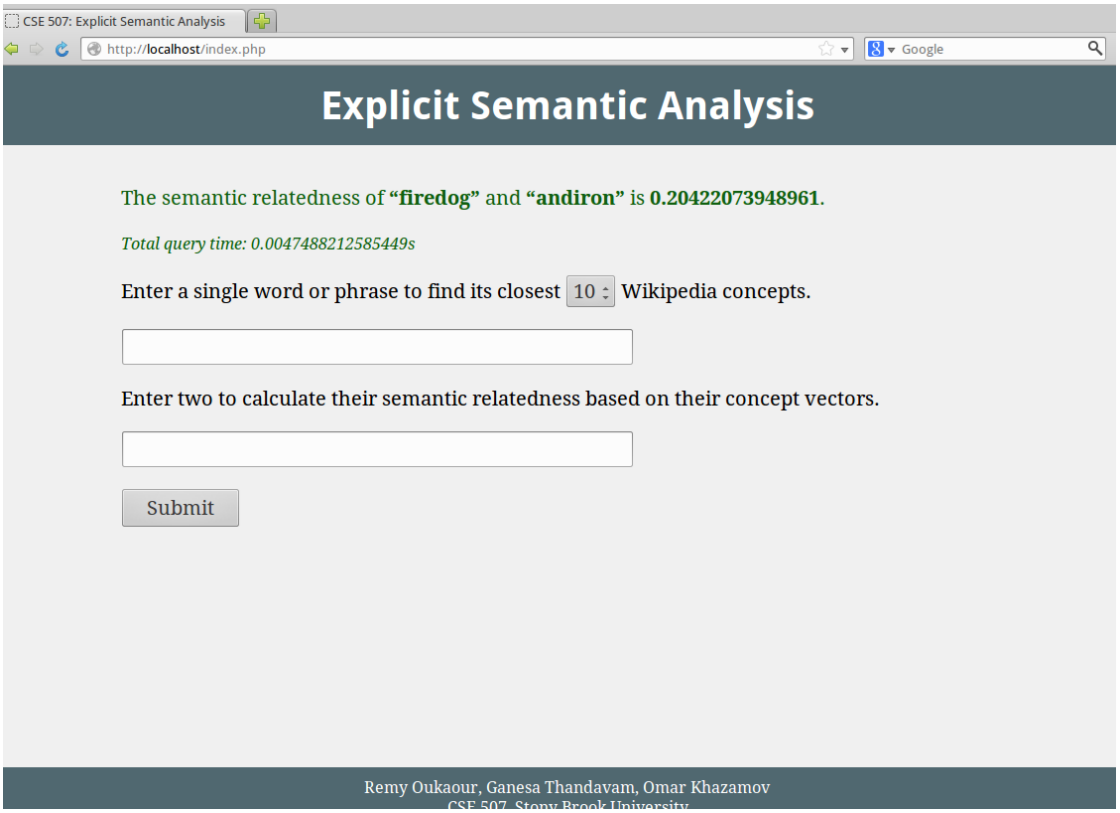

Figure 1: Evaluating the semantic relatedness of two words, "firedog" and "andiron."

