Efficient Logical Inference for Semantic Processing

Ran Tian* **Yusuke Miyao** **Takuya Matsuzaki**
National Institute of Informatics, Japan
{tianran,yusuke,takuya-matsuzaki}@nii.ac.jp

Abstract

Dependency-based Compositional Semantics (DCS) provides a precise and expressive way to model semantics of natural language queries on relational databases, by simple dependency-like trees. Recently abstract denotation is proposed to enable generic logical inference on DCS. In this paper, we discuss some other possibilities to equip DCS with logical inference, and we discuss further on how logical inference can help textual entailment recognition, or other semantic precessing tasks.

1 Introduction

Dependency-based Compositional Semantics (DCS) was proposed as an interface for querying relational databases by natural language. It features DCS trees as semantic representation, with a structure similar to dependency trees. In its basic version, a node of a DCS tree indicates a table in the database, and an edge indicates a join relation. Both ends of an edge are labeled by a field of the corresponding table (Liang et al., 2011). However, when DCS is applied to logical inference on unrestricted texts, it is unrealistic to assume an explicit database, because we cannot prepare a database for everything in the world. For this reason, DCS trees are detached from any specific relational database, in a way that each node of a DCS tree indicates a content word in a sentence (thus no fixed set of possible word labels for a DCS tree node), and each edge indicates a semantic relation between two words. Labels on the two ends of an edge, initially indicating fields of tables in a database, are considered as semantic roles of the corresponding words. Abstract denotation is proposed to capture the meaning of this abstract version of DCS tree, and a textual inference system based on abstract denotation is built (Tian et al., 2014).

It is quite natural to apply DCS trees, a simple and expressive semantic representation, to textual inference; however the use of abstract denotations to convey logical inference is somehow unusual. There are two seemingly obvious way to equip DCS with logical inference: (i) at the tree level, by defining a set of logically sound transformations of DCS trees; or (ii) at the logic level, by converting DCS trees to first order predicate logic (FOL) formulas and then utilizing a theorem prover. For (i), it may not be easy to enumerate all types of logically sound transformations, but tree transformations can be seen as an approximation of logical inference. For (ii), abstract denotation is more efficient than FOL formula, because abstract denotation eliminates quantifiers and meanings of natural language texts can be represented by atomic sentences.

To elaborate the above discussion and to provide more topics to the literature, in this paper we discuss the following four questions: (§2) How well can tree transformation approximate logical inference? (§3) With rigorous inference on DCS trees, where does logic contribute in the system of Tian et al. (2014)? (§4) Does logical inference have further potentials in Recognizing Textual Entailment (RTE) task? and (§5) How efficient is abstract denotation compared to FOL formula? We provide examples or experimental results to the above questions.

*Current affiliation of the first author: Graduate School of Information Sciences, Tohoku University, Japan. Email address: tianran@ecei.tohoku.ac.jp

Proceedings of the ACL 2014 Workshop on Semantic Parsing, pages 71–75,
Baltimore, Maryland USA, June 26 2014.

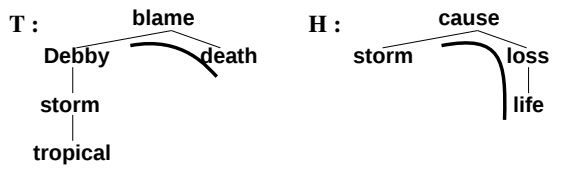

Figure 1: DCS trees of **T:** *Tropical storm Debby is blamed for death* and **H:** *A storm has caused loss of life*

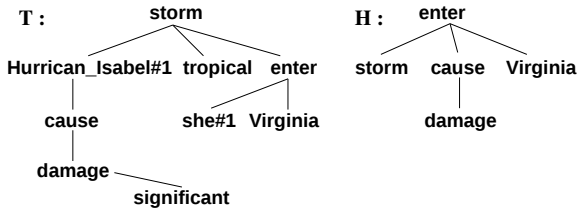

Figure 2: DCS trees with coreference

2 Tree transformation vs. logical inference

In the tree transformation based approach to RTE, it has been realized that some gaps between **T** and **H** cannot be filled even by a large number of tree transformation rules extracted from corpus (Bar-Haim et al., 2007a). For example in Figure 1, it is possible to extract the rule *blamed for death* → *cause loss of life*, but not easy to extract *tropical storm Debby* → *storm*, because "Debby" could be an arbitrary name which may not even appear in the corpus.

This kind of gaps was typically addressed by approximate matching methods, for example by counting common sub-graphs of **T** and **H**, or by computing a cost of tree edits that convert **T** to **H**. In the example of Figure 1, we would expect that **T** is "similar enough" (i.e. has many common sub-graphs) with **H**, or the cost to convert **T** into **H** (e.g. by deleting the node **Debby** and then add the node **storm**) is low. As for how similar is enough, or how the cost is evaluated, we will need a statistical model to train on RTE development set.

