Talking with the Theorem Prover to Interactively Solve Natural Language Inference

[1]**Atsushi Sumita**
University of Tokyo

[2]**Koji Mineshima**
Keio University

[3]**Yusuke Miyao**
University of Tokyo

[1]atsushisumita0421@gmail.com
[2]minesima@abelard.flet.keio.ac.jp
[3]yusuke@is.s.u-tokyo.ac.jp

Abstract

Solving natural language inference (NLI) with formal semantics and automated theorem proving has the merit of high precision and interpretability. However, they suffer from non-logical inference such as lexical inference. To overcome this weakness, we propose a human-in-the-loop mechanism in which the user provides non-logical knowledge to the theorem prover. We focus on the **subgoal** of a proof. A subgoal is a proposition that remains unproved by the theorem prover. Although the subgoal conveys what knowledge should be supplemented to the proof, it is a logical formula inscrutable to normal users. To make the subgoal understandable to the user, we propose a method to translate the subgoal into a natural language text. The resulting sentence is called a **readable subgoal**. The reaction to the readable subgoal is communicated to the theorem prover in the form of **temporary axiom**, a logical formula that conveys the content of the reaction. We experimented with our method by adding an interactive component to an existing NLI system, ccg2lambda. The annotators recruited by Amazon Mechanical Turk used the proposed system to solve SICK, a data set for NLI. Experiments show the interaction improved the recall rate of ccg2lambda from 32.1% to 44.1%, and the accuracy from 70.4% to 75.1%.

1 Introduction

Natural Language Inference (NLI) is a task to judge whether a text fragment entails another text fragment (Levy et al., 2013). The former fragment is called the *premise*, and the latter fragment is called the *hypothesis*. NLI has many potential applications such as question answering, summarization, machine translation, and so on (Dagan et al., 2010; White et al., 2017; Bora-Kathariya and Haribhakta, 2018; Demszky et al., 2018; Poliak et al., 2018; Thorne and Vlachos, 2018). Moreover, NLI itself has been considered as an important task to measure progress in natural language understanding (Wang et al., 2018; Wang et al., 2020).

We hypothesize that *interaction* between the machine and the human is useful for NLI.

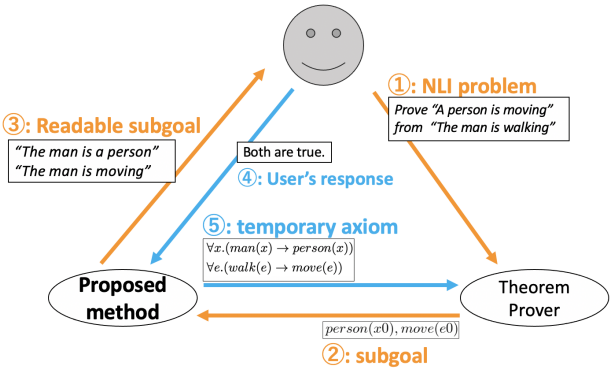

Figure 1: The user enters the premise and the hypothesis, which the semantic parser converts to semantic representations. This is passed to the automated theorem prover, which attempts to prove the hypothesis from the premise. If the proof does not succeed, it outputs the **subgoal**, which is the input to our proposed method. Our method converts the subgoal into **readable subgoal**, so the user can understand the content of the subgoal. The reaction of the user is fed back to the theorem prover as **temporary axioms**, so the proof can continue.

Generally speaking, humans are better at recognizing non-logical inference such as lexical inference or common sense reasoning, while machines are better at comprehensive search and verification over logical relationships. To evaluate this hypothesis, we propose an NLI system based on formal semantics and theorem prover, which asks the user to supplement knowledge when necessary. Importantly, this interaction is fully done *in natural language*, thus users without profession in formal semantics or theorem provers can use the system.

