DLSITE-2: Semantic Similarity Based on Syntactic Dependency Trees Applied to Textual Entailment

Daniel Micol, Óscar Ferrández, Rafael Muñoz, and Manuel Palomar
Natural Language Processing and Information Systems Group
Department of Computing Languages and Systems
University of Alicante
San Vicente del Raspeig, Alicante 03690, Spain
{dmicol, ofe, rafael, mpalomar}@dlsi.ua.es

Abstract

In this paper we attempt to deduce textual entailment based on syntactic dependency trees of a given text-hypothesis pair. The goals of this project are to provide an accurate and fast system, which we have called *DLSITE-2*, that can be applied in software systems that require a near-real-time interaction with the user. To accomplish this we use *MINIPAR* to parse the phrases and construct their corresponding trees. Later on we apply syntactic-based techniques to calculate the semantic similarity between text and hypothesis. To measure our method's precision we used the test text corpus set from *Second PASCAL Recognising Textual Entailment Challenge* (RTE-2), obtaining an accuracy rate of 60.75%.

1 Introduction

There are several methods used to determine textual entailment for a given text-hypothesis pair. The one described in this paper uses the information contained in the syntactic dependency trees of such phrases to deduce whether there is entailment or not. In addition, semantic knowledge extracted from WordNet (Miller et al., 1990) has been added to achieve higher accuracy rates.

It has been proven in several competitions and other workshops that textual entailment is a complex task. One of these competitions is *PASCAL Recognising Textual Entailment Challenge* (Bar-Haim et al., 2006), where each participating group develops a textual entailment recognizing system attempting to accomplish the best accuracy rate of all competitors. Such complexity is the reason why we use a combination of various techniques to deduce whether entailment is produced.

Currently there are few research projects related to the topic discussed in this paper. Some systems use syntactic tree matching as the textual entailment decision core module, such as (Katrenko and Adriaans, 2006). It is based on maximal embedded syntactic subtrees to analyze the semantic relation between text and hypothesis. Other systems use syntactic trees as a collaborative module, not being the core, such as (Herrera et al., 2006). The application discussed in this paper belongs to the first set of systems, since syntactic matching is its main module.

The remainder of this paper is structured as follows. In the second section we will describe the methods implemented in our system. The third one contains the experimental results, and the fourth and last discusses such results and proposes future work based on our actual research.

2 Methods

The system we have built aims to provide a good accuracy rate in a short lapse of time, making it feasible to be included in applications that require near-real-time responses due to their interaction with the user. Such a system is composed of few modules that behave collaboratively. These include tree construction, filtering, embedded subtree search and graph node matching. A schematic representation of the system architecture is shown in Figure 1.

TextGraphs-2: Graph-Based Algorithms for Natural Language Processing, pages 73–80,
Rochester, April 2007

