DSS: Text Similarity Using Lexical Alignments of Form, Distributional Semantics and Grammatical Relations

Diana McCarthy
Saarland University*
diana@dianamccarthy.co.uk

Spandana Gella
University of Malta
spandanagella@gmail.com

Siva Reddy
Lexical Computing Ltd.
siva@sivareddy.in

Abstract

In this paper we present our systems for the STS task. Our systems are all based on a simple process of identifying the components that correspond between two sentences. Currently we use words (that is word forms), lemmas, distributional similar words and grammatical relations identified with a dependency parser. We submitted three systems. All systems only use open class words. Our first system (`alignheuristic`) tries to obtain a mapping between every open class token using all the above sources of information. Our second system (`wordsim`) uses a different algorithm and unlike `alignheuristic`, it does not use the dependency information. The third system (`average`) simply takes the average of the scores for each item from the other two systems to take advantage of the merits of both systems. For this reason we only provide a brief description of that. The results are promising, with Pearson's coefficients on each individual dataset ranging from .3765 to .7761 for our relatively simple heuristics based systems that do not require training on different datasets. We provide some analysis of the results and also provide results for our data using Spearman's, which as a nonparametric measure which we argue is better able to reflect the merits of the different systems (`average` is ranked between the others).

1 Introduction

Our motivation for the systems entered in the STS task (Agirre et al., 2012) was to model the contribution of each linguistic component of both sentences to the similarity of the texts by finding an alignment. Ultimately such a system could be exploited for ranking candidate paraphrases of a chunk of text of any length. We envisage a system as outlined in the future work section. The systems reported are simple baselines to such a system. We have two main systems (`alignheuristic` and `wordsim`) and also a system which simply uses the average score for each item from the two main systems (`average`). In our systems we:

- only deal with open class words as tokens i.e. nouns, verbs, adjectives, adverbs. `alignheuristic` and `average` also use numbers
- assume that tokens have a 1:1 mapping
- match:
  - word forms
  - lemmas
  - distributionally similar lemmas
  - (`alignheuristic` and `average` only) argument or head in a matched grammatical relation with a word that already has a lexical mapping
- score the sentence pair based on the size of the overlap. Different formulations of the score are used by our methods

The paper is structured as follows. In the next section we make a brief mention of related work though of course there will be more pertinent related work presented and published at SemEval 2012. In section 3 we give a detailed account of the systems

[*] The first author is a visiting scholar on the Erasmus Mundus Masters Program in 'Language and Communication Technologies' (LCT, 2007–0060).

*First Joint Conference on Lexical and Computational Semantics (*SEM)*, pages 557–564,
Montréal, Canada, June 7-8, 2012. ©2012 Association for Computational Linguistics

and in section 4 we provide the results obtained on the training data on developing our systems. In section 5 we present the results on the test data, along with a little analysis using the gold standard data. In section 6 we conclude our findings and discuss our ideas for future work.

2 Related Work

Semantic textual similarity relates to textual entailment (Dagan et al., 2005), lexical substitution (McCarthy and Navigli, 2009) and paraphrasing (Hirst, 2003). The key issue for semantic textual similarity is that the task is to determine similarity, where similarity is cast as meaning equivalence. [1] In textual entailment the relation under question is the more specific relation of entailment, where the meaning of one sentence is entailed by another and a system needs to determine the direction of the entailment. Lexical substitution relates to semantic textual similarity though the task involves a lemma in the context of a sentence, candidate substitutes are not provided, and the relation at question in the task is one of substitutability. [2] Paraphrase recognition is a highly related task, for example using comparable corpora (Barzilay and Elhadad, 2003), and it is likely that semantic textual similarity measures might be useful for ranking candidates in paraphrase acquisition.

In addition to various works related to textual entailment, lexical substitution and paraphrasing, there has been some prior work explicitly on semantic text similarity. Semantic textual similarity has been explored in various works. Mihalcea et al. (2006) extend earlier work on word similarity using various WordNet similarity measures (Patwardhan et al., 2003) and a couple of corpus-based distributional measures: PMI-IR (Turney, 2002) and LSA (Berry, 1992). They use a measure which takes a summation over all tokens in both sentences. For each token they find the maximum similarity (WordNet or distributional) weighted by the inverse document frequency of that word. The distributional similarity measures perform at a similar level to the knowledge-based measures that use WordNet. Mohler and Mihalcea (2009) adapt this work for automatic short answer grading, that is matching a candidate answer to one supplied by the tutor. Mohler et al. (2011) take this application forward, combining lexical semantic similarity measures with a graph-alignment which considers dependency graphs using the Stanford dependency parser (de Marneffe et al., 2006) in terms of lexical, semantic and syntactic features. A score is then provided for each node in the graph. The features are combined using machine learning.