Figure 1 describes an example of an interactive session between the user and the system. The user inputs the premise and the hypothesis she wants to prove. In the example, the user asks the system to prove *A person is moving* from the premise *The man is walking*. The system compiles these sentences into logical representations, and attempts to prove that the hypothesis logically follows from the premise. The theorem prover does this by decomposing the proof into **subgoals**. If all subgoals are proved, the proof is completed. Otherwise, the system translates the unproved subgoal into natural language and returns it to the user. We refer to this natural language text as **readable subgoal**. In this case, the system identified the subgoals $\mathsf{person}(x_0)$ and $\mathsf{move}(e_0)$, and the readable subgoals are *The man is a person* and *The man is moving*. Then the system asks the user if the readable subgoal holds or not under the premise. Finally, the reaction of the user is translated into a logical axiom and is added to the theorem prover to continue the proof. The interaction in natural language enables the end-user to understand what kind of knowledge should be supplemented to the system, and dynamically respond to it.

Based on this mechanism, we expanded an existing NLI system ccg2lambda (Martínez-Gómez et al., 2016), and added an interaction mechanism. We evaluated the proposed system with SICK (Marelli et al., 2014), a data set that contains NLI problems combining logical inference and lexical inference. Annotators are recruited by Amazon Mechanical Turk as the user of the system. Experiments show the interaction improved the recall rate of ccg2lambda from 32.1% to 44.1%, and the accuracy from 70.4% to 75.1%. This suggests the machine benefits from interaction with the user. We observe that our method successfully solves NLI problems that require phrasal knowledge, which previous methods struggle to solve.

2 Related Works

2.1 Formal Semantics and Theorem Proving

One approach to NLI is methods based on formal semantics and automated theorem proving (Bos et al., 2004; Mineshima et al., 2015; Martínez-Gómez et al., 2016; Abzianidze, 2017). These methods have high interpretability since the inference process is explicitly modeled, and also is trustworthy since they produce very few false positives. Moreover, this approach can handle complex linguistic phenomena that deep learning based models struggle to solve (Geiger et al., 2018; Yanaka et al., 2019; Yanaka et al., 2021). Another advantage is that these approaches do not require parameter learning,[1] while achieving competitive results to deep learning-based approaches.

However, logic-based approaches suffer from a low recall rate, mainly caused by non-logical linguistic knowledge such as lexical inference, common sense reasoning, etc. Several studies proposed mechanisms to integrate knowledge base or knowledge base completion models to overcome this issue (Martínez-Gómez et al., 2017; Yanaka et al., 2018; Yoshikawa et al., 2019). For example, the proposition that *Apple is a fruit* is represented as $\forall x.(\mathsf{apple}(x) \rightarrow \mathsf{fruit}(x))$, which is not a logically derived theorem but an *axiom*, which the user has to provide from outside of the theorem prover. However, maintaining knowledge bases is costly and it is hard to enumerate all possible lexical relations including phrasal knowledge and neologism. Even a single presence of uncovered lexical relation or paraphrase makes the proof uncompleted. Note that lexical inference is context-sensitive (Levy and Dagan, 2016; Schmitt and Schütze, 2019), and a static knowledge base cannot handle this issue.

2.2 Natural Language Interface for Theorem Proving

In the context of automated theorem proving, several works propose presenting the output of theorem

[1] Stricly speaking, CCG parser requires parameter tuning to resolve the ambiguity of syntactic structure.

prover in natural language (Felty and Miller, 1988; Huang and Fiedler, 1996; Ganesalingam and Gowers, 2017). They hypothesize that the inscrutable output of the theorem prover is one of the obstacles for the theorem proving techniques to be widely adopted. These works employ rule-based systems to translate the output of the theorem prover into natural language text, so the user can better understand and communicate with the theorem prover.

This branch of work has a similar spirit to ours in that they aim to make the output of the theorem prover more accessible to the human user by the translation from formal language to natural language. The difference is that these methods aim to generate a complete human-style proof for mathematical theorems, while we aim to translate a proposition (i.e. subgoal) into a natural language text for better interaction. Additionally, to the best of our knowledge, our work is the first attempt to apply the idea of natural language interface to NLI.

3 Formal Semantics and Automated Theorem Proving

In this section, we briefly introduce formal semantics and automated theorem proving, since our method is based on these techniques.

