Approximate Searching for Distributional Similarity

James Gorman and **James R. Curran**
School of Information Technologies
University of Sydney
NSW 2006, Australia
{jgorman2,james}@it.usyd.edu.au

Abstract

Distributional similarity requires large volumes of data to accurately represent infrequent words. However, the nearest-neighbour approach to finding synonyms suffers from poor scalability. The Spatial Approximation Sample Hierarchy (SASH), proposed by Houle (2003b), is a data structure for approximate nearest-neighbour queries that balances the efficiency/approximation trade-off. We have intergrated this into an existing distributional similarity system, tripling efficiency with a minor accuracy penalty.

1 Introduction

With the development of WordNet (Fellbaum, 1998) and large electronic thesauri, information from lexical semantic resources is regularly used to solve NLP problems. These problems include collocation discovery (Pearce, 2001), smoothing and estimation (Brown et al., 1992; Clark and Weir, 2001) and question answering (Pasca and Harabagiu, 2001).

Unfortunately, these resources are expensive and time-consuming to create manually, and tend to suffer from problems of bias, inconsistency, and limited coverage. In addition, lexicographers cannot keep up with constantly evolving language use and cannot afford to build new resources for the many subdomains that NLP techniques are being applied to. There is a clear need for methods to extract lexical semantic resources automatically or tools that assist in their manual creation and maintenance.

Much of the existing work on automatically extracting resources is based on the *distributional hypothesis* that *similar words appear in similar contexts*. Existing approaches differ primarily in their definition of "context", e.g. the surrounding words or the entire document, and their choice of distance metric for calculating similarity between the vector of contexts representing each term. Finding synonyms using distributional similarity involves performing a nearest-neighbour search over the context vectors for each term. This is very computationally intensive and scales according to the vocabulary size and the number of contexts for each term. Curran and Moens (2002b) have demonstrated that dramatically increasing the quantity of text used to extract contexts significantly improves synonym quality. Unfortunately, this also increases the vocabulary size and the number of contexts for each term, making the use of huge datasets infeasible.

There have been many data structures and approximation algorithms proposed to reduce the computational complexity of nearest-neighbour search (Chávez et al., 2001). Many of these approaches reduce the search space by using clustering techniques to generate an index of near-neighbours. We use the Spacial Approximation Sample Hierarchy (SASH) data structure developed by Houle (2003b) as it allows more control over the efficiency-approximation trade-off than other approximation methods.

This paper describes integrating the SASH into an existing distributional similarity system (Curran, 2004). We show that replacing the nearest-neighbour search improves efficiency by a factor of three with only a minor accuracy penalty.

Proceedings of the ACL-SIGLEX Workshop on Deep Lexical Acquisition, pages 97–104,
Ann Arbor, June 2005. ©2005 Association for Computational Linguistics

2 Distributional Similarity

Distributional similarity systems can be separated into two components. The first component extracts the contexts from raw text and compiles them into a statistical description of the contexts each term appears in. The second component performs nearest-neighbour search or clustering to determine which terms are similar, based on distance calculations between their context vectors. The approach used in this paper follows Curran (2004).

2.1 Extraction Method

A *context relation* is defined as a tuple (w, r, w') where w is a term, which occurs in some grammatical relation r with another word w' in some sentence. We refer to the tuple (r, w') as an *attribute* of w. For example, (dog, diect-obj, walk) indicates that dog was the direct object of walk in a sentence.

Context extraction begins with a Maximum Entropy POS tagger and chunker (Ratnaparkhi, 1996). The Grefenstette (1994) relation extractor produces context relations that are then lemmatised using the Minnen et al. (2000) morphological analyser. The relations for each term are collected together and counted, producing a context vector of attributes and their frequencies in the corpus.

2.2 Measures and Weights

Both nearest-neighbour and cluster analysis methods require a distance measure that calculates the similarity between context vectors. Curran (2004) decomposes this measure into *measure* and *weight* functions. The *measure* function calculates the similarity between two weighted context vectors and the *weight* function calculates a weight from the raw frequency information for each context relation.

