Semantic Inference at the Lexical-Syntactic Level

Roy Bar-Haim and **Ido Dagan**
Computer Science Department
Bar-Ilan University
Ramat-Gan 52900, Israel
{barhair, dagan}@cs.biu.ac.il

Iddo Greental
Linguistics Department
Tel Aviv University
Ramat Aviv 69978, Israel
greenta@post.tau.ac.il

Eyal Shnarch
Computer Science Department
Bar-Ilan University
Ramat-Gan 52900, Israel
shey@cs.biu.ac.il

Abstract

Semantic inference is an important component in many natural language understanding applications. Classical approaches to semantic inference rely on complex logical representations. However, practical applications usually adopt shallower lexical or lexical-syntactic representations, but lack a principled inference framework. We propose a generic semantic inference framework that operates directly on syntactic trees. New trees are inferred by applying entailment rules, which provide a unified representation for varying types of inferences. Rules were generated by manual and automatic methods, covering generic linguistic structures as well as specific lexical-based inferences. Initial empirical evaluation in a Relation Extraction setting supports the validity of our approach.

Introduction

According to the traditional formal semantics approach inference is conducted at the logical level. Texts are first translated into some logical form and then new propositions are inferred from interpreted texts by a logical theorem prover. However, practical text understanding systems usually employ shallower lexical and lexical-syntactic representations, sometimes augmented with partial semantic annotations like word senses, named-entity classes and semantic roles. This state of affairs was clearly demonstrated in the recent PASCAL Recognizing Textual Entailment (RTE) Challenges (Dagan, Glickman, & Magnini 2006; Bar-Haim *et al.* 2006), a popular framework for evaluating application-independent semantic inference, where only a few systems applied logical inference (Raina, Ng, & Manning 2005; Tatu & Moldovan 2006; Bos & Markert 2006). While practical semantic inference is mostly performed over linguistic rather than logical representations, such practices are typically partial and quite ad-hoc, and lack a clear formalism that specifies how inference knowledge should be represented and applied. The current paper proposes a step towards filling this gap, by defining a principled semantic inference mechanism over parse-based representations.

Within the textual entailment setting a system is required to recognize whether a hypothesized statement h can be inferred from an asserted text t. Overall, the task consists of two different types of inference. Some inferences can be based on available knowledge, such as information about synonyms, paraphrases, world knowledge relationships etc. In the general case, however, some knowledge gaps arise and it is not possible to derive a complete "proof" based on available inference knowledge. Such situations are typically handled through approximate matching methods.

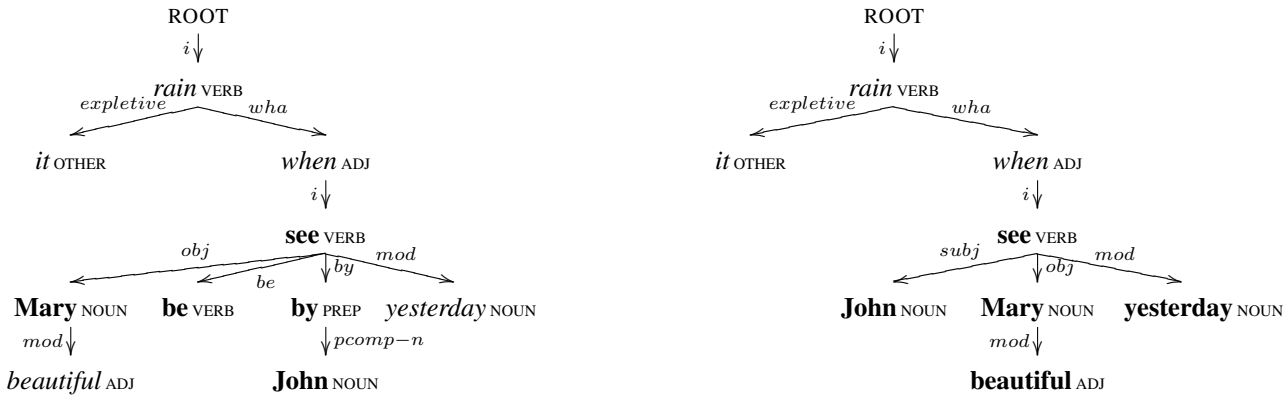
This paper focuses on the first type of knowledge-based inference. We define a proof system that operates over syntactic parse trees. New trees are derived using *entailment rules*, which provide a principled and uniform mechanism for incorporating a wide variety of critical inference knowledge. Notably, this approach allows easy incorporation of rules learned by unsupervised methods, which seems essential for scaling inference systems. Interpretation into stipulated semantic representations, which is often difficult and is inherently a supervised semantic task for learning, is circumvented altogether. Our overall research goal is to explore how far we can get with such an inference approach, and identify the scope in which semantic interpretation may not be needed.