Formal semantics is a theory of meaning which models the semantics of natural language by tools from logic. We employ Neo-Davidsonian semantics (Parsons, 1990; Maienborn, 2011), a formal theory for the meaning of language. The characteristics of this framework are that it can model references to events (such as adverbs referring to some event), while staying in the first order logic scheme. This is achieved by the two types of variables, *entity* and *event*. For example, following the standard Neo-Davidsonian approach, the sentence *John quickly eats the apple* is modeled as follows.

$$\begin{aligned}&\exists x_1, x_2, e.(\,\mathsf{John}(x_1) \wedge \mathsf{eat}(e) \wedge \mathsf{quick}(e)\\ &\wedge (\mathsf{Subj}(e) = x_1) \wedge \mathsf{apple}(x_2) \wedge (\mathsf{Obj}(e) = x_2))\end{aligned} \tag{1}$$

Here, x_1, x_2 are the entity variables, and e is the event variable. The content words appearing in the original sentence are modeled as a predicate that takes the relevant variables. The *semantic role* is modeled by the formula $\mathsf{Subj}(e) = x$, which denotes that the subject of the event e is the entity referred by x_1, and $\mathsf{Obj}(e) = x_2$, which denotes that the object of the event e is the entity referred by x_2. For more details for Neo-Davidsonian semantics, refer to (Parsons, 1990; Maienborn, 2011).

Now consider the following NLI problem.

T : A boy is throwing a ball.

H : A tall kid is pitching a ball in a yard.

By utilizing the framework of Neo-Davidsonian semantics, this NLI problem can be represented as the following implication relation.

$$\begin{aligned}&\exists x_0, x_1, e_0.(\mathsf{boy}(x_0) \wedge \mathsf{ball}(x_1) \wedge (\mathsf{Subj}(e_0) = x_0)\\ &\wedge(\mathsf{Obj}(e_0) = x_1) \wedge \mathsf{throw}(e_0)\\ &\Rightarrow \exists x_2, x_3, x_4, e_1.(\mathsf{kid}(x_2) \wedge \mathsf{tall}(x_2) \wedge \mathsf{ball}(x_3)\\ &\wedge(\mathsf{Subj}(e_1) = x_2) \wedge (\mathsf{Obj}(e_1) = x_3)\\ &\wedge\mathsf{pitch}(e_1) \wedge \mathsf{in}(e_1, x_4) \wedge \mathsf{yard}(x_4)\end{aligned} \tag{2}$$

Since these formulas are based on the standard first order logic, we can use an *automated theorem prover* off the shelf. The theorem prover decomposes the proof into smaller pieces, which are called **subgoals**. We employ Natural Deduction (Indrzejczak, 2010), a formal system for deduction. By the elimination rules of Natural Deduction, the initial subgoals are the individual logical formula, as shown below.

$$\begin{aligned}&\mathsf{kid}(x_2), \mathsf{tall}(x_2), \mathsf{ball}(x_3),\\ &\mathsf{Subj}(e_1) = x_2, \mathsf{Obj}(e_1) = x_3, \mathsf{pitch}(e_1),\\ &\mathsf{in}(e_1, x_4), \mathsf{yard}(x_4)\end{aligned} \tag{3}$$

When all of these subgoals are proved, the entire proof is constructed. The theorem prover solves the **unification** problem, in that it finds a substitution $\{x_0 := x_2, x_1 := x_3, e_0 := e_1\}$ to prove the subgoals. For this example, many subgoals are proved, such as $\mathsf{ball}(x_2)$ or $\mathsf{Subj}(e_1) = x_2$. However, the subgoal $\mathsf{kid}(x_2)$ is not proved. This is because we did not provide the axiom for the lexical knowledge, $\forall x.\mathsf{boy}(x) \rightarrow \mathsf{kid}(x)$, thus the theorem prover has no way proving the subgoal. As a result, the theorem prover fails to find a proof for the NLI problem in this case. Our method aims to solve this kind of inflexibility of the theorem prover, by enabling interaction.

4 Method

We hypothesize that the **interaction** with the human user and the theorem prover is effective for solving NLI problems. We aim to enable the communication from the theorem prover to the human *in natural language*, so the method can be applicable to the wider population without profession in formal semantics or theorem proving. Then the problem is how to communicate from one side to the other side. From the theorem prover, what information is lacking from the proof should be conveyed to the human, in the form of natural language. From the user, what is the correct decision to make for that information should be conveyed to the theorem prover, in the form of logical axioms.