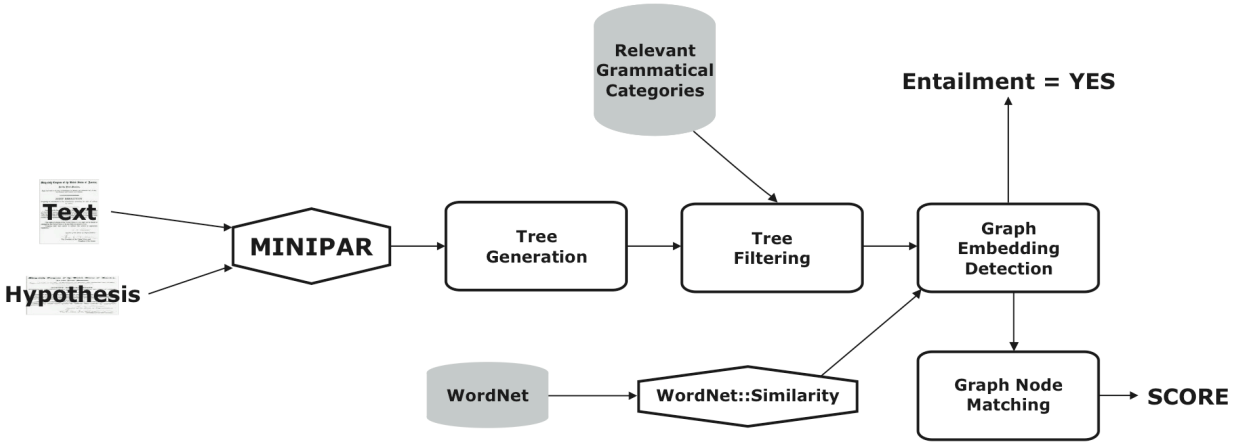


Figure 1: *DLSITE-2* system architecture.

Each of the steps or modules of *DLSITE-2* is described in the following subsections, that are numbered sequentially according to their execution order.

2.1 Tree generation

The first module constructs the corresponding syntactic dependency trees. For this purpose, *MINIPAR* (Lin, 1998) output is generated and afterwards parsed for each text and hypothesis of our corpus. Phrase tokens, along with their grammatical information, are stored in an on-memory data structure that represents a tree, which is equivalent to the mentioned syntactic dependency tree.

2.2 Tree filtering

Once the tree has been constructed, we may want to discard irrelevant data in order to reduce our system's response time and noise. For this purpose we have generated a database of relevant grammatical categories, represented in Table 1, that will allow us to remove from the tree all those tokens whose category does not belong to such list. The resulting tree will have the same structure as the original, but will not contain any stop words nor irrelevant tokens, such as determinants or auxiliary verbs. The whole list of ignored grammatical categories is represented in Table 2.

We have performed tests taking into account and discarding each grammatical category, which has allowed us to generate both lists of relevant and ignored grammatical categories.

Verbs, verbs with one argument, verbs with two arguments, verbs taking clause as complement, verb *Have*, verb *Be*
Nouns
Numbers
Adjectives
Adverbs
Noun-noun modifiers

Table 1: Relevant grammatical categories.

2.3 Graph embedding detection

The next step of our system consists in determining whether the hypothesis' tree is embedded into the text's. Let us first define the concept of embedded tree (Katrenko and Adriaans, 2006).

> **Definition 1: Embedded tree** A tree $T_1 = (V_1, E_1)$ is embedded into another one $T_2 = (V_2, E_2)$ iff
>
> 1. $V_1 \subseteq V_2$, and
> 2. $E_1 \subseteq E_2$
>
> where V_1 and V_2 represent the vertices, and E_1 and E_2 the edges.

In other words, a tree, T_1, is embedded into another one, T_2, if all nodes and branches of T_1 are present in T_2.

We believe that it makes sense to reduce the strictness of such a definition to allow the appearance of intermediate nodes in the text's branches that are

Determiners
Pre-determiners
Post-determiners
Clauses
Inflectional phrases
Preposition and preposition phrases
Specifiers of preposition phrases
Auxiliary verbs
Complementizers

Table 2: Ignored grammatical categories.

not present in the corresponding hypothesis' branch, which means that we allow partial matching. Therefore, a match between two branches will be produced if all nodes of the first one, namely $\theta_1 \in E_1$, are present in the second, namely $\theta_2 \in E_2$, and their respective order is the same, allowing the possibility of appearance of intermediate nodes that are not present in both branches. This is also described in (Katrenko and Adriaans, 2006).

To determine whether the hypothesis' tree is embedded into the text's, we perform a top-down matching process. For this purpose we first compare the roots of both trees. If they coincide, we then proceed to compare their respective child nodes, which are the tokens that have some sort of dependency with their respective root token.