The remainder of the paper presents our inference framework, the incorporated entailment rules, which address both generic linguistic structures and lexical-based inferences, an initial evaluation that supports the proposed approach, and some comparison to related work.

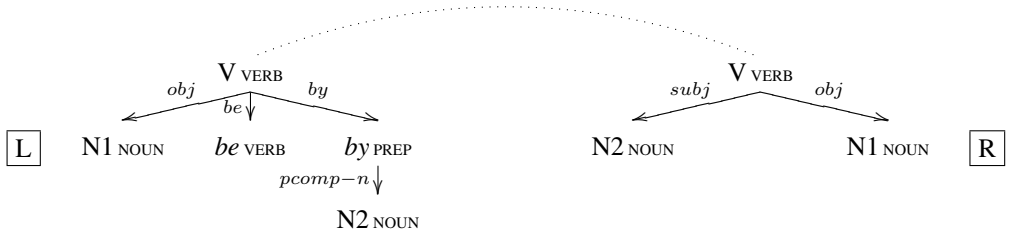
Inference Framework

Given two syntactically parsed text fragments, termed *text* (t) and *hypothesis* (h), the goal of the inference system (or *prover*) is to determine whether t entails h. The prover tries to generate h from t by applying *entailment rules* that aim to transform t into h, through a sequence of intermediate parse trees. If such a proof is found, the prover concludes that entailment holds.

Like logic-based systems, our inference framework is composed of *propositions* and *inference rules*. The propositions include t (the assumption), h (the goal), and intermediate premises inferred during the proof. The inference (entailment) rules define how new propositions are derived from previously established ones.



(a) Passive-to-active tree transformation



(b) Passive to active substitution rule. The dotted arc represents alignment.

Figure 1: Application of an inference rule. POS and relation labels are based on Minipar (Lin 1998)

Propositions

The general inference framework assumes that propositions are represented by some form of parse trees. In this paper we focus on dependency tree representation, which is often preferred to capture directly predicate-argument relations (Figure 1(a)). Nodes represent words and hold a set of features and their values. These features include the word lemma and part-of-speech, and additional features that may be added during the proof process. Edges are annotated with dependency relations.

Entailment Rules

At each step of the proof an entailment rule generates a *derived* tree d from a *source* tree s. A rule '$L \rightarrow R$' is primarily composed of two templates, termed *left-hand-side* (L), and *right-hand-side* (R). *Templates* are dependency subtrees which may contain *variables*. Figure 1(b) shows an entailment rule, where V, $N1$ and $N2$ are common variables shared by L and R. L specifies the subtree of s to be modified, and R specifies the new generated subtree. Rule application consists of the following steps:

L matching The prover first tries to match L in s. L is *matched* in s if there exists a one-to-one node mapping function f from L to s, such that: (i) For each node u in L, $f(u)$ has the same features and feature values as u. Variables match any lemma value in $f(u)$. (ii) For each edge $u \rightarrow v$ in L, there is an edge $f(u) \rightarrow f(v)$ in s, with the same dependency relation. If matching fails, the rule is not applicable to s. Otherwise, successful matching induces *variable binding* $b(X)$, for each variable X in L, defined as the full subtree rooted in $f(X)$ if X is a leaf, and $f(X)$ alone otherwise. We denote by l the subtree in s to which L was mapped (as illustrated in bold in the left part of Figure 1(a)).

