NTNU-CORE: Combining strong features for semantic similarity

Erwin Marsi, Hans Moen, Lars Bungum, Gleb Sizov, Björn Gambäck, André Lynum
Norwegian University of Science and Technology
Department of Computer and Information and Science
Sem Sælands vei 7-9
NO-7491 Trondheim, Norway
{emarsi,hansmoe,bungum,sizov,gamback,andrely}@idi.ntnu.no

Abstract

The paper outlines the work carried out at NTNU as part of the *SEM'13 shared task on Semantic Textual Similarity, using an approach which combines shallow textual, distributional and knowledge-based features by a support vector regression model. Feature sets include (1) aggregated similarity based on named entity recognition with WordNet and Levenshtein distance through the calculation of maximum weighted bipartite graphs; (2) higher order word co-occurrence similarity using a novel method called "Multi-sense Random Indexing"; (3) deeper semantic relations based on the RelEx semantic dependency relationship extraction system; (4) graph edit-distance on dependency trees; (5) reused features of the TakeLab and DKPro systems from the STS'12 shared task. The NTNU systems obtained 9th place overall (5th best team) and 1st place on the SMT data set.

1 Introduction

Intuitively, two texts are semantically similar if they roughly mean the same thing. The task of formally establishing semantic textual similarity clearly is more complex. For a start, it implies that we have a way to formally represent the intended meaning of all texts in all possible contexts, and furthermore a way to measure the degree of equivalence between two such representations. This goes far beyond the state-of-the-art for arbitrary sentence pairs, and several restrictions must be imposed. The Semantic Textual Similarity (STS) task (Agirre et al., 2012, 2013) limits the comparison to isolated sentences only (rather than complete texts), and defines similarity of a pair of sentences as the one assigned by human judges on a 0–5 scale (with 0 implying no relation and 5 complete semantic equivalence). It is unclear, however, to what extent two judges would agree on the level of similarity between sentences; Agirre et al. (2012) report figures on the agreement between the authors themselves of about 87–89%.

As in most language processing tasks, there are two overall ways to measure sentence similarity, either by data-driven (distributional) methods or by knowledge-driven methods; in the STS'12 task the two approaches were used nearly equally much. Distributional models normally measure similarity in terms of word or word co-occurrence statistics, or through concept relations extracted from a corpus. The basic strategy taken by NTNU in the STS'13 task was to use something of a "feature carpet bombing approach" in the way of first automatically extracting as many potentially useful features as possible, using both knowledge and data-driven methods, and then evaluating feature combinations on the data sets provided by the organisers of the shared task.

To this end, four different types of features were extracted. The first (Section 2) aggregates similarity based on named entity recognition with WordNet and Levenshtein distance by calculating maximum weighted bipartite graphs. The second set of features (Section 3) models higher order co-occurrence similarity relations using Random Indexing (Kanerva et al., 2000), both in the form of a (standard) sliding window approach and through a novel method called "Multi-sense Random Indexing" which aims to separate the representation of different senses of a term

*Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task*, pages 66–73, Atlanta, Georgia, June 13-14, 2013.

from each other. The third feature set (Section 4) aims to capture deeper semantic relations using either the output of the RelEx semantic dependency relationship extraction system (Fundel et al., 2007) or an in-house graph edit-distance matching system. The final set (Section 5) is a straight-forward gathering of features from the systems that fared best in STS'12: TakeLab from University of Zagreb (Šarić et al., 2012) and DKPro from Darmstadt's Ubiquitous Knowledge Processing Lab (Bär et al., 2012).

As described in Section 6, Support Vector Regression (Vapnik et al., 1997) was used for solving the multi-dimensional regression problem of combining all the extracted feature values. Three different systems were created based on feature performance on the supplied development data. Section 7 discusses scores on the STS'12 and STS'13 test data.

2 Compositional Word Matching

Compositional word matching similarity is based on a one-to-one alignment of words from the two sentences. The alignment is obtained by maximal weighted bipartite matching using several word similarity measures. In addition, we utilise named entity recognition and matching tools. In general, the approach is similar to the one described by Karnick et al. (2012), with a different set of tools used. Our implementation relies on the ANNIE components in GATE (Cunningham et al., 2002) and will thus be referred to as `GateWordMatch`.