In order to add more flexibility to our system, we do not require the pair of tokens to be exactly the same, but rather set a threshold that represents the minimum similarity value between them. This is a difference between our approach and the one described in (Katrenko and Adriaans, 2006). Such a similarity is calculated by using the *WordNet::Similarity* tool (Pedersen et al., 2004), and, concretely, the Wu-Palmer measure, as defined in Equation 1 (Wu and Palmer, 1994).

$$Sim(C_1, C_2) = \frac{2N_3}{N_1 + N_2 + 2N_3} \qquad (1)$$

where C_1 and C_2 are the synsets whose similarity we want to calculate, C_3 is their least common superconcept, N_1 is the number of nodes on the path from C_1 to C_3, N_2 is the number of nodes on the path from C_2 to C_3, and N_3 is the number of nodes on the path from C_3 to the root. All these synsets and distances can be observed in Figure 2.

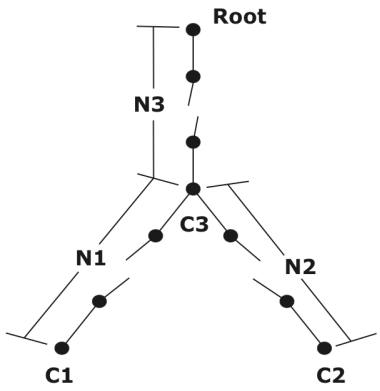


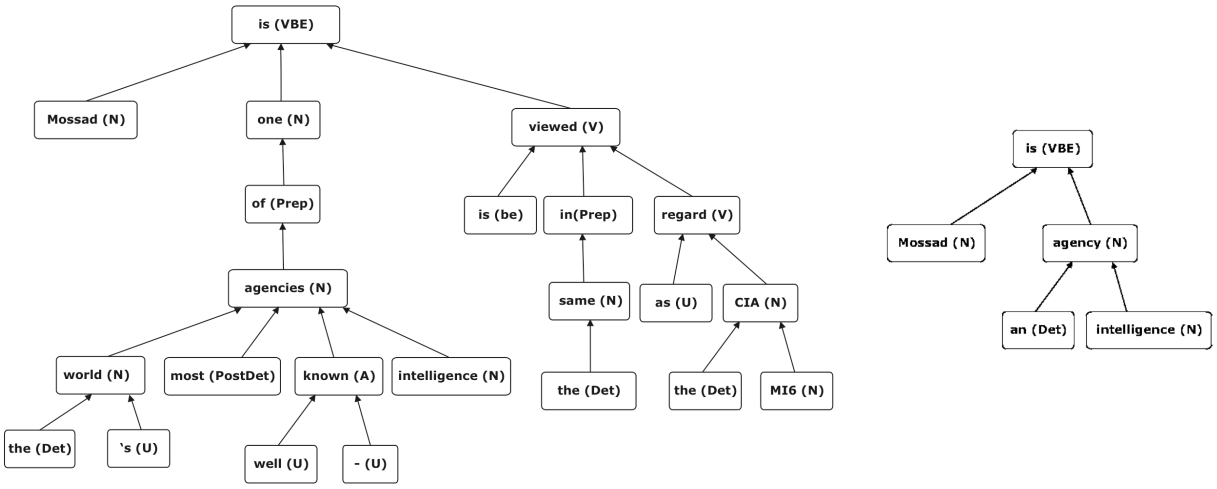
Figure 2: Distance between two synsets.

If the similarity rate is greater or equal than the established threshold, which we have set empirically to 80%, we will consider the corresponding hypothesis' token as suitable to have the same meaning as the text's token, and will proceed to compare its child nodes in the hypothesis' tree. On the other hand, if such similarity value is less than the corresponding threshold, we will proceed to compare the children of such text's tree node with the actual hypothesis' node that was being analyzed.

The comparison between the syntactic dependency trees of both text and hypothesis will be completed when all nodes of either tree have been processed. If we have been able to find a match for all the tokens within the hypothesis, the corresponding tree will be embedded into the text's and we will believe that there is entailment. If not, we will not be able to assure that such an implication is produced and will proceed to execute the next module of our system.