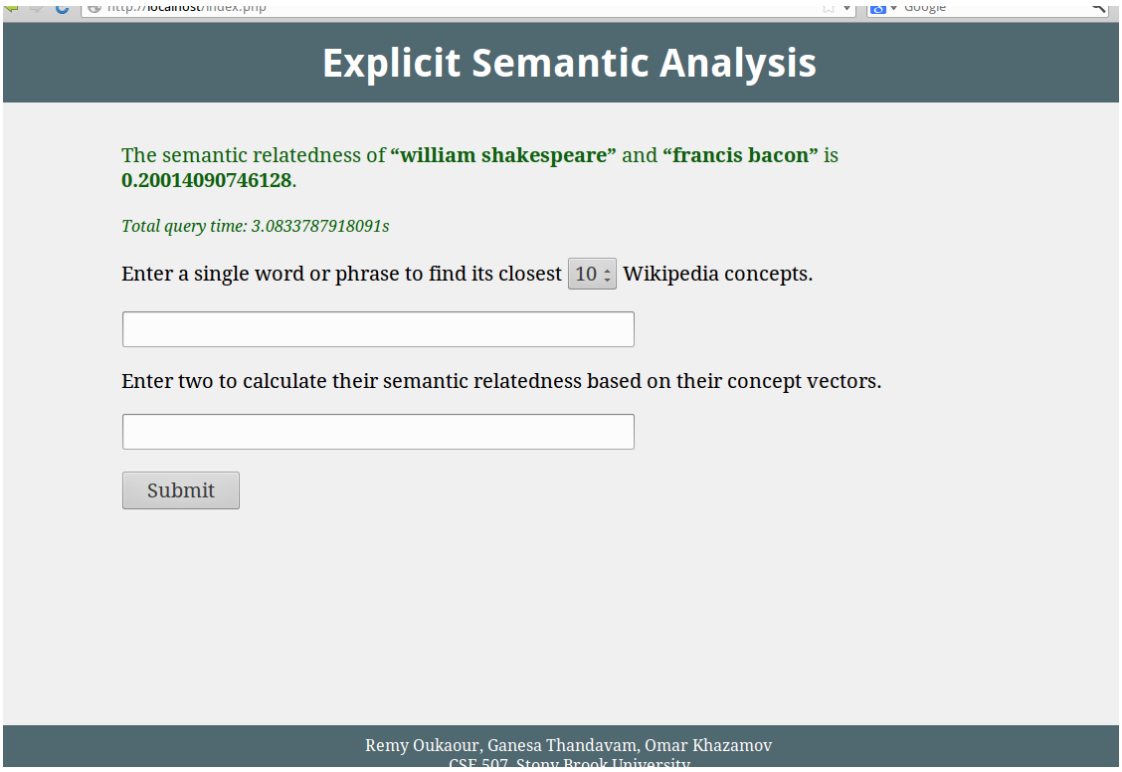

Figure 2: Evaluating the semantic relatedness of two phrases, "William Shakespeare" and "Francis Bacon."

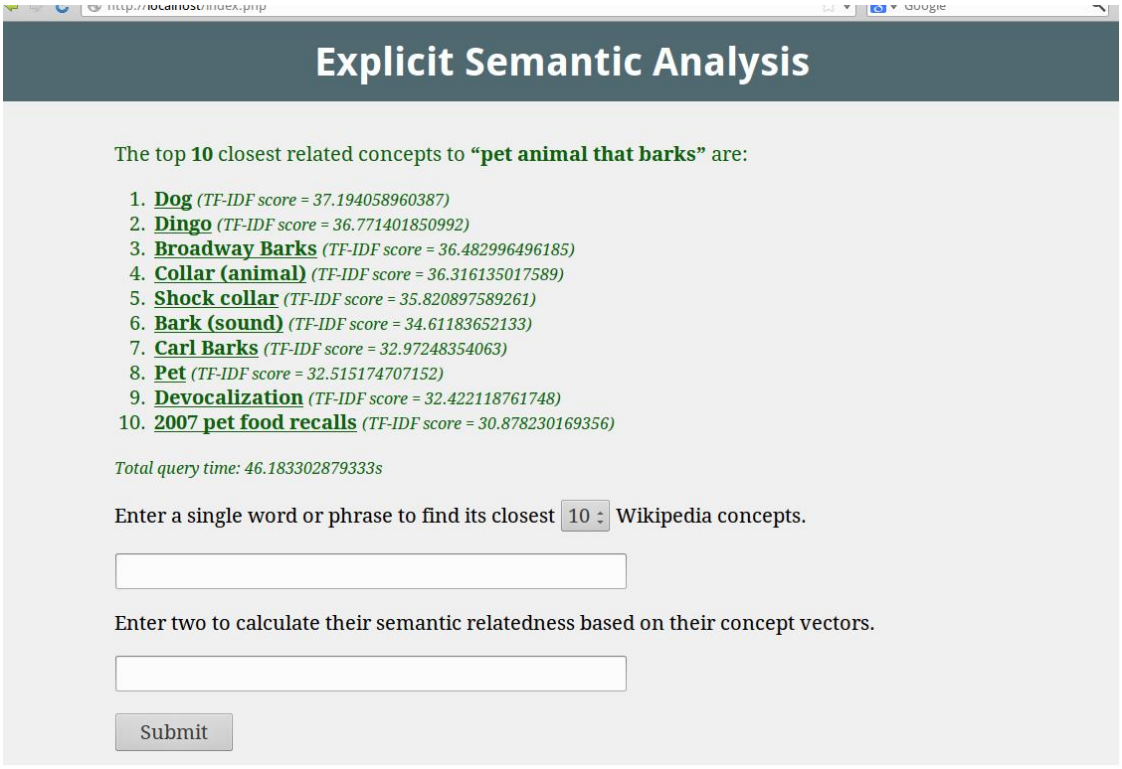

Figure 3: Reverse dictionary output for "pet animal that barks"; it correctly outputs "Dog."