The processing pipeline for `GateWordMatch` is: (1) tokenization by ANNIE English Tokeniser, (2) part-of-speech tagging by ANNIE POS Tagger, (3) lemmatization by GATE Morphological Analyser, (4) stopword removal, (5) named entity recognition based on lists by ANNIE Gazetteer, (6) named entity recognition based on the JAPE grammar by the ANNIE NE Transducer, (7) matching of named entities by ANNIE Ortho Matcher, (8) computing WordNet and Levenstein similarity between words, (9) calculation of a maximum weighted bipartite graph matching based on similarities from 7 and 8.

Steps 1–4 are standard preprocessing routines. In step 5, named entities are recognised based on lists that contain locations, organisations, companies, newspapers, and person names, as well as date, time and currency units. In step 6, JAPE grammar rules are applied to recognise entities such as addresses, emails, dates, job titles, and person names based on basic syntactic and morphological features. Matching of named entities in step 7 is based on matching rules that check the type of named entity, and lists with aliases to identify entities as "US", "United State", and "USA" as the same entity.

In step 8, similarity is computed for each pair of words from the two sentences. Words that are matched as entities in step 7 get a similarity value of 1.0. For the rest of the entities and non-entity words we use *LCH* (Leacock and Chodorow, 1998) similarity, which is based on a shortest path between the corresponding senses in WordNet. Since word sense disambiguation is not used, we take the similarity between the nearest senses of two words. For cases when the WordNet-based similarity cannot be obtained, a similarity based on the Levenshtein distance (Levenshtein, 1966) is used instead. It is normalised by the length of the longest word in the pair. For the STS'13 test data set, named entity matching contributed to 4% of all matched word pairs; LCH similarity to 61%, and Levenshtein distance to 35%.

In step 9, maximum weighted bipartite matching is computed using the Hungarian Algorithm (Kuhn, 1955). Nodes in the bipartite graph represent words from the sentences, and edges have weights that correspond to similarities between tokens obtained in step 8. Weighted bipartite matching finds the one-to-one alignment that maximizes the sum of similarities between aligned tokens. Total similarity normalised by the number of words in both sentences is used as the final sentence similarity measure.

3 Distributional Similarity

Our distributional similarity features use Random Indexing (RI; Kanerva et al., 2000; Sahlgren, 2005), also employed in STS'12 by Tovar et al. (2012); Sokolov (2012); Semeraro et al. (2012). It is an efficient method for modelling higher order co-occurrence similarities among terms, comparable to Latent Semantic Analysis (LSA; Deerwester et al., 1990). It incrementally builds a term co-occurrence matrix of reduced dimensionality through the use of a sliding window and fixed size *index vectors* used for training *context vectors*, one per unique term.

A novel variant, which we have called "Multi-

sense Random Indexing" (MSRI), inspired by Reisinger and Mooney (2010), attempts to capture one or more "*senses*" per unique term in an unsupervised manner, each sense represented as an individual vector in the model. The method is similar to classical sliding window RI, but each term can have multiple context vectors (referred to as "*sense vectors*" here) which are updated individually. When updating a term vector, instead of directly adding the index vectors of the neighbouring terms in the window to its context vector, the system first computes a separate *window vector* consisting of the sum of the index vectors. Then cosine similarity is calculated between the window vector and each of the term's sense vectors. Each similarity score is in turn compared to a set *similarity threshold*: if no score exceeds the threshold, the sentence vector is added as a new separate sense vector for the term; if exactly one score is above the threshold, the window vector is added to that sense vector; and if multiple scores are above the threshold, all the involved senses are merged into one sense vector, together with the window vector. This accomplishes an incremental clustering of senses in an unsupervised manner while retaining the efficiency of classical RI.

