Natural Language as the Basis for Meaning Representation and Inference

Ido Dagan[1], Roy Bar-Haim[1], Idan Szpektor[1], Iddo Greental[2], and Eyal Shnarch[1]

[1] Bar Ilan University, Ramat Gan 52900, Israel,
dagan@cs.biu.ac.il
[2] Tel Aviv University, Tel Aviv 69978, Israel

Abstract. Semantic inference is an important component in many natural language understanding applications. Classical approaches to semantic inference rely on logical representations for meaning, which may be viewed as being "external" to the natural language itself. However, practical applications usually adopt shallower lexical or lexical-syntactic representations, which correspond closely to language structure. In many cases, such approaches lack a principled meaning representation and inference framework. We describe a generic semantic inference framework that operates directly on language-based structures, particularly 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 and potential of our approach. Additionally, such inference is shown to improve the critical step of unsupervised learning of entailment rules, which in turn enhances the scope of the inference system.
This paper corresponds to the invited talk of the first author at CICLING 2008.

1 Introduction

It has been a common assumption that the structure of natural language is not suitable to formally represent meanings and to conduct inferences over them. Indeed, according to the traditional formal semantics approach inference is conducted at a logical level. Texts are first translated, or *interpreted*, into some logical form and then new propositions are inferred from interpreted texts by a logical theorem prover. Meaning and inference are thus captured by representations that are "external" to the language itself, and are typically independent of the structure of any particular natural language.

However, practical text understanding systems usually employ shallower lexical and lexical-syntactic representations, which clearly correspond to the structure of the particular natural language being processed. Such representations are 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 [1–3], a popular framework for evaluating application-independent semantic inference, where only a few systems applied logical inference [4–6].

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. This paper describes a step towards filling this gap, by defining a principled semantic inference mechanism over natural language based representations, particularly parse-based structures (originally presented in [7]). In fact, the formulation of the textual entailment task as a touchstone for semantic processing, which is not bound to any particular extra-linguistic representation, encourages the development of semantic formalisms like ours which operate directly over relatively immediate natural language structures.

Within the textual entailment setting a semantic inference 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 (see [8] for an initial integration of approximate matching methods within our framework, applied to the RTE-3 benchmark). We define a proof system that operates over syntactic parse trees, which are the basis for representing both sentence meaning and inference knowledge. New trees are derived using *entailment rules*, which provide a principled and uniform mechanism for incorporating a wide variety of inference knowledge types. 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 practical scope within which semantic interpretation is not needed. This goal is somewhat analogous to that of the Natural Logic paradigm [9, 10], which attempted to model monotonicity-based inference using natural language rather than logical representation. While similar modeling of such phenomena may be embedded in our framework as well, we are mostly interested in modeling a broader range of phenomena which are most relevant for applied semantic inference.

The first sections of this paper present our formulation and implementation of the inference framework, and its evaluation on an inference task. In addition, we show (in Section 6) how the inference mechanism can be utilized also to improve unsupervised learning of entailment rules, which in turn enhances the potential scope of the inference system itself (originally presented in [11]; see this paper and [7] for additional detail about our and related works).

2 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.

2.1 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.

2.2 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 upper-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 lower-right part of Figure 1(a) correspond to r after alignment copying. *yesterday* was copied to r due to the alignment of its parent verb node.

