Language Resources and Evaluation manuscript No.
(will be inserted by the editor)

A Syntax-based Rule-base for Textual Entailment

Amnon Lotan · Asher Stern · Ido Dagan ·
Jonathan Berant

Received: date / Accepted: date

Abstract Insert your abstract here. Include keywords, PACS and mathematical subject classification numbers as needed.

Keywords Syntactic Rules · Entailment Rule-base · Textual Entailment · Entailment Rules · Inference · NLP

1 Introduction

Textual entailment (TE) is the semantic inference task which takes two text fragments, termed text (T) and hypothesis (H), and determines whether T entails H, in the sense that a human reader would judge that H is most likely true, given T (Dagan et al, 2013) . In recent years TE has been established as an important research area within Natural Language Processing (NLP). This is because it can help solve many semantic inference tasks such as Question

Amnon Lotan
Department of Linguistics, Tel Aviv University, Tel Aviv, Israel
E-mail: amnonlot@post.tau.ac.il

Asher Stern
Computer Science Department, Bar Ilan University, Ramat Gan, Israel
E-mail: astern7@gmail.com

Ido Dagan
Computer Science Department, Bar Ilan University, Ramat Gan, Israel
E-mail: dagan@cs.biu.ac.il

Jonathan Berant
Stanford University
E-mail: joberant@stanford.edu

Answering (QA), Information Extraction (IE), Relation Extraction, Summarization, Semantic Parsing and educational applications (Harabagiu and Hickl, 2006; Ravichandran and Hovy, 2002; Shinyama and Sekine, 2006; Romano et al, 2006; Harabagiu et al, 2007; Nielsen and Ward, 2007; Berant and Liang, 2014). Notably, the annual PASCAL Recognizing Textual Entailment (RTE) challenges (Bentivogli et al, 2009, 2010b) serve as the main benchmark for evaluating TE applications.

Generally speaking, TE systems try to detect (or rule out) entailment, by assessing how the elements of the hypothesis can be "covered" by the information that is expressed in the text or can be implied from it. While simple systems attempt only at lexical matching between T and H (Shnarch et al, 2012; Clark and Harrison, 2010; MacKinlay and Baldwin, 2009), it has been shown that systems that consider also the syntactic structures in T and H yield improved performance (Cabrio et al, 2008; Wang and Neumann, 2008). Though these systems benefit from syntactic analysis, they need to cope with the challenge of a variety of distinct syntactic structures that express or imply the same meaning.

Some previous works have taken various measures to address this challenge. For instance, Bar-Haim et al (2007) and de Salvo Braz et al (2005) built banks of syntax-based entailment rules, that can convert text fragments with certain predefined structures into equivalent or entailed structures. For example, they encode the equivalence *Man bites dog* $\Leftrightarrow$ *Dog is bitten by man.* Using these rules, an entailment system has the means to straightforwardly compare H against many different syntactic forms of T. We consider this to be a sound approach, but these rule-bases suffer from several practical drawbacks: they are limited in scope, not publicly available, and their development lacked substantial performance evaluation.

In this paper, we present a novel comprehensive knowledge resource of syntax-based inference rules. It offers the broadest scope of rules to date, covering content of the previous work as well as many novel rules, making it more than twice as large, motivated by empirical analysis that is based on the target RTE datasets. The resource is publicly-available[1], including documentation and an edit utility, which allows for maintaining and altering the existing rules, as well as developing extensions. Furthermore, its potential and empirical contribution to the TE task is methodically evaluated, via ablation

[1] `http://u.cs.biu.ac.il/~nlp/resources/downloads/syntacx-based-rulebase-for-textual-entailment`

tests that isolate its impact on system performance, as well as via quantitative and qualitative error analyses.

To represent our rules, we use a generic parse-tree transformation formalism, which is one of the leading approaches for TE, similar to those of Bar-Haim et al (2007), Heilman and Smith (2010) and Stern et al (2010). The formalism defines entailment rules, as well as how a rule can be *applied* to a text and generate new consequents. Thus, our rule-base can be used by different types of inference architectures. We evaluated it in BiuTee (Stern and Dagan, 2011), a general purpose entailment engine, which can employ many different rule-bases within this framework (lexical, lexical syntactic rules etc.).

Our knowledge resource encompasses a wide variety of syntactic phenomena, relevant to the conversion between semantically-equivalent forms, as well as to the extraction of generic implications. These include: substituting one sentential construction with an equivalent (e.g. active vs. passive voice), possessive constructions, complementizer insertion and deletion, coordination manipulation, extracting IS-A relations (e.g. from appositions), determiner substitutions and case correction.

The rules are designed based on the Stanford dependency relations standard (de Marneffe and Manning, 2008), as implemented by the EasyFirst parser (Goldberg and Elhadad, 2010). Thus, they should be compatible as-is with any inference system that uses that parser. Since different parsers implement the Stanford representation differently, using the rules in a system with a different compliant parser may yield poor performance. Based on our experience, we estimate that the necessary adaptations to accommodate for replacing a parser would be rather minimal.

Some particular earlier works are sources for rules, as follows. Amoia and Gardent (2008) developed a suite of 36 inference patterns revolving around English adjectives, out of which 9 are generic and were adapted into our formalism. Hearst (1992) and later Pantel et al (2004) presented 15 syntactic patterns that detect hyponyms (IS-A relations) in general domain natural English texts, most of which were similarly incorporated in our rule-base. Finally, Bar-Haim et al (2007) attempted to develop a comprehensive generic rule-base, with rules dealing with 7 different syntactic categories. As opposed to the other sources for rules, they also presented a formalism that defines entailment rules and their application, which we adopt in this investigation.

Besides existing sources, novel rules account for about half of our inventory, and as mentioned above, their design is mostly based on an investigation of the generic operations that would be most useful in the RTE challenge datasets.

As opposed to all previous work in this field, we report a series of evaluations and analyses to assess the resource. They can be divided into three types. Firstly, they estimate the general potential of syntax-based resources to help in resolving RTE datasets. Here we show that indeed syntactic rules play a substantial part in entailment. Secondly, they measure the rule-base's recall and precision. The recall is estimated by comparing against a set of gold standard transformations, at 70%. The precision of the rules is defined as how often a rule application yields a correct entailment, and showed to be 94%. Thirdly, they analyse its impact on an actual RTE system, in particular the above-mentioned system BiuTee, via error analysis and ablation tests. We report that the resource demonstrably improves the performance of a concrete entailment engine, adding 2.9% to the F1 score on the RTE5 dataset.

We've made our knowledge resource publicly available. The distribution package offers all the rules reported and evaluated below, as well as a separate collection of several more rules that are in themselves precise (see below), but which did not perform well in the ablation tests, and therefore were set aside. We make a point of retaining the latter, since our empirical experience led us to conclude that distinct entailment tasks, and distinct genres of text, tend to benefit from different sets of entailment rules.

The rules are given in XML format, and come with a graphical editor that allows for an intuitive tree-based display, for ease of maintenance and modification, and for defining further rules. A component in BiuTee is capable of loading the rules onto Java objects that may be used in its open source environment, in compliance with our formalism. In order to offer more portability, BiuTee also provides a command line utility that converts rules from our XML format to CoNLL format[2]. Developed as part of this work, the software is accompanied by a user guide, giving full documentation of all aspects of the rule-base and its usage.

This work is structured as follows. Section 2 discusses the aforementioned previous work. Section 3 lays out the formalism of the resource: sentence representation and the formal definition for entailment rules, and for the way rules can be applied to generate entailments. Section 4 presents the rules them-

[2] `http://ilk.uvt.nl/conll/#dataformat`

selves, illustrating examples for each of its main categories. Section 5 reports the extensive analyses and evaluations mentioned above, and Section 6 offers some conclusions.

2 Previous Work

2.1 The use of Syntactic Knowledge in Entailment Systems

Many research approaches represent and analyse T and H at the lexical level, with no syntactic analysis (Shnarch et al, 2012; Clark and Harrison, 2010; MacKinlay and Baldwin, 2009). Hence, they all lack the ability to make inferences based on structure. For instance, *The horse was eating by the hay* and *The horse was eaten by the hay* would seem equivalent lexically, but their syntactic structures are distinct.

Many other approaches rely on some kind of syntactic representation of text (see below), and so they have the advantageous ability to identify equivalent, or contradicting, meanings that are expressed in the syntax, rather than at the lexical level (Cabrio et al, 2008; Wang and Neumann, 2008; Bar-Haim et al, 2007; de Salvo Braz et al, 2005). However, these systems face the challenge of sentences that are different in syntactic form, but express related meanings. For instance, *Mica wants this car* and *This is the car Mica wants* both mean the same.

One way, in which syntax-sensitive systems rise to this challenge, is to predefine a collection of *transformations* that allow conversion from one syntactic form to another, while preserving, or entailing, the meaning. Such transformations can be used to convert T into a form more similar to H, or vice versa.

Below, we mention several entailment systems that incorporate syntax-based transformations, usually termed syntax-based *entailment rules*.

De Salvo Braz et al (2005) incorporated syntax- and semantic-based entailment rules in an entailment system. In their system, entailment rules are applied over hybrid syntacto-semantic structures called *concept graphs*. When the template side (left hand side, LHS) of a rule is matched in the concept graph (e.g., a pattern matching a passive voice construction), the graph is augmented with an instantiation of the right hand side (RHS) of the rule (e.g., an active voice construction). After several iterations of rule applications, the system attempts to embed the hypothesis in the augmented graph.

Bar-Haim et al (2007) presented another entailment system that applies all matching rules on a syntactic parse tree of T, and iteratively on all consequents, thus representing all possible entailments within a data structure, called a *compact forest* of consequent trees. Compact forests are a scalable data structure, capable of representing a large set of distinct parse trees. Then, a binary classifier determines whether the compact forest contains a tree that is similar enough to the hypothesis tree, which would constitute a positive entailment.

Hickl (2008) derived from a given $T - H$ pair a small set of consequents that he terms *discourse commitments*. These consequents are based on syntax (coordinations, appositions, relative clauses etc.), co-reference, predicate-complement structure, the extraction of certain relations, and several paraphrases acquired from the Web. The commitments were generated by several different tools and techniques. Pairs of commitments derived from T and H were fed into the next stages of the RTE system - lexical alignment and entailment classification.

All the above works share these disadvantages: the syntax-based rules were relatively few, and were not made publicly available.

In comparison, our research presents a comprehensive syntax-based rule-base, over twice as large as its counterparts. It is motivated by an RTE dataset analysis, while covering the contents of each of the previous works (see below). Furthermore, this study reports a series of methodical qualitative and quantitative evaluations, that measure and the rule-base's potential, coverage of the required syntactic operations, the accuracy of its rules, and empirical usefulness for a particular entailment system. As mentioned above, they show it measures highly in all these aspects.

2.2 Utilized sources of Syntax-based Inference Rules

As noted, many of the rules presented in this study are novel, based on an analysis of the required syntax-based rules in an RTE dataset, as well as on some well known syntactic equivalences and entailments. Other rules were adopted from several papers from the Semantics and Natural Language Inference literatures, as follows.

Amoia and Gardent (2008) produced an adjective-oriented test suite of 36 adjectival inference patterns, based on substantial literature about classifica-

tion of adjectives, on WordNet relations (Fellbaum, 1998; Miller, 1995) and on their own semantic classification of English adjectives. Their goal was to conduct an in-depth study of this narrow slice of inference problems, and to create a resource supporting the evaluation of computational systems handling natural language inference. In our rule-base, presented in Section 4, only 9 of their inference patterns are included, since the rest are not purely syntactic, i.e., they depend on lexical information, and thus fall outside the scope of the phenomena handled by our resource. For instance, we adopt the following transformations:

This is *Adj* for a *N* ⇒ This is an *Adj N*

- *This is* ***expensive*** *for a* ***carpet*** ⇒ *This is an* ***expensive carpet***

N_1 is *Adj* as a N_2 ⇒ N_1 *is a Adj* N_2

- ***John*** *is* ***good*** *as a* ***cook*** ⇒ ***John*** *is a* ***good cook***

Inspired by their work, we composed a novel set of similar rules in the adverbial domain.

Hearst (1992) outlined a way to discover hyponyms (IS-A relations) in general domain natural English texts, using generic syntactic patterns. A few such patterns were incorporated in the rule-base here, for instance:

I like tomato juice ***and other*** *delicacies* ⇒ *Tomato juice is a delicacy*

Pantel et al (2004) present and evaluate several similar patterns, also used here.

