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

Steve Klein
Meaningful Machines, LLC
steve@applecoreholdings.com

David Miller
Meaningful Machines, LLC
dave@applecoreholdings.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 the literature and to thesaurus look up for multi-word phrases. Moreover, the results show that the statistical scoring functions and overall scalability of the system are more important than language specific NLP tools. The method is language-independent and practically useable due to accuracy and real-time performance via parallel decomposition.

Introduction

Synonymy is recognized as having various degrees that range from complete contextual substitutability or *absolute synonymy* through to near-synonymy or *plesionymy* (Curran 2004). Hirst (1995) summarizes the definition of plesionyms (near-synonyms) as words that are close in meaning, not fully inter-substitutable but varying in their shades of denotation, connotation, implicature, emphasis or register. The above definition can be extended to multi-word phrases, for example the pair "extremely difficult" and "very challenging". In particular, synonymy is a much narrower subset as compared to the general task of paraphrasing, as the latter may encompass many forms of semantic relationships (Barzilay and McKeown 2001).

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 many 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 help in text mining, such as finding 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.

The importance of near-synonymy has been noted by many researchers, such as Metzler and Hovy (2011) in many tasks such as processing Twitter feeds. It is also crucial in information retrieval, especially if recall truly matters, where searching for synonyms of queries may be of high value. For instance if one wants "*cheap housing*" then also searching for "affordable homes" might prove useful. Or if typing "*heart attack*" one might also want "cardiac arrest" or "heart failure" to also be searched via query expansion. Search engines are starting to offer 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 (scalable) system is essential since a precompiled database (Ganitkevitch et al. 2013) no matter how large, cannot achieve full coverage.

This paper develops a new 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, relying only on a large monolingual corpus, and thus can be applied to any language without the need of pre-existing linguistic or lexical resources. Human judgments confirm that the method is able to extract some absolute synonyms and larger numbers of near-synonyms.

Single Word Phrases

Some distributional approaches cluster contexts into a set of induced "senses" (Schütze 1998; Reisinger and Mooney 2010), others dynamically modify a word's vector according to each given context (Mitchell and Lapata 2008; Thater et al. 2009). Therefore, in addition to traditional word similarity, they also try to address polysemy. For example, Reisinger and Mooney (2010) use an average prototype vector for each cluster to produce a set of vectors for each word. Extending statistical language models, neural language models (Bengio et al. 2003; Mnih and Hinton 2007; Collobert and Weston 2008) predict the next word given the previously seen words, based on 2-10 grams. Huang et al. (2012) rely on neural networks and use the ranking-loss training objective proposed by Collobert and Weston (2008), but also incorporate additional context to train word embeddings. They account for homonymy and polysemy by learning multiple embeddings per word (Reisinger and Mooney 2010). We compare directly with their well-developed word embedding method via our scoring functions (see section Experiments).

Most of the vector-based models used to represent semantics in NLP are evaluated on standard datasets such as WordSim-353 (Finkelstein et al. 2001), Miller and Charles (1991), and the SemEval 2007 Lexical Substitution Task by McCarthy and Navigli (2007). Typically, cosine similarity is used to rank the data, and these ranks are then correlated with human judgments and/or gold standards using for instance the Spearman correlation. Whereas these models may improve the performance on supervised NLP tasks such as named entity recognition and chunking (Dhillon et al. 2011), they are unable to extract (or represent) *absolute synonymy* (Zgusta 1971) and perform far inferior to our methods in extracting (or representing) *plesionymy* (Hirst 1995) even at the individual word level.

Multi-Word Phrases

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.

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 distributional similarity features to extract lexical, and phrasal paraphrases. Some other approaches (Paşca 2005; Lin and Pantel 2001; Berant et al. 2012) differ from ours in that, they use manually coded linguistic patterns to align only specific text fragment contexts to generate paraphrases (Paşca 2005), and require language specific resources such as part-of-speech taggers (Paşca 2005) and parsers (Lin and Pantel 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 a 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. Our work expands on these, relying on information theory and statistics. Moreover, NeSS is the first method to reach practical usability due to higher accuracy and real-time on-line performance via its efficient parallel algorithms.