Next, we will present a text-hypothesis pair sample where the syntactic dependency tree of the hypothesis (Figure 3(b)) is embedded into the text's (Figure 3(a)). The mentioned text-hypothesis pair is the following:

> **Text:** *Mossad is one of the world's most well-known intelligence agencies, and is often viewed in the same regard as the CIA and MI6.*
> **Hypothesis:** *Mossad is an intelligence agency.*



(a) *Mossad is one of the world's most well-known intelligence agencies, and is often viewed in the same regard as the CIA and MI6.*

(b) *Mossad is an intelligence agency.*

Figure 3: *Representation of a hypothesis' syntactic dependency tree that is embedded into the text's.*

As one can see in Figure 3, the hypothesis' syntactic dependency tree represented is embedded into the text's because all of its nodes are present in the text in the same order. There is one exception though, that is the word *an*. However, since it is a determinant, the filtering module will have deleted it before the graph embedding test is performed. Therefore, in this example the entailment would be recognized.

2.4 Graph node matching

Once the embedded subtree comparison has finished, and if its result is negative, we proceed to perform a graph node matching process, termed alignment, between both the text and the hypothesis. This operation consists in finding pairs of tokens in both trees whose lemmas are identical, no matter whether they are in the same position within the tree. We would like to point out that in this step we do not use the *WordNet::Similarity* tool.

Some authors have already designed similar matching techniques, such as the ones described in (MacCartney et al., 2006) and (Snow et al., 2006). However, these include semantic constraints that we have decided not to consider. The reason of this decision is that we desired to overcome the textual entailment recognition from an exclusively syntactic perspective. Therefore, we did not want this module to include any kind of semantic knowledge.

The weight given to a token that has been found in both trees will depend on the depth in the hypothesis' tree and the token's grammatical relevance. The first of these factors depends on an empirically-calculated weight that assigns less importance to a node the deeper it is located in the tree. This weight is defined in Equation 2. The second factor gives different relevance depending on the grammatical category and relationship. For instance, a verb will have the highest weight, while an adverb or an adjective will have less relevance. The values assigned to each grammatical category and relationship are also empirically-calculated and are shown in Tables 3 and 4, respectively.

Grammatical category	**Weight**
Verbs, verbs with one argument, verbs with two arguments, verbs taking clause as complement	1.0
Nouns, numbers	0.75
Be used as a linking verb	0.7
Adjectives, adverbs, noun-noun modifiers	0.5
Verbs *Have* and *Be*	0.3

Table 3: Weights assigned to the grammatical categories.

Grammatical relationship	Weight
Subject of verbs, surface subject, object of verbs, second object of ditransitive verbs	1.0
The rest	0.5

Table 4: Weights assigned to the grammatical relationships.

Let τ and λ represent the text's and hypothesis' syntactic dependency trees, respectively. We assume we have found members of a synset, namely β, present in both τ and λ. Now let γ be the weight assigned to β's grammatical category (defined in Table 3), σ the weight of β's grammatical relationship (defined in Table 4), μ an empirically-calculated value that represents the weight difference between tree levels, and δ_β the depth of the node that contains the synset β in λ. We define the function $\phi(\beta)$ as represented in Equation 2.

$$\phi(\beta) = \gamma \cdot \sigma \cdot \mu^{-\delta_\beta} \tag{2}$$

The value obtained by calculating the expression of Equation 2 would represent the relevance of a synset in our system. The experiments performed reveal that the optimal value for μ is 1.1.

For a given pair (τ, λ), we define the set ξ as the one that contains the synsets present in both trees:

$$\xi = \tau \cap \lambda \;\; \forall \alpha \in \tau, \beta \in \lambda \tag{3}$$

Therefore, the similarity rate between τ and λ, denoted by the symbol ψ, would be defined as:

$$\psi(\tau, \lambda) = \sum_{\nu \in \xi} \phi(\nu) \tag{4}$$

One should note that a requirement of our system's similarity measure would be to be independent of the hypothesis length. Thus, we must define the normalized similarity rate, as shown in Equation 5.

$$\overline{\psi(\tau, \lambda)} = \frac{\psi(\tau, \lambda)}{\sum_{\beta \in \lambda} \phi(\beta)} = \frac{\sum_{\nu \in \xi} \phi(\nu)}{\sum_{\beta \in \lambda} \phi(\beta)} \tag{5}$$

Once the similarity value, $\overline{\psi(\tau, \lambda)}$, has been calculated, it will be provided to the user together with the corresponding text-hypothesis pair identifier. It will be his responsibility to choose an appropriate threshold that will represent the minimum similarity rate to be considered as entailment between text and hypothesis. All values that are under such a threshold will be marked as not entailed. For this purpose, we suggest using a development corpus in order to obtain the optimal threshold value, as it is done in the RTE challenges.

