Unsupervised Phrasal Near-Synonym Generation from Text Corpora

Dishan Gupta
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213 USA
dishang@cs.cmu.edu

Jaime Carbonell
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213 USA
jgc@cs.cmu.edu

Anatole Gershman
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213 USA
anatole.gershman@gmail.com

Abstract

Unsupervised discovery of synonymous phrases is useful in a variety of tasks ranging from text mining and search engines to semantic analysis and machine translation. This paper presents an unsupervised corpus-based conditional model: Near-Synonym System (NeSS) for finding phrasal synonyms and near synonyms that requires only a large monolingual corpus. The method is based on maximizing information-theoretic combinations of shared contexts and is parallelizable for large-scale processing. An evaluation framework with crowd-sourced judgments is proposed and results are compared with alternate methods, demonstrating considerably superior results to alternatives in the literature and to thesaurus look up for multi-word phrases. The method is language-independent making it applicable to any language with sufficient monolingual electronic text.

1 Introduction

Phrasal near-synonym extraction is extremely important in domains such as natural language processing, information retrieval, text summarization, machine translation, and other AI tasks. Whereas finding near-synonyms for individual words or possibly very common canned phrases may involve no more than a thesaurus lookup, the general case of finding near-synonymous multi-word phrases requires a generative process based on analysis of large corpora. For instance our method finds the following synonyms/near-synonyms for "*it is fair to say*": "it's safe to say", "we all understand", "it's pretty clear", "we believe", "it's well known", "it's commonly accepted", and so on. The meanings of these phrases are quite close, yet that is not the case for most of their corresponding words individually. Moreover, for proper nouns our method finds orthographic variants (after all they are the best synonyms) as well as descriptive near-synonyms, e.g. for "*Al Qaeda*" it finds: "Al Qaida", "Al-Qaeda network", "jihadist group", "terrorist organization", "Bin Laden's followers". It is clear how near-synonym phrases can help in text mining, such as finding all (or most) occurrences of entities of interest in text corpora or text streams, and discovering relations expressed in different ways in large and diverse natural language corpora.

If one were issuing an information retrieval query, especially if recall truly matters, searching for synonyms of queries live may be of high value. For instance if one wants "*cheap housing*" then also searching for "affordable real estate" might prove useful. Or if typing "*heart attack*" one might also want "cardiac arrest" or "heart failure" to also be searched via query expansion. To some extent search engines are starting to offer such expanded search automatically, but in so far as one can observe, only via highly-related single-word substitutions. Moreover, to emulate a phrasal thesaurus, a live system is essential since, a precompiled database (Ganitkevitch et al., 2013) no matter how large, cannot achieve full coverage as the search space for potential grammatically coherent phrases tends to be prohibitively large.

This paper develops a method for discovering near synonym phrases based on common surrounding context relying on an extension of Harris' Distributional Hypothesis (Harris, 1985) – the more instances of common context, the more specific said context, and the longer the shared contexts, the stronger the potential synonymy relation. This unsupervised shared-context method relies only on a large monolingual corpus, and thus can be applied to any language without the need of pre-existing linguistic or lexical resources. We present the method, inspired by Carbonell et al. (2006), after outlining related work.

2 Related Work

The NLP literature addressing semantic similarity at the phrasal level is fairly sparse. Compositional distributional semantic methods attempt to formalize the meaning of compound words by applying a vector composition function on the vectors associated with its constituent words (Mitchell and Lapata, 2008; Widdows, 2008; Reddy et al, 2011), but they do not address phrasal synonymy, and instead focus on tasks such as forming NN-compounds. More importantly, the phrases (compounds) are treated as consisting of individual constituent words rather than as distinct entities, thus ignoring an essential fact that semantics of the whole might be quite different from that of its constituents. Our model, deals with phrases directly as separate entities at every level of computation.

A few approaches address phrases without breaking them into the constituting words. Barzilay and McKeown (2001) use parallel resources to construct paraphrase pairs. They include a wide variety of semantic relationships as paraphrase categories, such as siblings or hyperonyms. Ganitkevitch et al. (2013) use the bilingual pivoting technique (Bannard and Callison-Burch, 2005) along with monolingual distributional similarity features to extract lexical, phrasal and syntactic paraphrases. They officially released a collection of 220 million English paraphrase pairs known as the Paraphrase Database (PPDB) 1.0. Some other approaches (Pasca, 2005; Lin and Pantel, 2001; Berant et al., 2001) differ from ours in that, they use manually coded linguistic patterns to align only specific text fragment contexts to generate paraphrases (Pasca, 2005), and require language specific resources such as part-of-speech taggers (Pasca, 2005) and parsers (Lin and Pante, 2001). Furthermore, the latter two only find alternate constructions with the same content words, such as "X manufactures Y" infers "X's Y factory" (Lin and Pantel, 2001). Near-synonyms with a distinct set of words such as "makes ends meet" and "pay the bills" are undetectable by their methods.