The systems we propose likewise use lexical similarity and dependency relations, but in a simple heuristic formulation without a man-made thesaurus such as WordNet and without machine learning.

3 Systems

We lemmatize and part-of-speech tag the data using TreeTagger (Schmid, 1994). We process the tagged data with default settings of the Malt Parser (Nivre et al., 2007) to dependency parse the data. All systems make use of a distributional thesaurus which lists distributionally similar lemmas (‘neighbours’) for a given lemma. This is a thesaurus constructed using log-dice (Rychlý, 2008) and UkWaC (Ferraresi et al., 2008). [3] Note that we use only the top 20 neighbours for any word in all the methods described below. We have not experimented with varying this threshold.

In the following descriptions, we refer to our sentences as $s1$ and $s2$ and these open classed tokens within those sentences as $t_i \in s1$ and $t_j \in s2$ where each token in either sentence is represented by a word (w), lemma (l), part-of-speech (p) and grammatical relation (gr), identified by the Malt parser, to its dependency head at a given position (hp) in the same sentence.

3.1 `alignheuristic`

This method uses nouns, verbs, adjectives, adverbs and numbers. The algorithm aligns words (w), or lemmas (l) from left to right from $s1$ to $s2$ and vice

[1] See the guidelines given to the annotators at `http://www.cs.columbia.edu/~weiwei/workshop/instructions.pdf`

[2] This is more or less semantic equivalence since the annotators were instructed to focus on meaning `http://www.dianamccarthy.co.uk/files/instructions.pdf`.

[3] This is the ukWaC distributional thesaurus available in Sketch Engine (Kilgarriff et al., 2004) at `http://the.sketchengine.co.uk/bonito/run.cgi/first\_form?corpname=preloaded/ukwac2`

versa ( ). If there is no alignment for words or lemmas then it does the same matching process ($s1$ given $s2$ and vice versa) for distributionally similar neighbours using the distributional thesaurus mentioned above ( ) and also another matching process looking for a corresponding grammatical relation identified with the Malt parser in the other sentence where the head (or argument) already has a match in both sentences ( ).

A fuller and more formal description of the algorithm follows:

1. retain nouns, verbs (not *be*), adjectives, adverbs and numbers in both sentences $s1$ and $s2$.

2. :

    (a) look for word matches

    - $w_i \in s1$ to $w_j \in s2$, left to right i.e. the first matching $w_j \in s2$ is selected as a match for w_i.
    - $w_j \in s2$ to $w_i \in s1$, left to right i.e. the first matching $w_i \in s1$ is selected as a match for w_j

    (b) and then lemma matches for any $t_i \in s1$ and $t_j \in s1$ not matched in steps 2a

    - $l_i \in s1$ to $l_j \in s2$, left to right i.e. the first matching $l_j \in s2$ is selected as a match for l_i.
    - $l_j \in s2$ to $l_i \in s1$, left to right i.e. the first matching $l_i \in s1$ is selected as a match for l_j

3. using only $t_i \in s1$ and $t_j \in s2$ not matched in the above steps:

    - : match lemma and PoS ($l + p$) with the distributional thesaurus against the top 20 most similar lemma-pos entries. That is:

        (a) For $l + p_i \in s1$, if not already matched at step 2 above, find the most similar words in the thesaurus, and match if one of these is in $l + p_j \in s2$, left to right i.e. the first matching $l + p_j \in s2$ to any of the most similar words to $l + p_i$ according to the thesaurus is selected as a match for $l + p_i \in s1$.