R instantiation An instantiation of R, which we denote r, is generated in two steps: (i) creating a copy of R; (ii) replacing each variable X with a copy of its binding $b(X)$ (as set during L matching). In our example this results in the subtree *John saw beautiful Mary*.

Alignment copying Part of the rule definition is an *alignment* relation between pairs of nodes in L and R that specifies which modifiers in l that are not part of the rule structure need to be copied to the generated r. Formally, for any two nodes u in l and v in r whose matching nodes in L and R are aligned, we copy the daughter subtrees of u in s, which are not already part of l, to become daughter subtrees of v in r. The bold nodes in the right part of Figure 1(b) correspond to r after alignment copying. *yesterday* was copied to r due to the alignment of its parent verb node.

Derived tree generation by rule type Our formalism has two methods for generating the derived tree: *substitution* and *introduction*, as specified by the rule type. With *substitution* rules, the derived tree d is obtained by making a local modification to the source tree s. Except for this modification s and d are identical (a typical example is a lexical rule, such as *buy* $\rightarrow$ *purchase*). For this type, d is formed by copying s while replacing l (and the descendants of l's

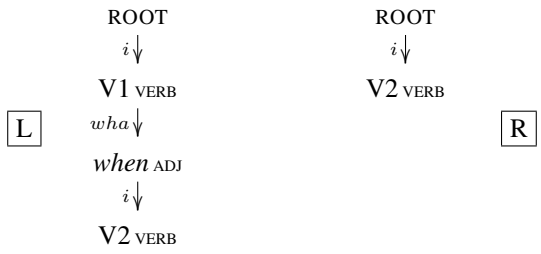


Figure 2: Temporal clausal modifier extraction (introduction rule)

nodes) with r. This is the case for the passive rule. The right part of Figure 1(a) shows the derived tree for the passive rule application. By contrast, *introduction* rules are used to make inferences from a subtree of s, while the other parts of s are ignored and do not effect d. A typical example is inference of a proposition embedded as a relative clause in s. In this case the derived tree d is simply taken to be r. Figure 2 presents such a rule which enables to derive propositions that are embedded within temporal modifiers. Note that the derived tree does not depend on the main clause. Applying this rule to the right part of Figure 1(b) yields the proposition *John saw beautiful Mary yesterday*.

Annotation Rules

Annotation rules add features to parse tree nodes, and are used in our system to annotate negation and modality. Annotation rules do not have an R, but rather each node of L may contain annotation features. If L is matched in a tree then the annotations are copied to the matched nodes. Annotation rules are applied to the original text *t*, and to each inferred premise, prior to any entailment rule application. Since the annotated features would be checked during subsequent L matching, these additional features may block inappropriate subsequent rule applications, such as for negated predicates.

Template Hypotheses

For many applications it is useful to allow the hypothesis h to be a template rather than a proposition, that is, to contain variables. The variables in this case are existentially quantified: t entails h if there exists a proposition h', obtained from h by variable instantiation, so that t entails h'. The obtained variable instantiations may stand for sought answers in questions or slots to be filled in relation extraction. For example, applying this framework in a question-answering setting, the question *Who killed Kennedy?* may be translated into the hypothesis *X killed Kennedy*. A successful proof of h from the sentence *"The assassination of Kennedy by Oswald shook the nation"* would instantiate *X* with *Oswald*.

Rules for Generic Linguistic Structures

Based on the above framework we have manually created a rule base for generic linguistic phenomena. The current rule base was developed under the assumption that the hypothesis h has a relatively simple structure and is positive (non-negated) and non-modal, which is often the case in applications such as question answering and information extraction. Accordingly, the rules aim to simplify and decompose the source proposition, and to block inference from negated and modal predicates.

Syntactic-Based Rules

These rules capture entailment inferences associated with common syntactic structures. The rules have three major functions: (1) simplification and canonization of the source tree (categories 6 and 7 in Table 1); (2) extracting embedded propositions (categories 1, 2, 3); (3) inferring propositions from non-propositional subtrees of the source tree (category 4).

Polarity-Based Rules

Consider the following two examples:

John *knows* that Mary is here $\Rightarrow$ Mary is here.
John *believes* that Mary is here $\not\Rightarrow$ Mary is here.