Perhaps the most relevant prior work is Carbonell et al. (2006) and Metzler and Hovy (2011). Carbonell et al. (2006) briefly introduce totally-heuristic approach for the same problem to aid their context-based MT system. That work used the number of distinct contexts and their length to estimate near-synonymy. Meltzer and Hovy (2011) use similar methods and point-wise mutual information but also distribute the process using Hadoop.

3 The Near-Synonym System

The Near Synonym System (NeSS) introduces a new method which differs from other approaches in that it does not require parallel resources, unlike (Barzilay and McKeown, 2001; Lin et al., 2003; Callison-Burch et al., 2006; Ganitkevitch et al., 2013) nor does it use pre-determined sets of manually coded patterns (Lin et al., 2003; Pasca, 2005). In addition to producing synonyms or near-synonyms for individual words, it also operates at the phrase level. NeSS captures semantic similarity via n-gram distributional methods that implicitly preserve local syntactic structure without the aid of POS-tagging/chunking/parsing, making the underlying method language independent. NeSS is a Webserver, which functions as a live near-synonym search engine.

3.1 Parallel Suffix Arrays

When working on such a gigantic data set, it is absolutely necessary to have a scalable system. NeSS uses two important techniques to gain considerable speed at run-time: (1) suffix arrays, and (2) parallel computing. Suffix arrays (Manber and Myers, 1993), are powerful pattern-matching data structures that use an augmented form of binary search to locate all the occurrences of a string pattern within a corpus. They address queries such as, "Is W a substring of A?" in time $O(P + \log N)$, where P is the length of W and N is the length of A (Manber and Myers, 1993).

In our case, A is a sequence of word tokens, and $P \ll N$ since W is a phrase and A is a large corpus. Specifically, we first create a global dictionary (vocabulary) of whitespace-delineated word tokens from a large monolingual corpus and then split it into a fixed number (depending on hardware specifications) of equally sized parts. Each corpus split is sorted according to the global dictionary to construct a suffix array of its own. While starting the NeSS Webserver, all the corpus splits, the corresponding suffix arrays and the dictionary are all loaded into memory. As we will show in Section 3.2, our algorithm lends itself naturally to parallelization and thus, we put each suffix array on a separate thread of its own at run-time, which leads to further scalability.

3.2 NeSS Run-Time Architecture

We use the term "*query phrase*" to denote the input phrase for which we want to find synonyms or near synonyms.

Phase I, Context Collection and Filtering: NeSS uses the local contexts surrounding the query phrase as features to the conditional model to capture both semantic and syntactic information. A local context consists of:

1. Left context which we call "left", is a 3 to 4-gram token to the immediate left of the query phrase,
2. Right context which we call "right", is a 3 to 4-gram token to the immediate right of the query phrase (longer *n*-grams may further improve results),
3. Paired left & right context which we call "cradle", combining left and right contexts of the same query phrase occurrence with "placeholder" (OOV) words in the middle of the candidate near-synonyms.

We iterate over each occurrence of the query phrase in the data and collect the corresponding local context at each instance to form three sets of distinct lefts, distinct rights and distinct cradles, respectively. To compute *CQR* (Section 4), during iteration we also store the frequency of each context with the query phrase as well as the frequency of the query phrase in the data. Since the contexts can be collected independent of each other, we run this operation in parallel on the multi-threaded suffix arrays. After all the threads terminate, we combine the individual frequencies obtained through each suffix array to get the total frequency for the query phrase with each context as well as the total frequency of the query phrase in the entire data.

Phase II, Candidate Collection and Filtering: Once all the local contexts of the query phrase have been mined, we iterate over all the instances (excluding the ones corresponding to the query phrase itself) of each left, right and cradle in the data to collect a set of near-synonym candidate phrases. Minimum and maximum candidate lengths permitted are given by:

$$CL_{min} = 1$$
$$CL_{max} = d_1 * QL + d_0$$

where QL is query phrase length, d_0 and d_1 are constant parameters set to 2 and 2, respectively. To compute *CCS* and *k* (Section 4), we also store the frequency of each candidate with each context, the frequency of each context and the frequency of each candidate in the data. Similar to Phase I, the candidates can be collected independent of each other, and we run this operation in parallel on the multi-threaded suffix arrays. Again, once all the threads terminate, we combine the individual frequencies obtained through each suffix array to get the three total frequencies.