NeSS: 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; Paşca, 2005). NeSS captures semantic similarity via *n*-gram distributional methods that implicitly preserve local syntactic structure without parsing, making the underlying method language independent. NeSS is a Web-server, which functions as a live near-synonym phrasal generator.

NeSS relies on suffix arrays, and parallel computing for real-time performance with massive corpora. Suffix arrays (Manber and Myers 1993) use an augmented form of binary trees to seek all 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 $P = |W|$ and $N = |A|$. Given a large text $A = a_0 a_1 a_2 \dots a_{N-1}$, of length N, let $A_i = a_i a_{i+1} \dots a_{N-1}$ denote the suffix of A that starts at position i. A suffix array is then a lexicographically sorted array, Pos, such that $Pos[k]$ is the start of the k^{th}lexicographically smallest suffix in the set $\{A_0, A_1, A_2 \dots A_{N-1}\}$. That is:

$$A_{Pos[0]} < A_{Pos[1]} < \cdots < A_{Pos[N-1]}$$

is the lexicographical ordering. Since it is sorted, it can

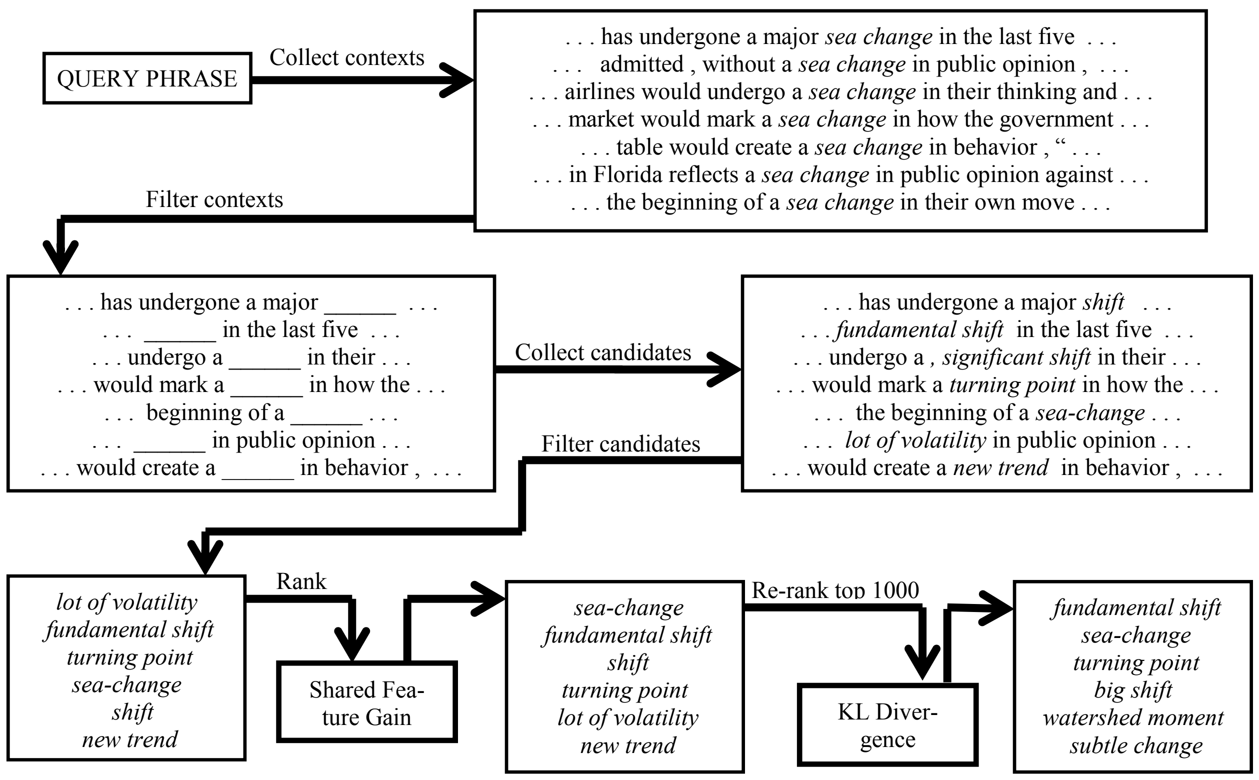
QUERY PHRASE
Collect contexts
. . . has undergone a major sea change in the last five . . .
. . . admitted , without a sea change in public opinion , . . .
. . . airlines would undergo a sea change in their thinking and . . .
. . . market would mark a sea change in how the government . . .
. . . table would create a sea change in behavior , " . . .
. . . in Florida reflects a sea change in public opinion against . . .
. . . the beginning of a sea change in their own move . . .
Filter contexts
. . . has undergone a major ______ . . .
. . . ______ in the last five . . .
. . . undergo a ______ in their . . .
. . . would mark a ______ in how the . . .
. . . beginning of a ______ . . .
. . . ______ in public opinion . . .
. . . would create a ______ in behavior , . . .
Collect candidates
. . . has undergone a major shift . . .
. . . fundamental shift in the last five . . .
. . . undergo a , significant shift in their . . .
. . . would mark a turning point in how the . . .
. . . the beginning of a sea-change . . .
. . . lot of volatility in public opinion . . .
. . . would create a new trend in behavior , . . .
Filter candidates
lot of volatility
fundamental shift
turning point
sea-change
shift
new trend
Rank
Shared Feature Gain
sea-change
fundamental shift
shift
turning point
lot of volatility
new trend
Re-rank top 1000
KL Divergence
fundamental shift
sea-change
turning point
big shift
watershed moment
subtle change