While syntax-based transformations have been addressed in these and other works to some extent, so far only Bar-Haim et al (2007) and Bar-Haim (2010) have attempted to develop a high coverage generic rule-base. Our syntax-based rule-base was initially modelled after theirs, which was available to us. However, as mentioned above, our resource possesses over twice as many rules and addresses more syntactic phenomena. Beyond this quantitative contribution, it is the first to be made publicly available, with tools for editing and augmenting it. As an additional methodological contribution, we present a manual dataset analysis, conducted to assess the potential of syntax-based resources in general, in-depth evaluations of the quality of our resource, and error analyses inspecting how a state of the art entailment system utilizes it.

3 Formalism of The Generic Syntax-Based Rule-Base

This section lays out how texts are represented in our system, the formalism defining the rule-base, and how it may be used in an entailment system.

3.1 Sentence Representation

This subsection describes the standards, tools and conventions our rule-base uses to represent texts.

Our approach assumes that T and H are represented by dependency parse trees, detailed as follows. Two example dependency trees are shown in Figure 1(b). Nodes represent words and hold a set of features, including the word lemma and part-of-speech (POS). Edges are annotated with syntactic dependency relations (subject, auxiliary, etc.). From here on, we will use T and H to denote either the text and hypothesis, or their respective parse tree representations.

The syntactic parser we use to generate parse trees from plain text is EasyFirst (Goldberg and Elhadad, 2010), which is state of the art. It complies with two commonly used standards: the Penn Tree Bank standard for POS tags (Marcus et al, 1993), and the Stanford dependency relations standard for dependency parse representation (de Marneffe and Manning, 2008).

Each of our rules is defined using two parse tree templates (see below), represented using Stanford dependencies, while adopting a reduced version of the Penn Tree Bank POS tag set, comprised of: NOUN, VERB, ADJECTIVE, ADVERB, PREPOSITION, DETERMINER, PRONOUN, PUNCTUATION, and OTHER[3]. The conversion from Penn POS tag set to the reduced POS tag set is straightforward, e.g., NN(simple noun) $\rightarrow$ NOUN, NNP(proper noun) $\rightarrow$ NOUN, VBN(participle verb) $\rightarrow$ VERB etc. The full conversion table is in Appendix B.

In addition, we disregard surface word forms in favour of their lemmas. This helps us assume a further degree of abstraction in the tree representation, in which we do not deal with variation in tense, aspect, mood, number and person. On the other hand, this means our rules cannot make inferences that are sensitive to any of the above morphological features. For instance, in our

[3] This set is slightly more coarse than the one proposed by Petrov et al (2012), with 9 vs. 12 tags

framework, the following text fragments all give identical entailments: *Dana plays, Dana play, Dana has been playing, Dana playing, Dana to play.*

Nonetheless, in the future, more linguistic data may be added to the structure of rule nodes and/or edges, thus expanding the formalism of our entailment rules (defined below), and enabling the rules to address more fine-grained linguistic phenomena.

3.2 Entailment Rules

This subsection presents the function of entailment rules, while the next subsection gives a more detailed definition.

Linguistic knowledge required for inference is represented in our framework as entailment rules, which encode parse tree transformations. Each rule application on the source text tree T generates a new consequent sentence (represented as another parse tree). As a framework, we adopt Bar-Haim et al (2007)'s definition, illustrated in Figure 1(a), with a sample entailment rule, representing a passive-to-active transformation. Nonetheless, for convenience and brevity, many entailment rules are presented in this study in an intuitive textual format, rather than with full trees.

From a knowledge representation and usage perspective, entailment rules provide a simple unifying formalism for representing and applying a very broad range of inference knowledge. Some examples of this breadth are illustrated in Table 1. We say that an entailment rule is *syntactic*, *syntax-based* or *generic*, when it contains no lexical data, i.e., no nouns, verbs, adjectives or adverbs. In contrast, word lists in the closed-set parts of speech like pronouns and prepositions are allowed. In addition, using information about generic morphemes is also permissible, e.g., using the adverbial suffix *ly* to derive (most) adverbs from their matching adjectives, as in *rapid* $\rightarrow$ *rapidly.*

Given the syntactically parsed tree T of the source text, and a set of entailment rules, this formalism defines how to compute the set of consequents (entailed trees) derivable from the text using the rules. Each consequent is obtained through a sequence of rule applications, each generating an intermediate parse tree, similar to a proof process in logic. According to the formalism, a text T is judged as entailing a hypothesis H, if H is a consequent of T.

Table 1 Representing diverse knowledge types as entailment rules

Rule Type	Sources	Examples
Syntax-based	Typically manually-composed	Active/passive, apposition, relative clause, coordination
Lexical-Syntactic	Learned with unsupervised algorithms, and derived automatically by integrating information from lexical resources like WordNet and Nomlex	X's wife, Y, $\Rightarrow$ X is married to Y, X bought Y $\Rightarrow$ Y was sold to X, X is a maker of Y $\Rightarrow$ X produces Y
Lexical	Extracted from large manually constructed lexical resources like WordNet, Wikipedia	Steal $\Rightarrow$ take, Albanian $\Rightarrow$ Albania, Janis Joplin $\Rightarrow$ singer, Amazon $\Rightarrow$ South America

3.3 Overview of Entailment Rule Definition and Application

This subsection gives an overview of the formalism for applying entailment rules (described above) on parse trees, which we assume entailment systems follow. For the full technical details, see Appendix A. As an illustration, Figure 1(a) shows passive-to-active transformation rule, and Figure 1(b) shows its application.

A rule '$L \Rightarrow R$' is composed of two *templates*, L on the left-hand-side (LHS) and R on the right-hand-side (RHS). Templates are dependency parse subtrees which may contain *variable* nodes alongside regular parse tree nodes. These variables are regular nodes that have no specified lemma, POS tag and/or relation (between node and parent), so that they match any value for the unspecified data types. A pair of corresponding variables on both sides of a rule (bearing the same variable name) are said to be *aligned*, meaning that, when the rule is applied, the contents and subtree of the LHS variable is copied to the RHS. For instance, in Figure 1(a) the two *N1* variables are aligned, as are two pair of *N2*s and *V*s.

A rule may be *applied* to the tree of a parsed text, in which case *matches* between the entire LHS pattern and parts of the tree are sought. A valid match is essentially some subtree of the tree, which corresponds to the entire LHS, both structurally and by the content of each corresponding node - requiring equivalence in POS, relation to parent and (for non-variables) lemma. If no match is found, the rule application terminates. Otherwise, each match triggers

an *RHS instantiation*, where data in the tree is copied to the RHS template via the rule's alignments, if any, so that the RHS contains no variables, and a new consequent tree is generated, as follows.

At this point, there are two different ways for the consequent generation to complete, depending on whether the *rule type* is substitution, or extraction. If substitution, the instantiated RHS is embedded within a copy of the original tree, replacing the subtree that was matched against the LHS, like in the example in Figure 1 (b). If extraction, the instantiated RHS is left as is, constituting a new tree, whose root is the root of the instantiation. For example, applying the rule in Figure 2 to the lower tree of Figure 1(b) yields the proposition *John saw beautiful Mary yesterday.*

In Section 4, when presenting our rule-base, it is stated at the beginning of each subsection whether it is *substitution* or *extraction.*

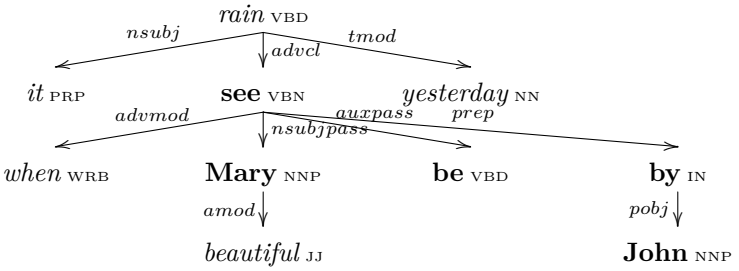
4 A Generic Entailment Rule-base

This section presents the rules of our resource, categorized by syntactic phenomena. It contains over 60 rule *schemas*, which compile into 226 formal rules (see below). For considerations of scope and fluency, each category is described in summary, with only a sample of its rules detailed and discussed. The full resource is available for download[4], along with its full specification, further documentation, and software for formulating new rules and for compiling rules into Java and CoNLL representations.

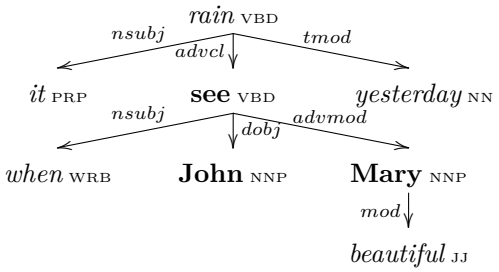
In order to acquire independent empirical motivation for new rules, we performed a preliminary dataset analysis, as follows. First, we manually derived proofs for 60 T - H pairs from a textual entailment oriented dataset, RTE5 main task (Bentivogli et al, 2009), and annotated in detail all inference phenomena involved in those proofs. We then observed the usage counts of syntactic operations, and motivated several new rules by them. This informal experiment's method is the same as in the final data analysis reported in Subsection 5.1. We mention this empirical motivation with regards to specific rules in this section, where relevant.

This section is built as follows: Subsection 4.1 describes our graphical rule editing tool, and the set of conventions with which we use it to represent many similar rules in one compact diagram, known as RULE SCHEMAS; Then, we de-

[4] `http://www.cs.biu.ac.il/~nlp/downloads/syntacx-based-rulebase-for-textual-entailment`

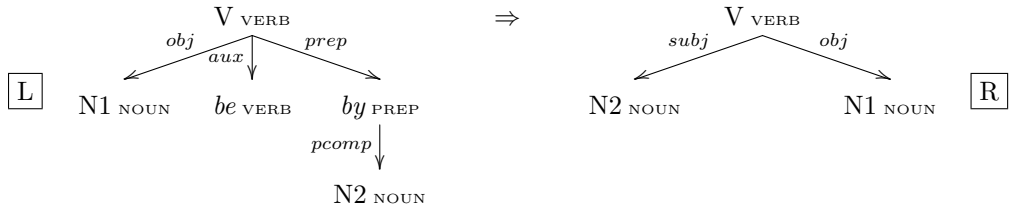


Source: it rained when beautiful Mary was seen by John yesterday



Derived: it rained when John saw beautiful Mary yesterday

(a) Passive-to-active tree transformation



(b) Passive to active substitution rule.

Fig. 1 An inference rule in 1(b), and one of its possible applications in 1(a). POS and relation labels are based on the Stanford dependency standard. N1, N2 and V are variables, which are implicitly aligned between L and R, as described in Bar-Haim et al (2009).

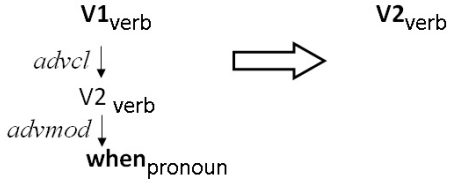


Fig. 2 Temporal clausal modifier extraction (extraction rule)

scribe the rules themselves; Subsection 4.2 presents our complementizer insertion and deletion rules; Subsection 4.3 presents our coordination construction rules; Subsection 4.4 presents our IS-A relation extraction rules; Subsection 4.5 presents our possessive construction rules; Subsection 4.6 presents our general sentential constructions rules; Subsection 4.7 presents our relative clause rules; Subsection 4.8 presents our determiners related rules; and Subsection 4.9 presents our last rules, which correct errors in pronoun case marking left by the application of other rules.

4.1 Rule Editing Tool and Compact Schema

Since our representation of text and of rules is fine grained, while naturally the underlying inference pattern behind each rule is more general, it is common to have to define up to several dozen formal rules in order to capture all the various forms in which the pattern may occur. For instance, the operation of swapping places between a pair of conjuncts (e.g., *Telma and Louise* $\Longleftrightarrow$ *Louise and Telma*) is essentially the same whether the conjuncts are nouns, verbs, adjectives or adverbs, but it would take 4 different formal rules to cover all those variations. Hence, in order to allow capturing similar syntactic structures in one file, and, at once, to make a voluminous bank of manual rules more scalable and simple to understand, our graphical rule-editing tool and compilation component support a schematic representation of rules, called RULE SCHEMA, that is more abbreviated than our formal rules.