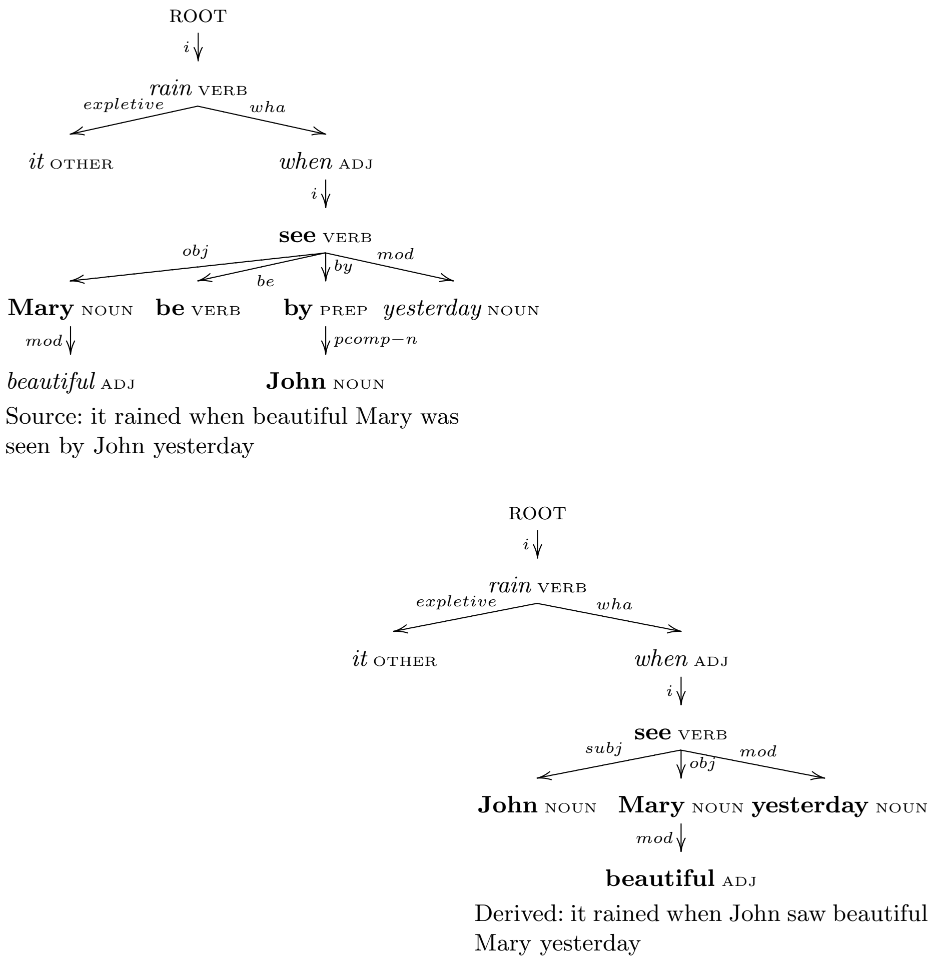
Derived tree generation by rule type Our formalism has two methods for generating the derived tree: *substitution* and *introduction*, as specified by the rule type. With *substitution* rules, the derived tree d is obtained by making a local modification to the source tree s. Except for this modification 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 nodes) with r. This is the case for the passive rule. The lower-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 lower-right part of Figure 1(a) yields the proposition *John saw beautiful Mary yesterday.*

2.3 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 of rules, these additional features may block inappropriate subsequent rule applications, such as for negated predicates.

2.4 Template Hypotheses

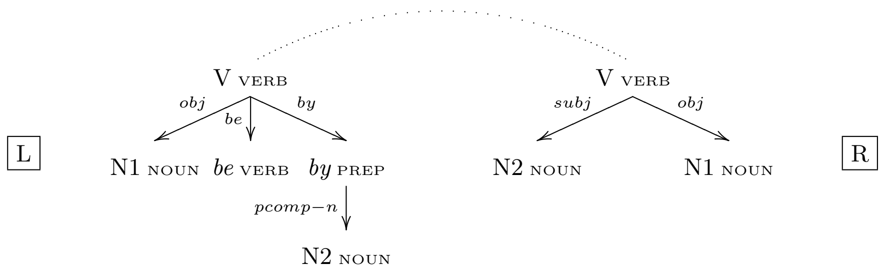
For many applications it is useful to allow the hypothesis h to be a template rather than a proposition, that is, to contain variables. The variables in this case are existentially quantified: t entails h if there exists a proposition h', obtained from h by variable instantiation, so that t entails h'. The obtained variable instantiations may stand, for example, for sought answers in questions or slots to



Source: it rained when beautiful Mary was seen by John yesterday

Derived: it rained when John saw beautiful Mary yesterday

(a) Passive-to-active tree transformation



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

Fig. 1. Application of an inference rule. POS and relation labels are based on Minipar [12].

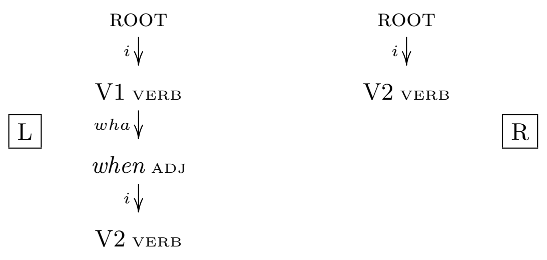


Fig. 2. Temporal clausal modifier extraction (introduction rule)

be filled in relation extraction applications. For example, applying this framework in a question-answering setting, the question *Who killed Kennedy?* may be translated into the template 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*.