The SASH requires a distance measure that preserves metric space (see Section 4.1). For these experiments we use the JACCARD (1) measure and the TTEST (2) weight, as Curran and Moens (2002a) found them to have the best performance in their comparison of many distance measures.

$$\frac{\sum_{(r,w')} \min(\mathrm{wgt}(w_m, *_r, *_{w'}), \mathrm{wgt}(w_n, *_r, *_{w'}))}{\sum_{(r,w')} \max(\mathrm{wgt}(w_m, *_r, *_{w'}), \mathrm{wgt}(w_n, *_r, *_{w'}))} \quad (1)$$

$$\frac{p(w, r, w') - p(*, r, w')p(w, *, *)}{\sqrt{p(*, r, w')p(w, *, *)}} \quad (2)$$

3 Nearest-neighbour search

The simplest algorithm for finding synonyms is nearest-neighbour search, which involves pairwise vector comparison of the target term with every term in the vocabulary. Given an n term vocabulary and up to m attributes for each term, the asymptotic time complexity of nearest-neighbour search is $O(n^2m)$. This is very expensive with even a moderate vocabulary and small attribute vectors making the use of huge datasets infeasible.

3.1 Heuristic

Using cutoff to remove low frequency terms can significantly reduce the value of n. In these experiments, we used a cutoff of 5. However, a solution is still needed to reduce the factor m. Unfortunately, reducing m by eliminating low frequency contexts has a significant impact on the quality of the results.

Curran and Moens (2002a) propose an initial heuristic comparison to reduce the number of full $O(m)$ vector comparisons. They introduce a bounded vector (length k) of *canonical* attributes, selected from the full vector, to represent the term. The selected attributes are the most strongly weighted verb attributes: Curran and Moens chose these relations as they generally constrain the semantics of the term more and partake in fewer idiomatic collocations.

If a pair of terms share at least one canonical attribute then a full similarity comparison is performed, otherwise the terms are not considered similar. If a maximum of p positive results are returned, our complexity becomes $O(n^2k + npm)$, which, since k is constant, is $O(n^2 + npm)$.

4 The SASH

The SASH approximates a nearest-neighbour search by pre-computing some of the near-neighbours of each node (terms in our case). It is arranged as a multi-leveled pyramid, where each node is linked to its (approximate) near-neighbours on the levels above and below. This produces multiple paths between nodes, allowing the SASH to shape itself to the data set (Houle, 2003a). This graph is searched by finding the near-neighbours of the target node at each level. The following description is adapted from Houle (2003b).

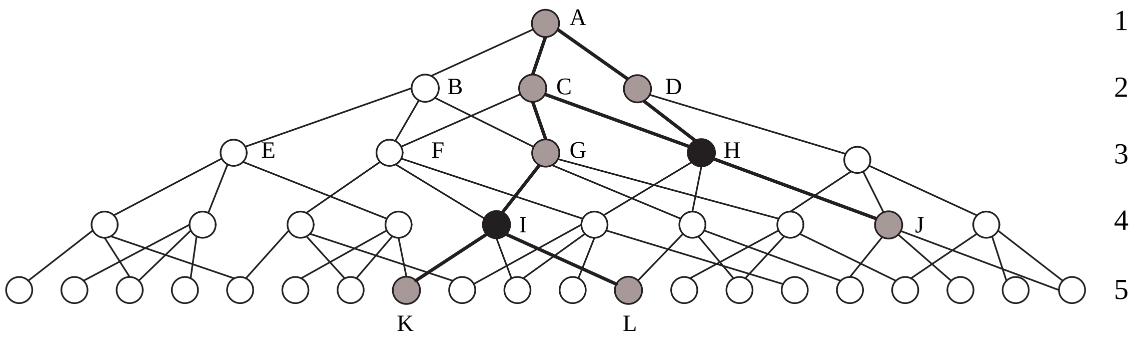

Figure 1: A SASH, where $p = 2$, $c = 3$ and $k = 2$

4.1 Metric Spaces

The SASH organises nodes that can be measured in *metric space*. Although it is not necessary for the SASH to work, only in this space can performance be guaranteed. Our meaures produce a *metric-like* space for the terms derived from large datasets.