Figure 3 shows how we represent our active/passive conversion rule (discussed in Subsection 4.6), and demonstrates some of the main features that make the rule schema more compact than the formal rule representation.

At the top, the ruleType=substitution label means that this is a substitution rule, in contrast to an extraction rule (See Section 3.3). The two small nodes on the top labeled LHS and RHS are artificial nodes serving to point at the actual root nodes of the LHS and RHS, their two respective children. Notice how each LHS node must specify the relation (rel), POS tag (tag) and lemma (lemma) that it matches against. As exceptions, only the two roots lack a rel, and only variable nodes lack a lemma. There are also three alignment relations (arrows labelled copy), stretching between pairs of corresponding nodes from LHS to RHS. An alignment relation indicates that the contents of the matched text node, along with its unmatched children (those *not* represented in the LHS), will be copied to the RHS counterpart, when generating the consequent tree.

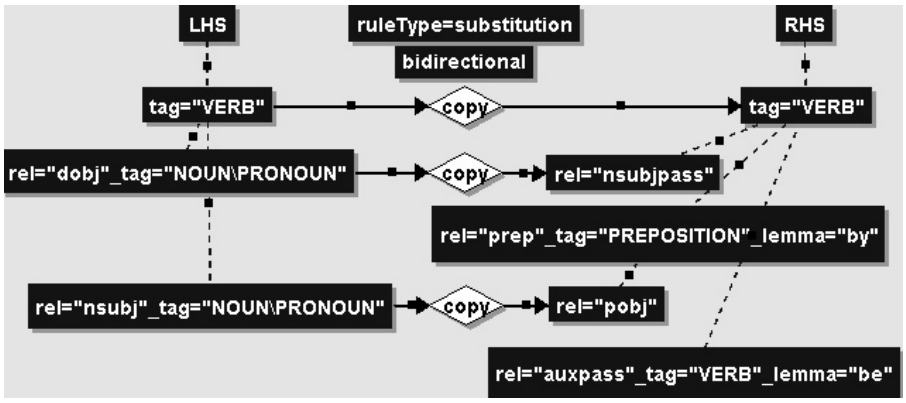


Fig. 3 Graphical representation of the active-passive rule schema. See elaborations in the text.

Now we turn to explain some of the traits of our compact rule schema. Formally, each RHS node must specify its rel, tag and lemma, like LHS nodes. However, in a schema RHS node, any one of these properties can be omitted, provided that it is specified in an *aligned* LHS node. In these cases, at compilation step, the missing properties in the RHS node are supplemented using the properties of the aligned LHS node. For instance, the two bottom RHS nodes that have alignments, do not specify their tag. Furthermore, the bidirectional label means that this schema actually represents two rules, one for each direction of entailment, and the compiler will produce the two respective unidirectional rules out of this one schema.

In addition, this rule exhibits multiple choice, in two of its nodes' properties, denoted by a '\' (meaning *OR*) between values, in order to compact several nearly identical sub-cases into a single representation. Specifically, since we know it is possible for the subject (nsubj) to be either a noun or a pronoun, e.g., ***Tim*** *wrote a paper and later* ***he*** *published it* , it follows that both options must be listed in the tag property of the LHS's subject node. The same goes for the direct object (dobj).

A multiple choice property value specifies two (or more) unambiguous rules, each containing only one of the listed options. It then follows that the given LHS subject and object nodes have the combined effect of defining 4 rules, one for each member in the Cartesian product of their multiple choice tag values. Together with the bidirectional property, this schema represents a total of 8 concrete rules.

In other cases, we may want an LHS node to match against not one or two POSs, but many, or even all, POSs. Therefore we defined a wildcard value: tag="*", shorthand for "all possible values". rel="*" is also possible. One additional feature of the rule schema is exemplified in Subsection 4.7.

Once a schema is completed manually, using the graphical editing tool, we run it through a compilation procedure that produces several formal rules, and outputs them in CoNLL format. Accordingly, throughout this section rules are described in terms of compact schemas. For further documentation and code, the reader should refer to the download package.

4.2 Complementizer Insertion/Deletion

(*Substitution*) Complementizers are used in English to open subordinate clauses, and they constitute a closed lexical class that can be contained in a short list (*that, who, when...*). For many clausal constructions, which can usually be straightforwardly identified in the parse tree, the complementizer can either be present (materialized) or absent (implied) - without altering meaning. This motivates five rules in this category that insert and delete complementizers, such as the following:

(1) Insert or delete any relative pronoun + *be* at a position introducing a reduced object relative passive clause. Relative pronouns include *that, which, who* and *whom*.

(a) *The horse* ***which was*** *raced past the barn fell*

⇕

(b) *The horse raced past the barn fell*

Notice that the entailment *The horse which was* ***being*** *raced past the barn fell* is also entailed by (1a), since our formalism ignores grammatical aspect.

Figure 4 shows the graphical representation of (1). The LHS tree is designed to match against sentences like (1a) using four nodes. The match criteria of the root LHS node consists solely of requiring that the part of speech (POS) tag be a (any) noun. Its child node requires a "relative clause modifier" (rcmod) syntactic relation to its parent, and must be a verb. The two children of the rcmod are not variables, but rather concrete words. The left child is either the word *who* or *which*, the only two possible relative pronouns for this structure. Accordingly, it takes a PRONOUN POS tag (which is the reduced category for 'WH pronouns', see Appendix B), and a nsubjpass relation, since it is

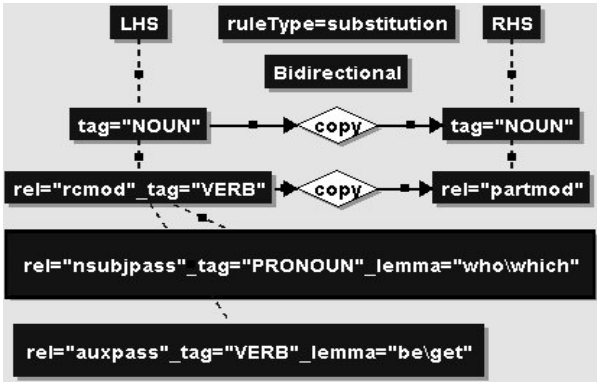


Fig. 4 The graphical representation of the reduced relative clause rule.

the nominal subject of a passive clause. Its sibling represents any of the two common English passive auxiliary verbs "to be" or "to get", with the auxpass relation.

On the other side, the RHS tree (which in this bidirectional schema also represents the LHS of the reversed rule) needs only two variable nodes to capture reduced relative clauses like in (1b). Its root node matches against any noun, and its child is a verb, which (according to the parser's analysis) heads a participle modifier clause node (partmod).

(1) also exemplifies caveats in the rule. For example, it is prone to creating many ungrammatical sentences from T, e.g.,

* *The horse* ***who was*** *raced past the barn fell*

Nonetheless, it is almost guaranteed that these would never be matched against a given H, taken from a natural language corpus. Therefore, it seems beneficial to accept such caveats, for the sake of better coverage. We employ a similar rationale in other rules, where we prefer to improve coverage at the cost of generating ungrammatical constructions.

Similar rules in this category were defined with other complementizers, such as *that, when, whom*, in similar constructions, with or without the *be* auxiliary.

4.3 Coordinations

(*Substitution*) The coordination (conjunction) structure coordinates two or more syntactically equivalent phrases into a single phrase. For instance, the sentence

He likes her and she likes him

exhibits a coordination of two clausal phrases (CPs). In the Stanford dependency standard, conjunction words (*and, or, but...*) are all identifiable by their CC relation.

Conjunct Deletion Semantically and syntactically speaking, each conjunct is (usually) omissible, say in:

Two students and one teacher sing $\Rightarrow$ *Two students sing*

Logically, this reasoning is wrong in many cases, for example:

Exactly two students and one teacher sing $\not\Rightarrow$ *Exactly two students sing*

However, currently we cannot distinguish such cases, because we use syntactic representations that do not handle semantic phenomena like quantification and scope. On the other hand, an inspection of occurrences of coordinations in TE datasets (in the preliminary analysis and in other sources) shows that problematic cases are scarce, and therefore conjunct deletion is expected to be predominately beneficial as is. Our assumption of high empirical accuracy, for these and all other rules, is tested and validated in Subsection 5.2.2.

Motivated by this observation and by similar work by Bar-Haim et al (2009), we specified several rules that delete any one conjunct in any distinguishable level of coordination - clausal, verbal, nominal, prepositional, adjectival and adverbial.

Conjunct Swapping Much like deleting a conjunct, in most cases swapping two conjuncts preserves entailment, for instance,

(2a) *Tom stayed home, but Jerry went out* $\Leftrightarrow$ *Gerry went out, but Tom stayed home*

but not

(2b) *Tom got lost, but Gerry found him* $\not\Leftrightarrow$ *Gerry found him, but Tom got lost*

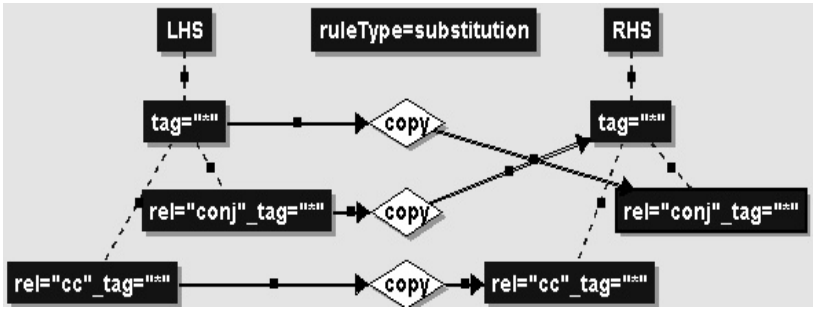


Fig. 5 The "swap conjuncts" rule, in its simple variation. The POSs are not important to this pattern, so all the rule's nodes use the `tag="*"` wildcard.

We still assume that this operation is empirically beneficial, for motivations similar to the previous case, and developed three rules to address it. Each of these rules is tailored to a different variant of coordination.

Figure 5 shows the first of the three. It addresses the structure of (2a), which is the simplest. To understand it, it is important first to note the way the Stanford dependencies standard represents coordinations. This is reflected in the three nodes of the LHS tree, as follows:

a) A node representing the first conjunct, with two children, defined in b) and c).
b) A child with the `cc` syntactic relation, stating that it is the conjunction function word.
c) A child with the `conj` relation. This is the second conjunct. If there are further conjuncts, each is represented by another such child of the first conjunct.

At first the LHS of the rule attempts to match against a coordination in the text, i.e., against the first conjunct and any one of the others. Afterwards, the rule swaps the positions of the two conjuncts in the RHS. If a coordination in a given text has three or more conjuncts, then matches of successive applications of the rule can yield all possible permutations of conjuncts.

4.4 IS-A Implications

(*Extraction*) This section presents extraction rules that extract IS-A relations out of T. Such canonical IS-A relations are straightforward to exploit in many

settings. What's more, in the RTE datasets in particular, many Hs correspond to IS-A extractions[5].

Implied IS-A relations with Hearst Patterns The 10 rules in this group are adapted from Hearst (1992) and Pantel et al (2004). Each describes a composite noun phrase (NP) structure containing at least two NP components, extracts them, and uses them in a new (possibly reversed) copular sentence. The NP structures are:

1. NP1 such as NP2
2. Such NP1 as NP2
3. NP1 or other NP2
4. NP1 and other NP2
5. NP1 known as NP2
6. NP1 especially NP2
7. NP1 like NP2
8. NP1 including NP2
9. NP1 is a NP2
10. NP1 a NP2

For instance, applying the rule of pattern no. 1 will yield extractions like:

I like builders such as Bob $\Rightarrow$ *Bob is a builder*

Apposition Notice that the last Hearst pattern describes the generic apposition structure, labelled by the parser with the syntactic relation appos, and dealt with in this rule:

(3) Apposition to copula - extract an NP and its apposition to an independent IS-A construction, in **both** orders.
   (a) *EU enlargement commissioner lent support to* ***prominent Turkish novelist, Orhan Pamuk***
   $\Downarrow$
   (b) *Orhan Pamuk is prominent Turkish novelist*
   $\Downarrow$
   (c) *Prominent Turkish novelist is Orhan Pamuk*

[5] We note that this might be an artefact of the choices made by the RTE dataset annotators, when manually picking the $T - H$ examples

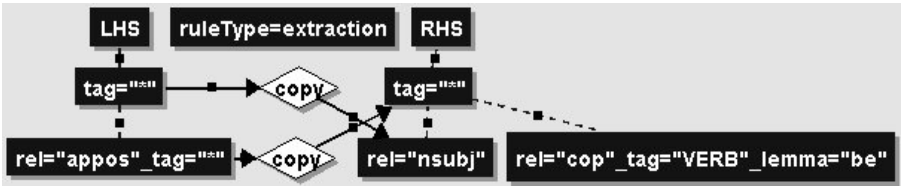


Fig. 6 Graphical representation of the apposition to reversed-copula rule. The ruleType = extraction label says this is an extraction rule. An apposition structure is identified straightforwardly, by a child node with the appos relation (in the LHS). Because there is no need to inspect the POSs, most nodes have a wildcard * tag.