From the theorem prover to the user, we focus on the **subgoal** produced by the theorem prover and propose to translate them into natural language text. We call the natural language text which represents the meaning of the subgoal as **readable subgoal**. From the user to the theorem prover, the reaction to the readable subgoal should be compiled into a logical axiom for the theorem prover to continue the proof. This axiom can be *temporary* in that it only holds in the current context, rather than a general axiom that holds across context (which would be much more difficult to obtain.). We define **temporary axiom** as the logical axiom which represents the semantics of the user's response to the readable subgoal. The temporary axiom is passed to the theorem prover so it can continue the proof.

Consider the formula (2) as an example. We know which subgoals are not proved from the theorem prover, shown in formula (3). By applying the substitution $\{x_1 := x_3, x_2 := x_4, e_1 := e_2\}$ found by the theorem prover, we obtain the following.

$$\mathsf{tall}(x_0), \mathsf{pitch}(e_0), \mathsf{in}(e_1, ?x), \mathsf{yard}(?x)$$

The symbol $?x$ indicates there was no corresponding entity in the premise (i.e. the unification process could not find a mapping of the variable x_4 with a variable in the premise.) These are the input of our proposed system. The main problem is how to create the readable subgoal from the subgoal. Because the subgoal is an abstract artifact to model natural language meaning, it itself does not correspond to any natural language expression. Thus, we need to identify what information has to be restored for the subgoal to be meaningful to the human user. Below we discuss how to generate the readable subgoal, and also the corresponding temporary axiom.

4.1 From the prover to the user

The problem of the translation from the subgoal to natural language text is that the subgoal is an abstract representation of meaning, and lacks many aspects that cannot be omitted from a valid natural language sentence. For example, the subgoal does not contain the subject of the sentence or functional words that connect the words to form a sentence. The following describes how we approach this problem and generate a readable subgoal from the premise, hypothesis, and subgoal.

4.1.1 Identification of Necessary Variables

The first step is to identify the set of variables that are necessary for the construction of the readable subgoal. Consider the case of $\mathsf{tall}(x_0)$. We cannot just say *The theorem prover cannot prove: tall*. The human user needs to know what is asserted to be *tall*. In other words, we need to identify the entity which represents the subject of the readable subgoal. In this case, the entity asserted to be tall is x_0, which can be used as the subject of the sentence.

When the argument of the subgoal is an event type variable, the subject is determined by the semantic role. For example, for the subgoal $\mathsf{pitch}(e_0)$ which has a verb predicate, the entity for the subject is tracked from the term $\mathsf{Subj}(e_0) = x_0$.

Observe that when the subgoal takes an event variable as the argument, it is necessary to include variables related to the event by semantic role. Consider the case of $\mathsf{pitch}(e_1)$. We cannot just say *The theorem prover cannot prove: pitch*. The human user needs to know what is pitched, in order to interpret the subgoal. We know this by $\mathsf{Obj}(e_0) = x_0$, and we know we should include the variable x_0 as well.

As a result, we identify $\{Subgoal : tall, Subj : x_0\}$, and $\{Subgoal : pitch, Subj : x_0, Obj : x_1\}$ for the above mentioned examples.

4.1.2 Translation of the Variables

Next, we need to translate each variable into natural language so the user can understand what the

content of the variables is. In this step, we identify the dictionary which maps each variable to a word in the premise.

For entity type variables, we select the noun which takes the entity variable as an argument, since the noun is typically the name of the entity. There may be several nouns associated with one entity variable, so we must choose one from them. We choose the first noun appearing in the premise, since the first noun tends to play a role to introduce the entity to the reader, and is an appropriate way to describe what the entity is. As a result, for the subgoal $\mathsf{tall}(x_0)$, we have $\{Subgoal : \text{“}tall\text{”}, Subj : \text{“}boy\text{”}\}$.