A domain D is a *metric space* if there exists a function $dist : D \times D \rightarrow R^{\geq 0}$ such that:

1. $dist(p, q) \geq 0 \ \forall \ p, q \in D$ (*non-negativity*)
2. $dist(p, q) = 0$ iff $p = q \ \forall \ p, q \in D$ (*equality*)
3. $dist(p, q) = dist(q, p) \ \forall \ p, q \in D$ (*symmetry*)
4. $dist(p, q) + dist(q, r) \geq dist(p, r)$ $\forall \ p, q, r \in D$ (*triangle inequality*)

We invert the similarity measure to produce a distance, resulting in condition 2 not being satisfied since $dist(p, p) = x$, $x > 0$. For most measures x is constant, so $dist(p, q) > dist(p, p)$ if $p \neq q$ and p and q do not occur in exactly the same contexts. For some measures, e.g. DICE, $dist(p, p) > dist(p, q)$, that is, p is closer to q than it is to itself. These do not preserve metric space in any way, so cannot be used with the SASH.

Chávez et al. (2001) divides condition 2 into:

5. $dist(p, p) = 0 \ \forall \ p \in D$ (*reflexivity*)
6. $dist(p, q) > 0$ iff $p \neq q \ \forall \ p, q \in D$ (*strict positiveness*)

If strict positiveness is not satisfied the space is called *pseudometric*. In theory, our measures do not satisfy this condition, however in practice most large datasets will satisfy this condition.

4.2 Structure

The SASH is a directed, edge-weighted graph with the following properties:

- Each term corresponds to a unique node.
- The nodes are arranged into a hierarchy of levels, with the bottom level containing $\frac{n}{2}$ nodes and the top containing a single root node. Each level, except the top, will contain half as many nodes as the level below. These are numbered from 1 (top) to h.
- Edges between nodes are linked from consecutive levels. Each node will have at most p *parent* nodes in the level above, and c *child* nodes in the level below.
- Every node must have at least one parent so that all nodes are reachable from the root.

Figure 1 shows a SASH which will be used below.

4.3 Construction

The SASH is constructed iteratively by finding the nearest parents in the level above. The nodes are first randomly distributed to reduce any clustering effects. They are then split into the levels described above, with level h having $\frac{n}{2}$ nodes, level 2 at most c nodes and level 1 having a single root node.

The root node has all nodes at level 2 as children and each node at level 2 has the root as its sole parent. Then for each node in each level i from 3 to h, we find the set of p nearest parent nodes in level $(i - 1)$. The node then asks that parent if it can be a child. As only the closest c nodes can be children of a node, it may be the case that a requested parent rejects a child.

DIST	c	LOAD TIME
RANDOM	16	21.0hr
RANDOM	64	15.6hr
RANDOM	128	21.1hr
FOLD1500	16	50.2hr
FOLD1500	64	33.4hr
FOLD1500	128	25.7hr
SORT	16	75.5hr
SORT	64	23.8hr
SORT	128	33.8hr

Table 1: Load time distributions and values of c

If a child is left without any parents it is said to be *orphaned*. Any orphaned nodes must now find the closest node in the above level that has fewer than c children. Once all nodes have at least one parent, we move to the next level. This proceeds iteratively through the levels.

4.4 Search

Searching the SASH is also performed iteratively. To find the k nearest neighbours of a node q, we first find the k nearest neighbours at each level. At level 1 we take the single root node to be nearest. Then, for each level after, we find the k nearest unique children of the nodes found in the level above. When the last level has been searched, we return the closest k nodes from all the sets of near neighbours returned.

In Figure 1, the filled nodes demonstrate a search for the near-neighbours of some node q, using $k = 2$. Our search begins with the root node A. As we are using $k = 2$, we must find the two nearest children of A using our similarity measure. In this case, C and D are closer than B. We now find the closest two children of C and D. E is not checked as it is only a child of B. All other nodes are checked, including F and G, which are shared as children by B and C. From this level we chose G and H. We then consider the fourth and fifth levels similarly.

At this point we now have the list of near nodes A, C, D, G, H, I, J, K and L. From this we chose the two nodes closest to q: H and I marked in black. These are returned as the near-neighbours of q.

k can also be varied at each level to force a larger number of elements to be tested at the base of the SASH using, for instance, the equation:

$$k_i = \max\{\, k^{1-\frac{h-i}{\log_2 n}}, \frac{1}{2}pc \,\} \tag{3}$$

We use this geometric function in our experiments.

