Annotating by Proving using SemAnTE

Assaf Toledo¹ Stavroula Alexandropoulou¹ Sophie Chesney²
Robert Grimm¹ Pepijn Kokke¹ Benno Kruit³
Kyriaki Neophytou¹ Antony Nguyen¹ Yoad Winter¹

¹ - Utrecht University ² - University College London ³ - University of Amsterdam

{a.toledo,s.alexandropoulou,y.winter}@uu.nl
sophie.chesney.10@ucl.ac.uk,{pepijn.kokke,bennokr}@gmail.com
{r.m.grimm,k.neophytou,a.h.nguyen}@students.uu.nl

Abstract

We present SemAnTE, a platform for marking and substantiating a semantic annotation scheme of textual entailment according to a formal model. The platform introduces a novel approach to annotation by providing annotators immediate feedback whether the data they mark are substantiated: for positive entailment pairs, the system uses the annotations to search for a formal logical proof that validates the entailment relation; for negative pairs, the system verifies that a counter-model can be constructed. By integrating a web-based user-interface, a formal lexicon, a lambda-calculus engine and an off-the-shelf theorem prover, this platform facilitates the creation of annotated corpora of textual entailment. A corpus of several hundred annotated entailments is currently in preparation using the platform and will be available for the research community.

1 Introduction

The Recognizing Textual Entailment (RTE) challenges (Dagan et al., 2006) advance the development of systems that automatically determine whether an entailment relation obtains between a naturally occurring text \( T \) and a manually composed hypothesis \( H \). The RTE corpus (Bar Haim et al., 2006; Giampiccolo et al., 2008), which is currently the only available resource of textual entailments, marks entailment candidates as positive/negative.¹ For example:

**Example 1**

- **T:** The book contains short stories by the famous Bulgarian writer, Nikolai Haitov.
- **H:** Nikolai Haitov is a writer.²
- **Entailment:** Positive

This categorization does not indicate the linguistic phenomena that underlie entailment or their contribution to inferential processes. In default of a gold standard identifying linguistic phenomena triggering inferences, entailment systems can be compared based on their performance, but the inferential processes they employ to recognize entailment are not directly accessible and consequently cannot be either evaluated or improved straightforwardly.

We address this problem by elucidating some of the central inferential processes underlying entailments in the RTE corpus, which we model formally within a standard semantic theory. This allows us not only to indicate linguistic phenomena that are involved in the recognition of entailment by speakers, but also to provide formal proofs that substantiate the annotations and explain how the

¹Pairs of sentences in RTE 1-3 are categorized in two classes: yes- or no-entailment; pairs in RTE 4-5 are categorized in three classes: entailment, contradiction and unknown. We label the judgments yes-entailment from RTE 1-3 and entailment from RTE 4-5 as positive, and the other judgments as negative.

²Pair 622 from the development set of RTE 2.
modeled phenomena interact and contribute to the recognition process. In this sense we adopt an
Annotating by Proving approach to textual entailment
annotation.

The annotation work is done using the Se-
ManTE (Semantic Annotation of Textual Entail-
ment) platform, which incorporates a web-based
user-interface, a formal lexicon, a lambda-calculus
engine and an off-the-shelf theorem prover. We
are currently using this platform to build a new
corpus of several hundred annotated entailments
comprising both positive and negative pairs. We
decided to focus on the semantic phenomena of
appositive, restrictive and intersective modification
as these semantic phenomena are prevalent in the
RTE datasets and can be annotated with high
consistency, and as their various syntactic expres-
sions can be captured by a limited set of concepts.3
In the future, we plan to extend this semantic model
to cover other, more complex phenomena.

2 Semantic Model

To model entailment in natural language, we
assume that entailment describes a preorder on
sentences. Thus, any sentence trivially entails itself
(reflexivity); and given two entailments \( T_1 \Rightarrow H_1 \)
and \( T_2 \Rightarrow H_2 \) where \( H_1 \) and \( T_2 \) are identical
sentences, we assume \( T_1 \Rightarrow H_2 \) (transitivity). We
use a standard model-theoretical extensional se-
manitics, based on the simple partial order on the
domain of truth-values. Each model \( M \) assigns
sentences a truth-value in the set \( \{0, 1\} \). Such a
Tarskian theory of entailment is considered ade-
quate if the intuitive entailment preorder on sentences
can be described as the pairs of sentences \( T \) and \( H \) whose truth-values \( [T]^M \) and \( [H]^M \) sat-
sify \( [T]^M \leq [H]^M \) for all models \( M \).