It was neglected that some combinations of tree edits are logical (while some are not). The entailment pair in Figure 1 can be easily treated by logical inference, as long as the apposition *tropical storm* = *Debby* is appropriately handled. In contrast to graph matching or tree edit models which theoretically admit arbitrary tree transformation, logical inference clearly discriminate sound transformations from unsound ones. In this sense, there would be no need to train on RTE data.

When coreference is considered, logically sound tree transformations can be quite complicated. The following is a modified example from RTE2-dev:

T: *Hurricane Isabel, which caused significant damage, was a tropical storm when she entered Virginia.*

H: *A storm entered Virginia, causing damage.*

The corresponding DCS trees are shown in Figure 2. Though the DCS trees of **T** and **H** are quite different, **H** can actually be proven from **T**. Note the coreference between *Hurricane Isabel* and *she*, suggesting us to copy the subtree of **Hurricane_Isabel** to **she**, in a tree edit approach. This is not enough yet, because the head **storm** in **T** is not placed at the subject of **cause**. The issue is indeed very logical: from "*Hurricane Isabel* = *she*", "*Hurricane Isabel* = *storm*", "*she* = subject of *enter*" and "*Hurricane Isabel* = subject of *cause*", we can imply that "*storm* = subject of *enter* = subject of *cause*".

3 Alignment with logical clues

Tian et al. (2014) proposed a way to generate on-the-fly knowledge to fill knowledge gaps: if **H** is not proven, compare DCS trees of **T** and **H** to generate path alignments (e.g. *blamed for death* ∼ *cause loss of life*, as underscored in Figure 1); evaluate the path alignments by a similarity score function; and path alignments with a score greater than a threshold (0.4) are accepted and converted to inference rules.

The word vectors Tian et al. (2014) use to calculate similarities are reported able to capture semantic compositions by simple additions and subtractions (Mikolov et al., 2013). This is also the case when used as knowledge resource for RTE, for example the similarities between **blamed+death** and **cause+loss+life**, or between **found+shot+dead** and **killed**, are computed > 0.4.

However, generally such kind of similarity is very noisy. Tian et al. (2014) used some logical clues to filter out irrelevant path alignments, which helps to keep a high precision. To evaluate the effect of such logical filters, we compare it with some other alignment strategies, the performance of which on RTE5-test data is shown in Table 1.

Each strategy is described in the following.

Strategy	Prec.	Rec.	Acc.
LogicClue + Inference	**69.9**	55.0	**65.7**
LexNoun + Inference	64.2	57.3	62.7
LexNoun + Coverage	57.1	75.0	59.3
NoFilter + Coverage	54.2	**87.7**	56.8

Table 1: Comparison of different alignment strategies

LogicClue + Inference This is the system of Tian et al. (2014)[1], which use logical clues to filter out irrelevant path alignments, and apply accepted path alignments as inference rules.

LexNoun + Inference The same system as above, except that we only align paths between lexically aligned nouns. Two nouns are aligned if and only if they are synonyms, hyponyms or derivatively related in WordNet.

LexNoun + Coverage As above, paths between lexically aligned nouns are aligned, and aligned paths with similarity score > 0.4 are accepted. If all nodes in **H** can be covered by some accepted path alignments, then output "Y". This is very similar to the system described in Bar-Haim et al. (2007b).

NoFilter + Coverage Same as above, but all paths alignments with similarity score > 0.4 are accepted.

4 How can logical inference help RTE?

Logical inference is shown to be useful for RTE, as Tian et al. (2014) demonstrates a system with competitive results. However, despite the expectation that all entailment matters can be explained logically, our observation is that currently logical inference only fills very limited short gaps from **T** to **H**. The logical phenomena easily addressed by Tian et al. (2014)'s framework, namely universal quantifiers and negations, seems rare in PASCAL RTE data. Most heavy lifting is done by distributional similarities between phrases, which may fail in complicated sentences. An especially complex example is:

T: *Wal-Mart Stores Inc. said Tuesday that a Massachusetts judge had granted its motion to decertify a class action lawsuit accusing the world's largest retailer of denying employees breaks.*
H: *Employee breaks had been denied by a motion granted by a Massachusetts judge.*

[1] http://kmcs.nii.ac.jp/tianran/tifmo/

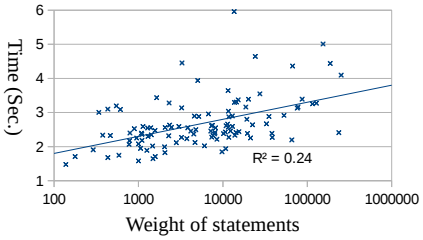

Figure 3: Time of forward-chaining (seconds) in our system, plotted on weights of statements (logarithmic scale).