4.5 Complexity

When measuring the time complexity, we consider the number of distance measurements as these dominate the computation. If we do not consider the problem of assigning parents to orphans, for n nodes, p parents per child, at most c children per parent and a search returning k elements, the loose upper bounds are:

SASH construction

$$pcn \log_2 n \tag{4}$$

Approx. k-NN query (uniform)

$$ck \log_2 n \tag{5}$$

Approx. k-NN query (geometric)

$$\frac{k^{1+\frac{1}{\log_2 n}}}{k^{\frac{1}{\log_2 n}}-1} + \frac{pc^2}{2}\log_2 n \tag{6}$$

Since the average number of children per node is approximately $2p$, practical complexities can be derived using $c = 2p$.

In Houle's experiments, typically less than 5% of computation time was spent assigning parents to orphans, even for relatively small c. In some of our experiments we found that low values of c produced significantly worse load times that for higher values, but this was highly dependant on the distribution of nodes. Table 1 shows this with respect to several distributions and values of c.

5 Evaluation

The simplest method of evaluation is direct comparison of the extracted synonyms with a manually-created gold standard (Grefenstette, 1994). However, on small corpora, rare direct matches provide limited information for evaluation, and thesaurus coverage is a problem. Our evaluation uses a combination of three electronic thesauri: the Macquarie (Bernard, 1990), Roget's (Roget, 1911) and Moby (Ward, 1996) thesauri.

With this gold standard in place, it is possible to use precision and recall measures to evaluate the quality of the extracted thesaurus. To help overcome the problems of direct comparisons we use several measures of system performance: direct matches (DIRECT), inverse rank (INVR), and precision of the top n synonyms (P(n)), for n = 1, 5 and 10.

INVR is the sum of the inverse rank of each matching synonym, e.g. matching synonyms at ranks 3, 5 and 28 give an inverse rank score of $\frac{1}{3} + \frac{1}{5} + \frac{1}{28}$, and with at most 100 synonyms, the maximum INVR score is 5.187. Precision of the top n is the percentage of matching synonyms in the top n extracted synonyms.

The same 70 single-word nouns were used for the evaluation as in Curran and Moens (2002a). These were chosen randomly from WordNet such that they covered a range over the following properties:

frequency Penn Treebank and BNC frequencies;

number of senses WordNet and Macquarie senses;

specificity depth in the WordNet hierarchy;

concreteness distribution across WordNet subtrees.

For each of these terms, the closest 100 terms and their similarity score were extracted.

6 Experiments

The contexts were extracted from the non-speech portion of the British National Corpus (Burnard, 1995). All experiments used the JACCARD measure function, the TTEST weight function and a cutoff frequency of 5. The SASH was constructed using the geometric equation for k_i described in Section 4.4. When the heuristic was applied, the TTESTLOG weight function was used with a canonical set size of 100 and a maximum frequency cutoff of 10,000.

The values 4–16, 8–32, 16–64, and 32–128 were chosen for p and c. This gives a range of branching factors to test the balance between *sparseness*, where there is potential for erroneous fragmentation of large clusters, and *bushiness*, where more tests must be made to find near children. The $c = 4p$ relationship is derived from the simple hashing rule of thumb that says that a hash table should have roughly twice the size required to store all its elements (Houle, 2003b).

DIST	FREQUENCY		# RELATIONS	
	Mean	Median	Mean	Median
RANDOM	342	18	126	13
FOLD500	915	865.5	500	500
FOLD1000	2155	1970.5	1001	1001.5
FOLD1500	3656	3444	1506	1510.5
SORT	44753	37937.5	8290	7583.5