We use annotations to link between textual
representations in natural language and model-
theoretic representations. This link is established
by marking the words and structural configurations
in \( T \) and \( H \) with lexical items that encode semantic meanings for the linguistic phenomena
that we model. The lexical items are defined for-
ma tally in a lexicon, as illustrated in Table 1 for ma-
jor lexical categories over type:s e for entities, t
for truth-values, and their functional compounds.

3This conclusion is based on an analysis of RTE 1-4, in
which these modification phenomena were found to occur in
80.65% of the entailments and were annotated with cross-
annotator agreement of 68% on average.

<table>
<thead>
<tr>
<th>Category</th>
<th>Type</th>
<th>Example</th>
<th>Denotation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proper Name</td>
<td>e</td>
<td>Dan</td>
<td>dan</td>
</tr>
<tr>
<td>Indef. Article</td>
<td>(et)(et)</td>
<td>a</td>
<td>A</td>
</tr>
<tr>
<td>Def. Article</td>
<td>(et)e</td>
<td>the</td>
<td>t</td>
</tr>
<tr>
<td>Copula</td>
<td>(et)(et)</td>
<td>is</td>
<td>is</td>
</tr>
<tr>
<td>Noun</td>
<td>et</td>
<td>book</td>
<td>book</td>
</tr>
<tr>
<td>Intrans. verb</td>
<td>et</td>
<td>sit</td>
<td>sit</td>
</tr>
<tr>
<td>Pred. Conj.</td>
<td>(et)((et)(et))</td>
<td>contain</td>
<td>contain</td>
</tr>
<tr>
<td>Res. Adj.</td>
<td>(et)(et)</td>
<td>short</td>
<td>( R_m(\text{short}) )</td>
</tr>
<tr>
<td>Exist. Quant.</td>
<td>(et)(et)t</td>
<td>some</td>
<td>SOME</td>
</tr>
</tbody>
</table>

Table 1: Lexicon Illustration

Denotations that are assumed to be arbitrary are
given in boldface. For example, the intransitive verb \( \text{sit} \) is assigned the type \( et \), which describes
functions from entities to a truth-values, and its
denotation \( \text{sit} \) is an arbitrary function of this type.
By contrast, other lexical items have their denota-
tions restricted by the given model \( M \). As illus-
trated in Figure 1, the coordinator \( \text{and} \) is assigned
the type \( (et)((et)(et)) \) and its denotation is a func-
tion that takes a function \( A \) of type \( et \) and returns
a function that takes a function \( B \), also of type \( et \),
and returns a function that takes an entity \( x \) and
returns 1 if and only if \( x \) satisfies both \( A \) and \( B \).

\[
A = \text{IS} = \lambda A_{et}. A \\
\text{AND} = \lambda A_{et}. \lambda x.e. M(A(x)) \\
\text{WHOA} = \lambda A_{et}. \lambda x.e. (\lambda y.y = x \land A(x)) \\
\text{SOME} = \lambda A_{et}. \lambda B_{et}. \exists x. A(x) \land B(x) \\
\text{AND} = \lambda A_{et}. \lambda B_{et}. \lambda x.e. A(x) \land B(x)
\]

Figure 1: Functions in the Lexicon

By marking words and syntactic constructions
with lexical items, annotators indicate the under-
lying linguistic phenomena in the data. Further-
more, the formal foundation of this approach al-
 lows annotators to verify that the entailment re-
lation (or lack thereof) that obtains between the
textual forms of \( T \) and \( H \) also obtains between their respective semantic forms. This verification
ensures that the annotations are sufficient in the
sense of providing enough information for recog-
nizing the entailment relation based on the semantic
abstraction. For example, consider the simple
entailment Dan sat and sang \( \Rightarrow \text{Dan sang} \) and
assume annotations of Dan as a proper name, sat
and sang as intransitive verbs and \( \text{and} \) as predi-
cate conjunction. The formal model can be used
to verify these annotations by constructing a proof
as follows: for each model \( M \):
\[ [\text{Dan [sat [and sang]]}]^M = ((\text{AND}(\text{sing}))(\text{sit}))(\text{dan}) \]
\[ \text{analysis} = (((\lambda A_{et}.\lambda B_{et}.\lambda x_{cx}.A(x) \land (\text{sing}))(\text{sit}))(\text{dan}) \]
\[ = \text{sit(dan)} \land \text{sing(dan)} \text{ func. app. to sing}, \text{ sit and dan} \]
\[ \leq \text{sing(dan)} \text{ def. of } \land \]
\[ = [\text{Dan sang}]^M \text{ def. of } \land \]

3 Platform Architecture

The platform’s architecture is based on a client-server model, as illustrated in Figure 2.