3 Rules for Generic Linguistic Structures

Based on the above framework for entailment rules 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. The various rule categories we developed are summarized in Table 1 and explained below.

3.1 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).

3.2 Polarity-Based Rules

Consider the following two examples:

John *knows* that Mary is here ⇒ Mary is here.
John *believes* that Mary is here ⇏ Mary is here.

Valid inference of propositions embedded as verb complements depends on the

#	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. (⇏ What we've seen is actual costs come down.)

Table 1. Summary of rule base for generic linguistic structures, and examples of their application.

verb properties, and the polarity of the context in which the verb appears (positive, negative, or unknown) [13]. We extracted from the polarity lexicon of Nairn et al. (see Acknowledgments) 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.

3.3 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*.

3.4 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.

4 Lexical-Syntactic Rules

Lexical-Syntactic entailment rules include open-class lexical components within varying syntactic structures. Accordingly these rules are numerous compared to the generic rules of the previous section, and have been acquired either from available large-scale lexicographic resources or automatically (e.g. paraphrases). We incorporated several sources of such rules, as described below (see [8] for our use of WordNet as an additional resource for lexical entailment rules).

4.1 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 [14] from Nomlex, a hand-coded database of English nominalizations [15], and from WordNet.

4.2 Automatically Learned Rules

DIRT [16] and TEASE [17] are two state-of-the-art unsupervised algorithms that learn lexical-syntactic inference rules.[3] 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.

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

5 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 RE setting, the system needs to identify in sentences pairs of arguments for a target semantic relation (e.g. *X buy Y*).

5.1 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 law was confirmed by the parliament* is found to prove the instantiated hypothesis *parliament approve law*. Finally, a sample of such sentence-hypothesis pairs are judged manually for true entailment. The process was repeated to compare different system configurations.

We tested 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,[4] 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 hypothesis. While backward chaining the DIRT/TEASE and nominalization rules, this process generates a set of intermediate 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 the DIRT/TEASE rule 'X confirm $Y \rightarrow X$ approve Y', and then further generate the template *confirmation of Y by X*, through the corresponding 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.

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 the generic rules do not modify open-class words). To that end we apply a forward

[4] 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.

search that applies the generic rules, starting from a candidate sentence s and trying to derive (prove) 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 [12] 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").

5.2 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, where h has to be strictly generated from t rather than embedded in t. The only inference rule available in the basic Proof configuration 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.[5] 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 would be equal to the yield divided

[5] 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.

Morpho-Syntactic Variations	
X compose $Y \rightarrow X$ write Y	X is composed by $Y \rightarrow X$ write Y
X accuse $Y \leftrightarrow X$ blame Y	X's accusation of $Y \leftrightarrow X$ blame Y
X acquire $Y \rightarrow X$ obtain Y	acquisition of Y by $X \rightarrow Y$ is obtained by X

Table 3. Examples of learned rules that differ only in their morpho-syntactic structure.

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 in any corpus.

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.

6 Applying entailment inference during rule learning

This section describes how certain forms of entailment inference, of the types described earlier, can be utilized also to improve unsupervised learning of entailment rules (see [11] for further detail). In this setting, entailment inference yields "canonical forms" of various template variations, thus enhancing the statistical evidence that underlies learning and removing redundancies amongst the learned rules.

A noticeable phenomenon of lexical-syntactic templates is that they have many morpho-syntactic variations, which (largely) represent the same predicate and are semantically equivalent. For example, 'X compose Y' can be expressed also by 'Y is composed by X' or 'X's composition of Y'. Current learning algorithms ignore this morpho-syntactic variability. They treat these variations as semantically different and learn rules for each variation separately. This leads to several undesired consequences. First, statistics for a particular semantic predicate are scattered among different templates. This may result in insufficient statistics for learning a rule in any of its variations. Second, though rules may be learned in several variations (see Table 3), in most cases only a small part of the morpho-syntactic variations are learned. Thus, an inference system that uses only these learned rules would miss recognizing a substantial number of variations of the sought predicate.