3 Experimental results

The experimental results shown in this paper were obtained processing a set of text-hypothesis pairs from RTE-2. The organizers of this challenge provide development and test corpora to the participants, both of them containing 800 pairs manually annotated for logical entailment. It is composed of four subsets, each of them corresponding to typical true and false entailments in different tasks, such as Information Extraction (IE), Information Retrieval (IR), Question Answering (QA), and Multi-document Summarization (SUM). For each task, the annotators selected the same amount of true entailments as negative ones (50%-50% split).

The organizers have also defined two measures to evaluate the participating systems. All judgments returned by the systems will be compared to those manually assigned by the human annotators. The percentage of matching judgments will provide the *accuracy* of the system, i.e. the percentage of correct responses. As a second measure, the *average precision* will be computed. This measure evaluates the ability of the systems to rank all the pairs in the corpus according to their entailment confidence, in decreasing order from the most certain entailment to the least. *Average precision* is a common evaluation measure for system rankings that is defined as shown in Equation 6.

$$AP = \frac{1}{R} \sum_{i=1}^{n} E(i) \frac{\#correct\_up\_to\_pair\_i}{i} \tag{6}$$

where n is the amount of the pairs in the test corpus, R is the total number of positive pairs in it, i ranges over the pairs, ordered by their ranking, and $E(i)$ is defined as follows:

$$E(i) = \begin{cases} 1 & \text{if the } i-th \text{ pair is positive,} \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$

As we previously mentioned, we tested our system against RTE-2 development corpus, and used the test one to evaluate it.

First, Table 5 shows the accuracy (ACC) and average precision (AP), both as a percentage, obtained processing the development corpus from RTE-2 for a threshold value of 68.9%, which corresponds to the highest accuracy that can be obtained using our system for the mentioned corpus. It also provides the rate of correctly predicted true and false entailments.

Task	**ACC**	**AP**	**TRUE**	**FALSE**
IE	52.00	51.49	54.00	50.00
IR	55.50	58.99	32.00	79.00
QA	57.50	54.72	53.00	62.00
SUM	65.00	81.35	39.00	91.00
Overall	**57.50**	**58.96**	**44.50**	**70.50**

Table 5: Results obtained for the development corpus.

Next, let us show in Table 6 the results obtained processing the test corpus, which is the one used to compare the different systems that participated in RTE-2, with the same threshold as before.

Task	**ACC**	**AP**	**TRUE**	**FALSE**
IE	50.50	47.33	75.00	26.00
IR	64.50	67.67	59.00	70.00
QA	59.50	58.16	80.00	39.00
SUM	68.50	75.86	49.00	88.00
Overall	**60.75**	**57.91**	**65.75**	**55.75**

Table 6: Results obtained for the test corpus.

As one can observe in the previous table, our system provides a high accuracy rate by using mainly syntactical measures. The number of text-hypothesis pairs that succeeded the graph embedding evaluation was three for the development corpus and one for the test set, which reflects the strictness of such module. However, we would like to point out that the amount of pairs affected by the mentioned module will depend on the corpus nature, so it can vary significantly between different corpora.

Let us now compare our results with the ones that were achieved by the systems that participated in RTE-2. One should note that the criteria for such ranking is based exclusively on the accuracy, ignoring the average precision value. In addition, each participating group was allowed to submit two different systems to RTE-2. We will consider here the best result of both systems for each group. The mentioned comparison is shown in Table 7, and contains only the systems that had higher accuracy rates than our approach.