As data for training the models we used the CLEF 2004–2008 English corpus (approx. 130M words). Our implementation of RI and MSRI is based on JavaSDM (Hassel, 2004). For classical RI, we used stopword removal (using a customised versions of the English stoplist from the Lucene project), window size of 4+4, dimensionality set to 1800, 4 non-zeros, and unweighted index vector in the sliding window. For MSRI, we used a similarity threshold of 0.2, a vector dimensionality of 800, a non-zero count of 4, and window size of 5+5. The index vectors in the sliding window were shifted to create *direction vectors* (Sahlgren et al., 2008), and weighted by distance to the target term. Rare senses with a frequency below 10 were excluded. Other sliding-window schemes, including unweighted non-shifted vectors and *Random Permutation* (Sahlgren et al., 2008), were tested, but none outperformed the sliding-window schemes used.

Similarity between sentence pairs was calculated as the normalised maximal bipartite similarity between term pairs in each sentence, resulting in the following features: (1) `MSRI-Centroid`: each term is represented as the sum of its sense vectors; (2) `MSRI-MaxSense`: for each term pair, the sense-pair with max similarity is used; (3) `MSRI-Context`: for each term, its neighbouring terms within a window of 2+2 is used as context for picking a single, max similar, sense from the target term to be used as its representation; (4) `MSRI-HASenses`: similarity between two terms is computed by applying the Hungarian Algorithm to all their possible sense pair mappings; (5) `RI-Avg`: classical RI, each term is represented as a single context vector; (6) `RI-Hungarian`: similarity between two sentences is calculated using the Hungarian Algorithm. Alternatively, sentence level similarity was computed as the cosine similarity between sentence vectors composed of their terms' vectors. The corresponding features are (1) `RI-SentVectors-Norm`: sentence vectors are created by summing their constituent terms (i.e., context vectors), which have first been normalized; (2) `RI-SentVectors-TFIDF`: same as before, but TF*IDF weights are added.

4 Deeper Semantic Relations

Two deep strategies were employed to accompany the shallow-processed feature sets. Two existing systems were used to provide the basis for these features, namely the RelEx system (Fundel et al., 2007) from the OpenCog initiative (Hart and Goertzel, 2008), and an in-house graph-edit distance system developed for plagiarism detection (Røkenes, 2013).

RelEx outputs syntactic trees, dependency graphs, and *semantic frames* as this one for the sentence "*Indian air force to buy 126 Rafale fighter jets*":

```
Commerce_buy:Goods(buy,jet)
Entity:Entity(jet,jet)
Entity:Name(jet,Rafale)
Entity:Name(jet,fighter)
Possibilities:Event(hyp,buy)
Request:Addressee(air,you)
Request:Message(air,air)
Transitive_action:Beneficiary(buy,jet)
```

Three features were extracted from this: first, if there was an exact match of the frame found in s_1 with s_2; second, if there was a partial match until the first argument (`Commerce_buy:Goods(buy)`; and third if there was a match of the frame category

(Commerce_buy:Goods).

In STS'12, Singh et al. (2012) matched Universal Networking Language (UNL) graphs against each other by counting matches of relations and universal words, while Bhagwani et al. (2012) calculated WordNet-based word-level similarities and created a weighted bipartite graph (see Section 2). The method employed here instead looked at the graph edit distance between dependency graphs obtained with the Maltparser dependency parser (Nivre et al., 2006). Edit distance is the defined as the minimum of the sum of the costs of the edit operations (insertion, deletion and substitution of nodes) required to transform one graph into the other. It is approximated with a fast but suboptimal algorithm based on bipartite graph matching through the Hungarian algorithm (Riesen and Bunke, 2009).

5 Reused Features

The TakeLab '*simple*' system (Šarić et al., 2012) obtained 3rd place in overall Pearson correlation and 1st for normalized Pearson in STS'12. The source code[1] was used to generate all its features, that is, *n-gram overlap*, *WordNet-augmented word overlap*, *vector space sentence similarity*, *normalized difference*, *shallow NE similarity*, *numbers overlap*, and *stock index* features.[2] This required the full LSA vector space models, which were kindly provided by the TakeLab team. The word counts required for computing Information Content were obtained from Google Books Ngrams.[3]