Valid inference of propositions embedded as verb complements depends on the verb properties, and the polarity of the context in which the verb appears (positive, negative, or unknown) (Nairn, Condoravdi, & Karttunen. 2006). We extracted from the polarity lexicon of Nairn et al. a list of verbs for which inference is allowed in positive polarity context, and generated entailment rules for these verbs (category 8 in Table 1). The list was complemented with a few reporting verbs, such as *say* and *announce*, since information in the news domain is often given in reported speech, while the speaker is usually considered reliable.

Negation and Modality Annotation Rules

We use annotation rules to mark negation and modality of predicates (mainly verbs), based on their descendent modifiers. Since annotation rules may capture subtrees of any size, we can use them to identify negation and modality phenomena in complex subtrees where the source of the phenomenon is not in the immediate daughter node of the predicate. Negation rules identify full and contracted verbal negation, as well as negation implied by certain determiners and nouns. Modality rules identify modality expressed by the use of modal verbs such as *should*, as well as conditional sentences and modal adverbials. Category 9 in Table 1 illustrates a negation rule, annotating the verb *seen* for negation due to the presence of *never*.

Generic Default Rules

Generic default rules are used to define default behavior, in situations where no case-by-case rules are available. We used one default rule that allows removal of any modifiers from nodes. Desirably, specific rules should be specified in future work to capture more precisely many cases that are currently handled by this default rule.

Lexical-Syntactic Rules

Lexical-Syntactic rules include open-class lexical components within varying syntactic structures. Accordingly these

#	Category	Example: source	Example: derived
1	Conjunctions	Helena's very experienced and has played a long time on the tour.	⇒ Helena has played a long time on the tour.
2	Clausal modifiers	But celebrations were muted as many Iranians observed a Shi'ite mourning month.	⇒ Many Iranians observed a Shi'ite mourning month.
3	Relative clauses	The assailants fired six bullets at the car, which carried Vladimir Skobtsov.	⇒ The car carried Vladimir Skobtsov.
4	Appositives	Frank Robinson, a one-time manager of the Indians, has the distinction for the NL.	⇒ Frank Robinson is a one-time manager of the Indians.
5	Determiners	The plaintiffs filed **their** lawsuit last year in U.S. District Court in Miami.	⇒ The plaintiffs filed **a** lawsuit last year in U.S. District Court in Miami.
6	Passive	We have been approached by the investment banker.	⇒ The investment banker approached us.
7	Genitive modifier	Malaysia's crude palm oil output is estimated to have risen by up to six percent.	⇒ The crude palm oil output of Malasia is estimated to have risen by up to six percent.
8	Polarity	Yadav was forced to resign.	⇒ Yadav resigned.
9	Negation, modality	What we've **never** seen is actual costs come down.	What we've never $\overline{\text{seen}}$ is actual costs come down. ($\not\Rightarrow$ What we've seen is actual costs come down.)

Table 1: Summary of rule base for generic linguistic structures.

rules are numerous compared to the generic rules of the previous section, and have been acquired either lexicographically or automatically (e.g. paraphrases). We incorporated several sources of such rules.

Nominalization Rules

Entailment rules such as 'X's acquisition of $Y \rightarrow X$ acquired Y' capture the relations between verbs and their nominalizations. These rules were derived automatically (Ron 2006) from Nomlex, a hand-coded database of English nominalizations (Macleod *et al.* 1998), and from WordNet.

Automatically Learned Rules

DIRT (Lin & Pantel 2001) and TEASE (Szpektor *et al.* 2004) are two state-of-the-art unsupervised algorithms that learn lexical-syntactic inference rules.[1] Some of the learned rules are linguistic paraphrases, e.g. 'X confirm $Y \rightarrow X$ approve Y', while others capture world knowledge, e.g. 'X file lawsuit against $Y \rightarrow X$ accuse Y'. These algorithms do not learn the entailment direction, which reduces their accuracy when applied in any given direction. For each system, we considered the top 15 bi-directional rules learned for each template.