Participant	**Accuracy**
(Hickl et al., 2006)	75.38
(Tatu et al., 2006)	73.75
(Zanzotto et al., 2006)	63.88
(Adams, 2006)	62.62
(Bos and Markert, 2006)	61.62
DLSITE-2	60.75

Table 7: Comparison of some of the teams that participated in RTE-2.

As it is reflected in Table 7, our system would have obtained the sixth position out of twenty-four participants, which is an accomplishment considering the limited number of resources that it has built-in.

Since one of our system's modules is based on (Katrenko and Adriaans, 2006), we will compare their results with ours to analyze whether the modifications we introduced perform correctly. In RTE-2, they obtained an accuracy rate of 59.00% for the test corpus. The reason why we believe we have achieved better results than their system is due to the fact that we added semantic knowledge to our graph embedding module. In addition, the syntactic dependency trees to which we have applied such a module have been previously filtered to ensure that they do not contain irrelevant words. This reduces the system's noise and allows us to achieve higher accuracy rates.

In the introduction of this paper we mentioned that one of the goals of our system was to provide

a high accuracy rate in a short lapse of time. This is one of the reasons why we chose to construct a light system where one of the aspects to minimize was its response time. Table 8 shows the execution times[1] of our system for both development and test text corpora from RTE-2. These include total and average[2] response times.

	Development	Test
Total	1045	1023
Average	1.30625	1.27875

Table 8: *DLSITE-2* response times (in seconds).

As we can see, accurate results can be obtained using syntactic dependency trees in a short lapse of time. However, there are some limitations that our system does not avoid. For instance, the tree embedding test is not applicable when there is no verb entailment. This is reflected in the following pair:

> **Text:** *Tony Blair, the British Prime Minister, met Jacques Chirac in London.*
> **Hypothesis:** *Tony Blair is the British Prime Minister.*

The root node of the hypothesis' tree would be the one corresponding to the verb *is*. Since the entailment here is implicit, there is no need for such a verb to appear in the text. However, this is not compatible with our system, since *is* would not match any node of the text's tree, and thus the hypothesis' tree would not be found embedded into the text's.

The graph matching process would not behave correctly either. This is due to the fact that the main verb, which has the maximum weight because it is the root of the hypothesis' tree and its grammatical category has the maximum relevance, is not present in the text, so the overall similarity score would have a considerable handicap.

The example of limitation of our system that we have presented is an apposition. To avoid this specific kind of situations that produce an undesired behavior in our system, we could add a preprocessing module that transforms the phrases that have the structure X, Y, Z into X *is* Y, *and* Z. For the shown example, the resulting text and hypothesis would be as follows:

> **Text:** *Tony Blair is the British Prime Minister, and met Jacques Chirac in London.*
> **Hypothesis:** *Tony Blair is the British Prime Minister.*

The transformed text would still be syntactically correct, and the entailment would be detected since the hypothesis' syntactic dependency tree is embedded into the text's.

4 Conclusions and future work

The experimental results obtained from this research demonstrate that it is possible to apply a syntactic-based approach to deduce textual entailment from a text-hypothesis pair. We can obtain good accuracy rates using the discussed techniques with very short response times, which is very useful for assisting different kinds of tasks that demand near-real-time responses to user interaction.

The baseline we set for our system was to achieve better results than the ones we obtained with our last participation in RTE-2. As it is stated in (Ferrández et al., 2006), the maximum accuracy value obtained by then was 55.63% for the test corpus. Therefore, our system is 9.20% more accurate compared to the one that participated in RTE-2, which represents a considerable improvement.

The authors of this paper believe that if higher accuracy rates are desired, a step-based system must be constructed. This would have several preprocessing units, such as negation detectors, multi-word associators and so on. The addition of these units would definitely increase the response time preventing the system from being used in real-time tasks.