Consequently, we propose using an entailment module that recognizes generic morphological and syntactic regularities (morpho-syntactic entailments) during learning time, and learn only canonical forms of templates and rules. Then, applying morpho-syntactic entailments also at inference time, in conjunction with the learned lexical-based canonical rules (as described earlier in this paper), guarantees the coverage of all morpho-syntactic variations of a given canonical rule.

Our proposed approach poses two advantages. First, the statistics from the different morpho-syntactic variations accumulate for a single (canoncial) template form. The improved statistics may result, for example, in learning more rules. Second, the learning output does not contain redundancies due to variations of the same predicate. Additionally, the evaluation of learning algorithms becomes more accurate, since rules learned for different variations of the same predicate are not counted seperately.

For the purpose of generating canonical templates during learning we implemented a morpho-syntactic entailment module that applies some of the inferences of the complete entailment system. These include syntactic rules for major syntactic phenomena (like passive and conjunctions) and morphological rules that address nominalizations. In the remainder of this section we provide first some necessary background about entailment rule learning; then we describe how learning is enhanced with the canonization module and evaluate its impact.

6.1 Background – Entailment Rule Learning

Many algorithms for automatically learning entailment rules and paraphrases (which can be viewed as bidirectional entailment rules) were proposed in recent years. These methods recognize templates in texts and identify entailment relations between them based on shared features.

Template	Single-feature Approach (DIRT)		Anchor-Set Approach
	X-vector Features	Y-vector Features	Common Features
X compose Y	Bach, Beethoven Mozart, he	symphony, music sonata, opera	{X=‘Mozart’; Y=‘Jupiter symphony’}, {X=‘Bach’; Y=‘Sonata Abassoonata’}
X write Y	Tolstoy, Bach, author, Mozart, he	symphony, anthem, sonata, book, novel	

Table 4. Examples for features of the anchor set and single-feature approaches for two related templates.

These algorithms may be divided into two types. The prominent approach identify an entailment relation between two templates by finding variable instantiation tuples, termed here *anchor-sets*, that are common to both templates [18–21, 17, 22]. Anchor-sets are complex features, consisting of several terms, labelled by their corresponding variables. Table 4 (right column) presents common anchor-sets for the related templates ‘X compose Y’ and ‘X write Y’. Typically, only few common anchor-sets are identified for each entailment relation.

A different *single-feature* approach is proposed by the DIRT algorithm [16]. It uses simple, less informative but more frequent features. It constructs a feature vector for each variable of a given template, representing the context words that fill the variable in the different occurrences of the template in the corpus. Two templates are identified as semantically related if they have similar vectors. Table 4 shows examples for features of this type. DIRT parses a whole corpus and limits the allowed structures of templates only to paths in the parse graphs, connecting nouns at their ends.

In this paper we report experiments with the TEASE algorithm [17]. It is an unsupervised algorithm that acquires entailment relations from the Web for given input templates using the anchor-set approach, where at least two common anchor-sets were required for learning a relation. Sentences were parsed using the Minipar dependency parser [12].

For a given input template I, the algorithm can be viewed as learning a list of output templates $\{O_j\}_1^{n_I}$, where n_I is the number of templates learned for I. Each output template is suggested as holding an entailment relation with the input template, but most current algorithms do not specify the entailment direction(s). Thus, each pair $\{I, O_j\}$ induces two candidate directional entailment rules: ‘$I \rightarrow O_j$’ and ‘$O_j \rightarrow I$’.

The learned entailment rules and paraphrases can be used at *inference time* in applications such as IE [19, 23, 24] and QA [16, 18, 25], where matched rules deduce new target predicate instances from texts (like the ‘compose → write’ example in the beginning of Section **??**). As shown in previous evaluations the precision of algorithms like DIRT and TEASE is still limited [16, 20, 17, 26]. Currently, utilizing their output in applications may require manual filtering of the learned rules, and the algorithms’ utility is reflected mainly by the amount of correct rules they learn.

Current methods for learning lexical-syntactic rules do not address the morpho-syntactic variability at learning time. Thus, they learn rules separately for each template variation. This results in either learning redundant rules (see Table 3) or missing some of the relevant rules that occur in a corpus. Moreover, some rules might not be learned in any variation. For example, if for each of the rules 'X acquire $Y \rightarrow X$ own Y', 'Y is acquired by $X \rightarrow X$ own Y' and 'X's acquisition of $Y \rightarrow X$ own Y' there is just anecdotal, but insufficient, statistical evidence then none of them will be learned. On the other hand, accumulating the statistics for all rules together may suffice to enable their learning.