This rule was also motivated by the the preliminary manual analysis.

In the above example, the indefinite determiner *a* is missing from the two entailments. This is because our generic formalism considers lemmas instead of surface forms, which means an **a** and an **an** are indistinguishable, as are cases where the NP is plural and has no determiner. Therefore, determiners are never generated in the first place.

Figure 6 shows the apposition rule, for the case of the reversed order of the copular sentence, i.e., (3a) ⇒ (3c). The RHS in the figure describes a canonical copuar sentence, like (3b) and (3c). Here, the LHS main NP (with tag="*") becomes the RHS subject, the LHS apposition becomes the main predicate, and it's accompanied by a *to be* copular (cop) verb.

Other substitution rules in this category swap between an NP and its apposition, or delete an apposition, e.g.:

- *EU enlargement commissioner lent support to* ***prominent Turkish novelist, Orhan Pamuk*** ⇔ *EU enlargement commissioner lent support to* ***Orhan Pamuk, prominent Turkish novelist***
- *EU enlargement commissioner lent support to* ***prominent Turkish novelist, Orhan Pamuk*** ⇒ *EU enlargement commissioner lent support to* ***Orhan Pamuk***

4.5 Possessive Constructions

(*Substitution and extraction*) There are two popular constructions in English to express possession. One uses an apostrophe-*s* marker, as in *a plane's wings*, while another can be called an "of-construction", as in *the wings of a plane*.

The two are usually semantically equivalent, so we developed a bidirectional substitution rule to convert one to another, shown in (4).

(4) Substitute an apostrophe-*s* construction with an "of-construction" and vice versa.
   (a) *Pamuk is Turkey**'s** best known writer* $\Longleftrightarrow$ *Pamuk is the best known writer **of** Turkey*

In many cases, the same "of-construction" can be taken to have adjectival meaning, rather than possessive, as in

*The flood **of a** coal mine killed 10 in China* $\Longleftrightarrow$ *The **coal mine flood** killed 10 in China*

Another rule, with a compound rather than a possessive construction, performs this transformation.

In addition, we notice that a possessive construction immediately implies a HAS-A relation between its two components, similarly to the way an apposition implies an IS-A relation. So we developed an extraction rule to extract independent HAS-A statements out of apostrophe-*s* constructions, shown in (5) below. Notice how a similar HAS-A extraction from an "of-construction" is redundant, since an inference engine is expected to be able to perform (4) and then (5) sequentially.

(5) Extract both noun components of an apostrophe-'s' construction to an independent HAS-A copula.
   (a) *Pamuk is Turkey's best known writer* $\Rightarrow$ *Turkey has a best known writer*

4.6 Sentential Constructions

(*Substitution*) This subsection covers a relatively large number of bidirectional substitution rules, converting between diverse sentential constructions: active vs. passive, adjectival constructions, adverbial constructions and cleft constructions. These rules are aimed to convert some common constructions to and from the canonical subject-verb-object construction, where possible. This kind of "star topology" design, that maintains each construction at one step of entailment application away from its most canonical counterpart, minimizes and simplifies the chain of rule applications required to prove any complex inference.

Furthermore, most of the entailment rules described here have only theoretical motivation, and, to the best of our knowledge, have little or no empirical justification in the current RTE datasets. As an exception, the first rule does occur frequently, according to our preliminary data analysis.

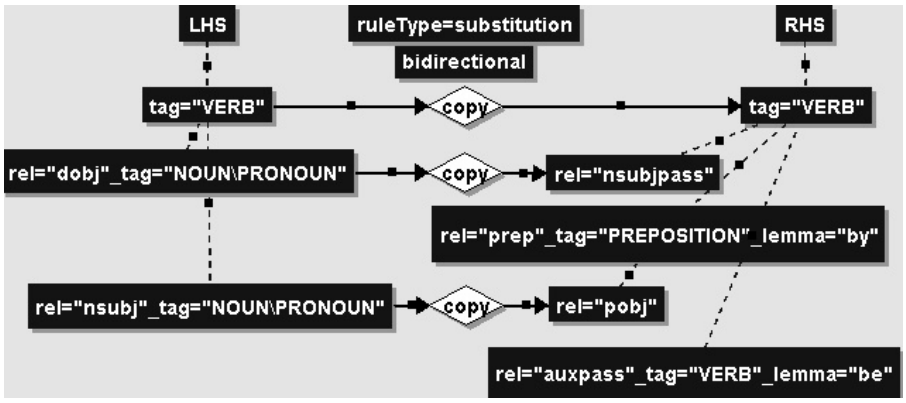


Fig. 7 Graphical representation of the active-passive rule schema. It has multiple choice in two properties, in order to compact several nearly identical sub-cases into one representation.

Active-Passive Rule This rule converts between active and passive voice, e.g.:

(6) *Some DNA evidence solved the case* $\Longleftrightarrow$ *The case was solved by some DNA evidence*

Notice there are many verbs for which the same transformation generates ungrammatical or unintelligible sentences, e.g.:

(7) *Lee escaped the police* $\Longleftrightarrow$ **The police was escaped by Lee*

Still, like in the discussion about the "delete conjunct" rule in Subsection 4.3, we prefer to have rules that have broader coverage, even if they produce ungrammatical sentences, since this almost always does not cause false positive entailments empirically. The rule schema is shown in Figure 7.

Adjectival Constructions The adjectival rules are an adaptation of work by Amoia and Gardent (2008), where they formulate dozens of syntax-based, lexical syntactic and semantic inference rules, all involving adjectives. We implemented and enhanced 9 of those cases that fit into our generic syntactic level framework. Like other rules in our study, these are not meant to be linguistically accurate, but rather are only expected to generate valid entailments on most RTE-style texts.

Two of the rules are:

(8) Predicative Attributive Construction
This NP is Adj $\Longleftrightarrow$ This is Adj NP
(a) (adjective) *This table is red* $\Longleftrightarrow$ *This is a red table*

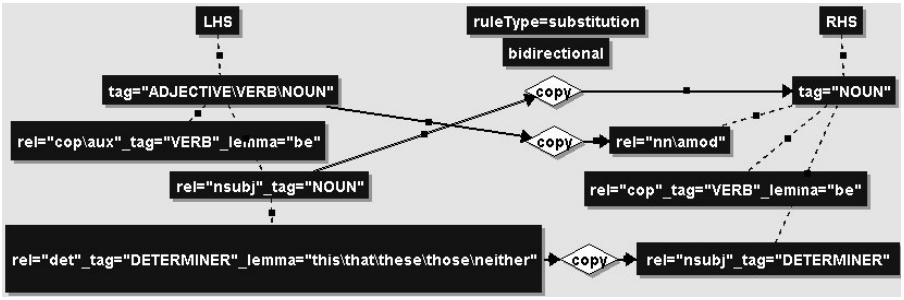


Fig. 8 Graphical representation of the predicative attributive rule

(b) (gerund) *This chair is rocking* $\Longleftrightarrow$ *This is a rocking chair*

(c) (noun) *This horse is a gift* $\Longleftrightarrow$ *This is a gift horse*

(9) Tough-Construction

NP is Adj to V $\Longleftrightarrow$ V NP is Adj $\Longleftrightarrow$ It is Adj to V NP

(a) *John is easy to please* $\Longleftrightarrow$ *Pleasing John is easy* $\Longleftrightarrow$ *It is easy to please John*

Figure 8 defines a generalized rule that captures (8a), (8b) and (8c). It exhibits a few features common to many other rules in this subsection, meant to maximize the language diversity of matched texts, while obtaining a compact representation and minimizing false entailments.

First, note that the adjectival pattern in (8a) does not seem to match against examples (8b) and (8c), which are verbal and nominal sentences (and note that, according to the Stanford dependencies representation, the head of a nominal/copular clause is the predicate, dominating the copular verb). To accommodate for texts like this, we defined the POS tag of the root of the LHS node to be a choice between adjective, verb and noun, by setting tag="ADJECTIVE\VERB\NOUN". Like in the active-passive rule, this node's mapped RHS counterpart has no explicit tag value, so that the tag is copied through the alignment at compilation time. Notice the root node of the RHS must remain NOUN in all these examples.

Another general aspect of rule design that is demonstrated here, is that loosening the match criteria of one node may require loosening other nodes as a result. In this case, comparing the parse trees of the above three examples, we noticed that, according to the Stanford dependency relations standard, the relation of the auxiliary verb *to be* is cop (copular) in adjectival and nominal sentences, while in verbal sentences it aux (auxiliary). Hence the rel="cop\aux" property in the auxiliary verb node in the LHS. The RHS's auxiliary verb node

only has the rel="cop" option, because the root of the nominal copular sentence is always a noun.

For similar reasons, the RHS node mapped to the LHS root has multiple choices of syntactic relation, between noun modifier and adjectival modifier, set as rel="nn\amod". The first option, when combined with the NOUN tag, matches parse trees of nominal examples like (8c). The second option, when combined with ADJECTIVE or VERB, matches parses of (8a) or (8b) type, respectively.

As the last point of interest in this rule schema, notice that the bottom node of the LHS has a multiple choice of words, covering the closed set of demonstrative pronouns, set by lemma="this\that\these\those\neither", reflecting the fact that any demonstrative can fit into this inference pattern.

In summary, this use of multiple value choice yields a compact representation of several similar rules into one manageable schema. Unfortunately, it also means that some anomalous rules will be compiled, for instance, one with a node that is both a nominal modifier by its syntactic relation and a verb by its POS tag. Luckily, in this case we can neglect the effect of such rules, because they can only be matched against specific ungrammatical texts, which are scarce in natural language datasets.

An equivalent rule exists in the rule-base for the Tough-Constructions described in (9).

Adverbial Constructions This subsection also covers a set of 4 novel bidirectional adverbial rules, inspired by the above adjectival rules, which convert between sentential constructions involving adverbs. Some of these rules convert parts of speech, between corresponding adjectives and adverbs, for instance:

(10) Passive Adverbial

NP is Adv $V_{passive}$ $\Longleftrightarrow$ NP is Adj_{adv} to V

(a) *John is easily pleased* $\Longleftrightarrow$ *John is easy to please*

In (10), Adj_{adv} denotes the adjective derived from the corresponding adverb. The seemingly lexical derivation between adverbs and adjectives is possible in a generic syntax-based knowledge resource, without the aid of a lexicon. This is done by exploiting a generalized assumption that, in most of these derivations, the form of an adverb differs from its corresponding adjective solely in its POS tag and the *-ly* suffix of its lemma.