Evaluation

As the current work is concerned with performing exact proofs, we should evaluate its precision over text-hypothesis pairs for which a complete proof chain is found, using the available rules. We note that the PASCAL RTE datasets are not suitable for this purpose. These rather small datasets include many pairs for which entailment recognition requires approximate matching, as currently it is not realistic to assume sufficient knowledge that will enable a complete exact proof. As an alternative we chose a Relation Extraction (RE) setting, for which complete proofs can be achieved for a large number of corpus sentences. In this setting, the system needs to identify in sentences pairs of arguments for a target semantic relation (e.g. *X buy Y*).

Evaluation Process

We use a sample of test template hypotheses that correspond to typical RE relations, such as *X approve Y*. We then identify in a large test corpus sentences from which an instantiation of the test hypothesis is proved. For example, the sentence *the budget was approved by the parliament* is found to prove the instantiated hypothesis *parliament approve budget*. Finally, a sample of such sentences-hypothesis pairs are judged manually for true entailment. The process was repeated to compare different system configurations.

We aimed to test hypotheses that are covered by all our lexical-syntactic resources. Since the publicly available output of TEASE is much smaller than the other resources, we selected from this resource 9 transitive verbs that may correspond to typical RE predicates,[2] forming test templates by adding subject and object variable nodes.

For each test template h we need to identify in the corpus sentences from which it is proved. To find efficiently proof chains that generate h from corpus sentences we combined forward and backward (Breadth-First) search over the available rules. First, backward search is used over the lexical-syntactic rules, starting with rules whose right-hand-side is identical to the test template. While backward chaining the DIRT/TEASE and nominalization rules, this process generates a set of templates t_i, all of them proving (deriving) h. For example, for the hypothesis *X approve Y* we may generate the template *X confirm Y*, through backward application of a DIRT/TEASE rule, and then further generate the template *confirmation of Y by X*, through a nominalization rule. Since the templates t_i are generated by lexical-syntactic rules, which modify open-class lexical items, they may be considered as "lexical expansions" of h.

[1] Their output is publicly available at the ACLWiki Textual Entailment Resources Pool.

[2] The verbs are *approach, approve, consult, lead, observe, play, seek, sign, strike*.

#	Configuration	Precision	Yield
1	BASELINE	67.0%	2,414
2	PROOF	78.5%	1,426
3	+GEN	74.8%	2,967
4	+GEN+LEXSYN	23.6%	18,809

Table 2: Empirical evaluation - results.

Next, for each specific t_i we generate a search engine query composed of the open-class words in t_i. This query fetches from the corpus candidate sentences, from which t_i might be proven using the generic linguistic rules (recall that these rules do not modify open-class words). To that end we apply a forward search that applies the generic rules, starting from a candidate sentence s and trying to derive t_i by a sequence of rule applications. If successful, this process instantiates the variables in t_i with the appropriate variable bindings to elements in s. Consequently, we know that, under the same variable instantiations, h can be proved from s (since s derives t_i which in turn derives h).

The above search for sentences that prove each test template was performed over the Reuters RCV1 corpus, CD#2, applying Minipar (Lin 1998) for parsing. Through random sampling we obtained 30 sentences that prove each of the 9 test templates, yielding a total of 270 pairs of a sentence and an instantiated hypothesis for each of the four tested configurations (1080 pairs overall). These pairs were split for entailment judgment between two human annotators. The annotators achieved, on a sample of 100 shared examples, agreement of 87%, and a Kappa value of 0.71 (corresponding to "substantial agreement").

Results

We tested 4 configurations of the proof system:

1. BASELINE The baseline configuration follows the prominent approach in graph-based entailment systems (see next section): the system simply tries to embed the given hypothesis anywhere in the text tree, while only modality or negation (detected by the annotation rules) may block embedding.
2. PROOF: The basic configuration of our prover. h has to be strictly generated from t, rather than embedded in t. The only inference rule available is the default rule for removing modifiers (annotation rules are active as in BASELINE).
3. +GEN: As PROOF, plus generic linguistic rules.
4. +GEN+LEXSYN: As +GEN, plus lexical-syntactic rules.