Table 2: Top 3 SASH level averages with $c = 128$

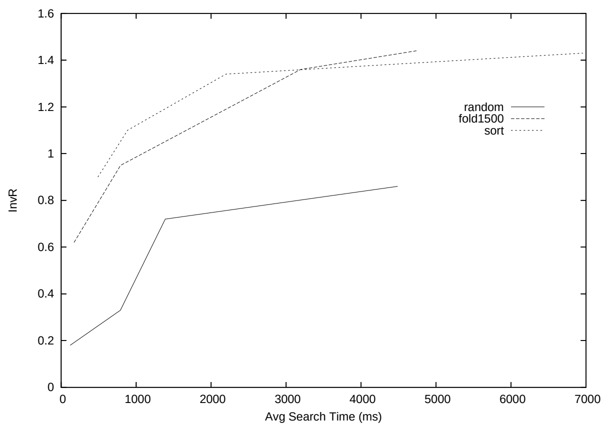

Figure 2: INVR against average search time

Our initial experiments showed that the random distribution of nodes (RANDOM) in SASH construction caused the nearest-neighbour approximation to be very inaccurate for distributional similarity. Although the speed was improved by two orders of magnitude when $c = 16$, it achieved only 13% of the INVR of the naïve implementation. The best RANDOM result was less than three times faster then the naïve solution and only 60% INVR.

In accordance with Zipf's law the majority of terms have very low frequencies. Similarity measurements made against these low frequency terms are less reliable, as accuracy increases with the number of relations and their frequencies (Curran and Moens, 2002b). This led to the idea that ordering the nodes by frequency before generating the SASH would improve accuracy.

The SASH was then generated with the highest frequency terms were near the root so that the initial search paths would be more accurate. This has the unfortunate side-effect of slowing search by up to four times because comparisons with high frequency terms take longer than with low frequency terms as they have a larger number of relations.

DIST	c	DIRECT	P(1)	P(5)	P(10)	INVR		SEARCH TIME
NAIVE		2.83	49%	41%	32%	1.43		12217ms
RANDOM	16	0.17	9%	6%	3%	0.18	13%	120ms
RANDOM	64	1.09	30%	21%	15%	0.72	50%	1388ms
RANDOM	128	1.53	31%	24%	20%	0.86	60%	4488ms
SORT	16	1.51	33%	25%	20%	0.90	63%	490ms
SORT	64	2.55	47%	38%	31%	1.34	94%	2197ms
SORT	128	2.81	49%	41%	33%	1.43	100%	6960ms

Table 3: Evaluation of different random and fully sorted distributions

This led to updating our original frequency ordering idea by recognising that we did not need the *most* accurately comparable terms at the top of the SASH, only *more* accurately comparable terms than those randomly selected.

As a first attempt, we constructed SASHs with frequency orderings that were *folded* about a chosen number of relations $\mathcal{M}$. For each term, if its number of relations m_i was greater than $\mathcal{M}$, it was given a new ranking based on the score $\frac{\mathcal{M}^2}{m_i}$. In this way, very high and very low frequency terms were pushed away from the root. The folding points this was tested for were 500, 1000 and 1500. There are many other node organising schemes we are yet to explore.

The frequency distributions over the top three levels for each ordering scheme are shown in Table 2. Zipf's law results in a large difference between the mean and median frequency values in the RANDOM results: most of the nodes have low frequency, but some high frequency results push the mean up. The four-fold reduction in efficiency for SORT (see Table 3) is a result of the mean number of relations being over 65 times that of RANDOM.

Experiments covering the full set of permutations of these parameters were run, with and without the heuristic applied. In the cases where the heuristic rejected pairs of terms, the SASH treated the rejected pairs as being as infinitely far apart. In addition, the brute force solutions were generated with (NAIVE HEURISTIC) and without (NAIVE) the heuristic.

We have assumed that all weights and measures introduce similar distribution properties into the SASH, so that the best weight and measure when performing a brute-force search will also produce the best results when combined with the SASH. Future experiments will explore SASH behaviour with other similarity measures.

7 Results

Table 3 presents the results for the initial experiments. SORT was consistently more accurate than RANDOM, and when $c = 128$, performed as well as NAIVE for all evaluation measures except for direct matches. Both SASH solutions outperformed NAIVE in efficiency.

The trade-off between efficiency and approximation accuracy is evident in these results. The most efficient result is 100 times faster than NAIVE, but only 13% accurate on INVR, with 6% of direct matches. The most accurate result is 100% accurate on INVR, with 99% of direct matches, but is less than twice as fast.

Table 4 shows the trade-off for folded distributions. The least accurate FOLD500 result is 30% accurate but 50 times faster than NAIVE, while the most accurate is 87% but less than two times faster. The least accurate FOLD1500 result is 43% accurate but 71 times faster than NAIVE, while the most accurate is 101% and two and half times faster. These results show the impact of moving high frequency terms away from the root.