For the same reason, we select the first verb which takes the event type variable as an argument for the name of the event. As a result, for the subgoal $\mathsf{pitch}(e_0)$, we have $\{Subgoal : \text{“}pitch\text{”}, Subj : \text{“}boy\text{”}, Obj : \text{“}ball\text{”}\}$.

Note that this method fails when there are multiword expressions. We use this method as an approximation since multiword expressions are mostly removed in the data set we use. In general, the additional process to chunk multiword expression should be necessary.

4.1.3 Word Order and Functional Words

At this point, we have the words which consist of the readable subgoal and their semantic roles. In the current example, $\{Subgoal : \text{“}tall\text{”}, Subj : \text{“}boy\text{”}\}$, $\{Subgoal : \text{“}pitch\text{”}, Subj : \text{“}boy\text{”}, Obj : \text{“}ball\text{”}\}$.

The final step is to select a valid word order and appropriate functional words. Since the readable subgoal is a very simple declarative sentence, the task of generating a sentence is relatively a simple task. We employ a simple rule-based system to determine the correct sentence structure. The system selects the correct copula and determiner and determines the correct word order. In the current example, we output the readable subgoal *The boy is tall* for $\mathsf{tall}(x_0)$, and *The boy is pitching a ball* for $\mathsf{pitch}(e_0)$. Further details of the template is summarized in the appendix.

4.1.4 Subgoal with Ununified Variables

Sometimes, the argument of the subgoal is not unified to any variables in the premise. For example, in formula 2, the prepositional phrase *in the yard* introduces a new entity *yard*. Since there is no corresponding entity in the premise, this entity is not unified to any variable in the premise (i.e. the subgoal is ununified).

Actually, these ununified subgoals can be ignored. These ununified variables appear because some event variable introduces a new entity, but in the procedure explained in section 4.1.1, we collect these arguments in the set of variables. Thus, the information of the ununified subgoal is already contained in the readable subgoal for the event variable which introduced the ununified variable. In the current example, observe that the readable subgoal for $\mathsf{in}(e_1, ?x)$ is *The boy is in a yard*, which asks about the subgoal $\mathsf{yard}(?x)$ at the same time.

4.1.5 Hypothesis with Negation

When the hypothesis is negated, the subgoals are more complicated. In theorem proving, negation can be modeled as follows.

$$\neg A(x) \Leftrightarrow (\forall x.A(x) \Rightarrow \bot)$$

Consider the negation of **H** in the current example; *The tall kid is not pitching a ball in a yard.* Then only one subgoal is produced, which is

$$\begin{aligned}&(\forall x_5, x_6, x_7, e_2.(\mathsf{kid}(x_5) \wedge \mathsf{tall}(x_5) \wedge \mathsf{ball}(x_6)\\ &\wedge (\mathsf{Subj}(e_1) = x_6) \wedge (\mathsf{Obj}(e_1) = x_6) \wedge \mathsf{pitch}(e_2)\\ &\wedge \mathsf{in}(e_2, x_7) \wedge \mathsf{yard}(x_7)) \Rightarrow \bot\end{aligned} \tag{4}$$

This subgoal is clearly more complicated than those in formulae (4). The subgoal becomes monolithic and is more difficult to translate in natural language expression.

To alleviate this problem, we use the following procedure. First, we remove every negation from the hypothesis, by recursively checking each term. Then, because the hypothesis is not negated, we can reuse the above-mentioned process to generate the readable subgoal. In this case, subgoals are decomposed enough and contain only one predicate, and still useful for the original problem where the hypothesis is negated. For example, the readable subgoal *The boy is pitching a ball* results in $\forall e.\mathsf{throw}(e) \Rightarrow \mathsf{pitch}(e)$, which is a useful axiom when proving the subgoal (4).

4.2 From the User to the Prover

After the readable subgoals are generated, they are shown to the human user. The human reacts to the subgoal, and that reaction should be represented as a temporary axiom so the theorem prover can retry the proof with the new information. Below we describe what reaction user can choose, and how the decision is converted into a temporary axiom.