The other 3 rules are:

(11) *In* Construction
NP V Adv $\Longleftrightarrow$ NP is Adj_{adv} in Ving
(a) *John cooks splendidly* $\Longleftrightarrow$ *John is splendid in cooking*

(12) Predicative Attributive Construction
NP_{subj} V NP_{obj} Adv $\Longleftrightarrow$ NP_{subj} is Adj_{adv} to V NP_{obj}
(a) *John took this job foolishly* $\Longleftrightarrow$ *John was foolish to take this job*

(13) Passive Construction with Subject
NP_1 Adv V NP_2 $\Longleftrightarrow$ NP_2 is Adv $\text{V}_{passive}$ by NP_1 $\Longleftrightarrow$ NP_2 is Adj_{adv} for NP_1 to V
(a) *Mary easily pleases John* $\Longleftrightarrow$ *John is easily pleased by Mary* $\Longleftrightarrow$ *John is easy for Mary to please*

Cleft Constructions The last contribution to this subsection is based on some of the substantial literature available on cleft constructions[6]. It contains 6 rules that substitute well known cleft constructions with the canonical subject-verb-object (SVO) construction, and vice versa, presented here:

(14) WH-Cleft
(a) *What he wanted to buy was a Fiat* $\Longleftrightarrow$ *He wanted to buy a Fiat*

(15) Reversed WH-Cleft/Pseudo-Cleft
(a) *A Fiat is what he wanted to buy* $\Longleftrightarrow$ *He wanted to buy a Fiat*

(16) It-Cleft
(a) *It was Henry that kissed Rosie* $\Longleftrightarrow$ *Henry kissed Rosie*

(17) All-Cleft
(a) *All he wanted to buy was a Fiat* $\Longleftrightarrow$ *He only/just wanted to buy a Fiat*

(18) Inferential Cleft - Negative
(a) *It is not that he loves her* $\Longleftrightarrow$ *He doesn't love her*

(19) Inferential Cleft - Positive
(a) *It's just that he loves her* $\Longleftrightarrow$ *He loves her*

4.7 Extracting Relative Clauses

(*Extraction*) This subsection discusses a few rules that extract relative clauses to independent sentences.

[6] For example: http://en.wikipedia.org/wiki/Cleft sentence

It is well known that every *definite* noun phrase (NP) is *presupposed*, i.e., NPs modified by *the, this, of* etc. This signifies that the existence of every definite NP is entailed from the text, and, in case the NP has a relative clause, the clause's body is entailed along with the modified NP as its subject or object. This kind of entailment persists even if the external argument is negated (*not, never...*), is questioned (*perhaps, might...*) or is conditioned (*if X then...*). For example:

(20) *Isn't* ***the*** *beer that you brought cold?* $\Rightarrow$ *You brought beer*

In contrary, most indefinite NPs are not presupposed, for example:

(21) ***A*** *dollar saved is a dollar earned* $\not\Rightarrow$ *A dollar was saved*

Ideally, we wish to form rules that extract relative clauses to independent sentences, like in (20), but not like (21). In practice, although we could form a few rules that detect definiteness of nouns, it is impossible to combine them all into one rule that also performs the clause extraction. Furthermore, under our formalism, we have neither the tools for annotating a feature like definiteness, nor can we form an entailment rule that matches against such an annotation.

So, instead of abandoning these rules altogether, while not breaching our simple formalism, we incorrectly assume in these rules that *all* relative clauses are presupposed. This assumption is justified by extensive informal empirical tests on RTE datasets, and also by the formal rule-base evaluations, reported in Section 5.

According to the Stanford dependency standard, most relative clause constructions (like in (20)) can be characterized by three nodes:

1. The modified external noun
2. The main predicate of the clause, with an rcmod (relative clause modifier) relation to the external noun
3. (optional) The complementizer (*which, who* or *that*), with a subject or object relation to the clausal predicate, which is the same relation that should hold in the extracted entailment, from the noun to the predicate

In contrast, reduced participle relative clauses (like in (21)) are characterized by the partmod (participle modifier) relation from the clausal predicate to the modified noun.

The first rule, that covers many cases of relative clause constructions like in (20), where the modified noun is either the subject or the direct object (with no prepositions) of the external clause, is illustrated in Figure 9. The figure

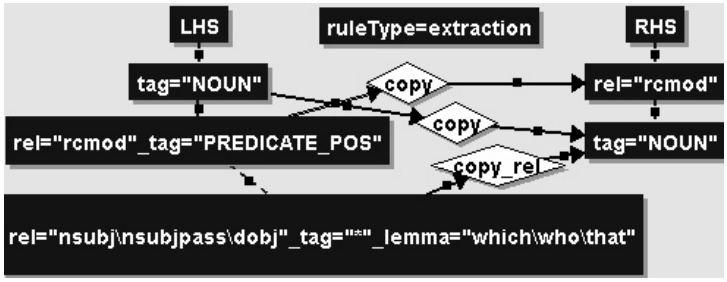


Fig. 9 Graphical representation of the main relative clause extraction rule. In the second LHS node from the top PREDICATE_POS is an idiom for the multi choice of all possible predicate POS: "VERB\NOUN\PRONOUN\ADJECTIVE". The copy_rel label in the bottom diamond is not an alignment (used at the RHS instantiation step), but rather it specifies that its LHS node's rel (after expanding its multiple choices) is copied to the RHS node at rule compilation stage. This creates an unambiguous formal rule with just that rel in both nodes.

warrants a few observations. In the LHS, the top node is the head noun that is modified by the relative clause. The second node is the main predicate of the clause, using the idiom PREDICATE_POS that stands for the multi choice of all possible predicate POS: "VERB \ NOUN \ PRONOUN \ ADJECTIVE". The third is the complementizer of the clause, which, in this type of relative clause, can be either the subject, passive subject, or direct object of the predicate. Also, as mentioned above, this syntactic relation determines the relation from the RHS noun to its predicate, while that RHS noun is aligned to a different LHS node (the head noun). To implement this requirement, the relation is copied to the RHS noun without alignment, at compilation time, over the copy_rel labeled arrow.

With this last feature we can summarize three different ways, by which the properties of a schema RHS node are set at compilation stage, in decreasing order of precedence:

1. property values are copied through a copy_rel, copy_tag or copy_lemma arrow (without alignment)
2. parameter values are written explicitly in the RHS node (as in most figures in this section)
3. parameter values are copied from an aligned LHS (with a COPY arrow)

Notice that this rule, like others in this category, does not cover many cases, in which the complementizer is not materialised, like in entailing (c) directly from (a) here:

a. *I would like that book on the top shelf*
   ⇓

b. *I would like that book* ***that is*** *on the top shelf*

⇓

c. *That book is on the top shelf*

This is deliberate, since complementizer insertion and deletion is covered independently by the rules in Subsection 4.2. Therefore, for entailments such as this, it is necessary to apply an intermediate complementizer insertion rule in (b).

Another rule extracts clauses out of the reduced relative clause construction, e.g.

Roberta found the pack, ***hidden*** *underneath the morning paper*

4.8 Substituting Determiners

(*Substitution*) This subsection features 8 rules that convert determiners, *a, the, some, not all, other, seven, Tom's, etc.* Most entailments between determiner phrases are unidirectional, and most of the rules had at least some motivation in our preliminary data analysis. To our experience, the most widely applied rule from this group is:

(22) Substitute any of the determiners *the, that, this, other, another, these, those, the other* and *some* with *a*

(a) *Pluto is like* ***other*** *Kuiper Belt objects* ⇒ *Pluto is like* ***a*** *Kuiper Belt object*

The next rule might seem to generate bad sentences in practically every application:

(23) Substitute *an* with *a*, and vice versa

(a) *Pluto is like* ***a*** *Kuiper Belt objects* ⟺ *Pluto is like* ***an*** *Kuiper Belt object*

However, since swapping between these two indefinite articles changes surface form but not meaning, this rule is useful in many common use cases, where a vowel-initial noun is replaced by a consonant-initial noun, thus raising the need to apply this rule. One such use case is when employing our resource in tandem with a lexical-based rule-base (such as Wordnet), as in:

But Brutus is an honourable man ⇒ *But Brutus is* ***an respectable*** *man*

Here our generic rule is beneficial to correct the error created by the lexical rule. Another common use case, relevant to many inference systems, is when T and H have different indefinite articles, and the system must assess the similarity between them, e.g.:

T *But Brutus is an honourable man*
H *But Brutus is a respectable man*

In this case, just applying our rule may be is beneficial in making the two propositions more alike, without altering the meaning or employing other knowledge resources.

4.9 Case Correction

(*Substitution*) The need for the next two rules follows from many pronoun case errors created by other rules, e.g., the following error can be caused by an application of the active-to-passive rule:

They were escorted by the police $\Leftrightarrow$ *The police escorted* ***they***

This was also noted by Bar-Haim et al (2007), who first formed these rules that detect incorrect pronoun case, and correct it, by substituting the appropriate lemmas. Since the closed set of pronouns is the only class of English nouns with visible case marking, the rules can be implemented generically, without the aid of a lexicon.

(24) Substitute an accusative pronoun in subject position with its nominative counterpart
   (a) *Her took the car* $\Rightarrow$ *She took the car*

(25) Substitute a nominative pronoun in object position with its accusative counterpart
   (a) *The car was taken by she* $\Rightarrow$ *The car was taken by her*

5 Evaluation and Analysis

We analysed and evaluated our approach and the new resource in three major aspects: 1) the potential impact of a resource of this type in the RTE datasets (Subsection 5.1); 2) the resource's quality in terms of recall and precision (Subsection 5.2); and 3) its impact on a particular inference system (Subsection 5.3). To address these three aspects, we conducted a series of analytical and empirical experiments.

5.1 Manual Analysis of a Syntax-Based Resource Potential

Our first analysis estimates the potential contribution of generic syntax-based inference rules to the TE task, i.e., how common are generic syntax-based inference phenomena in a manually computed "gold standard" of $T-H$ proofs (in opposed to imperfect machine-aided proofs). In addition, we would like to give a quantitative prediction as to the relative usefulness of different kinds of rules.

We conducted our analysis over the most recent RTE dataset that was available to us at the time, RTE6 Dev (Bentivogli et al, 2010b), as RTE is the main benchmark for TE technology. For the purposes of this analysis, it suffices to describe RTE as a collection of several thousand T - H pairs. Most are not entailing (negative pairs), while about 5.6% of the pairs are positive.

We randomly sampled 50 positive T - H pairs, and manually constructed their "gold standard" entailment proofs as a sequence of transformations that transforms T to H, following the transformation-based inference paradigm of Stern and Dagan (2011), which follows Bar-Haim et al (2007) in its use of entailment rules. Each proof consists of a chain of entailments preserving atomic transformations, applied to the syntactic parse tree of T, ending in a tree identical to H's tree (exact match). See example proof in Table 2.

The range of linguistic transformations we allowed ourselves to use can be broadly categorized according to their operational mechanism (*substitution* vs. *extraction*), and by the linguistic knowledge they use.

In terms of linguistic knowledge, a rule may rely either on syntactic knowledge (i.e., the rules described in this work), lexical or lexical-syntactic knowledge, on coreference resolution (replacing an anaphor with its antecedent) or bridging. Bridging is still a vague term in the literature, and we take it to mean transformations that introduce a subtree, copied from another tree within the context of T, which is implicitly referred to within T, see (Mirkin et al, 2010).

An example for each rule category we employed is given in Table 3. For convenience, we demonstrate these categories with plain text and not with parse trees.

In cases where we could not come up with a plausible full proof for a pair, using the above transformation-based methodology, it was marked "hard", set aside, and replaced by a new random positive pair, so that we ended up with 50 full proofs. Overall, we assessed 7 pairs to be "hard", and the following results refer to the remaining 50 fully annotated pairs.

Table 2 An example manually derived proof for a $T - H$ pair, from the RTE6 Dev main task dataset

H		**Original T**
Jill Carroll was seized by gunmen		Carroll's driver, quoted in a story posted on the Monitor's website, said gunmen jumped in front of the car, pulled Carroll from it, and drove off with their two captives all within 15 seconds
Rule	**Rule Category**	**Entailed T**
local coreference substitution	Coref	Carroll's driver, quoted in a story posted on the Monitor's website, said gunmen jumped in front of the car, pulled **Jill Carroll** from it, and drove off with their two captives all within 15 seconds
delete verbal conjunct	*Syntactic* Coordination	Carroll's driver, quoted in a story posted on the Monitor's website, said **gunmen pulled Jill Carroll** from it, and drove off with their two captives all within 15 seconds
pulled X from Y $\rightarrow$ *seized X*	Lexical Syntactic Substitution	Carroll's driver, quoted in a story posted on the Monitor's website, said **gunmen seized Jill Carroll**, and drove off with their two captives all within 15 seconds
active to passive	*Syntactic* Active-Passive	Carroll's driver, quoted in a story posted on the Monitor's website, said **Jill Carroll was seized by gunmen**, and drove off with their two captives all within 15 seconds
say X $\rightarrow$ *X*	Semantic Extraction	Jill Carroll was seized by gunmen

5.1.1 Usage of Syntax-Based Rules as a Whole

We first investigate the overall need for syntax-based operations by counting the $T - H$ pair proofs that contain syntax-based transformations, and how many such operations each proof contains.