4 Shared Feature Gain

In this section, we describe the scoring function NeSS's conditional model uses to rank candidates. For some query phrase $Q = q$ and candidate $Y = y$, we compute the score for lefts $(L(q))$:

$$S_L(y,q) = \sum_{x \in L(q)} CQR(x,q) CCS(y,x) k(y) Inf(x)$$

where,

$$CQR(x,q) = P(X = x | Q = q) = \frac{p(x,q)}{p(q)} = \frac{f(x,q)}{f(q)}$$

$$CCS(y,x) = P(Y = y | X = x) = \frac{p(y,x)}{p(x)} = \frac{f(y,x)}{f(x)}$$

$$k(y) = \big(f(y)\big)^{-d}$$

$$Inf(x) = a * w(x) + b * l(x) + c$$

Since content words are the most important factor in $Inf(x)$, the value of a was chosen to be larger than b and c. Also, while normalizing, setting d to unity devalued the score too much. After considerable experimentation, 2, 0.5, 1 and 0.5, for a, b, c and d, respectively, proved to be the best combination.

Similarly, we compute scores for rights and cradles with the same empirical settings for each of the corresponding parameters and combine the three to get the final score:

$$S(y,q) = HM + C_{cf}.S_C(y,q) \quad (1)$$

$$HM = 2\frac{S_L S_R}{S_L + S_R}$$

where C_{cf} is empirically set to 5 to boost the score for cradle matches and S_C is the cradle score.

5 Rank-Sensitive Evaluation

For our experiments, we chose a set of 54 query phrases which included no proper nouns which included an assortment of noun phrases, verb

Method	MRṢ(5)	MRṢ(10)	MRṢ(15)	MRṢ(20)
SF	2.35	2.19	2.12	2.00
PPDB	1.97	1.80	1.62	1.48
Mavuno	2.04	1.83	1.75	1.64
Thesaurus	1.18	1.09	1.00	0.95

Table 1. Significant MRṢ improvements over PPDB, Mavuno and Roget's Thesaurus, for 23 two word query phrases.

Method	MRṢ(5)	MRṢ(10)	MRṢ(15)	MRṢ(20)
SF	**2.15**	**1.99**	1.85	1.76
PPDB	1.65	1.57	1.48	1.38
Mavuno	1.85	1.76	1.71	1.65
Thesaurus	0.50	0.47	0.43	0.43

Table 2. Significant MRṢ improvements over PPDB, Mavuno and Roget's Thesaurus, for 16 greater than two word query phrases

phrases, non-compositional phrases and nouns, verbs, and adjectives. The query phrase set comprised 15 single word, 23 two word, and 16 greater than two word phrases.

In order to objectively evaluate the output of our model, we develop a normalized metric, the mean rank-sensitive score ($MR\dot{S}$), which devalues the annotated scores for higher ranks:

$$MR\dot{S}(n) = \frac{1}{|A|}\frac{1}{|Q|}\sum_{a\in A}\sum_{q\in Q}\frac{\sum_{r=1}^{n} S_r[\log_2(r+1)]^{-1}}{\sum_{r=1}^{n}[\log_2(r+1)]^{-1}}$$

where S_r is the annotated score, n is the cutoff at the n^{th} rank, r is the rank of the candidate and A is the set of raters. $MR\dot{S}$ takes into account missing results by padding the rating sequence with zeros for the missing values. Also, due to normalization $MR\dot{S}$ is insensitive to the length of the rating sequence, i.e., $MR\dot{S}(3)$ for 2, 2, 2 is equal to $MR\dot{S}(5)$ for 2, 2, 2, 2, 2.

The annotators (6 human judges) were asked to provide ratings on each query phrase-synonym candidate combination. The ratings scaled from 0-3, where 3 indicates absolute synonymy (Zgusta, 1971), 2 indicates near-synonymy (Hirst, 1995), 1 indicates some semantic correlation such as hypernymy, hyponymy or antonymy and 0 indicates no relationship.

To show the inadequacy of thesauri lookup, we compare our model to a self-manufactured baseline from the Roget's Thesaurus. Since, like all other thesauri it primarily contains single words, we take random combinations of elements in the synonym sets of individual words (except articles and auxiliary verbs) in the query phrase to construct candidates for each of the 54 query phrases, if the thesaurus fails to provide 20 of them directly. For instance, in *"strike a balance"* we randomly select "hammer" and "harmony" as synonyms for "strike" and "balance", respectively, to form "hammer a harmony" as a candidate. We assume 100% precision for single word thesaurus entries (it is a published thesaurus), and for the rest we again employ 3 human judges.