Figure 2 plots the trade-off using search time and INVR at c = 16, 32, 64 and 128. For $c = 16$ every SASH has very poor accuracy. By $c = 64$ their accuracy has improved dramatically, but their search time also increased somewhat. At $c = 128$, there is only a small improvement in accuracy, coinciding with a large increase in search time. The best trade-off between efficiency and approximation accuracy occurs at the knee of the curve where $c = 64$.

When $c = 128$ both SORT and FOLD1500 perform as well as, or slightly outperform NAIVE on some evaluation measures. These evaluation measures involve the rank of correct synonyms, so if the SASH

DIST	*c*	DIRECT	P(1)	P(5)	P(10)	INVR		SEARCH TIME
FOLD500	16	0.53	23%	11%	8%	0.43	30%	243ms
FOLD500	64	1.69	49%	29%	23%	1.09	76%	2880ms
FOLD500	128	2.29	50%	35%	27%	1.25	87%	6848ms
FOLD1000	16	0.61	29%	14%	9%	0.51	35%	228ms
FOLD1000	64	2.07	49%	36%	26%	1.21	84%	3192ms
FOLD1000	128	2.57	50%	39%	31%	1.40	98%	4330ms
FOLD1500	16	0.90	30%	17%	13%	0.62	43%	171ms
FOLD1500	64	2.36	57%	39%	30%	1.36	95%	3193ms
FOLD1500	**128**	**2.67**	**53%**	**42%**	**32%**	**1.44**	**101%**	**4739ms**

Table 4: Evaluation of folded distributions

approximation was to fail to find some incorrectly proposed synonyms ranked above some other correct synonyms, those correct synonyms would have their ranking pushed up. In this way, the approximation can potentially outperform the original nearest-neighbour algorithm.

From Tables 3 and 4 we also see that as the value of *c* increases, so does the accuracy across all of the experiments. This is because as *c* increases the number of paths between nodes increases and we have a solution closer to a true nearest-neighbour search, that is, there are more ways of finding the true nearest-neighbour nodes.

Table 5 presents the results of combining the canonical attributes heuristic (see Section 3.1) with the SASH approximation. This NAIVE HEURISTIC is 14 times faster than NAIVE and 97% accurate, with 96% of direct matches. The combination has comparable accuracy and is much more efficient than the best of the SASH solutions. The best heuristic SASH results used the SORT ordering with $c = 16$, which was 37 times faster than NAIVE and 2.5 times faster than NAIVE HEURISTIC. Its performance was statistically indistinguishable from NAIVE HEURISTIC.

Using the heuristic changes the impact of the number of children *c* on the SASH performance characteristics. It seems that beyond $c = 16$ the only significant effect is to *reduce* the efficiency (often to slower than NAIVE HEURISTIC).

The heuristic interacts in an interesting way with the ordering of the nodes in the SASH. This is most obvious with the RANDOM results. The RANDOM heuristic INVR results are eight times better than the full RANDOM results. Similar, though less dramatic, results are seen with other orderings. It appears that using the heuristic changes the clustering of nearest-neighbours within the SASH so that better matching paths are chosen and more noisy matches are eliminated entirely by the heuristic.

It may seem that there are no major advantages to using the SASH with the already efficient heuristic matching method. However, our experiments have used small canonical attribute vectors (maximum length 100). Increasing the canonical vector size allows us to increase the accuracy of heuristic solutions at the cost of efficiency. Using a SASH solution would offset some of this efficiency penalty. This has the potential for a solution that is more than an order of magnitude faster than NAIVE and is almost as accurate.

8 Conclusion

We have integrated a nearest-neighbour approximation data structure, the Spacial Approximation Sample Hierarchy (SASH), with a state-of-the-art distributional similarity system. In the process we have extended the original SASH construction algorithms (Houle, 2003b) to deal with the non-uniform distribution of words within semantic space.

We intend to test other similarity measures and node ordering strategies, including a more linguistic analysis using WordNet, and further explore the interaction between the canonical vector heuristic and the SASH. The larger 300 word evaluation set used by Curran (2004) will be used, and combined with a more detailed analyis. Finally, we plan to optimise our SASH implementation so that it is comparable with the highly optimised nearest-neighbour code.