Table 4 presents the distribution of the pairs according to the number of syntax-based rule applications included in their proofs (57 applications in total). We see that the majority of the pairs (78%) did contain at least one syntax-based application. On average, each pair required 1.16 transformations.

Table 3 Examples for inference rules and their application, one from each category used in the dataset annotation. T_1 is the text before application and T_2 is the entailed text.

Category	Example Rule	Text	
Syntactic Substitution	Swap a Noun with Its Apposition	T_1	My friend, Ben, is coming over
		T_2	Ben, my friend, is coming over
Syntactic Extraction	Extract Apposition to Copula	T_1	My friend, Ben, is coming over
		T_2	Ben is my Friend
Lexical Substitutions	*begin* $\Rightarrow$ *launch*	T_1	A peace process was **begun** by Pakistan and India
		T_2	A peace process was **launched** by Pakistan and India
Lexical Syntactic Substitution	*suffer from X* $\Rightarrow$ *have a X*	T_1	Peter Jennings announced that he **was suffering from** lung cancer
		T_2	Peter Jennings announced that he **had a** lung cancer
Lexical Syntactic Extractions	*in connection with X* $\Rightarrow$ *there was X*	T_1	Bagri was arrested **in connection with** a blast at the airport
		T_2	There was a blast at the airport
Semantic Extraction	*say that X* $\Rightarrow$ *X*	T_1	The paper **said** the economy is recovering
		T_2	The economy is recovering
Coreference Resolution and Bridging	Coreference Substitution	T_1	**The two countries** have been to war three times
		T_2	**Pakistan and India** have been to war three times

This analysis shows that generic syntax-based rules play a substantial part in the entailments of RTE datasets.

5.1.2 Profiling of Rules by Phenomena

In addition, we analysed the distribution of syntax-based rule applications over syntactic phenomena, to determine which rules are applied more frequently, and which are rare. To this end, we used the manual analysis to sum up the number of rules contained in the manually conducted proofs, grouped by syntactic phenomena.

The resulting distribution is given in the second column of Table 5, and it shows that some rule categories are used much more than others, while neither one is dominant. The third column is discussed in Subsection 5.3.

Table 4 The distribution of the solved T - H pairs, according to the number of syntax-based transformations each proof contains.

# Syntax-based Operations in Pair	# Pairs
None	11 (22%)
1	24 (48%)
2	12 (24%)
3	3 (6 %)

Table 5 Distribution of generic rule applications, grouped by syntactic phenomena, as used in the manual analysis, and by the BiuTee entailment system (see discussion in Subsection 5.3). The figures are percentages of the total syntax-based applications.

Syntactic Category	Manual Analysis	BiuTee Usage
Apposition	26.3%	6.7%
Relative Clause	19%	3.7%
Determiners	19%	26.3%
Possessives	14%	16.9%
Active/Passive	12.3%	35.9%
Coordination	5.3%	10.2%
Adjectival	3.5%	0%
Case Correction	0%	0.2%
IS-A implications	0%	0.1%

5.2 Resource quality

While the previous subsection evaluated the potential for our resource in the dataset, this subsection assesses its quality in terms of *recall* and *precision*. By *recall* we mean the number of syntax-based rules that are covered by the resource, out of all the syntax-based rules applied in the manual analysis. We also tested the *recall of the formalism*, which is the number of syntax-based rules permitted in the formalism (defined in Section 3), out of all syntax-based rules in the manual analysis. By *precision* we denote the probability that applying a syntax-based rule from the resource preserves entailment.

5.2.1 Recall

The recall measures are shown in Table 6. In total, we find that 30% of the listed applications require syntax-based rules which are not available in the

Table 6 Recall for each investigated syntactic category, according to the manual dataset analysis

Syntactic Category	# Apps in Manual Analysis	# Apps Covered by Formalism	# Apps Covered by Rule-base (Recall)
Apposition	15	15	13 (87%)
Relative Clause	11	11	7 (63%)
Determiners	11	11	7 (63%)
Possessive	8	8	3 (37%)
Active/Passive	7	7	7 (100%)
Coordination	3	3	3 (100%)
Adjectival	2	2	0 (0%)
Total	**57**	**57**	**40 (70%)**

rule-base, though all are allowed by the formalism. Therefore, the recall for the rule-base is estimated at 70%, while the recall for the formalism is estimated at 100%, for this sample.

Only 15 missing rules accounted for this gap, and we estimate that most of these are relatively rare and specific to this dataset, in comparison with the bulk of our existing rules, whose use is more widespread. Also, in Subsection 5.3.2 we demonstrate why the rest are likely to generate more false entailments than valid ones. For these reasons, we estimate it would require a significant effort, and dozens of new rules, to raise the recall measure on most similar datasets.

5.2.2 Precision

We judge a given rule application to preserve entailment, if the underlying meaning of the tree generated by this application is entailed by the underlying meaning of the source tree, on which it was applied. To this aim, we randomly sampled 100 syntax-based rule applications from our resource, which were performed by a concrete RTE system, BiuTee (Stern and Dagan, 2011), whose operation is described in Appendix C.

To produce the 100 samples of syntax-based rule applications for this analysis, we ran BiuTee on the RTE6 Test dataset, in one of its best configurations

Table 7 Precision (entailment preservation) analysis of a sample of 100 rule applications

Correct Applications		94%
Incorrect Applications		
	Rule Caveats	3%
	Parser Errors	3%

(Stern and Dagan, 2011), along with our syntax-based resource[7]. This configuration includes two knowledge resources, Wordnet (Fellbaum, 1998; Miller, 1995) and Directional Similarity Kotlerman et al (2010).

Out of the log of all rule applications in all of BIUTEE's proofs for the $T-H$ pairs in the test set, we randomly extracted 100 syntax-based rule applications, 50 from proofs of positive pairs, and 50 from negative pairs. These include both correct and incorrect proofs that the system generated, because we need to test for rule application precision in both contexts. We then judged whether each sampled rule application preserves entailment. Note that the more commonly used rules are more likely to be sampled and evaluated, and that this bias reflects how the resource is used in practice.

The results are shown in Table 7. Notice that the vast majority (94%) of rule applications are correct, i.e., their application preserves entailment. Of the few remaining errors, 3% are due to caveats in the rules themselves, mostly having to do with limitations of the formalism.

One such case involved the pair:

(26) T: *Kashmir is divided between India and Pakistan,* ***but*** *both claim the region in its entirety*

H: *Kashmir is claimed in full by India and Pakistan*

where T has a coordination construction at the matrix clause level. On this coordination, BIUTEE applied a rule that matches correctly to it, but which is meant to be applied only on *verbal* coordinations, where the coordinated elements share a single subject. The rule first swaps between the two elements, and then reattaches the first element's subject to the second one, e.g., *Kids laugh and cry easily* $\Leftrightarrow$ *Kids cry and laugh easily*. Since it was applied to a clausal coordination, it yielded an ungrammatical and non-entailed sentence:

[7] We excluded "Case Correction" rules (Section 4.9), because we do not consider case alteration to change meaning, i.e., *he* and *him* are equivalent in this analysis.

**Kashmir both claim the region in its entirety, but is divided between India and Pakistan*

With this and several similar rules, we accept such caveats in the rule's precision, in favour of higher coverage.

Furthermore, an additional 3% were errors caused by the parser, which produced ungrammatical or improbable parse trees, causing unexpected results when rules are applied over such erroneous parses.

In summary, in this subsection we have assessed the recall and precision of the resource at 70% and 94% respectively. This reassuring result stresses the good performance of the rule-base.

5.3 Impact on BiuTee

In the previous two subsections we discovered a high potential for syntax-based rule-base resources in the RTE task, and assessed the quality of our resource's recall and precision. This subsection investigates to what degree does a state-of-the-art RTE system, namely BiuTee, takes advantage of the resource's potential.

The data used hereafter were obtained from two runs of BiuTee: a test run, configured the same as in Subsection 5.2.2, and a baseline run, without the use of our resource, for comparison.

5.3.1 BiuTee *Usage*

In order to enhance the rule profiling results in Subsection 5.1.2 with an empirical parallel, we collected usage counts of the resource's rules from BiuTee's application logs. These results are given in the third column of Table 5. They represent a distribution similar to that of the manual analysis, in that no syntactic category accounts for most of the applications, and the fact that the *adjectival, case correction* and *IS-A implications* have negligible weight, and *determiners* and *possessives* have similar large weights. In the following subsection, we investigate and explain the more significant differences in the remaining categories, *apposition, relative clause* and *coordination.*

5.3.2 BiuTee*'s Utilization of the Rule-Base*

Subsection 5.1 provided us with an estimation for the potential usefulness of a syntax-based rule resource on a given dataset, in terms of the rules' usage

rates. Now we compare this estimation with the way BIUTEE actually applies rules from our resource, on the same dataset.

First we investigate the *coverage*, i.e., to what degree did the system reproduce the syntax-based applications used in the manual analysis, on the same $T - H$ proofs. This experiment may help reveal system components which hamper better use of the rule-base, and problems with the rules themselves.

Second, we investigate the way BIUTEE used the rules, as follows. Originally, each rule was designed to address a particular syntactic phenomenon. However, we observed that BIUTEE sometimes applies a rule, in cases where a human would judge that the phenomenon takes no part in the needed inference, like in the following example:

(27) **H**: *Ajaib Singh Bagri is **a** mill worker*

T: *...Ripudaman Singh Malik, 57, and Kamloops, British Columbia mill worker Ajaib Singh Bagri, 55, are charged with multiple counts of conspiracy ... in **the** world's worst airline terrorism act ...*

Rule: Substitute determiners, *the* → *a*

T': *...Ripudaman Singh Malik, 57, and Kamloops, British Columbia mill worker Ajaib Singh Bagri, 55, are charged with multiple counts of conspiracy ... in **a** world's worst airline terrorism act ...*

In this case the system chose to apply a rule that substitutes determiners, and later to move the generated article *a* to another part of the sentence, to make T resemble H more, but it did so without any linguistic justification.

Coverage Referring back to the manual analysis data of Subsection 5.1, we checked, for each syntax-based rule application, whether it was also applied by BIUTEE in the same proof. In those cases where BIUTEE did not use the predicted rule application, we documented the reason for the transformation's omission, based on the alternative proof that the system had found.

In the first line of Table 8 we see that in total, BIUTEE reproduced only about 26% of the expected syntax-based rule applications (15 applications). The rest of the table presents a breakdown of the reasons for transformation omissions, which explain the low coverage in the first row. Examples illustrating each listed cause follow.

Table 8 The distribution of cases in which BiuTee did not apply a syntax-based entailment rule, which was expected according to the manual annotation.

Applied / Why not applied	**% of manually identified syntax-based rule applications**
Applied by the system	26.3%
Unavailable rule, within the formalism	30%
System found alternative proof	15.8%
Parser error	10.5%
System lacking prerequisite rules	7%
Resource lacking prerequisite rules	5.3%
Unavailable rule, out of the formalism	3.5%

Unavailable rule, within the formalism - The largest contributors to the coverage gap are syntax-based transformations that had been found necessary or beneficial in the preliminary manual analysis of RTE5, and that are feasible according to our formalism, but were not included in the resource. This corresponds to the 70% recall result in Subsection 5.2.1. As stated there, the main reason for lacking these rules is either that they're inaccurate to the degree that they'd cause more false entailments than valid ones (and harm the precision), or because we simply did not conceive of them.

As an example, consider:

T: *... a suitcase exploded at Japan's Narita airport,* ***killing two baggage handlers*** *as they transferred it to Air India Flight 301 for Bangkok and Delhi...*

H: *Two baggage handlers were killed while transferring suitcases to an Air India Flight.*

Here the manual proof suggests applying a rule that converts the boldface segment of T from active voice into passive voice, without the presence of a subject, making it similar to H, i.e., *killing two handlers* $\Rightarrow$ *two handlers were killed*. This hypothetical rule would look like our existing active/passive rule (in the direction from active to passive), without the nodes representing its LHS and RHS subjects.

Nonetheless, a rule that matches such a loose participle construction does not exist in the rule-base, because it would be matched against many clauses

that have an overt subject and cause false entailments like *A blast killed two handlers* ⇒ **A blast two handlers were killed*. Note that the resource does have a similar rule for converting active to passive voice, but it requires the presence of a subject (Subsection 4.6). Example (26) in Subsection 5.2.2 offers a similar discussion about another rule-base rule.