6.2 The Morpho-Syntactic Canonization Module

In our scheme, we use a morpho-syntactic entailment module to transform lexical-syntactic template variations that occur in a text into their *canonical form*. This form, which we chose to be the active verb form with direct modifiers, is entailed by other template variations.

We implemented the canonization module based on a set of *canonization rules*, which are highly accurate morpho-syntactic entailment rules. Each rule represents one morpho-syntactic regularity that is eliminated when the rule is applied to a given template (see examples in Table 5 and Figure 3).

The current canonization rule collection consists of two types of rules: (a) syntactic-based rules; (b) morpho-syntactic nominalization rules. We next describe each rule type. As we use the Minipar parser, all rules are adapted to Minipar's output format.

Syntactic-based Rules These rules capture entailment patterns associated with common syntactic structures. Their function is to simplify and generalize the syntactic structure of a template.

In the current implementation we manually created the following simplification rules: (a) passive forms into active forms; (b) removal of conjunctions; (c) removal of appositions; (d) removal of abbreviations; (e) removal of set description by the 'such as' preposition. Table 5 presents some of the rules we created together with examples of their effect.

Nominalization Rules Entailment rules such as 'acquisition of Y by $X \rightarrow X$ acquire Y' and 'Y's acquisition by $X \rightarrow X$ acquire Y' capture the relations between verbs and their nominalizations. We automatically derived these rules from Nomlex, a hand-coded database of about 1000 English nominalizations [15], as described in [14]. These rules transform any nominal template in Nomlex into its related verbal form, preserving the semantics of the original template predicate. We chose the verbal form as the canonical form since for every predicate with specific semantic modifiers there is only one verbal active form in Nomlex, but typically several equivalent nominal forms.

Rule	Original Template	Simplified Template
passive to active	$X \xleftarrow{pcomp-n} by \xleftarrow{by-subj} find \xrightarrow{obj} Y$	$X \xleftarrow{subj} find \xrightarrow{obj} Y$
conjunction	$X \xleftarrow{subj} find \xrightarrow{obj} gold \xrightarrow{conj} Y$	$X \xleftarrow{subj} find \xrightarrow{obj} Y$
apposition	$X \xleftarrow{subj} find \xrightarrow{obj} protein \xrightarrow{appo} Y$	$X \xleftarrow{subj} find \xrightarrow{obj} Y$
abbreviation	$X \xleftarrow{subj} find \xrightarrow{obj} NDA \xrightarrow{spellout} Y$	$X \xleftarrow{subj} find \xrightarrow{obj} Y$

Table 5. Some of the syntactic rules used in our implementation, together with usage examples (the application of the second rule and the third rule is demonstrated in Figure 3).

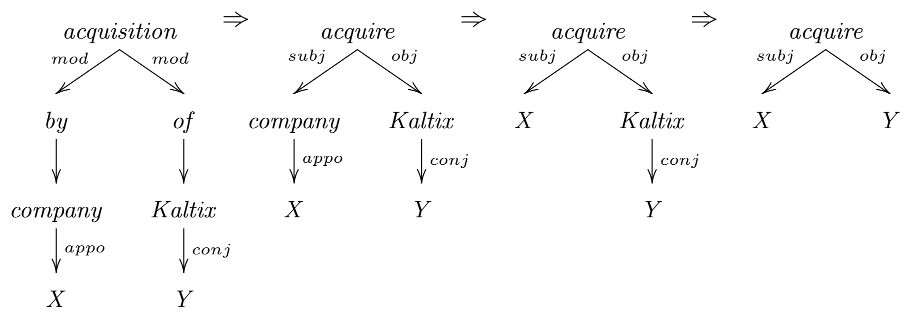


Fig. 3. Chaining of canonization rules that transforms the path template between the arguments {X='Google';Y='Sprinks'}, which occurs in the sentence "*We witnessed the acquisition of Kaltix and Sprinks by another growing company, Google*", into a canonized template form. The first rule applied is a nominalization rule, followed by removal of apposition and removal of conjunction (as described in Table 5). As can be seen, applying the rules in any order will result in the same final canonical form.