Figure 1: An overview of the NeSS run-time process design given an input phrase (sea change in the example)

locate all occurrences of a string pattern W within A by searching for the left and right boundaries of W in Pos, which takes two augmented binary searches, i.e., $2 * O(P + \log N)$ time. In our case, A is a sequence of word tokens, and $P \ll N$ since W is a phrase and A is a corpus.

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 as illustrated in Figure 1. At run-time, NeSS goes through two phases of development which we describe below.

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", defined similarly (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.

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 *contextual query phrase relevance* (see subsection Model Elements), 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 using multi-threaded suffix arrays.

Phase II, Candidate Collection and Filtering: We iterate over all the instances of each left, right and cradle in the data to collect a set of near-synonym candidate phrases, subject to minimum and maximum candidate lengths: $CL_{min} = 1$, $CL_{max} = d_1 * QL + d_0$, where QL is query phrase length, d_0 and d_1 are constant parameters. To compute *candidate contextual strength* and *normalization factors* (see subsection Model Elements), we also store the frequency of each candidate with each context, and their independently occurring frequencies, again using multi-threaded suffix arrays to expedite the process.

Computational Complexity: Considering a single suffix array, given a query phrase q, if N is the number of word tokens in the data, $f(q)$ the frequency of q, X the set of contexts (lefts, rights and cradles) of q, Y the set of mined near-synonyms candidates of q, $f(x)_{max}$ the highest frequency context in X, and XL_{max} the maximum permitted one-sided context length (in our case 4), then a tight upper bound on the run-time complexity of NeSS for q when only the shared feature gain function (see next two subsections) is used, can be expressed as:

$$O\big((QL + \log N + f(q)XL_{max}) + (XL_{max} + \log N + f(x)_{max}CL_{max})|X| + (CL_{max} + \log N)|Y|\big)$$

With parallel suffix arrays, the only difference in the above expression would be that, N, $f(q)$, and $f(x)_{max}$ would be defined local to the data corresponding one suffix array instead of the entire data. This run-time complexity is faster than other competing methods, for example, Meltzer and Hovy's (2011) method was $\sim 200X$ slower.

Model Elements

We propose a new conditional model to construct a probabilistic combination function, essentially measuring similarity between two entities based on a function over shared (common) set of features, as discussed below:

Contextual Query Phrase Relevance (CQR): Contextual Query Phrase Relevance (CQR) is a measure of how important the query phrase is to its contexts as compared to other phrases occurring together with them:

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

where p(•) and $f(\bullet)$ are probability and frequency points, respectively, in the distribution.