        (b) For $l + p_j \in s2$, if not already matched at step 2 above, find the most similar words in the thesaurus, and match if one of these is in $l + p_i \in s1$, left to right

    - : match the tokens, if not already matched at step 2 above, by looking for a head or argument relation with a token that has been matched at step 2 to a token with the inverse relation. That is:

        i For $t_i \in s1$, if not already matched at step 2 above, if $hp_i \in s1$ (the pointer to the head, i.e. parent, of t_i) refers to a token $t_x \in s1$ which has at t_k in $s2$, and there exists a $t_q \in s2$ with $gr_q = gr_i$ and $hp_q = t_k$, then match t_i with t_q

        ii For $t_i \in s1$, if not already matched at step 2 or the preceding step (3i) and if there exists another $t_x \in s1$ with a hp_x which refers to t_i (i.e. t_i is the parent, or head, of t_x) with a match between t_x and $t_k \in s2$ from step 2, [4] and where t_k has $gr_k = gr_x$ with hp_k which refers to t_q in $s2$, then match t_i with t_q [5]

        iii we do likewise in reverse for $s2$ to $s1$ and then check all matches are reciprocated with the same 1:1 mapping

Finally, we calculate the score $sim(s1, s2)$:

$$\frac{| \quad | + (wt \times | \quad + \quad |)}{|s1| + |s2|} \times 5 \qquad (1)$$

where wt is a weight of 0.5 or less (see below).

The *sim* score gives a score of 5 where two sentences have the same open class tokens, since matches in both directions are included and the denominator is the number of open class tokens in both sentences. The score is 0 if there are no matches. The thesaurus and grammatical relation matches are counted equally and are considered less important

[4] In the example illustrated in figure 1 and discussed below, t_i could be *rose* in the upper sentence ($s1$) and *Nasdaq* would be t_x and t_k.

[5] So in our example, from figure 1, t_i (*rose*) is matched with t_q (*climb*) as *climb* is the counterpart head to *rose* for the matched arguments (*Nasdaq*).

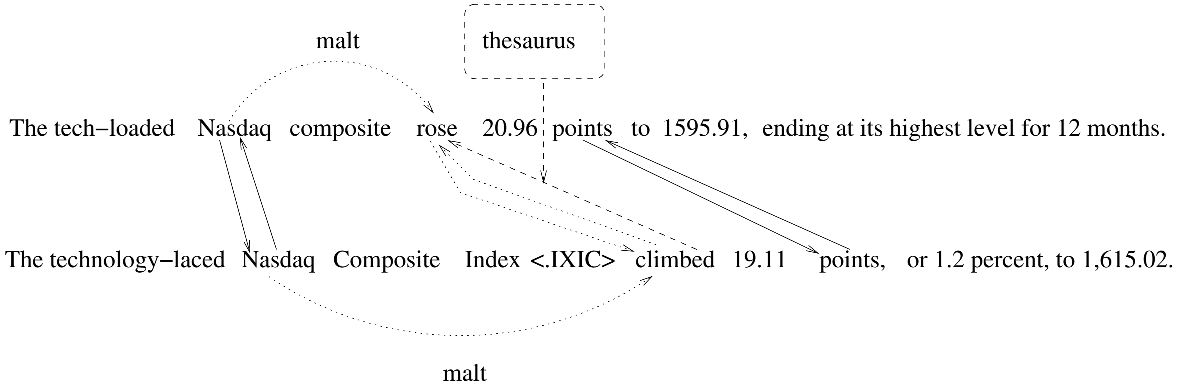
malt
thesaurus
The tech−loaded Nasdaq composite rose 20.96 points to 1595.91, ending at its highest level for 12 months.
The technology−laced Nasdaq Composite Index <.IXIC> climbed 19.11 points, or 1.2 percent, to 1,615.02.
malt

Figure 1: Example of matching with `alignheuristic`

for the score as the exact matches. We set *wt* to 0.4 for the official run, though we could improve performance by perhaps setting a bit lower as shown below in section 4.1.

Figure 1 shows an example pair of sentences from the training data in MSRpar. The solid lines show alignments between words. *Composite* and *composite* are not matched because the lemmatizer assumes that the former is a proper noun and does not decapitalise; we could decapitalise all proper nouns. The dotted arcs show parallel dependency relations in the sentences where the argument (*Nasdaq*) is matched by . The process therefore assumes the corresponding heads (*rise* and *climb*) align. In addition, finds a match from *climb* to *rise* as *rise* is in the top 20 most similar words (neighbours) in the distributional thesaurus. *climb* is not however in the top 20 for *rise* and so a link is not found in the other direction. We have not yet experimented with validating the thesaurus and grammatical relation processes together, though that would be worthwhile in future.