The DKPro system (Bär et al., 2012) obtained first place in STS'12 with the second run. We used the source code[4] to generate features for the STS'12 and STS'13 data. Of the string-similarity features, we reused the *Longest Common Substring*, *Longest Common Subsequence* (with and without normalization), and *Greedy String Tiling* measures. From the character/word n-grams features, we used *Character n-grams* ($n = 2, 3, 4$), *Word n-grams by Containment w/o Stopwords* ($n = 1, 2$), *Word n-grams by Jaccard* ($n = 1, 3, 4$), and *Word n-grams by Jaccard w/o Stopwords* ($n = 2, 4$). Semantic similarity measures include *WordNet Similarity* based on the Resnik measure (two variants) and *Explicit Semantic Similarity* based on WordNet, Wikipedia or Wiktionary. This means that we reused all features from DKPro run 1 except for *Distributional Thesaurus*.

[1] http://takelab.fer.hr/sts/

[2] We did not use content n-gram overlap or skip n-grams.

[3] http://storage.googleapis.com/books/ngrams/books/datasetsv2.html, version 20120701, with 468,491,999,592 words

[4] http://code.google.com/p/dkpro-similarity-asl/

6 Systems

Our systems follow previous submissions to the STS task (e.g., Šarić et al., 2012; Banea et al., 2012) in that feature values are extracted for each sentence pair and combined with a gold standard score in order to train a Support Vector Regressor on the resulting regression task. A postprocessing step guarantees that all scores are in the $[0, 5]$ range and equal 5 if the two sentences are identical. SVR has been shown to be a powerful technique for predictive data analysis when the primary goal is to approximate a *function*, since the learning algorithm is applicable to continuous classes. Hence support vector *regression* differs from support vector machine *classification* where the goal rather is to take a *binary* decision. The key idea in SVR is to use a cost function for building the model which tries to ignore noise in training data (i.e., data which is too close to the prediction), so that the produced model in essence only depends on a more robust subset of the extracted features.

Three systems were created using the supplied annotated data based on Microsoft Research Paraphrase and Video description corpora (MSRpar and MSvid), statistical machine translation system output (SMTeuroparl and SMTnews), and sense mappings between OntoNotes and WordNet (OnWN). The first system (NTNU1) includes all TakeLab and DKPro features plus the GateWordMatch feature with the SVR in its default setting.[5] The training material consisted of all annotated data available, except for the SMT test set, where it was limited to SMTeuroparl and SMTnews. The NTNU2 system is similar to NTNU1, except that the training material for OnWN and FNWN excluded MSRvid and that the SVR parameter C was set to 200. NTNU3 is similar to NTNU1 except that *all* features available are included.

[5] RBF kernel, $\epsilon = 0.1$, $C = \#samples$, $\gamma = \frac{1}{\#features}$

Data	NTNU1	NTNU2	NTNU3
MSRpar	0.7262	0.7507	0.7221
MSRvid	0.8660	0.8882	0.8662
SMTeuroparl	0.5843	0.3386	0.5503
SMTnews	0.5840	0.5592	0.5306
OnWN	0.7503	0.6365	0.7200
mean	0.7022	0.6346	0.6779

Table 1: Correlation score on 2012 test data

	NTNU1		NTNU2		NTNU3	
Data	r	n	r	n	r	n
Head	0.7279	11	0.5909	59	0.7274	12
OnWN	0.5952	31	0.1634	86	0.5882	32
FNWN	0.3215	45	0.3650	27	0.3115	49
SMT	0.4015	2	0.3786	9	**0.4035**	**1**
mean	0.5519	9	0.3946	68	0.5498	12

Table 2: Correlation score and rank on 2013 test data

7 Results

System performance is evaluated using the Pearson product-moment correlation coefficient (r) between the system scores and the human scores. Results on the 2012 test data (i.e., 2013 development data) are listed in Table 1. This basically shows that except for the `GateWordMatch`, adding our other features tends to give slightly lower scores (cf. NTNU1 vs NTNU3). In addition, the table illustrates that optimizing the SVR according to cross-validated grid search on 2012 training data (here $C = 200$), rarely pays off when testing on unseen data (cf. NTNU1 vs NTNU2).