Candidate Contextual Strength (CCS): Candidate Contextual Strength (CCS) is a measure of how strongly related the query phrase contexts are to the potential near synonym candidate phrases as compared to other local contexts surrounding them:

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

Normalization: In order to address base-level frequency variation among candidate phrases we introduce a normalization factor: $k(y) = \left(f(y)\right)^{-d}$, where d is a constant.

Contextual Information (Inf): Some contexts still carry more semantic information than others based on their content (e.g., type and/or number of words) and our model tries to take that into account. Therefore, $Inf(x) = a * w(x) + b * l(x) + c$, where $w(x)$ is the number of content words in context x, $l(x)$ is the length of x, and a, b and c are coefficients.

Shared Feature Gain Scoring Fuction

Combining the concepts described above, we compute the score, first for left contexts ($L(q)$):

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

The model also accounts for *displaced contextual matches*, that are essentially cradle matches but with the left and right matching at different instances of the query:

$$S'_L(y,q) = \sum_{x \in L(q)} \begin{cases} 2 * CQR.CCS.k.Inf & if\ x \in DL(q) \\ CQR.CCS.k.Inf & otherwise \end{cases}$$

where $DL(q)$ is a subset of $L(q)$ which qualifies as displaces lefts. Similarly, we compute scores for rights and cradles and combine the three to get the final score:

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

$$HM = 2\frac{S'_L S'_R}{S'_L + S'_R}$$

where $C_{cf} > 1$ to boost the score for cradle matches S_C.

Kullback-Leibler Divergence Scoring Function

KL divergence (Cover and Thomas 1991) is measure of the difference between two probability distributions. We use it to measure the information lost when the contextual distribution given a candidate is used to approximate the same contextual distribution given the query phrase:

$$S_L(y,q) = \sum_{x \in L(q)} p(x|q)\log(\frac{p(x|q)}{r(x|y)})$$

$$p(x|q) = \frac{p(x,q) + c'}{p(q) + c'|L(q)|} \ \& \ r(x|y) = \frac{r(x,y) + c''}{r(y) + c''|L(q)|}$$

$$c' = p(q)\frac{0.001}{|L(q)|} \ \& \ c'' = r(y)\frac{0.001}{|L(q)|}$$

where $L(q)$ represents the combined set of lefts for the query phrase and the candidate. As before, the ratio of the probabilities $p(\bullet)$ and $r(\bullet)$, can be interpreted as the ratio of frequencies. We apply smoothing and also compute the scores for the combined rights and combined cradles, then combine the three to get the final score:

$$S(y,q) = -S_L(y,q)S_R(y,q)S_C(y,q)^2 \qquad (2)$$

We re-score and re-rank the top 1000 scoring candidates generated by the shared feature gain using Equation 2.

Parameter Training

Equation 1 contains the parameters a, b, c, and d separately for S'_L, S'_R and S_C each along with the cradle boosting parameter C_{cf}, for a total of 13 parameters. One possible parameter training scheme, is to generate training data consisting of query phrases (Q), and pick near-synonym candidates rated as highly synonymous by human judges. A natural optimization objective would then be:

$$\sum\nolimits_{q \in Q} \sum\nolimits_{y \in Y(q)} \|S(y_{best}, q) - S(y,q)\|$$

with the constraint that all the parameters > 0. $S(y,q)$ is a product of two nonnegative convex functions, and is therefore convex. This makes the optimization objective a difference of two convex functions (DC class) and its direct optimization is reserved for future work. For the present we relied on multi-start coordinate ascent with binary search instead of increasing the linear step size increase. The parameters were trained on a set of 30 query phrases, separate from the ones used in the evaluation (see section Experiments).