4.2.1 Choice of the User

In principle, the user can react to the readable subgoal in free text, such as *Yes, that's true* or *I don't know, it depends*, and so on. However, as a simplification, we limit the choice the user can make to the following four choices; {*Yes, No, Unknown, Nonsense*}. Based on which is chosen, our system takes a different action, as described in the next section.

4.2.2 Creation of the Temporary Axiom

After the user responds to the readable subgoal, the result is fed back to the theorem prover. When the user responded by *Yes*, the readable subgoal is compiled into a logical axiom. For example, when the user responded to the readable subgoal *The boy is pitching a ball* by *true*, the following axiom is created.

$$\forall e.\mathsf{throw}(e) \rightarrow \mathsf{pitch}(e)$$

If the user chooses *No* instead, then the postcedent is negated;

$$\forall e.\mathsf{throw}(e) \rightarrow \neg\mathsf{pitch}(e)$$

Else if the user chooses *Unknown* or *Nonsense*, no action is taken since there is no new information available for the proof to proceed. Note that the above logical axioms are only valid under the current proof and not a global axiom (such as $\forall x, \mathsf{apple}(x) \rightarrow \mathsf{food}(x)$), which is desirable in terms of flexibility since the user can inject context-dependent axioms when they want.

5 Experiment

5.1 Data

We use SICK (Marelli et al., 2014) as the data set to evaluate our system. SICK mostly restricts the inference type to simple logical inference and lexical inference, and also removes complicated semantic phenomena such as a multiword expression or named entity recognition, so as to enable evaluation of NLI system based on compositional semantics. In addition, this data set is used in previous studies that attempt to combine logical inference and lexical inference (Martínez-Gómez et al., 2017; Yanaka et al., 2018), and it is desirable to compare the performance of our methods to them. SICK consists of train spilt and test split, both containing 5000 pairs of premise and hypothesis. There are three labels *entailment*, *contradiction*, *neutral* for each premise-hypothesis pair, and the task is to predict these three classes given the sentence pair.

Label	N	Ratio	Perfect Agreement rate
Yes	5489	43.6%	23.1%
No	3053	24.2%	7.8%
Unknown	2051	16.3%	4.5%
Nonsense	2008	15.9%	4.5%
Total	12600	1	40%

Table 1: The distribution of the labels provided by the annotators. *N* shows the number of labels annotated by the crowd workers. There was 4200 readable subgoals generated, and each was annotated by three workers, which results in total $4200 \times 3 = 12600$ labels. *Agreement rate* is the number of pairs where perfect agreement occurs among the three annotators, divided by the total number of readable subgoal generated, 3342.

5.2 Experimental setup

We use an A* CCG parsing model proposed by Yoshikawa et al. (2017) for assigning CCG tags, available at https://github.com/masashi-y/depccg. For the semantic parser, we use ccg2lambda (Martínez-Gómez et al., 2016), available at https://github.com/mynlp/ccg2lambda. For the theorem prover we use Coq (Bertot and Castéran, 2013). We used Amazon Mechanical Turk to recruit users to annotate the readable subgoals. Each readable subgoal was annotated by three workers. When the annotators all agree on the label we use the decision.[2] Otherwise, no further action is taken and the prediction is *neutral*.

5.3 Results

Table 1 shows the distribution of the labels annotators provided. We observe that, although we use

[2]We also experimented using majority vote resulted in noisy result, however, led to much lower precision and accuracy.

	Methods		Prec.	Rec.	Acc.
	KB	readable subgoal			
Baseline					56.6
ccg2lambda			99.9	32.1	70.4
ccg2lambda & abduction	✓		99.7	43.0	75.1
Ours		✓	98.9	44.1	75.1
Ours & abduction	✓	✓	98.9	45.0	75.5

Table 2: Results for SICK test split. Checkmarks indicate whether the system uses Knowledge Base for axiom injection (Martínez-Gómez et al., 2017) and/or readable subgoal (our method). *Prec.* show the precision, *Rec.* shows the recall rate, *Acc.* shows the accuracy.