System found alternative proof - Even when BIUTEE is able to repeat the entire proof from the manual analysis (i.e., the relevant entailment rules are available), its proof search algorithm may find another proof, without the syntax-based rule in question. Mostly, these alternative system proofs include linguistically unmotivated transformations, but still get an overall *score* that is just as good as the score of the proofs we constructed.

For instance, consider the following simplified proof:

T: ***The*** *blast occurred at a hotel in Taba*

H: ***A*** *blast occurred at a hotel in Taba*

Naturally, at this point in the manual analysis, we chose to apply the first "substitute determiners" rule from Subsection 4.8, which replaced *the* with *a*. However, at the same point the system chose to perform a DUPLICATE AND MOVE operation, copying the second *a* next to the position of the *the*, yielding the same end result (full $T - H$ match), while the overall score of the proof was as good as our proof's. Thus, the "substitute determiners" rule was not applied here, due to the alternative proof the system found.

Combining this set with the manual rule applications that were repeated by the system, we get an encouraging result, that in 42.1% of the cases, either the expected rule was used, or an alternative proof was found.

Parser error - Parser errors produce many flawed syntactic trees, which prevent matching with a desired syntax-based rule. These account for 10.5% of such cases. For example, consider the pair:

T: *All that changed Thursday, when two buses ... shuttled in each direction between* ***Srinagar and Muzaffarabad, the capitals of the Indian and Pakistan-administered Kashmir****...*

H: *Srinagar is the summer capital of Indian Kashmir.*

The entailment proof we had composed in the manual analysis includes a rule application that converts the apposition construction in T (in bold) into an

independent copular sentence, similar to H (the "Apposition to Copula" rule, Section 4.4). However, the parser used by BIUTEE (EasyFirst) produced an erroneous analysis for this sentence, in which the apposition relation between *Srinagar and Muzaffarabad* and *the capitals* is missing. In this situation, the "Apposition to Copula" rule could not be matched against the text, although it is available in the rule-base and is needed for a correct proof.

System lacking prerequisite rules - To illustrate this category, consider the simplified pair:

T: *Dick Cheney and Harry M. Whittington had been hunting ...*

H: *Harry M. Whittington is a hunter*

In the manual analysis, we constructed a 2-step proof for this pair:

1. Apply the hypothetical lexical syntactic substitution rule (containing these explicit words) *X hunts* $\Rightarrow$ *X is hunter(s)*, generating *Dick Cheney and Harry M. Whittington are hunters*
2. Apply the syntax-based rule "delete conjunct" (see Subsection 4.3), generating *Harry M. Whittington is a hunter*

Here, we neglect the effect of some auxiliaries and word inflections in T, as explained in Subsection 3.1. Since the knowledge resources loaded onto BIUTEE didn't have any the lexical syntactic rules similar to the one we hypothesized, apparently the system found no proof that featured the "delete conjunct" rule (with a high enough score). Therefore, this generic rule was not applied as expected.

Resource lacking prerequisite rules - This is just like the previous category, except that in these cases it is a syntactic rule that the engine lacks. This means that for each such case, there exists a corresponding case in the *Unavailable rule, within the formalism* category.

Unavailable rule, out of the formalism - As a demonstration for the last category, consider this simplified pair:

T: ***India, Pakistan*** *due to Hold Meetings*

H: ***Pakistan and India*** *were due to hold meetings*

T looks like the headline of a news item (short, capitalized words, no copula nor coordination words etc.), which hints us that the comma structure in boldface is a coordination. In this case, in our manual analysis we assumed the

existence of a generic rule that can substitute such comma constructions with coordinations, i.e., *India and Pakistan*. Appropriately, we proceeded to apply the conjectured rule, and then apply a "swap conjuncts" rule (see Subsection 4.3). These two applications yielded the desired construction *Pakistan and India*, and completed an almost full $T - H$ match.

Nonetheless, although the rule we conjectured is generic (does not depend on lexicon), it is impossible to implement within our formalism, since it is suitable only for sentences that are headlines, and we have no systematic way to detect headlines. Therefore, that rule application was annotated "not applied - unavailable rule, out of the formalism", while the application of the "swap conjuncts" rule was annotated "not applied - resource lacking prerequisite rules".

Justified vs. Spurious Rule Applications In our next experiment, we examined the cases in which BIUTEE applied rules spuriously, i.e., where the rule's syntactic phenomenon is not relevant to the correct inference chain. For example, consider the above-mentioned pair again:

T: *All that changed Thursday, when two buses ... shuttled in each direction between Srinagar and Muzaffarabad, the capitals of the Indian and Pakistan-administered Kashmir...*

H: *Srinagar is the summer capital of Indian Kashmir*

Applying an "active to passive" rule on T would be spurious, since a human would judge that converting between active and passive is irrelevant to this inference. Nonetheless, the system may still apply the rule if it would produce some side-effect, which can be exploited by the system to apply other rules, that ultimately help construct a (possibly erroneous) proof. We investigate how frequently such applications were performed.

In order to obtain a fuller account of spurious applications, we revisited the manual analysis of syntax-based rule applications from Section 5.2.2, focusing on the 46 entailment-preserving rule applications (according to the precision evaluation there), in proofs of positive pairs. For each rule-application we drew two new judgements: whether the application was spurious, and if so, why.

The results are presented in Table 9. First, we see that about half of the syntax-based rule applications were judged as justified and the other half were spurious. This may indicate that the main challenge for future work lies not

Table 9 The contribution of justified vs. spurious syntax-based system rule applications, based on 46 entailment-preserving applications

Justified rule use	48%
Spurious use	
System decision	50%
Parser Error	2%

Table 10 Global ablation test results: the top 2 rows show the F1, recall and precision measures of the system, run on the RTE6 dataset, without the syntax-based resource (baseline) and with it. The bottom two rows are based on the RTE5 dataset, and contain an accuracy measure.

Configuration	**Recall**	**Precision**	**F1**	**Accuracy**
Baseline RTE6	43.07%	55.37%	48.45%	N/A
Rule-base Test RTE6	43.17%	57.71%	49.4%	N/A
Baseline RTE5	70%	61.04%	65.21%	0.6333
Rule-base Test RTE5	73.66%	63.32%	68.1%	0.6483

in improving the rules themselves, but in developing better entailment system components that would utilize them properly.

5.3.3 Impact on system results

In this subsection we assess the impact of the resource on BiuTee's overall performance, beginning with a discussion on the global performance measures, and ending with a breakdown of the results into categories of $T - H$ pairs.

Global Ablation Results Table 10 shows the global F1, recall and precision measures of the system classifications without the syntax-based resource (baseline configuration, with the abovementioned WordNet and Bap knowledge resources) and with it, on the RTE5 and RTE6 datasets, which were the two most recent RTE datasets available to us at the time of the evaluation. We use the standard definitions of precision, recall and F1 used in NLP benchmarks.

We see that straightforwardly using the resource on RTE6 slightly enhances the overall performance, from 48.45% in the baseline run, to 49.4% in the rule-base test run. In addition, the resource is shown to be more effective on RTE5, improving the baseline F1 result from 65.21% to 68.1%. This result may signify that the style and composition of the dataset itself has a

large impact on the usefulness of the resource. All the above differences are statistically significant ($p < 0.02$ using Mcneamar test). Performance gains of nearly three percentage points are normal, and even above average of the positive effects of the resources that were typically measured in RTE ablation tests (to illustrate, see the RTE6 results in (Bentivogli et al, 2010a)) and the RTE Knowledge Resources page at `http://aclweb.org/aclwiki/index.php?title=RTE_Knowledge_Resources`).

Breakdown of Ablation Results Now we closely examine the resource's impact on the ratio of correct vs. mistaken system classifications (*positive/negative* entailment). The data is taken from an ablation test, comparing BIUTEE's classifications with and without the syntax-based rule-base, configured the same as in Section 5.2.2.

We compared each system classification (*positive/negative)* with the gold standard annotation (*true/false*), thus classifying each answer into one of four categories: *true-positive, false-positive, true-negative* or *false-negative.*

Based on these statistics, for each $T - H$ pair we combined the labels from the two runs, yielding eight composite categories, shown in the two left columns of Table 11. The left label describes the result from the run without syntax-based rules, while the right label describes the run with them. Notice that other combinations are not possible, e.g., a TP cannot become a FP by altering the classification. Presenting the answers this way can highlight the resource's impact in finer detail.

The main results of the ablation runs are presented in the third column of Table 11[8] , "Total System Classifications". The vast majority of classifications fall in the top 4 rows, which means they are not affected by the resource. The bottom 4 rows show that syntax-based rules affected merely 2% of the system classifications (91 out of 4353). Within those 91, use of the resource improved about two thirds (60 vs. 31) of the classifications (from FN to TP, and from FP to TN), while in a third of the cases it was the other way around (from TN to FP, and from TP to FN).

Finally, in order to enhance these statistics, we split the third column in two, with pairs whose proof involves some syntax-based operation (where the syntactic rule-base is employed) in the fourth column, and the rest in the

[8] The examination involves only 4535 pairs out of the total 19972 pairs in the RTE6 Test dataset, since BIUTEE judged the others to be non entailing based on an information retrieval ranking filter, which is used at a preprocessing stage.

Table 11 Ablation of BiuTee's classifications for RTE6 Test, with and without the syntactic rule-base, broken down into categories of $T - H$ pair proofs. The first column is the T(rue)/F(alse) and P(ositive)/N(egative) label the pair got in the baseline run's classification. The second is the label from the run with the rule-base. The third counts the total pairs that got the corresponding two labels. Out of those, the fourth column counts the pairs that the system used at least one syntactic rule in their proof (in the second run). The fifth counts all the rest.

Classification Label		Total System Classifications	Classifications for Proofs Involving Generic Transformations	Classifications for Proofs Not Involving Generic Transformations
w/o resource	with resource			
TP	TP	382	247	135
FP	FP	273	145	128
FN	FN	227	110	117
TN	TN	3562	1748	1814
Subtotal		**4444**	**2250**	**2194**
FN	TP	11	10	1
TN	FP	21	17	4
FP	TN	49	25	24
TP	FN	10	6	4
Subtotal		**91**	**58**	**33**
Total		**4535**	**2308**	**2227**

fifth. This information may indicate whether the presence of a syntax-based rule is likely to improve classifications or hamper them. Alternatively, if no such correlation is evident, it would indicate that the rules have a strong indirect impact on the system, by influencing the model that the classifier's machine learning algorithm learns. This may have a significant effect on the performance measures, even without applying a single rule from the resource.

The figures in the two right columns, where we see that the presence of syntactic rules does not strongly correlate with an improvement or worsening in the system's classifications, seem to suggest the latter conclusion. That is to say, the resource also has an indirect effect on the learned model, and how the system applies entailment rules from other knowledge resources.

Conclusion In this subsection we assessed the impact of the resource on BiuTee's overall performance. We used an ablation test to show the rule-base improves the F1 score by nearly 3%, on RTE5, and by nearly 1% on RTE6. In addition,

by observing the impact on categories of $T-H$ pairs, we found that alongside the direct role the syntactic rules play in improving entailment proofs, they play an equality influential indirect role, by affecting the way the machine learning algorithm uses other knowledge resources.

6 Conclusions

In this study we present a comprehensive novel generic syntax-based rule-base for the TE task, based on inputs from previous salient works on the topic, as well as novel contributions. Represented according to the Stanford dependencies standard and based on a well defined formalism, the rule-base is made publicly available for use with TE systems, including full documentation, and tools for GUI rule design and software compilation.

It offers a wide variety of over 60 generic entailment rules schemas, which compile to 226 concrete rules, that substitute between copious equivalent or entailing constructions in categories such as: active vs. passive, coordination, apposition, determiners, possessives, and case correction. Also, it can extract simplified IS-A and HAS-A implications from over a dozen generic patterns, and successfully decouples relative clauses. The rules are honed for high resilience to syntactic structure diversity, especially in RTE texts.