Chaining of Canonization Rules Each of the syntactic rules we implemented decreases the size of a template. In addition, nominalization rules can be applied only once for a given template, since no rule in our rule-set transforms a verbal template into one of its nominal forms. Thus, applying rules until no rule can apply is a finite process. In addition, each of our rules is independent of the others, operating on a different set of dependency relations. Consequently, chaining of canonization rules is a well-defined procedure, since applying any sequence of rules until no other rule can apply will result in the same final canonical template form. Figure 3 illustrates an example for rule chaining.

6.3 Applying the Canonization Module

When morpho-syntactic entailment rules are utilized at inference time (e.g. [23]), they recognize a closure of morpho-syntactic variations for a lexical-syntactic template. Accordingly, acquisition algorithms may learn just a single morpho-syntactic variation of an entailment rule.

With this modular scheme in mind, we propose to solve the learning problems discussed in Subsection 6.1 by utilizing the morpho-syntactic entailment module at learning time as well. We incorporate the module in the learning algorithm by converting each template variation occurrence in the learning corpus into an occurrence of a canonical template. Consequently, the learning algorithms operate only on canonical forms.

As discussed earlier, when canonization is used no morpho-syntactically redundant rules are learned, with respect to the variations that are recognized by the module. This makes the output more compact, both for storage and for use. In addition, the statistical reliability of learned rules may be improved. For example, rules that could not be learned for any particular variation may be learned now for the canonical form.

Methodologically, previous evaluations of learning algorithms reported accuracy relative to the redundant list of rules, which creates a bias for templates with many frequent variations. When this bias is removed and only truly different lexical-syntactic rules are assessed, evaluation is more efficient and accurate.

6.4 Evaluation

Human Judgements We have selected 20 different verbs and verbal phrases[6] as input templates for TEASE (see [11] for an additional evaluation based on the DIRT algorithm). We executed its baseline version (without canonization), denoted by $TEASE_b$, as well as the version with the canonization module, denoted by $TEASE_c$. The results of these two executions constitute our test-set rules. To enable the comparison between the outputs of the two version we converted the output templates of $TEASE_b$ into their canonical forms (that is, this canonization is not considered as part of the $TEASE_b$ algorithm, and was not exploited during learning).

As discussed above, TEASE does not learn the direction(s) of an entailment relation between an input template I and a learned output template O. Thus, we evaluated both candidate directional rules, '$I \rightarrow O$' and '$O \rightarrow I$'.

Rule Evaluation The prominent *rule based* approach for evaluating entailment rules is to present them to human judges, who assess whether each rule is correct or not. Generally, a rule is considered correct if the judge could think of reasonable contexts under which it holds. However, it is difficult to define explicitly when a learned rule should be considered correct under this methodology.

[6] The verbs are: accuse, approve, calculate, change, demand, establish, finish, hit, invent, kill, know, leave, merge with, name as, quote, recover, reflect, tell, worsen, write.

Instead, we follow the *instance based* evaluation methodology presented in [26]. By this methodology, each rule '$L \rightarrow R$' is evaluated by presenting the judges not only with the rule itself but rather with a sample of sentences that match its left hand side L. The judges then assess whether the rule is valid under each specific example sentence. The precision of a rule is computed by the percentage of examples for which entailment holds out of all "relevant" examples in the judged sample. The rule is then considered correct if its precision is higher than 0.8 (see [26] for details). This instance-based approach for human judgment was shown to be more reliable than the rule-based approach.

Evaluation Setup We separated the templates that were learned by $TEASE_c$ into two lists: (a) a *baseline-templates* list containing templates learned also by $TEASE_b$; (b) a *new-templates* list containing templates that were not learned by $TEASE_b$, but learned by $TEASE_c$ thanks to the improved accumulated statistics for canonical forms. In total, 3871 (unique) canonical templates were learned: 3309 in the baseline-templates list and 562 in the new-templates list. Inherently, every output template learned by $TEASE_b$ is also learned in its canonical form by $TEASE_c$, since its supporting statistics may only increase.

We randomly sampled 100 templates from each list and evaluated their correctness according to the human judgment methodology. To that end we retrieved 10 example sentences for each rule from the first CD of Reuters RCV1. Two judges, fluent English speakers, evaluated the examples. We randomly split the rules between the judges with 100 rules (942 examples) cross annotated for agreement measurement.