![Platform Architecture Diagram](image)

**Figure 2: Platform Architecture**

The user interface (UI) is implemented as a web-based client using Google Web Toolkit (Olson, 2007) and allows multiple annotators to access the RTE data, to annotate, and to substantiate their annotations. These operations are done by invoking corresponding remote procedure calls at the server side. Below we describe the system components as we go over the work-flow of annotating Example 1.

**Data Preparation**: We extract \(T\)-\(H\) pairs from the RTE datasets XML files and use the Stanford CoreNLP (Klein and Manning, 2003; Toutanova et al., 2003; de Marneffe et al., 2006) to parse each pair and to annotate it with part-of-speech tags. Consequently, we apply a naive heuristic to map the PoS tags to the lexicon. This process is called as part of the platform’s installation and when annotators need to simplify the original RTE data in order to avoid syntactic/semantic phenomena that the semantic engine does not support. For example, the bare plural short stories is simplified to some short stories as otherwise the engine is unable to determine the quantification of this noun.

**Annotation**: The annotation is done by marking the tree-leaves with entries from the lexicon. For example, short is annotated as a restrictive modifier (MR) of the noun stories, and contains is annotated as a transitive verb (V_2). In addition, annotators manipulate the tree structure to fix parsing mistakes and to add leaves that mark semantic relations. For instance, a leaf that indicates the apposition between the famous Bulgarian writer and Nikolai Haitov is added and annotated as WHO_A.

The server stores a list of all annotation actions. Figure 3 shows the tree-view, lexicon, prover and annotation history panels in the UI.

**Proving**: When annotating all leaves and manipulating the tree structures of \(T\) and \(H\) are done, the annotators use the prover interface to request a search for a proof that indicates that their annotations are substantiated. Firstly, the system uses lambda calculus reductions to create logical forms that represent the meanings of \(T\) and \(H\) in higher-order logic. At this stage, type errors may be reported due to erroneous parse-trees or annotations. In this case an annotator will fix the errors and re-run the proving step. Secondly, once all type errors are resolved, the higher-order representations are lowered to first order and Prover9 (McCune, 2010) is executed to search for a proof between the logical expressions of \(T\) and \(H\). The proofs are recorded in order to be included in the corpus release. Figure 4 shows the result of translating \(T\) and \(H\) to an input to Prover9.

4 Corpus Preparation

We have so far completed annotating 40 positive entailments based on data from RTE 1-4. The annotation is a work in progress, done by four Master students of Linguistics who are experts in the data and focus on entailments whose recognition relies on a mixture of appositive, restrictive or intersective modification. As we progress towards the compilation of a corpus of several hundred pairs, we extend the semantic model to support more inferences with less phenomena simplification.
formulas(assumptions).
all x0 (((writer(x0) & bulgarian(x0)) &
famous_writer_bulgarian(x0)) ↔ x0=c1).
all x0 (((stories(x0) & short_stories(x0)) & exists x1 (by_stories_short_stories(x1, x0) & (x1=c1 & x1=Nikolai_Haitov))) ↔ x0=c2). all x0 (book(x0) ↔ x0=c3).
contains(c2, c3).
end_of_list.
formulas(goals).
exists x0 (writer(x0) & x0=Nikolai_Haitov).
end_of_list.

5 Conclusions

We introduced a new concept of an annotation platform which implements an Annotating by Proving approach. The platform is currently in use by annotators to indicate linguistic phenomena in entailment data and to provide logical proofs that substantiate their annotations. This method guarantees that the annotations constitute a complete description of the entailment relation and can serve as a gold-standard for entailment recognizers. The new corpus will be publicly available.

Acknowledgments

The work of Stavroula Alexandropoulou, Robert Grimm, Sophie Chesney, Pepijn Kokke, Benno Kruit, Kyriaki Neophytou, Antony Nguyen, Assaf Toledo and Yoad Winter was supported by a VICI grant number 277-80-002 by the Netherlands Organisation for Scientific Research (NWO).

References