Future work can be related to the cases where no verb entailment is produced. For this purpose we propose to extract a higher amount of semantic information that would allow us to construct a characterized representation based on the input text, so that we can deduce entailment even if there is no apparent structure similarity between text and hypothesis. This would mean to create an abstract conceptualization of the information contained in the analyzed phrases, allowing us to deduce ideas that are not

[1] The machine we used to measure the response times had an Intel Core 2 Duo processor at 2GHz.

[2] Average response times are calculated diving the totals by the number of pairs in the corpus.

explicitly mentioned in the parsed text-hypothesis pairs.

In addition, the weights and thresholds defined in our system have been established empirically. It would be interesting to calculate those values by means of a machine learning algorithm and compare them to the ones we have obtained empirically. Some authors have already performed this comparison, being one example the work described in (MacCartney et al., 2006).

Acknowledgments

The authors of this paper would like to thank professors Borja Navarro and Rafael M. Terol for their help and critical comments.

This research has been supported by the undergraduate research fellowships financed by the Spanish Ministry of Education and Science, the project TIN2006-15265-C06-01 financed by such ministry, and the project ACOM06/90 financed by the Spanish Generalitat Valenciana.

References

Rod Adams. 2006. *Textual Entailment Through Extended Lexical Overlap*. In Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venice, Italy.

Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. *The Second PASCAL Recognising Textual Entailment Challenge*. In Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venice, Italy.

Johan Bos, and Katja Markert. 2006. *When logical inference helps determining textual entailment (and when it doesnt)*. In Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venice, Italy.

Óscar Ferrández, Rafael M. Terol, Rafael Muñoz, Patricio Martínez-Barco, and Manuel Palomar. 2006. *An approach based on Logic Forms and WordNet relationships to Textual Entailment performance*. In Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venice, Italy.

Jesús Herrera, Anselmo Peñas, Álvaro Rodrigo, and Felisa Verdejo. 2006. *UNED at PASCAL RTE-2 Challenge*. In Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venice, Italy.

Andrew Hickl, John Williams, Jeremy Bensley, Kirk Roberts, Bryan Rink, and Ying Shi. 2006. *Recognizing Textual Entailment with LCC's GROUNDHOG System*. In Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venice, Italy.

Sophia Katrenko, and Pieter Adriaans. 2006. *Using Maximal Embedded Syntactic Subtrees for Textual Entailment Recognition*. In Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venice, Italy.

Dekang Lin. 1998. *Dependency-based Evaluation of MINIPAR*. In Workshop on the Evaluation of Parsing Systems, Granada, Spain.

Bill MacCartney, Trond Grenager, Marie-Catherine de Marneffe, Daniel Cer, and Christopher D. Manning. 2006. *Learning to recognize features of valid textual entailments*. In Proceedings of the North American Association of Computational Linguistics (NAACL-06), New York City, New York, United States of America.

George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J. Miller. 1990. *Introduction to WordNet: An On-line Lexical Database*. International Journal of Lexicography 1990 3(4):235-244.

Ted Pedersen, Siddhart Patwardhan, and Jason Michelizzi. 2004. *WordNet::Similarity - Measuring the Relatedness of Concepts*. In Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL-04), Boston, Massachussets, United States of America.

Rion Snow, Lucy Vanderwende, and Arul Menezes. 2006. *Effectively using syntax for recognizing false entailment*. In Proceedings of the North American Association of Computational Linguistics (NAACL-06), New York City, New York, United States of America.

Marta Tatu, Brandon Iles, John Slavick, Adrian Novischi, and Dan Moldovan. 2006. *COGEX at the Second Recognizing Textual Entailment Challenge*. In Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venice, Italy.

Zhibiao Wu, and Martha Palmer. 1994. *Verb Semantics and Lexical Selection*. In Proceedings of the 32nd Annual Meeting of the Associations for Computational Linguistics, pages 133-138, Las Cruces, New Mexico, United States of America.

Fabio M. Zanzotto, Alessandro Moschitti, Marco Pennacchiotti, and Maria T. Pazienza. 2006. *Learning textual entailment from examples*. In Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment, Venice, Italy.