Qualitative and quantitative depth evaluations are reported, including a manual dataset analysis, that demonstrates the high potential for knowledge resources of this type in TE: an estimated 78% of the entailment proofs of $T-H$ pairs require, or could benefit from syntactic transformations. The rules' recall, i.e., coverage over the required set of syntax-based transformations, is estimated at 70%, benefiting from an inventory over twice as large as previous syntax-based resources. Their precision is proven to be very high, at 94%. We discuss and give an example of missing rules, that, if added, would increase recall, by generating entailments from frequent syntactic structures, but significantly harm precision, by generating non-entailments from similar structures that are indistinguishable. These results lead us to believe that, within the limits of our formalism, our syntax-based rule-base already fulfils much of its theoretical potential.

As a knowledge resource it significantly improves the F1 score of a concrete entailment engine, BIUTEE, by 2.9% on the RTE5 dataset, a contribution that is above the average for resources in this task. As further testimony of the resource's usefulness, when comparing the engine's usage of it to gold standard

entailment proofs, we find that in 42.1% of rule application, the engine either constructed a proof using the expected rule, or found an valid alternative proof. A further inspection of the proofs in which the engine chooses to use the rules reveals that in half of the cases, the engine applies a rule spuriously, where its syntactic phenomenon is not relevant to the correct inference chain. This may indicate that in order to improve the resource's performance, the main research effort should be directed to improving entailment system components that would employ it more effectively.

A Formal Definition and Application of Entailment Rules

This appendix defines entailment rules, and the formalism for applying them on parse trees, which we assume that entailment systems follow.

A rule '$L \Rightarrow R$' is composed of two *templates*, L on the left-hand-side (LHS) and R on the right-hand-side (RHS). Templates are dependency parse subtrees which may contain *variable* nodes. These are regular nodes that have no specified lemma, so that they match any lemma. Figure 1(a) shows active-to-passive transformation rule, and Figure 1(b) illustrates its application. Technically, there are two distinct types of rules: *substitution* and *extraction*, described ahead.

The rule application algorithm is given in Algorithm 1. One rule application, involving one *match* between the rule and a source tree s, generates one derived tree. Likewise, consecutive applications of one rule generate a set D of derived trees (consequents) from s, through the following steps:

1. *L matching* First, matches of L in the source tree s are sought. L is matched in s if there exists a one-to-one node mapping function f from L to s, such that:
   (a) For each node u in L, $f(u)$ matches the lemma and POS of u. Variables match any lemma value in $f(u)$.
   (b) 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. In our example in Figure1(b), the variable V is matched in the verb *see*, $N1$ is matched in *Mary* and $N2$ is matched in *John*. If matching succeeds, then the following is performed for each match found.
2. *R instantiation* A copy of R is generated and its variables are instantiated according to their matching nodes in L. In addition, a rule may specify alignments, defined as a partial function from L nodes to R nodes. An alignment indicates that for each modifier m of the source node that is not part of the rule structure, the subtree rooted at m should also be copied as a modifier of the target node. In addition to defining alignments explicitly, each variable in L is implicitly aligned to its counterpart in R. In our example, the alignment between the V nodes implies that *yesterday* (modifying *see*) should be copied to the generated sentence, and similarly *beautiful* (modifying *Mary*) is copied for $N1$.
3. *Derived tree generation* Let r be the instantiated R, along with its descendants, copied from L through alignment, and l be the subtree matched by L. The formalism has two methods for generating the derived tree d: *substitution* and *extraction*, as specified by

Input: a source tree s ; a rule $E : L \rightarrow R$
Output: a set D of derived trees

```
M ← the set of all matches of L in s
D ← ∅
for each: f ∈ M do
    l ← the subtree matched by L in s according to match f

    // R instantiation
    r ← a copy of R
    for each: variable v ∈ r do
        Instantiate v with f(v)
    end for
    for each: aligned pair of nodes u_L ∈ l and u_R ∈ r do
        for each: daughter m of u_L such that m ∉ l do
            Copy the subtree of s rooted in m under u_R in r, with the same dependency
            relation
        end for
    end for

    // Derived tree generation
    if substitution rule then
        d ← s copy with l (and the descendants of its nodes) replaced by r
    else introduction rule
        d ← r
    end if

    add d to D
end for
```

Algorithm 1: Applying a rule to a tree

the rule type. Substitution rules specify modification of a subtree of s, leaving the rest of s unchanged. Thus, 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, as well as for lexical rules such as '*buy* $\Rightarrow$ *purchase*'. By contrast, extraction rules are used to make inferences from a subtree of s, while the other parts of s are ignored and do not affect d. A typical example is inferring a proposition embedded as a temporal 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 bottom tree in Figure 1(b) yields the proposition *John saw beautiful Mary yesterday.*

B Mapping from Penn Tree Bank POS Set to the Reduced POS Set

Table 12 defines the conversion from the Penn Tree Bank POS set, as specified in the Stanford dependency relations standard, which we adopt, to our proprietary reduced POS set.

Table 12 The conversion from the Penn Tree Bank POS set, as specified in the Stanford dependency relations standard, to our proprietary reduced POS set

Reduced POS	**Penn POS**
VB	VERB
VBD	VERB
VBG	VERB
VBN	VERB
VBP	VERB
VBZ	VERB
NN	NOUN
NNS	NOUN
NNP	NOUN
NNPS	NOUN
JJ	ADJECTIVE
JJR	ADJECTIVE
JJS	ADJECTIVE
DT	DETERMINER
WDT	DETERMINER
PDT	DETERMINER
PRP	PRONOUN
PRP$	PRONOUN
WP	PRONOUN
WP$	PRONOUN
RB	ADVERB
RBR	ADVERB
RBS	ADVERB
WRB	ADVERB
CC	PREPOSITION
IN	PREPOSITION
RP	PREPOSITION
TO	PREPOSITION
RCB	PUNCTUATION
LCB	PUNCTUATION
LRB	PUNCTUATION
RRB	PUNCTUATION
LS	PUNCTUATION
SYM	OTHER
CD	OTHER
EX	OTHER
FW	OTHER
MD	OTHER
POS	OTHER
UH	OTHER

C Description of BiuTee, the Bar Ilan University Textual Entailment Engine

At the preprocessing stage, BiuTee parses all the texts and hypotheses using EasyFirst parser (Goldberg and Elhadad, 2010), and adds coreference information using ArkRef (Haghighi and Klein, 2009). Then, using machine learning, it finds the simplest and most reliable proofs to T - H pairs in the space of all possible proofs. The transformations are loaded from inference rule-base resources, such as our syntax-based resource, WordNet (Fellbaum, 1998; Miller, 1995), Wikipedia (Shnarch et al, 2009) and many others. In many cases, the resources are insufficient for the task, and therefore the system also uses a built-in set of "on the fly" transformations. These can manipulate a tree in any desirable way (adding, deleting, altering nodes and edges etc.), but have no linguistic justification. Hence the system tries to apply them as little as possible.

References

Amoia M, Gardent C (2008) A test suite for inference involving adjectives. In: Nicoletta Calzolari (Conference Chair) BMJMJOSPDT Khalid Choukri (ed) Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08), European Language Resources Association (ELRA), Marrakech, Morocco, http://www.lrec-conf.org/proceedings/lrec2008/

Bar-Haim R (2010) Semantic inference at the lexical-syntactic level. PhD thesis, Bar-Ilan Univeristy

Bar-Haim R, Dagan I, Greental I, Shnarch E (2007) Semantic inference at the lexical-syntactic level. In: Proceedings of AAAI, pp 871–876

Bar-Haim R, Berant J, Dagan I (2009) A compact forest for scalable inference over entailment and paraphrase rules

Bentivogli L, Magnini B, Dagan I, Dang HT, Giampiccolo D (2009) The fifth pascal recognizing textual entailment challenge. In: Preproceedings of the Text Analysis Conference (TAC)

Bentivogli L, Clark P, Dagan I, Dang HT, Giampiccolo D (2010a) The sixth pascal recognizing textual entailment challenge. In: Proceedings of TAC, Gaithersburg, Maryland

Bentivogli L, Clark P, Dagan I, Dang HT, Giampiccolo D (2010b) The sixth PASCAL recognizing textual entailment challenge. In: The Text Analysis Conference (TAC 2010)

Berant J, Liang P (2014) Semantic Parsing via Paraphrasing. In: Association for Computational Linguistics (ACL)

Cabrio E, Kouylekov M, Magnini B (2008) Combining specialized entailment engines for rte-4. In: Proceedings of TAC, Gaithersburg, Maryland

Clark P, Harrison P (2010) Blue-lite: a knowledge-based lexical entailment system for rte6. In: (to appear in) Proceedings of TAC, Gaithersburg, Maryland

Dagan I, Roth D, Sammons M, Zanzotto F (2013) Recognizing Textual Entailment: Models and Applications. Morgan and Claypool

Fellbaum C (ed) (1998) WordNet: An Electronic Lexical Database (Language, Speech, and Communication). MIT Press

Goldberg Y, Elhadad M (2010) An efficient algorithm for easy-first non-directional dependency parsing. In: Proc. of NAACL

Haghighi A, Klein D (2009) Simple coreference resolution with rich syntactic and semantic features. In: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Singapore, pp 1152–1161, URL http://www.aclweb.org/anthology/D/D09/D09-1120

Harabagiu S, Hickl A (2006) Methods for using textual entailment in open-domain question answering. In: Proceedings of ACL

Harabagiu S, Hickl A, Lacatusu F (2007) Satisfying information needs with multi-document summaries. Inf Process Manage 43:1619–1642

Hearst M (1992) Automatic acquisition of hyponyms from large text corpora. In: Proceedings of COLING

Heilman M, Smith NA (2010) Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In: HLT-NAACL

Hickl A (2008) Using discourse commitments to recognize textual entailment. In: COLING, pp 337–344

Kotlerman L, Dagan I, Szpektor I, Zhitomirsky-Geffet M (2010) Directional distributional similarity for lexical inference. Natural Language Engineering 16:359–389

MacKinlay A, Baldwin T (2009) A baseline approach to the rte5 search pilot. In: Proceedings of TAC, Gaithersburg, Maryland

Marcus MP, Santorini B, Marcinkiewicz MA (1993) Building a large annotated corpus of english: The penn treebank. COMPUTATIONAL LINGUISTICS 19(2):313–330

de Marneffe MC, Manning CD (2008) The stanford typed dependencies representation. In: COLING Workshop on Cross-framework and Cross-domain Parser Evaluation, URL pubs/dependencies-coling08.pdf

Miller GA (1995) Wordnet: A lexical database for english. Commun ACM 38(11):39–41

Mirkin S, Dagan I, Pado S (2010) Assessing the role of discourse references in entailment inference. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Uppsala, Sweden, pp 1209–1219, URL http://www.aclweb.org/anthology/P10-1123

Nielsen RD, Ward W (2007) A Corpus of Fine-Grained Entailment Relations. In: Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing, Association for Computational Linguistics, Prague, pp 28–35, URL http://www.aclweb.org/anthology/W/W07/W07-1405

Pantel P, Ravichandran D, Hovy EH (2004) Towards terascale semantic acquisition. In: COLING

Petrov S, Das D, McDonald R (2012) A universal part-of-speech tagset. In: Proc. of LREC

Ravichandran D, Hovy E (2002) Learning surface text patterns for a question answering system. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics

Romano L, Kouylekov M, Szpektor I, Dagan I, Lavelli A (2006) Investigating a generic paraphrase-based approach for relation extraction. In: Proceedings of EACL

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

Shinyama Y, Sekine S (2006) Preemptive information extraction using unrestricted relation discovery. In: Proceedings of the Human Language Technology Conference of the NAACL,

Main Conference

Shnarch E, Barak L, Dagan I (2009) Extracting lexical reference rules from wikipedia. In: Proceedings of ACL

Shnarch E, Dagan I, Goldberger J (2012) A probabilistic lexical model for ranking textual inferences. In: *SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012), Association for Computational Linguistics, Montréal, Canada, pp 237–245, URL http://www.aclweb.org/anthology/S12-1032

Stern A, Dagan I (2011) A confidence model for syntactically-motivated entailment proofs. In: Proceedings of the International Conference RANLP-2011

Stern A, Shnarch E, Lotan A, Mirkin S, Kotlerman L, Zeichner N, Berant J, Dagan I (2010) Rule chaining and approximate match in textual inference. In: (to appear in) Proceedings of TAC, Gaithersburg, Maryland

Wang R, Neumann G (2008) Relation validation via textual entailment. In: Proceedings of OBIES