3.2 `wordsim`

In this method, we first choose the shortest sentence based on the number of open words. Let $s1$ and $s2$ be the shortest and longest sentences respectively. For every lemma $l_i \in s1$, we find its best matching lemma $l_j \in s2$ using the following heuristics and assigning an alignment score as follows:

1. if l_i=l_j, then the alignment score of l_i ($algnscr(l_i)$) is one.
2. else $l_i \in s1$ is matched with a lemma $l_j \in s2$ with which it has the highest distributional similarity. [6] The alignment score, $algnscr(l_i)$ is the distributional similarity between l_i and l_j (which is always less than one).

The final sentence similarity score between the pair $s1$ and $s2$ is computed as

$$sim(s1, s2) = \frac{\sum_{l_i \in s1} algnscr(l_i)}{|s1|} \qquad (2)$$

3.3 `average`

This system simple uses the average score for each item from `alignheuristic` and `wordsim`. This is simply so we can make a compromise between the merits of the two systems.

4 Experiments on the Training Data

Table 1 displays the results on training data for the system settings as they were for the final test run. We conducted further analysis for the `alignheuristic` system and that is reported in the following subsection. We can see that while the `alignheuristic` is better on the MSRpar and SMT-eur datasets, the `wordsim` outperforms it on the MSRvid dataset, which contains shorter, simpler sentences. One reason might be that the `wordsim` credits alignments in one direction only and this works well when sentences are of a similar length but can loose out on the longer paraphrase and SMT data. This behaviour is

[6] Provided this is within the top 20 most similar words in the thesaurus.

	MSRpar	MSRvid	SMT-eur
alignheuristic	0.6015	0.6994	0.5222
wordsim	0.4134	0.7658	0.4479
average	0.5337	0.7535	0.5061

Table 1: Results on training data

confirmed by the results on the test data reported below in section 5, though we cannot rule out that other factors play a part.

4.1 alignheuristic

We developed the system on the training data for the purpose of finding bugs, and setting the weight in equation 1. During development we found the optimal weight for *wt* to be 0.4. [7] Unfortunately we did not leave ourselves sufficient time to set the weights after resolving the bugs. In table 1 we report the results on the training data for the system that we uploaded, however in table 2 we report more recent results for the final system but with different values of *wt*. From table 2 it seems that results may have been improved if we had determined the final value of *wt* after debugging our system fully, however this depends on the type of data as 0.4 was optimal for the datasets with more complex sentences (MSRpar and SMT-eur).

In table 3, we report results for alignheuristic with and without the distributional similarity thesaurus ( ) and the dependency relations ( ). In table 4 we show the total number of token alignments made by the different matching processes on the training data. We see, from table 4 that the MSRvid data relies on the thesaurus and dependency relations to a far greater extent than the other datasets, presumably because of the shorter sentences where many have a few contrasting words in similar syntactic relations, for example $s1$ *Someone is drawing.* $s2$ *Someone is dancing*. [8] We see from table 3 that the use of these matching processes is less accurate for MSRvid and that while improves performance, seems to degrade performance. From table 2 it would seem that on this type of data we would get the best results by reducing *wt* to a minimum, and perhaps it would make sense to drop . Meanwhile, on the longer more complex MSRpar and SMT-eur data, the less precise and are used less frequently (relative to the) but can be seen from table 3 to improve performance on both training datasets. Moreover, as we mention above, from table 2 the parameter setting of 0.4 used for our final test run was optimal for these datasets.

wt	MSRpar	MSRvid	SMT-eur
0.5	0.5998	0.6518	0.5290
0.4	0.6015	0.6994	0.5222
0.3	0.6020	0.7352	0.5146
0.2	0.6016	0.7577	0.5059
0.1	0.6003	0.7673	0.4964
0	0.5981	0.7661	0.4863

Table 2: Results for the alignheuristic algorithm on the training data: varying *wt*

	MSR	MSR	SMT
	par	vid	-eur
- +	0.6008	0.7245	0.5129
+ -	0.5989	0.7699	0.4959
- -	0.5981	0.7661	0.4863
+ +	0.6015	0.6994	0.5222

Table 3: Results for the alignheuristic algorithm on the training data: with and without and

	MSRpar	MSRvid	SMT-eur
	10960	2349	12155
	378	1221	964
	1008	965	1755

Table 4: Number of token alignments for the different matching processes

[7] We have not yet attempted setting the weight on alignment by relation and alignment by distributional similarity separately.