	Orig. 3 Sec.	Orig. 5 Min.	Red. 5 Min.
Proof found	8	16	82
Too many variables	5	24	3
Failed to find proof	0	1	3
Memory limit	0	2	0
Time out	86	57	13

Table 2: Proportion (%) of exit status of Prover9

The system of Tian et al. (2014) generated on-the-fly knowledge to join several fragments in **T** and wrongly proved **H**. In examples of such complexity, distributional similarity is no longer reliable. However, it may be possible to build a priori logical models at the meta level, such as on epistemic, intentional and reportive attitudes. The models then can provide signals for semantic parsing to connect the logic to natural language, such as the words "*grant*", "*decertify*", and "*accuse*" in the above example. We hope this approach can bring new progress to RTE and other semantic processing tasks.

5 Efficiency of abstract denotations

To evaluate the efficiency of logical inference on abstract denotations, we took 110 true entailment pairs from RTE5 development set, which are also pairs that can be proven with on-the-fly knowledge. We plot the running time of Tian et al. (2014)'s inference engine (single-threaded) on a 2.27GHz Xeon CPU, with respect to the weighted sum of all statements[2], as shown in Figure 3. The graph shows all pairs can be proven in 6 seconds, and proof time scales *logarithmically* on weight of statements.

On the other hand, we converted statements on abstract denotations into FOL formulas, and tried to prove the same pairs using Prover9,[3] a popu-

[2] If a statement is translated to FOL formula, the weight of this statement equals to the weighted sum of all predicates in the FOL formula, where an n-ary predicate is weighted as n.

[3] www.cs.unm.edu/~mccune/prover9/

lar FOL theorem prover. As the result turns out (Table 2), only 8% of the pairs can be proven in 3 seconds (the "Orig. 3 Sec." column), and only 16% pairs can be proven in 5 minutes (the "Orig. 5 Min." column), showing severe difficulties for an FOL prover to handle textual inferences with many (usually hundreds of) on-the-fly rules. As such, we use Tian et al. (2014)'s inference engine to pin down statements that are actually needed for proving **H** (usually just 2 or 3 statements), and try to prove **H** by Prover9 again, using only necessary statements. Proven pairs in 5 minutes then jump to 82% (the "Red. 5 Min." column), showing that a large number of on-the-fly rules may drastically increase computation cost. Still, nearly 20% pairs cannot be proven even in this setting, suggesting that traditional FOL prover is not suited for textual inference.

6 Conclusion and future work

We have discussed the role that logical inference could play in RTE task, and the efficiency of performing inference on abstract denotations. Though currently logical inference contributes at places that are somehow inconspicuous, there is the possibility that with some meta level logical models and the methodology of semantic parsing, we can build systems that understand natural language texts deeply: logic implies (in)consistency, which is in turn used as signals to produce more accurate semantic interpretation. And after all, as there may be many possible variations of semantic representations, it is good to have an efficient inference framework that has the potential to connect them. It would be exciting if we can combine different types of structured data with natural language in semantic processing tasks. Directions of our future work are described below.

Improvement of similarity score To calculate phrase similarities, Tian et al. (2014) use the cosine similarity of sums of word vectors, which ignores syntactic information. We plan to add syntactic information to words by some supertags, and learn a vector space embedding for this structure.

Integration of FreeBase to RTE It would be exciting if we can utilize the huge amount of FreeBase data in RTE task. Using the framework of abstract denotation, meanings of sentences can be explained as relational database queries; to convert it to FreeBase data queries is like relational to ontology schema matching. In order to make effective use of FreeBase data, we also need to recognize entities and relations in natural language sentences. Previous research on semantic parsing will be very helpful for learning such mapping.

Winograd Schema Challenge (WSC) As the RTE task, WSC (Levesque et al., 2012) also provides a test bed for textual inference systems. A Winograd schema is a pair of similar sentences but contain an ambiguity of pronouns that is resolved in opposite ways. A complicated partial example is:

> *Michael decided to freeze himself in cryo-stasis even though his father was against it, because he hopes to be unfrozen in the future when there is a cure available.*

The logical interplay among *decided*, *hopes*, *even though*, *because*, and the realization that *he* is coreferent to *Michael* (but not *his father*) is intriguing. By working on the task, we hope to gain further understanding on how knowledge can be gathered and applied in natural language reasoning.

Acknowledgments This research was supported by the Todai Robot Project at National Institute of Informatics.

References

Roy Bar-Haim, Ido Dagan, Iddo Greental, and Eyal Shnarch. 2007a. Semantic inference at the lexical-syntactic level. In *Proceedings of AAAI 2007*.

Roy Bar-Haim, Ido Dagan, Iddo Greental, Idan Szpektor, and Moshe Friedman. 2007b. Semantic inference at the lexical-syntactic level for textual entailment recognition. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Knowledge Representation and Reasoning Conference*.

Percy Liang, Michael Jordan, and Dan Klein. 2011. Learning dependency-based compositional semantics. In *Proceedings of ACL 2011*.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *Proceedings of NAACL 2013*.

Ran Tian, Yusuke Miyao, and Matsuzaki Takuya. 2014. Logical inference on dependency-based compositional semantics. In *Proceedings of ACL 2014*.