Table 2 shows the official results on the test data. These are generally in agreement with the scores on the development data, although substantially lower. Our systems did particularly well on SMT, holding first and second position, reasonably good on headlines, but not so well on the ontology alignment data, resulting in overall 9th (NTNU1) and 12th (NTNU3) system positions (5th best team). Table 3 lists the correlation score and rank of the ten best individual features per STS'13 test data set, and those among the top-20 overall, resulting from linear regression on a single feature. Features in boldface are genuinely new (i.e., described in Sections 2–4).

Overall the character n-gram features are the most informative, particularly for HeadLine and SMT. The reason may be that these not only capture word overlap (Ahn, 2011), but also inflectional forms and spelling variants.

The (weighted) distributional similarity features based on NYT are important for HeadLine and SMT, which obviously contain sentence pairs from the news genre, whereas the Wikipedia based feature is more important for OnWN and FNWN. WordNet-based measures are highly relevant too, with variants relying on path length outperforming those based on Resnik similarity, except for SMT.

As is to be expected, basic word and lemma unigram overlap prove to be informative, with overall unweighted variants resulting in higher correlation. Somewhat surprisingly, higher order n-gram overlaps ($n > 1$) seem to be less relevant. Longest common subsequence and substring appear to work particularly well for OnWN and FNWN, respectively.

`GateWordMatch` is highly relevant too, in agreement with earlier results on the development data. Although treated as a single feature, it is actually a combination of similarity features where an appropriate feature is selected for each word pair. This "vertical" way of combining features can potentially provide a more fine-grained feature selection, resulting in less noise. Indeed, if two words are matching as named entities or as close synonyms, less precise types of features such as character-based and data-driven similarity should not dominate the overall similarity score.

It is interesting to find that MSRI outperforms both classical RI and ESA (Gabrilovich and Markovitch, 2007) on this task. Still, the more advanced features, such as `MSRI-Context`, gave inferior results compared to `MSRI-Centroid`. This suggests that more research on MSRI is needed to understand how both training and retrieval can be optimised. Also, LSA-based features (see `tl.weight-dist-sim-wiki`) achieve better results than both MSRI, RI and ESA. Then again, larger corpora were used for training the LSA models. RI has been shown to be comparable to LSA (Karlgren and Sahlgren, 2001), and since a relatively small corpus was used for training the RI/MSRI models, there are reasons to believe that better scores can be achieved by both RI- and MSRI-based features by using more training data.