DIST	c	DIRECT	P(1)	P(5)	P(10)	INVR		SEARCH TIME
NAIVE HEURISTIC		2.72	49%	40%	32%	1.40		827ms
RANDOM	16	2.61	50%	40%	31%	1.39	99%	388ms
RANDOM	64	2.72	49%	40%	32%	1.40	100%	1254ms
RANDOM	128	2.71	49%	40%	32%	1.40	100%	1231ms
FOLD1500	16	2.53	49%	40%	31%	1.36	97%	363ms
FOLD1500	64	2.72	49%	40%	32%	1.40	100%	900ms
FOLD1500	128	2.72	49%	40%	32%	1.40	100%	974ms
SORT	**16**	**2.78**	**49%**	**40%**	**32%**	**1.41**	**100%**	**323ms**
SORT	64	2.73	49%	40%	32%	1.40	100%	1238ms
SORT	128	2.73	49%	40%	32%	1.40	100%	1049ms

Table 5: Evaluation of different distributions using the approximation

The result is distributional similarity calculated three times faster than existing systems with only a minor accuracy penalty.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful feedback and corrections. This work has been supported by the Australian Research Council under Discovery Project DP0453131.

References

John R. L. Bernard, editor. 1990. *The Macquarie Encyclopedic Thesaurus*. The Macquarie Library, Sydney, Australia.

Peter F. Brown, Vincent J. Della Pietra, Peter V. deSouza, Jennifer C. Lai, and Robert L. Mercer. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–479, December.

Lou Burnard, editor. 1995. *Users Reference Guide British National Corpus Version 1.0*. Oxford University Computing Services.

Edgar Chávez, Gonzalo Navarro, Ricardo Baeza-Yates, and José L. Marroquín. 2001. Searching in metric spaces. *ACM Computing Surveys*, 33(3):273–321, September.

Stephen Clark and David Weir. 2001. Class-based probability estimation using a semantic hierarchy. In *Proceedings of the Second Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 95–102, Pittsburgh, PA USA, 2–7 June.

James R. Curran and Marc Moens. 2002a. Improvements in automatic thesaurus extraction. In *Proceedings of the Workshop of the ACL Special Interest Group on the Lexicon (SIGLEX)*, pages 59–66, Philadelphia, USA, 12 July.

James R. Curran and Marc Moens. 2002b. Scaling context space. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 231–238, Philadelphia, USA, 7–12 July.

James R. Curran. 2004. *From Distributional to Semantic Similarity*. Ph.D. thesis, University of Edinburgh.

Christiane Fellbaum, editor. 1998. *WordNet: an electronic lexical database*. The MIT Press, Cambridge, MA USA.

Gregory Grefenstette. 1994. *Explorations in Automatic Thesaurus Discovery*. Kluwer Academic Publishers, Boston, USA.

Michael E. Houle. 2003a. Navigating massive data sets via local clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 547–552, Washington, DC, USA, 24–27 August.

Michael E. Houle. 2003b. SASH: a saptial approximation sample hierarchy for similarity search. Technical Report RT0517, IBM Reasearch, Tokyo Research Laboratory, Yamato Kanagawa, Japan, March.

Guido Minnen, John Carroll, and Darren Pearce. 2000. Robust applied morphological generation. In *Proceedings of the First International Natural Language Generation Conference*, pages 201–208, Mitzpe Ramon, Israel, 12–16 June.

Marius Pasca and Sanda Harabagiu. 2001. The informative role of wordnet in open-domain question answering. In *Proceedings of the Workshop on WordNet and Other Lexical Resources: Applications, Extensions and Customizations*, pages 138–143, Pittsburgh, PA USA, 2–7 June.

Darren Pearce. 2001. Synonymy in collocation extraction. In *Proceedings of the Workshop on WordNet and Other Lexical Resources: Applications, Extensions and Customizations*, pages 41–46, Pittsburgh, PA USA, 2–7 June.

Adwait Ratnaparkhi. 1996. A maximum entropy part-of-speech tagger. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 133–142, 17–18 May.

Peter Roget. 1911. *Thesaurus of English words and phrases*. Longmans, Green and Co., London, UK.

Grady Ward. 1996. *Moby Thesaurus*. Moby Project.