Results First, we measured the redundancy in the rules learned by $TEASE_b$, that is, the percentage of output templates that could be removed due to redundancy of morpho-syntactic variations, to be 6.2% per input template on average. This redundancy was eliminated using the canonization module.

Next, we evaluated the quality of the sampled rules using two scores: (1) micro average ***Precision***, the percentage of correct templates out of all learned templates, and (2) average ***Yield***, the average expected number of correct templates learned for each input template, as extrapolated based on the sampling proportion. The results are presented in Table 6. The agreement between the judges was measured by the Kappa value [27], obtaining a 0.67 score (corresponding to substantial agreement).

We expect $TEASE_c$ to learn new rules using the canonization module. In our experiment, 5.8 more correct templates were learned on average per input template by $TEASE_c$. This corresponds to an increase of 11.6% in average Yield (see Table 6). Examples of new correctly learned templates are shown in Table 7.

There is a slight decrease in precision when using $TEASE_c$. One possible reason is that the new templates are usually learned from very few occurrences of different variations, accumulated for the canonical templates. Thus, they may have a somewhat lower precision in general. Overall, the significant increase in

Template List	Avg. Precision	Avg. Yield
$TEASE_b$	30.1%	49.8
$TEASE_c$	28.7%	55.6

Table 6. Average Precision and Yield of the output lists.

Input Template	Learned Template
X accuse Y	X blame Y
X approve Y	X take action on Y
X demand Y	X call for Y, X in demand for Y
X establish Y	X open Y
X hit Y	X slap Y
X invent Y	grant X patent on Y, X is co-inventor of Y
X kill Y	X hang Y, charge X in death of Y
X named as Y	hire X as Y, select X as Y
X quote Y	X cite Y
X tell Y	X persuade Y, X say to Y
X worsen Y	X impair Y

Table 7. Examples for correct templates that TEASE learned only after using canonization rules.

Yield is much more important, especially if the learned rules are later filtered manually.

6.5 Analysis

Parser errors are one of the main reasons that variations are sometimes not transformed into their canonical form. These errors result in different parse trees for the same syntactic construct. Thus, several parser-dependent rules may be needed to capture the same syntactic structure. Moreover, it is difficult to design canonization rules for some parsing errors, since the resulting parse trees consist of structures that are common to other irrelevant templates. For example, when Minipar chooses the head of the conjunct '*Y*' in "*The interaction between X and Y will not hold for long*" to be '*interaction*' rather than '*X*', the appropriate canonization rule cannot be applied. These errors affect both the learning phase, where statistics are not accumulated for the appropriate canonical form, and the inference phase, where variations of a canonical rule template are not recognized.

Finally, we note that the reported results correspond only to the phenomena captured by our currently implemented canonization rules. Adding rules that cover more morpho-syntactic phenomena is expected to increase the benefit of the canonization scheme. For example, there are many nominalizations that are not specified in the current Nomlex version, but can be found in other resources, such as WordNet [28].

7 Conclusions

In this paper we presented a framework for semantic inference at the lexical-syntactic level, suggesting that natural language based structures may be suitable for meaning representation and inference. Our formalism was found suitable for describing a wide spectrum of inference knowledge, in the form 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. Finally, we showed that the inference module can be utilized also for improving unsupervised acquisition of entailment rules through canonization, which in turn enhances the scope of the inference system.

In future work we plan to enhance the framework to allow inference from multiple sentences. We will also investigate integration of the proof system with different methods for approximate matching, which would enable its application in additional settings, as was demonstrated initially in [8] for the RTE-3 benchmark.

Acknowledgements

We are grateful to Cleo Condoravdi for making the polarity lexicon developed at PARC available for this research. We also thank Dan Roth for helpful discussions, and Chen Erez for her help in the experiments. This work was partially supported by ISF grant 1095/05, the Israel Internet Association (ISOC-IL) grant 9022, the Israeli Ministry of Industry and Trade under the NEGEV Consortium (www.negev-initiative.org) and the IST Programme of the European Community under the PASCAL Network of Excellence IST-2002-506778.