Features	HeadLine		OnWN		FNWN		SMT		Mean	
	r	n	r	n	r	n	r	n	r	n
CharacterNGramMeasure-3	0.72	2	0.39	2	0.44	3	**0.70**	**1**	**0.56**	**1**
CharacterNGramMeasure-4	0.69	3	0.38	5	0.45	2	0.67	6	0.55	2
CharacterNGramMeasure-2	**0.73**	**1**	0.37	9	0.34	10	0.69	2	0.53	3
tl.weight-dist-sim-wiki	0.58	14	0.39	3	**0.45**	**1**	0.67	5	0.52	4
tl.wn-sim-lem	0.69	4	**0.40**	**1**	0.41	5	0.59	10	0.52	5
GateWordMatch	0.67	8	0.37	11	0.34	11	0.60	9	0.50	6
tl.dist-sim-nyt	0.69	5	0.34	28	0.26	23	0.65	8	0.49	7
tl.n-gram-match-lem-1	0.68	6	0.36	16	0.37	8	0.51	14	0.48	8
tl.weight-dist-sim-nyt	0.57	17	0.37	14	0.29	18	0.66	7	0.47	9
tl.n-gram-match-lc-1	0.68	7	0.37	10	0.32	13	0.50	17	0.47	10
MCS06-Resnik-WordNet	0.49	26	0.36	22	0.28	19	0.68	3	0.45	11
TWSI-Resnik-WordNet	0.49	27	0.36	23	0.28	20	0.68	4	0.45	12
tl.weight-word-match-lem	0.56	18	0.37	16	0.37	7	0.50	16	0.45	13
MSRI-Centroid	0.60	13	0.36	17	0.37	9	0.45	19	0.45	14
tl.weight-word-match-olc	0.56	19	0.38	8	0.32	12	0.51	15	0.44	15
MSRI-MaxSense	0.58	15	0.36	15	0.31	14	0.45	20	0.42	16
GreedyStringTiling-3	0.67	9	0.38	6	0.31	15	0.34	29	0.43	17
ESA-Wikipedia	0.50	25	0.30	38	0.32	14	0.54	12	0.42	18
WordNGramJaccard-1	0.64	10	0.37	12	0.25	25	0.33	30	0.40	19
WordNGramContainment-1-stopword	0.64	25	0.38	7	0.25	24	0.32	31	0.40	20
RI-Hungarian	0.58	16	0.33	31	0.10	34	0.42	22	0.36	24
RI-AvgTermTerm	0.56	20	0.33	32	0.11	33	0.37	28	0.34	25
LongestCommonSubstring	0.40	29	0.30	39	0.42	4	0.37	27	0.37	26
ESA-WordNet	0.11	43	0.30	40	0.41	6	0.49	18	0.33	29
LongestCommonSubsequenceNorm	0.53	21	0.39	4	0.19	27	0.18	37	0.32	30
MultisenseRI-ContextTermTerm	0.39	31	0.33	33	0.28	21	0.15	38	0.29	33
MultisenseRI-HASensesTermTerm	0.39	32	0.33	34	0.28	22	0.15	39	0.29	34
RI-SentVectors-Norm	0.34	35	0.35	26	-0.01	51	0.24	35	0.23	39
RelationSimilarity	0.31	39	0.35	27	0.24	26	0.02	41	0.23	40
RI-SentVectors-TFIDF	0.27	40	0.15	50	0.08	40	0.23	36	0.18	41
GraphEditDistance	0.33	38	0.25	46	0.13	31	-0.11	49	0.15	42

Table 3: Correlation score and rank of the best features

8 Conclusion and Future Work

The NTNU system can be regarded as continuation of the most successful systems from the STS'12 shared task, combining shallow textual, distributional and knowledge-based features into a support vector regression model. It reuses features from the TakeLab and DKPro systems, resulting in a very strong baseline.

Adding new features to further improve performance turned out to be hard: only GateWordMatch yielded improved performance. Similarity features based on both classical and innovative variants of Random Indexing were shown to correlate with semantic textual similarity, but did not complement the existing distributional features. Likewise, features designed to reveal deeper syntactic (graph edit distance) and semantic relations (RelEx) did not add to the score.

As future work, we would aim to explore a vertical feature composition approach similar to GateWordMatch and contrast it with the "flat" composition currently used in our systems.

Acknowledgements

Thanks to TakeLab for source code of their '*simple*' system and the full-scale LSA models. Thanks to the team from Ubiquitous Knowledge Processing Lab for source code of their DKPro Similarity system.

References

Agirre, E., Cer, D., Diab, M., and Gonzalez-Agirre, A. (2012). SemEval-2012 Task 6: A pilot on semantic textual similarity. In *SEM (2012), pages 385–393.

Agirre, E., Cer, D., Diab, M., Gonzalez-Agirre, A., and Guo, W. (2013). *SEM 2013 Shared Task: Semantic textual similarity, including a pilot on typed-similarity. In **SEM 2013: The Second Joint Conference on Lexical and Computational Semantics*. Association for Computational Linguistics.

Ahn, C. S. (2011). *Automatically detecting authors' native language*. PhD thesis, Monterey, California. Naval Postgraduate School.

Banea, C., Hassan, S., Mohler, M., and Mihalcea, R. (2012). UNT: a supervised synergistic approach to semantic text similarity. In *SEM (2012), pages 635–642.

Bär, D., Biemann, C., Gurevych, I., and Zesch, T. (2012). UKP: Computing semantic textual similarity by combining multiple content similarity measures. In *SEM (2012), pages 435–440.

Bhagwani, S., Satapathy, S., and Karnick, H. (2012). sranjans : Semantic textual similarity using maximal weighted bipartite graph matching. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 579–585, Montréal, Canada. Association for Computational Linguistics.

Cunningham, H., Maynard, D., Bontcheva, K., and Tablan, V. (2002). GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 168–175, Philadelphia, Pennsylvania. ACL.