For each system configuration we measure *precision*, the percentage of examples judged as correct (entailing), and average *extrapolated yield*, which is the expected number of truly entailing sentences in the corpus that would be proved as entailing by the system.[3] We note that, similar to IR evaluations, it is not possible to compute true recall in our setting since the total number of entailing sentences in the corpus is not known (recall is equal to the yield divided by this total). However, it is straightforward to measure *relative* recall differences among different configurations based on the yield. Thus, using these two measures estimated from a large corpus it is possible to conduct robust comparison between different configurations, and reliably estimate the impact of different rule types. Such analysis is not possible with the RTE datasets, which are rather small, and their hand-picked examples do not represent the actual distribution of linguistic phenomena.

The results are reported in Table 2. First, it is observed that the requirement for exact proof rather than embedding improves the precision considerably over the baseline (by 11.5%), while reducing the yield by nearly 40%. Remarkably, using the generic inference rules, our system is able to gain back the lost yield in PROOF and further surpass the yield of the baseline configuration. In addition, a higher precision than the baseline is obtained (a 7.8% difference), which is significant at a $p < 0.05$ level, using z test for proportions. This demonstrates that our principled proof approach appears to be superior to the more heuristic baseline embedding approach, and exemplifies the contribution of our generic rule base. Overall, generic rules were used in 46% of the proofs.

Adding the lexical-syntactic rules the prover was able to increase the yield by a factor of six(!). This shows the importance of acquiring lexical-syntactic variability patterns. However, the precision of DIRT and TEASE is currently quite low, causing overall low precision. Manual filtering of rules learned by these systems is currently required in order to obtain reasonable precision .

Error analysis revealed that for the third configuration (+GEN), a significant 65% of the errors are due to parsing errors, most notably incorrect dependency relation assignment, incorrect POS assignment, incorrect argument selection, incorrect analysis of complex verbs (e.g. *play down* in the text vs. *play* in the hypothesis) and ungrammatical sentence fragments. Another 30% of the errors represent conditionals, negation and modality phenomena, most of which could be handled by additional rules, some making use of more elaborate syntactic information such as verb tense. The remaining, and rather small, 5% of the errors represent truly ambiguous sentences which would require considerable world knowledge for successful analysis.

Related Work

Most previous work on lexical-syntactic entailment focused on approximate matching. In particular, many works tried to directly match or embed the hypothesis within the text, using tree-edit distance or other cost functions to measure the "distance" between the text and hypothesis (Kouylekov & Magnini 2005; Haghighi, Ng, & Manning 2005) . Rather limited amount of inference knowledge was utilized (as observed at the RTE-2 Challenge), typically to determine cost values. These mechanisms do not provide a clear separation between approximate matching heuristics and justified inferences based on available knowledge.

[3] The extrapolated yield for a specific template is calculated as the number of sample sentences judged as entailing, multiplied by the sampling proportion. The average is calculated over all test templates.

An initial theoretical proposal for an inference system based on lexical-syntactic rules was outlined in (Dagan & Glickman 2004). The current work may be viewed as a realization of that general direction.

(de Salvo Braz *et al.* 2005) were the first to incorporate augmented syntactic-based entailment rules in a comprehensive entailment system. In their system, entailment rules are applied as one of several inference mechanisms, over hybrid syntactic-semantic structures called *concept graphs*. When the left hand side of a rule is matched in the concept graph, the graph is augmented with an instantiation of the right hand side of the rule, creating a complex structure whose semantics were not fully specified. Eventually, their system attempts to embed the hypothesis in the graph. By contrast, we presented a clearly formalized framework which is based solely on entailment rules, derives a single proposition at a time, and fully generates the hypothesis itself rather than heuristically embedding it in the text.

(Romano *et al.* 2006) investigated simple application of a small set of generic syntactic-based entailment rules together with lexical-syntactic entailment rules produced by TEASE. They tested their method on a single relation (protein interaction). While following their general approach, the current work substantially extends their preliminary work by introducing a detailed inference formalism and a much richer spectrum of entailment rules.

Finally, none of the earlier works has presented a robust component-wise evaluation of a variety of entailment rule sources, based on samples from a large corpus.