[8] Note that the alignheuristic algorithm is symmetrical with respect to $s1$ and $s2$ so it does not matter which is which.

run	**ALL**	**MSRpar**	**MSRvid**	**SMT-eur**	**On-WN**	**SMT-news**
`alignheuristic`	.5253 (60)	.5735 (24)	.7123 (53)	.4781 (25)	.6984 (7)	.4177 (38)
`average`	.5490 (58)	.5020 (48)	.7645 (41)	.4875 (16)	.6677(14)	.4324 (31)
`wordsim`	.5130 (61)	.3765 (75)	.7761 (34)	.4161 (58)	.5728 (59)	.3964 (48)

Table 5: Official results: Rank (out of 89) is shown in brackets

run	**ALL**	**MSRpar**	**MSRvid**	**SMT-eur**	**On-WN**	**SMT-news**	average ρ
`alignheuristic`	0.5216	0.5539	0.7125	0.5404	0.6928	0.3655	0.5645
`average`	0.5087	0.4818	0.7653	0.5415	0.6302	0.3835	0.5518
`wordsim`	0.4279	0.3608	0.7799	0.4487	0.4976	0.3388	0.4756

Table 7: Spearman's ρ for the 5 datasets, 'all' and the average coefficient across the datasets

run	mean	Allnrm
`alignheuristic`	0.6030 (21)	0.7962 (42)
`average`	0.5943 (26)	0.8047 (35)
`wordsim`	0.5287 (55)	0.7895 (49)

Table 6: Official results: Further metrics suggested in discussion. Rank (out of 89) is shown in brackets

5 Results

Table 5 provides the official results for our submitted systems, along with the rank on each dataset. The results in the 'all' column which combine all datasets together are at odds with our intuitions. Our systems were ranked higher in every individual dataset compared to the 'all' ranking, with the exception of `wordsim` and the MSRpar dataset. This 'all' metric is anticipated to impact systems that have different settings for different types of data however we did not train our systems to run differently on different data. We used exactly the same parameter settings for each system on every dataset. We believe Pearson's measure has a significant impact on results because it is a parametric measure and as such the shape of the distribution (the distribution of scores) is assumed to be normal. We present the results in table 6 from new metrics proposed by participants during the post-results discussion: Allnrm (normalised) and mean (this score is weighted by the number of sentence pairs in each dataset). [9] The Allnrm score, proposed by a participant during the discussion phase to try and combat issues with the 'all' score, also does not accord with our intuition given the ranks of our systems on the individual datasets. The mean score, proposed by another participant, however does reflect performance on the individual datasets. Our `average` system is ranked between `alignheuristic` and `wordsim` which is in line with our expectations given results on the training data and individual datasets.

As mentioned above, an issue with the use of Pearson's correlation coefficient is that it is parametric and assumes that the scores are normally distributed. We calculated Spearman's ρ which is the non-parametric equivalent of Pearson's and uses the ranks of the scores, rather than the actual scores. [10] The results are presented in table 7. We cannot calculate the results for other systems, and therefore the ranks for our system, since we do not have the other system's outputs but we do see that the relative performance of our system on 'all' is more in line with our expectations: `average`, which simply uses the average of the other two systems for each item, is usually ranked between the other two systems, depending on the dataset. Spearman's 'all' gives a similar ranking of our three systems as the mean score. We also show average ρ. This is a macro average of the Spearman's value for the 5 datasets without weighting by the number of sentence pairs. [11]

[9] Post-results discussion is archived at `http://groups.google.com/group/sts-semeval/topics`

[10] Note that Spearman's ρ is often a little lower than Pearson's (Mitchell and Lapata, 2008)

[11] We do recognise the difficulty in determining metrics on a new pilot study. The task organisers are making every effort to make it clear that this enterprise is a pilot, not a competition and that they welcome feedback.

6 Conclusions

The systems were developed in less than a week including the time with the test data. There are many trivial fixes that may improve the basic algorithm, such as decapitalising proper nouns. There are many things we would like to try, such as validating the dependency matching process with the thesaurus matching. We would like to match larger units rather than tokens, with preferences towards the longer matching blocks. In parallel to the development of `alignheuristic`, we developed a system which measures the similarity between a node in the dependency tree of $s1$ and a node in the dependency tree of $s2$ as the sum of the lexical similarity of the lemmas at the nodes and the similarity of its children nodes. We did not submit a run for the system as it did not perform as well as `alignheuristic`, probably because the score focused on structure too much. We hope to spend time developing this system in future.