Experiments

Method	MRŞ(5)	MRŞ(10)	MRŞ(15)	MRŞ(20)
SF	2.35	2.19	2.12	2.00
KL	**2.45**	**2.28**	**2.17**	**2.08**
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 MRŞ improvements for both scoring functions (SF and KL) over PPDB, Mavuno and Roget's Thesaurus, for 23 two word query phrases

Method	MRŞ(5)	MRŞ(10)	MRŞ(15)	MRŞ(20)
SF	**2.15**	**1.99**	1.85	1.76
KL	2.10	1.99	**1.89**	**1.84**
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 MRŞ improvements for both scoring functions (SF and KL) over PPDB, Mavuno and Roget's Thesaurus, for 16 greater than two word query phrases

The Gigaword Corpus

We selected the very large English Gigaword Fifth Edition (Parker et al. 2011), a comprehensive archive of newswire text data, for our experiments. The corpus was split into 32 equal parts with a suffix array constructed from each split. Since, the server hardware can support up to 32 (16x2) threads in parallel, each suffix array operates on a separate thread of its own. We used 37.5% of the data (12 suffix arrays, ~1.51 billion words) for our experiments. The full Gigaword may have yielded better results, but would have run slower.

Rank-Sensitive Evaluation

For our experiments, we chose a set of 54 randomly selected query phrases including 15 single word, 23 two word phrases, and 16 longer phrases[1]. For each query phrase, 20 near-synonym candidates were generated using each of the two scoring functions and baselines. The annotators (6 human judges) were asked to provide ratings on each query phrase-synonym candidate combination. The ratings scaled from 0-3 (Rubenstein and Goodenough 1965), 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. The inter-annotator agreement was measure to be $\kappa = 0.43$ (Fleiss 1971), using binary categories for ratings 2, 3, and 0, 1, respectively, which is moder-

[1] The query phrases, annotations and other results can be downloaded at http://www.cs.cmu.edu/~dishang/.

Method	MRŞ(5)	MRŞ(10)	MRŞ(15)	MRŞ(20)
SF	2.22	2.00	1.90	1.79
KL	1.98	1.84	1.76	1.65
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 MRŞ improvements for both scoring functions (SF and KL) over PPDB, Mavuno and H&S Model, for 15 single word query phrases

ate. When the two categories were modified to 1, 2, 3 vs 0, it measured $\kappa = 0.90$ which is almost perfect agreement (Landis and Koch 1977).

We extended the standard performance measures: Mean Average Precision (MAP) and Normalized Discounted Cumulative Gain (nDCG). We did not use MAP directly because it is rank-insensitive, and is valid only for binary (0 or 1, relevant or irrelevant) rating scale. In the case of nDCG, even though it does take ordering into account it does not penalize for inferior results. For example, in our experiments the rating sequence 2, 2, 2 for the top 3 retrieval results of a query phrase would get a higher score as compared to the sequence 3, 2, 3, whereas the latter is clearly superior in quality. Besides this, nDCG does not penalize for missing results (recall) either. Our normalized metric, the mean rank-sensitive score ($MRŞ$), which devalues the annotated scores for lower ranks (further from top rank) is:

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

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. $MRŞ$ takes into account missing results by padding the rating sequence with zeros for the missing values. Also, due to normalization $MRŞ$ is insensitive to the length of the rating sequence, i.e., $MRŞ(3)$ for 2, 2, 2 is equal to $MRŞ(5)$ for 2, 2, 2, 2, 2.

Multi-Word and Single Word Comparisons

Can NeSS really perform better than thesauri, at least for multi-word phrases, and other systems in the literature?

Roget's Thesaurus: To show the inadequacy of thesauri lookup for phrasal synonyms, we compare our model to a baseline from the Roget's Thesaurus. Since, like all other thesauri it primarily contains single words, we combine elements in the synonym sets of individual words in the query phrase to construct candidates for each of the 54 query phrases. For instance, in *"strike a balance"* we randomly select "hammer" and "harmony" as synonyms for "strike" and "balance", respectively, to form "hammer a harmony"