Evaluation

We used extrinsic evaluation to compare our application with existing ones. To evaluate its performance for semantic relatedness, we used the WordSimilarity-353 dataset, a collection of 353 pairs of words along with their average human judgment of similarity on a scale from 1 to 10. We computed the Pearson correlation coefficient of our method's similarity scores and the human scores, as well as scores for competing method including WordNet and the original ESA implementation using a Wikipedia corpus from 2005.

We also tested the use of ESA for word sense disambiguation. Using 27 instances from the SemEval-2010 Task 17 dataset for domain-specific English WSD, we replaced the measure of gloss overlap in the baseline Lesk similarity algorithm with our ESA relatedness measure.

Finally, we manually compared the output of our application with existing reverse dictionaries, in order to find which queries it handled better or worse.

Results

Method	**Correlation with human**
WikiRelate!	0.19–0.48
WordNet	0.33–0.35
Our ESA	0.49
Roget's Thesaurus	0.55
LSA (Latent Semantic Analysis)	0.56
Gabrilovich et al. ESA	0.75

Table 2: WordSimilarity-353 correlations with human judgments.

Our implementation of ESA outperformed the WordNet and WikiRelate! algorithms for computing semantic relatedness; we achieved a correlation of 0.49 with humans, compared to 0.19 to 0.48 for WikiRelate! and 0.34 average for WordNet. However, the original ESA implementation by Gabrilovich et al. [1] reached 0.75 correlation. We believe this is due to increased noise in our larger dataset outweighing the larger amount of relevant information compared with the 2005 Wikipedia.

Method	**Precision**
Baseline (Lesk)	0.29
Extended Lesk	0.29
Lesk + ESA	0.51
Extended Lesk + ESA	0.59

Table 3: Word sense disambiguation precision results.

Baseline Lesk and extended Lesk performed similarly due to the small size of the data set. For larger data sets the difference is considerable; however, due to limitations on the running time of our ESA relatedness function, we had to restrict ourselves to the smaller set. Adding ESA to the extended Lesk algorithm nearly doubled its precision from 29% to 59%.