Ultimately, we envisage a system that:

- can have non 1:1 mappings between tokens, i.e. a phrase may be paraphrased as a word for example *blow up* may be paraphrased as *explode*
- can map between sequences of non-contiguous words for example the words in the phrase *blow up* may be separated but mapped to the word *explode* as in *the bomb exploded* ↔ *They blew it up*
- has categories (such as equivalence, entailment, negation, omission ...) associated with each mapping. Speculation, modality and sentiment should be indicated on any relevant chunk so that differences can be detected between candidate and referent
- scores the candidate using a function of the scores of the mapped units (as in the systems described above) but alters the score to reflect the category as well as the source of the mapping, for example entailment without equivalence should reduce the similarity score, in contrast to equivalence, and negation should reduce this still further

Crucially we would welcome a task where annotators would also provide a score on sub chunks of the sentences (or arbitrary blocks of text) that align along with a category for the mapping (equivalence, entailment, negation etc..). This would allow us to look under the hood at the text similarity task and determine the reason behind the similarity judgments.

7 Acknowledgements

We thank the task organisers for their efforts in organising the task and their readiness to take on board discussions on this as a pilot. We also thank the SemEval-2012 co-ordinators.

References

Agirre, E., Cer, D., Diab, M., and Gonzalez-Agirre, A. (2012). Semeval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval 2012), in conjunction with the First Joint Conference on Lexical and Computational Semantics (*SEM 2012)*.

Barzilay, R. and Elhadad, N. (2003). Sentence alignment for monolingual comparable corpora. In Collins, M. and Steedman, M., editors, *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 25–32.

Berry, M. (1992). Large scale singular value computations. *International Journal of Supercomputer Applications*, 6(1):13–49.

Dagan, I., Glickman, O., and Magnini, B. (2005). The pascal recognising textual entailment challenge. In *Proceedings of the PASCAL First Challenge Workshop*, pages 1–8, Southampton, UK.

de Marneffe, M.-C., MacCartney, B., and Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. In *To appear at LREC-06*.

Ferraresi, A., Zanchetta, E., Baroni, M., and Bernardini, S. (2008). Introducing and evaluating ukwac, a very large web-derived corpus of english. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco.

Hirst, G. (2003). Paraphrasing paraphrased. Invited talk at the Second International Workshop on Paraphrasing, 41st Annual Meeting of the Association for Computational Linguistics.

Kilgarriff, A., Rychlý, P., Smrz, P., and Tugwell, D. (2004). The sketch engine. In *Proceedings of Euralex*, pages 105–116, Lorient, France. Reprinted in Patrick Hanks (ed.). 2007. Lexicology: Critical concepts in Linguistics. London: Routledge.

McCarthy, D. and Navigli, R. (2009). The English lexical substitution task. *Language Resources and Evaluation Special Issue on Computational Semantic Analysis of Language: SemEval-2007 and Beyond*, 43(2):139–159.

Mihalcea, R., Corley, C., and Strapparava, C. (2006). Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of the American Association for Artificial Intelligence (AAAI 2006)*, Boston, MA.

Mitchell, J. and Lapata, M. (2008). Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT*, pages 236–244, Columbus, Ohio. Association for Computational Linguistics.

Mohler, M., Bunescu, R., and Mihalcea, R. (2011). Learning to grade short answer questions using semantic similarity measures and dependency graph alignments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 752–762, Portland, Oregon, USA. Association for Computational Linguistics.

Mohler, M. and Mihalcea, R. (2009). Text-to-text semantic similarity for automatic short answer grading. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 567–575, Athens, Greece. Association for Computational Linguistics.

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., and Marsi, E. (2007). Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Patwardhan, S., Banerjee, S., and Pedersen, T. (2003). Using measures of semantic relatedness for word sense disambiguation. In *Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2003)*, Mexico City.

Rychlý, P. (2008). A lexicographer-friendly association score. In *Proceedings of 2nd Workshop on Recent Advances in Slavonic Natural Languages Processing, RASLAN 2008*, Brno.

Schmid, H. (1994). Probabilistic Part-of-Speech Tagging Using Decision Trees. In *Proceedings of the International Conference on New Methods in Language Processing*, pages 44–49, Manchester, UK.

Turney, P. D. (2002). Mining the web for synonyms: Pmi-ir versus lsa on toefl. *CoRR*, cs.LG/0212033.