Deerwester, S., Dumais, S., Furnas, G., Landauer, T., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Fundel, K., Küffner, R., and Zimmer, R. (2007). RelEx - Relation extraction using dependency parse trees. *Bioinformatics*, 23(3):365–371.

Gabrilovich, E. and Markovitch, S. (2007). Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of The Twentieth International Joint Conference for Artificial Intelligence.*, pages 1606–1611.

Hart, D. and Goertzel, B. (2008). Opencog: A software framework for integrative artificial general intelligence. In *Proceedings of the 2008 conference on Artificial General Intelligence 2008: Proceedings of the First AGI Conference*, pages 468–472, Amsterdam, The Netherlands, The Netherlands. IOS Press.

Hassel, M. (2004). JavaSDM package.

Kanerva, P., Kristoferson, J., and Holst, A. (2000). Random indexing of text samples for latent semantic analysis. In Gleitman, L. and Josh, A., editors, *Proceedings of the 22nd Annual Conference of the Cognitive Science Society*, page 1036. Erlbaum.

Karlgren, J. and Sahlgren, M. (2001). From Words to Understanding. In Uesaka, Y., Kanerva, P., and Asoh, H., editors, *Foundations of real-world intelligence*, chapter 26, pages 294–311. Stanford: CSLI Publications.

Karnick, H., Satapathy, S., and Bhagwani, S. (2012). sranjans: Semantic textual similarity using maximal bipartite graph matching. In *SEM (2012), pages 579–585.

Kuhn, H. (1955). The Hungarian method for the assignment problem. *Naval research logistics quarterly*, 2:83–97.

Leacock, C. and Chodorow, M. (1998). Combining local context and WordNet similarity for word sense identification. *WordNet: An electronic lexical ...*.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10(8):707–710.

Nivre, J., Hall, J., and Nilsson, J. (2006). Maltparser: A data-driven parser-generator for dependency parsing. In *In Proc. of LREC-2006*, pages 2216–2219.

Reisinger, J. and Mooney, R. (2010). 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*, number June, pages 109–117.

Riesen, K. and Bunke, H. (2009). Approximate graph edit distance computation by means of bipartite graph matching. *Image and Vision Computing*, 27(7):950–959.

Røkenes, H. (2013). Graph-Edit Distance Applied to the Task of Detecting Plagiarism. Master's thesis, Norwegian University of Science and Technology.

Sahlgren, M. (2005). An introduction to random indexing. In *Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering, TKE*, volume 5.

Sahlgren, M., Holst, A., and Kanerva, P. (2008). Permutations as a Means to Encode Order in Word Space. *Proceedings of the 30th Conference of the Cognitive Science Society*.

Šarić, F., Glavaš, G., Karan, M., Šnajder, J., and Bašić, B. D. (2012). TakeLab: systems for measuring semantic text similarity. In *SEM (2012), pages 441–448.

*SEM (2012). *Proceedings of the First Joint Conference on Lexical and Computational Semantics (*SEM)*, volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation, Montreal, Canada. Association for Computational Linguistics.

Semeraro, G., Aldo, B., and Orabona, V. E. (2012). UNIBA: Distributional semantics for textual similarity. In *SEM (2012), pages 591–596.

Singh, J., Bhattacharya, A., and Bhattacharyya, P. (2012). janardhan: Semantic textual similarity using universal networking language graph matching. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 662–666, Montréal, Canada. Association for Computational Linguistics.

Sokolov, A. (2012). LIMSI: learning semantic similarity by selecting random word subsets. In *SEM (2012), pages 543–546.

Tovar, M., Reyes, J., and Montes, A. (2012). BUAP: a first approximation to relational similarity measuring. In *SEM (2012), pages 502–505.

Vapnik, V., Golowich, S. E., and Smola, A. (1997). Support vector method for function approximation, regression estimation, and signal processing. In Mozer, M. C., Jordan, M. I., and Petsche, T., editors, *Advances in Neural Information Processing Systems*, volume 9, pages 281–287. MIT Press, Cambridge, Massachusetts.