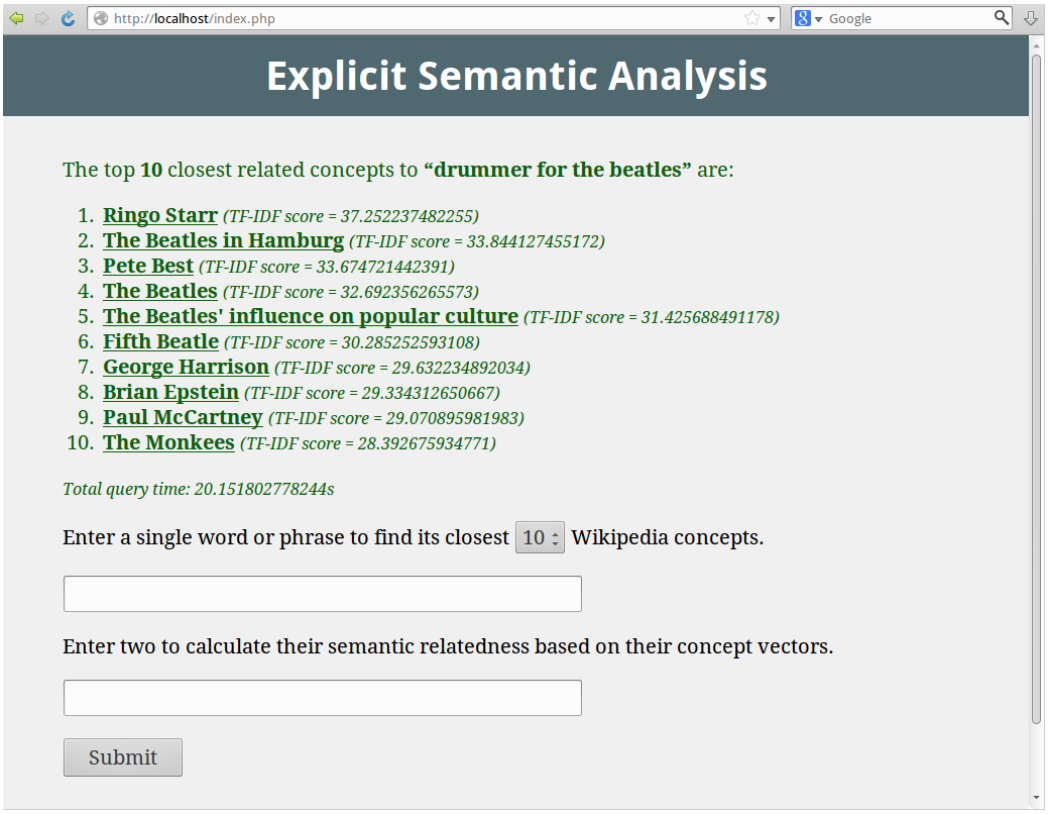

Figure 4: Reverse dictionary results for "drummer for the Beatles" (Ringo Starr).

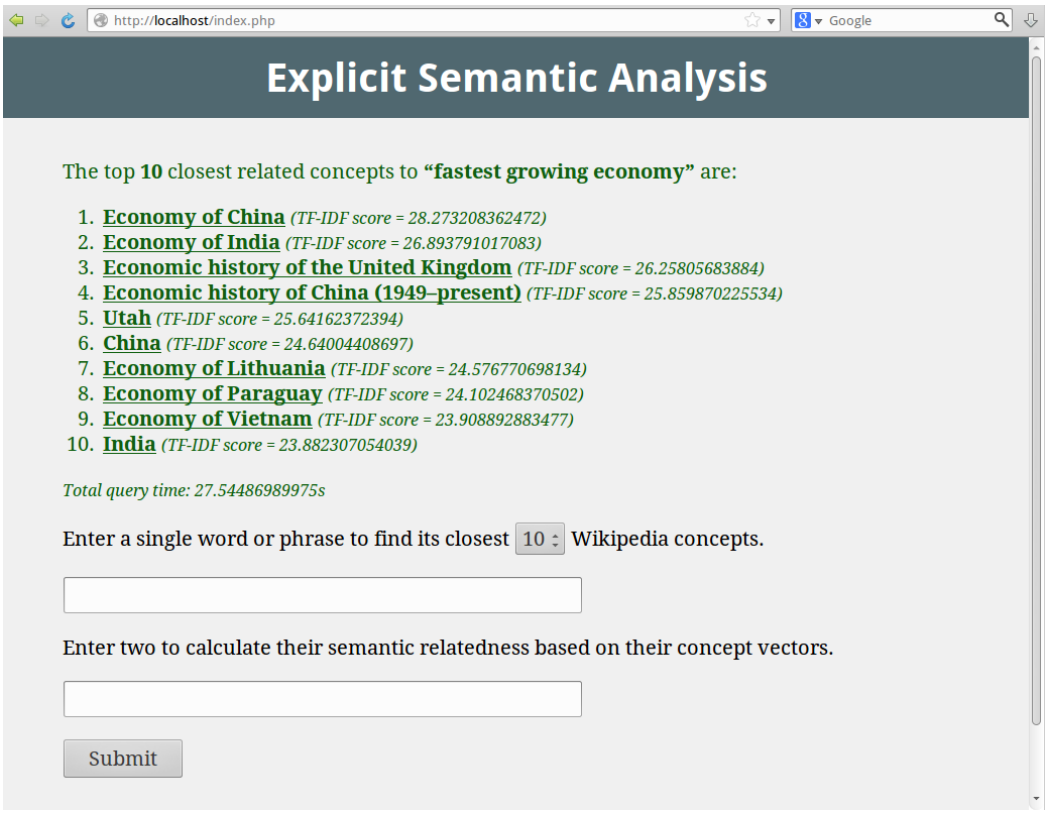

Figure 5: Reverse dictionary results for "fastest growing economy" (China).