Method	MRṢ(5)	MRṢ(10)	MRṢ(15)	MRṢ(20)
SF	2.22	2.00	1.90	1.79
PPDB	1.42	1.30	1.23	1.16
Mavuno	2.00	1.79	1.64	1.55
Thesaurus	**2.88**	**2.83**	**2.81**	**2.80**
H&S	0.27	0.29	0.28	0.26

Table 3. Significant MRṢ improvements over PPDB, Mavuno and H&S Model, for 15 single word query phrases

In addition to the thesaurus, we also compare our methods to two other multi-word systems: (1) PPDB (Ganitkevitch et al., 2013), and (2) Mavuno (Meltzer and Hovy, 2011). We also compare to Huang & Socher's (H&S, Huang et al., 2012) single prototype word embeddings. The results are shown in Tables 1, 2 and 3.

6 Concluding Remarks

We introduced a new unsupervised method for discovering phrasal near synonyms from large monolingual unannotated corpora and an evaluation method that generalizes precision@k for ranked lists of results based on multiple human numerical judgments, weighing more heavily the top of the ranked list. Our methods are based on combining elements of frequentist statistics, information theory, and scalable algorithms. Our NeSS method significantly outperforms previous automated synonym finding methods on both the lexical and phrasal level, and outperforms thesaurus-based methods for multi-word (phrasal) synonym generation, in terms of precision as well as recall. The evaluation demonstrates that scalable algorithms when combined with statistics and information theory are more effective in capturing semantics as compared to approaches that are unscalable or NLP intensive.

References

Bannard, C., & Callison-Burch, C. (2005, June). Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics* (pp. 597-604). Association for Computational Linguistics.

Barzilay, R., & McKeown, K. R. (2001, July). Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics* (pp. 50-57). Association for Computational Linguistics.

Berant, J., Dagan, I., & Goldberger, J. (2012). Learning entailment relations by global graph structure optimization. *Computational Linguistics*, *38*(1), 73-111.

Callison-Burch, C., Koehn, P., & Osborne, M. (2006, June). Improved statistical machine translation using paraphrases. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics* (pp. 17-24). ACL.

Carbonell, J. G., Klein, S., Miller, D., Steinbaum, M., Grassiany, T., & Frey, J. (2006). Context-based machine translation. The Association for Machine Translation in the Americas.

Ganitkevitch, J., Van Durme, B., & Callison-Burch, C. (2013). Ppdb: The paraphrase database. In *Proceedings of NAACL-HLT* (pp. 758-764).

Harris, Z. S. (1985). Distributional structure. In: Katz, J. J. (ed.) *The Philosophy of Linguistics.* New York: Oxford University Press. pp 26-47.

Hirst, G. (1995, March). Near-synonymy and the structure of lexical knowledge. In *AAAI Symposium on Representation and Acquisition of Lexical Knowledge: Polysemy, Ambiguity, and Generativity* (pp. 51-56).

Huang, E. H., Socher, R., Manning, C. D., & Ng, A. Y. (2012, July). Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1* (pp. 873-882). ACL.

Lin, D., & Pantel, P. (2001, August). DIRT@ SBT@ discovery of inference rules from text. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 323-328). ACM.

Lin, D., Zhao, S., Qin, L., & Zhou, M. (2003, August). Identifying synonyms among distributionally similar words. In *IJCAI* (pp. 1492-1493).

Manber, U., & Myers, G. (1993). Suffix arrays: a new method for on-line string searches. *siam Journal on Computing*, *22*(5), 935-948.

Metzler, D., & Hovy, E. (2011, August). Mavuno: a scalable and effective Hadoop-based paraphrase acquisition system. In *Proceedings of the Third Workshop on Large Scale Data Mining: Theory and Applications* (p. 3). ACM.

Mitchell, J., & Lapata, M. (2008, June). Vector-based Models of Semantic Composition. In *ACL* (pp. 236-244).

Paşca, M. (2005). Mining paraphrases from self-anchored web sentence fragments. In *Knowledge Discovery in Databases: PKDD 2005* (pp. 193-204). Springer Berlin Heidelberg.

Reddy, S., Klapaftis, I. P., McCarthy, D., & Manandhar, S. (2011). Dynamic and Static Prototype Vectors for Semantic Composition. In *IJCNLP* (pp. 705-713).

Widdows, D. (2008, March). Semantic vector products: Some initial investigations. In *Second AAAI Symposium on Quantum Interaction* (Vol. 26, p. 28th).

Zgusta, L. (1971). *Manual of lexicography* (Vol. 39). Publishing House of the Czechoslovak Academy of Sciences, Prague, Czech Republic, 1971.