Conclusion

This paper defined a novel framework for semantic inference at the lexical-syntactic level. Our formalism was found suitable for describing a wide spectrum of entailment rules, both automatically derived and manually created. We also presented a much-needed evaluation methodology for individual components in knowledge-based inference systems. The empirical results demonstrate that our exact proof approach is feasible for real-world applications such as relation extraction, and outperforms the more heuristic common practice of hypothesis embedding. We plan to enhance our framework to allow inference from multiple sentences, as well as to incorporate additional types of rules, such as lexical rules (e.g. *dog* $\rightarrow$ *animal*). Future research will also investigate integration of the proof system with different methods for approximate matching, which would enable its application in additional settings.

Acknowledgments

This work was partially supported by ISF grant 1095/05, the IST Programme of the European Community under the PASCAL Network of Excellence IST-2002-506778 and the Israel Internet Association (ISOC-IL), grant 9022. We are grateful to Cleo Condoravdi for making the polarity lexicon developed at PARC available for this research. We also thank Dan Roth and Idan Szpektor for helpful discussions.

References

Bar-Haim, R.; Dagan, I.; Dolan, B.; Ferro, L.; Giampiccolo, D.; Magnini, B.; and Szpektor, I. 2006. The Second PASCAL Recognising Textual Entailment Challenge. In *Second PASCAL Challenges Workshop on Recognizing Textual Entailment*.

Bos, J., and Markert, K. 2006. When logical inference helps determining textual entailment (and when it doesn't). In *Second PASCAL Challenges Workshop on Recognizing Textual Entailment*.

Dagan, I., and Glickman, O. 2004. Probabilistic textual entailment: Generic applied modeling of language variability. PASCAL workshop on Text Understanding and Mining.

Dagan, I.; Glickman, O.; and Magnini, B. 2006. The PASCAL Recognising Textual Entailment Challenge. In Quiñonero-Candela et al., ed., *MLCW 2005, LNAI Volume 3944*. Springer-Verlag. 177–190.

de Salvo Braz, R.; Girju, R.; Punyakanok, V.; Roth, D.; and Sammons, M. 2005. An inference model for semantic entailment in natural language. In *AAAI*, 1043–1049.

Haghighi, A. D.; Ng, A. Y.; and Manning, C. D. 2005. Robust textual inference via graph matching. In *Proceedings of EMNLP 2005, 387–394* .

Kouylekov, M., and Magnini, B. 2005. Tree edit distance for textual entailment. In *Recent Advances in Natural Language Processing (RANLP)*.

Lin, D., and Pantel, P. 2001. Discovery of inference rules for question answering. *Natural Language Engineering* 4(7):343–360.

Lin, D. 1998. Dependency-based evaluation of minipar. In *Proceedings of the Workshop on Evaluation of Parsing Systems at LREC 1998*.

Macleod, C.; Grishman, R.; Meyers, A.; Barrett, L.; and Reeves, R. 1998. Nomlex: A lexicon of nominalizations. In *EURALEX*.

Nairn, R.; Condoravdi, C.; and Karttunen., L. 2006. Computing relative polarity for textual inference. In *Proceedings of International workshop on Inference in Computational Semantics (ICoS-5)*.

Raina, R.; Ng, A. Y.; and Manning, C. D. 2005. Robust textual inference via learning and abductive reasoning. In *AAAI*, 1099–1105.

Romano, L.; Kouylekov, M.; Szpektor, I.; Dagan, I.; and Lavelli, A. 2006. Investigating a generic paraphrase-based approach for relation extraction. In *Proceedings of EACL 2006*, 409–416.

Ron, T. 2006. Generating entailment rules based on online lexical resources. Master's thesis, Computer Science Department, Bar-Ilan University, Ramat-Gan, Israel.

Szpektor, I.; Tanev, H.; Dagan, I.; and Coppola, B. 2004. Scaling web-based acquisition of entailment relations. In *Proceedings of EMNLP 2004*, 41–48.

Tatu, M., and Moldovan, D. 2006. A logic-based semantic approach to recognizing textual entailment. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, 819–826.