References

1. Dagan, I., Glickman, O., Magnini, B.: The pascal recognising textual entailment challenge. In: Lecture Notes in Computer Science, Volume 3944. Volume 3944. (2006) 177–190
2. Bar-Haim, R., Dagan, I., Dolan, B., Ferro, L., Giampiccolo, D., Magnini, B., Szpektor, I.: The second pascal recognising textual entailment challenge. In: Second PASCAL Challenge Workshop for Recognizing Textual Entailment. (2006)
3. Giampiccolo, D., Magnini, B., Dagan, I., Dolan, B.: The third pascal recognizing textual entailment challenge. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing. (2007)
4. Raina, R., Ng, A.Y., Manning, C.D.: Robust textual inference via learning and abductive reasoning. In: Proceedings of AAAI. (2005)
5. Tatu, M., Moldovan, D.: A logic-based semantic approach to recognizing textual entailment. In: Proceedings of COLING-ACL. (2006)

6. Bos, J., Markert, K.: When logical inference helps determining textual entailment (and when it doesn't). In: Proceedings of the Second PASCAL Challenges Workshop on Recognizing Textual Entailment. (2006)
7. Bar-Haim, R., Dagan, I., Greental, I., Shnarch, E.: Semantic inference at the lexical-syntactic level. In: Proceedings of AAAI. (2007)
8. Bar-Haim, R., Dagan, I., Greental, I., Szpektor, I., Friedman, M.: Semantic inference at the lexical-syntactic level for textual entailment recognition. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing. (2007)
9. Valencia, V.S.: Parsing-driven inference: natural logic. Linguistic Analysis **25** (1995) 258–285
10. MacCartney, B., Manning, C.D.: Natural logic for textual inference. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, Prague, Association for Computational Linguistics (2007) 193–200
11. Szpektor, I., Dagan, I.: Learning canonical forms of entailment rules. In: Proceedings of RANLP. (2007)
12. Lin, D.: Dependency-based evaluation of minipar. In: Proceedings of the Workshop on Evaluation of Parsing Systems at LREC. (1998)
13. Nairn, R., Condoravdi, C., Karttunen., L.: Computing relative polarity for textual inference. In: Proceedings of ICoS-5. (2006)
14. Ron, T.: Generating entailment rules using online lexical resources. Masterś thesis, Computer Science Department, Bar Ilan University (2006)
15. Macleod, C., Grishman, R., Meyers, A., Barrett, L., Reeves, R.: Nomlex: A lexicon of nominalizations. In: EURALEX. (1998)
16. Lin, D., Pantel, P.: Discovery of inference rules for question answering. In: Natural Language Engineering. Volume 7(4). (2001) 343–360
17. Szpektor, I., Tanev, H., Dagan, I., Coppola, B.: Scaling web-based acquisition of entailment relations. In: Proceedings of EMNLP. (2004)
18. Ravichandran, D., Hovy, E.: Learning surface text patterns for a question answering system. In: Proceedings of ACL. (2002)
19. Shinyama, Y., Sekine, S., Kiyoshi, S., Grishman, R.: Automatic paraphrase acquisition from news articles. In: Proceedings of HLT. (2002)
20. Barzilay, R., Lee, L.: Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In: Proceedings of HLT-NAACL. (2003)
21. Quirk, C., Brockett, C., Dolan, W.: Monolingual machine translation for paraphrase generation. In: Proceedings of EMNLP. (2004)
22. Sekine, S.: Automatic paraphrase discovery based on context and keywords between ne pairs. In: Proceedings of IWP. (2005)
23. Romano, L., Kouylekov, M., Szpektor, I., Dagan, I., Lavelli, A.: Investigating a generic paraphrase-based approach for relation extraction. In: Proceedings of EACL. (2006)
24. Sekine, S.: On-demand information extraction. In: Proceedings of the COLING/ACL Main Conference Poster Sessions. (2006)
25. Harabagiu, S., Hickl, A.: Methods for using textual entailment in open-domain question answering. In: Proceedings of ACL. (2006)
26. Szpektor, I., Shnarch, E., Dagan, I.: Instance-based evaluation of entailment rule acquisition. In: Proceedings of ACL. (2007)
27. Cohen, J.: A coefficient of agreement for nominal scales. Educational and Psychological Measurement **20** (1960) 37–46
28. Fellbaum, C., ed.: WordNet: An Electronic Lexical Database. Language, Speech and Communication. MIT Press (1998)