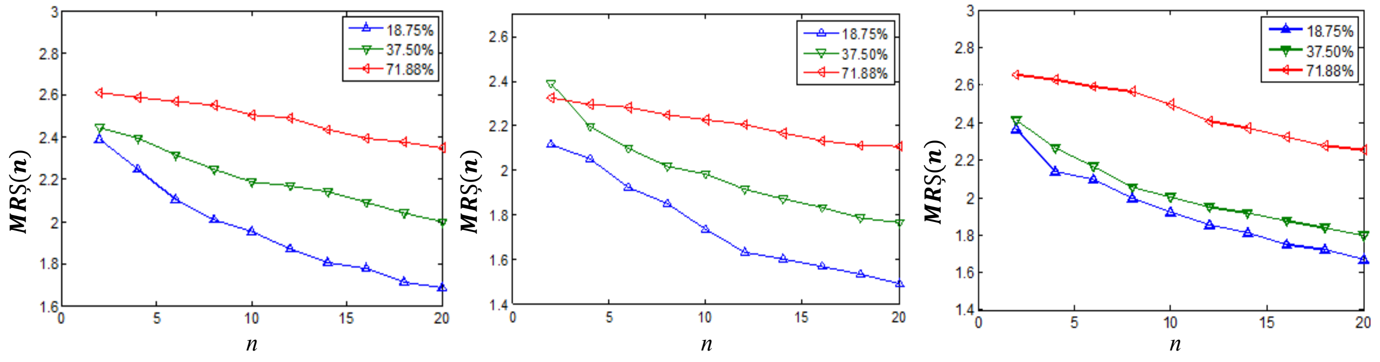

Figure 2: $\boldsymbol{MRŞ(n)}$ plots for 2 word (left), > 2 word (middle) and single word (right) phrases, using shared feature gain function, for 18.75%, 37.50% and 71.88% of the Gigaword corpus. NeSS's retrieval quality improves with increasing corpus size.

as a candidate. We assume 100% thesaurus precision for single word thesaurus entries (it is a published thesaurus), and for the rest we again employ 3 human judges. Tables 1, 2 and 3 compare the $MRŞ$ scores for shared feature gain, KL divergence and the thesaurus for two word, greater than two word and single word query phrases, separately. We can clearly see the performance advantage of our approach increases in query phrase length. Like the rating scale, $MRŞ$ ranges from 0 to 3 and thus, a difference of more than 1 and 1.3 at each cutoff for two word and greater than two word query phrases, respectively, further signifies the considerable superiority of our methods over thesauri composition. Note that both functions peak at the two word level, and shared feature gain performs stronger for single word queries whereas KL divergence takes the lead for longer ones.

Since $MRŞ$ is insensitive to cutoff point due to normalization, the observation that both our scoring functions produce greater scores at stricter cutoffs (i.e. lower values of n) implies that our model is able to discriminate stronger semantic matches from relatively weaker ones and ranks the highly synonymous candidates higher.

The Paraphrase Database: We also compare our methods to the machine translation technique by Ganitkevitch et al. (2013), PPDB 1.0. The English portion of PPDB 1.0 contains over 220 million paraphrases. We extracted the top 20 near-synonyms for our 54 query phrases from the 73 million phrasal and 8 million lexical paraphrase pairs, using the Annotated Gigaword distributional similarity scores provided in the database for ranking the candidates. Again, 6 human judges provided the ratings. Again, from Tables 1, 2 and 3, it is clear that our methods are better at ranking and recall at every cutoff point as well as phrase length. Considering the fact that NeSS operates on a monolingual corpus, does not require any NLP specific resources, and is a live retrieval system, as compared to PPDB which is none of the above, this is quite a significant result.

Mavuno: We also compare with Mavuno, an open-source Hadoop-based scalable paraphrase acquisition toolkit developed by Meltzer and Hovy (2011). Specifically, they define the context of a phrase as the concatenation of the n-grams to the immediate left and right of the phrase, and set the minimum and maximum lengths of an n-gram context to be 2 and 3, respectively, but they use point-wise mutual information weighted (Lin and Pantel 2001) phrase vectors, to compute cosine similarity as a measure of relatedness between two phrases. That is,

$$\cos(p,p') = \sum_{c\in C(p)} \frac{pmi(p,c) * pmi(p',c)}{\sqrt{\sum_{c'\in C(p)} pmi^2(p,c') * \sum_{c''\in C(p')} pmi^2(p',c'')}}$$

where $C(p)$ represents the context vector of phrase p.