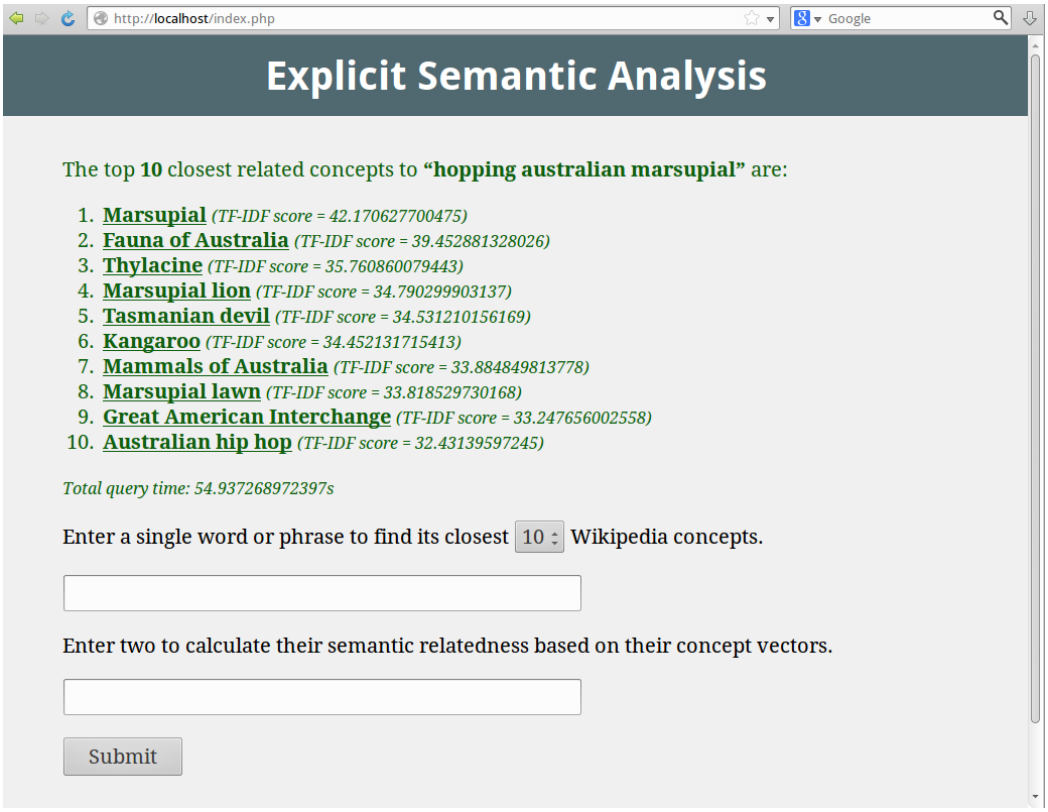

Figure 6: Reverse dictionary results for "hopping Australian marsupial" (kangaroo).

For reverse dictionary lookup, we found that our ESA implementation performs best for fact-related queries rather than dictionary definitions. Most Wikipedia articles are about people, places, and things, so inputting the definition of a verb or adjective is unlikely to succeed. On the other hand, using a recent database gives it up-to-date knowledge for factual queries about current events.

Limitations

One limitation of our tool is its response time. We reduced the 40 GB Wikipedia database to a 12 GB inverted index, which had to be read from disk every time.

The ESA implementation by Gabrilovich et al. [1] started with 4 GB of Wikipedia data. This would not only be faster to parse and query, but it would have fewer concepts and thus a greater percentage of relevant concepts (since the first Wikipedia articles created were likely to be the most relevant). This is likely why their implementation achieved 75% precision for WSD while ours reached 49%.

To deal with our noisy Wikipedia concepts, we realized the need for performing dimensionality reduction on our feature vectors. A machine learning algorithm such as singular value decomposition (SVD) would be helpful for this task.

It is also important to note that ESA itself has some inherent limitations. With many concepts, noise is introduced; with few concepts, it is difficult to represent all the different meanings that a word may have. As such, dimensionality reduction could have an adverse impact on the accuracy of our approach. Furthermore, ESA lacks a hierarchy structure on its concepts, which WordNet has. Organizing the concepts into a tree would more closely relate to how humans generalize ideas and view relations between words.

Future Work

In order to reduce the noise in our data, we plan to perform a SVD or principal component analysis on our current database in order to isolate the most useful concepts. We expect this to improve the precision of WSD using a dimensionality-reduced database.

As mentioned above, a limitation of ESA is its lack of relations between concepts. The semantic relations between WordNet's synsets—synonym, antonym, hyponym, hypernym, and so on—add valuable context to it apart from the content of the dictionary glosses. One possibility for accomplishing this with ESA would be to parse Wikipedia category data and use it to build a concept tree. This data is not easily parsed without multiple passes over the raw database, which is why we did not attempt it.

Conclusion

The idea of extracting world knowledge based on Wikipedia concepts seems to work well for a reverse dictionary implementation, as well as for less direct applications like word sense disambiguation. We observed the limitations of representing the semantics of a word without hierarchical relations to other words. In the future, we hope to improve upon the ESA approach by imposing semantic relations between the concepts, possibly by using Wikipedia categories; as well as to reduce the size of our data without compromising accuracy by using SVD/PCA.

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.