We re-implemented the above scoring function in NeSS on our data (37.5% of the preprocessed English Gigaword Fifth Edition). The results shown in Tables 1, 2 and 3 demonstrate that both our scoring functions are superior.

Word embedding: Recently, "word embedding" neural network approaches have been quite popular in building vector representations that capture semantic word relationships. To gauge their effectiveness, we compare with the single prototype word embeddings trained by Huang and Socher (2012). From the $MRŞ$ comparisons in Table 3, it is clear that the H&S model is inadequate for the task of synonym extraction. We are unable to make comparisons to their multi-prototype model because they trained it only for 6162 most frequent words in the vocabulary. We also tried to train the word embedding model on 10% of the Gigaword corpus, but the task proved to be infeasible, since it would take about 2-years.

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 judgments, weighing more heavily the top of the ranked list. Our methods in NeSS are based on combining elements of frequentist statistics, information theory, and scalable algorithms. NeSS 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. Suggested future work includes:

- Testing NeSS on multiple languages, since it contains no English-specific assumptions or knowledge.
- Fully parallelizing NeSS as an efficient cloud-based phrasal synonym server.
- Task based evaluations, such as web search.

Acknowledgments

We would like to thank Meaningful Machines for the financial support that enabled this work to come to fruition.

References

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.

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.

Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C., Jaz K, Hofmann, T., Poggio, T., & Shawe-taylor, J. (2003). A neural probabilistic language model. *Journal of Machine Learning Research* (pp. 1137-1155).

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.

Collobert, R., & Weston, J. (2008, July). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning* (pp. 160-167). ACM.

Cover, T. M., & Thomas, J. A. (2012). *Elements of information theory*. John Wiley & Sons.

Curran, J. R. (2004). From distributional to semantic similarity. *Technical Report*.

Dhillon, P., Foster, D. P., & Ungar, L. H. (2011). Multi-view learning of word embeddings via cca. In *Advances in Neural Information Processing Systems*(pp. 199-207).

Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., & Ruppin, E. (2001, April). Placing search in context: The concept revisited. In*Proceedings of the 10th international conference on World Wide Web* (pp. 406-414). ACM.

Fleiss, J. L. (1971). Measuring nominal scale agreement among many raters.*Psychological bulletin*, *76*(5), 378.

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.

Landis, J. R., & Koch, G. G. (1977). The measurement of observer agreement for categorical data. *biometrics*, 159-174.

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.

McCarthy, D., & Navigli, R. (2007, June). Semeval-2007 task 10: English lexical substitution task. In *Proceedings of the 4th International Workshop on Semantic Evaluations* (pp. 48-53). Association for Computational Linguistics.

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.

Miller, G. A., & Charles, W. G. (1991). Contextual correlates of semantic similarity. *Language and cognitive processes*, *6*(1), 1-28.

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

Mnih, A., & Hinton, G. (2007, June). Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning* (pp. 641-648). ACM.

Parker, R., Graff, D., Kong, J., Chen, K., and Maeda, K. (2011). English Gigaword Fifth Edition. *Linguistic Data Consortium*, Philadelphia.

Pasca, 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).

Reisinger, J., & Mooney, R. J. (2010, June). Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (pp. 109-117). Association for Computational Linguistics.

Rubenstein, H., & Goodenough, J. B. (1965). Contextual correlates of synonymy. *Communications of the ACM*, *8*(10), 627-633.

Schütze, H. (1998). Automatic word sense discrimination. *Journal of Computational linguistics*, *24*(1), 97-123.

Thater, S., Dinu, G., & Pinkal, M. (2009, August). Ranking paraphrases in context. In *Proceedings of the 2009 Workshop on Applied Textual Inference*(pp. 44-47). Association for Computational Linguistics.

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.