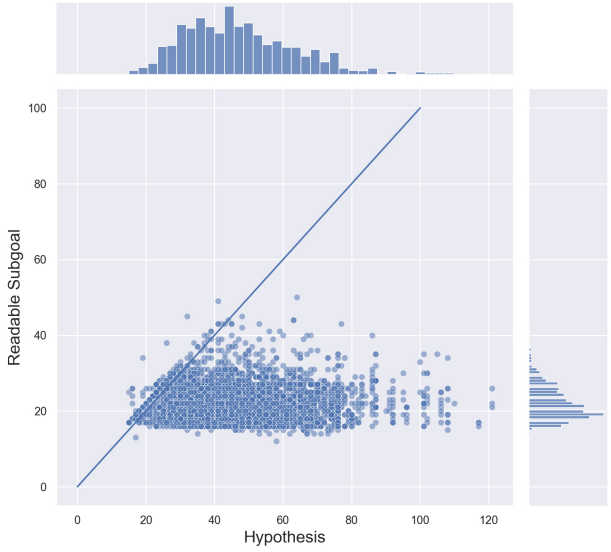

Figure 2: The character length of the original hypothesis (x-axis) vs readable subgoal (y-axis). The readable subgoals are mostly shorter than the original hypothesis, which indicates the readable subgoal decomposes the original problem and makes the problem easier to process.

perfect agreement as to the aggregation criteria, the inter-annotator agreement rate is rather low. In total, 40% of the readable subgoals were used as a valid response. Note that the agreement rate for *Yes* is much higher than other labels. The reason for this may be that there is some ambiguity in deciding between *No, Unknown, Nonsense*.

Figure 2 shows the character lengths of the readable subgoal and the original hypothesis. We see that most of the hypothesis is distributed in the range 30-60, while most readable subgoals are shorter than 30 characters. This shows that in most cases the readable subgoal decomposes the original problem into smaller pieces, and makes the problem easier to process.

The evaluation metrics of the prediction quality are summarized in Table 2. The baseline is the majority class. In SICK, *neutral* is the majority class, which consists of 56.6% of this subset of the data. The second row is the result obtained from ccg2lambda (Martínez-Gómez et al., 2016), where no axiom injection mechanism was used. It achieves near-perfect precision for this subset of the data, and it has better accuracy than the majority class baseline. However, the recall rate is 32.1% which is relatively low, leading to degraded accuracy. The third row is the result when ccg2lambda is enhanced with the word level abduction mechanism (Martínez-Gómez et al., 2017), which leads to a better recall rate of 43.0%. The fourth row shows the result of our proposed method. Our method achieves the best recall rate of 44.1%, with slightly low precision, leading to comparable accuracy with the word abduction mechanism. The fifth row shows the result when the word level abduction mechanism and the proposed method were combined, by using the readable subgoal only when the abduction mechanism could not prove entailment. The recall rate and the accuracy is the best among the compared methods, which indicates the interaction process can solve problems that existing methods struggle to solve.

In the next section, we see instances where our method predicts the correct label but the existing abduction mechanism fails, and see if they have different tendencies.

Sentence Pair	Gold.	Pred.	Readable subgoal
T: A man with a helmet painted red is riding a blue motorcycle down the road. **H**: A motorcyclist with a red helmet is riding a blue motorcycle down the road.	Ent.	Ent.	The helmet is red. The man is a motorcyclist.
T: A surfer is riding the wave. **H**: A rider is surfing the wave	Ent.	Ent.	The surfer is surfing the wave. The surfer is a rider.
T: A man is speaking on a stage **H**: A man is speaking on a podium	Unk.	Ent.	The stage is a podium

Table 3: *Sentence Pair* shows the premise-hypothesis pair from SICK. *Gold.* shows the correct label of the NLI problem. *Pred.* shows the predicted label of our system. *The readable subgoal* shows the readable subgoal generated by our system.

5.4 Positive and Negative Examples

We conducted an additional experiment by repeating the same experiment on the randomly sampled 1000 pairs from the train split of SICK. We focus on pairs where our method correctly predicts entailment, but the existing method could not (positive examples). These examples are the cause of the increase in the recall rate. We also provide some examples where the system outputs false positive predictions (negative examples). These examples are the cause of a decrease in precision. Examples are shown in Table 3.

The pair in the first row shows a positive example. The required knowledge here is phrasal, and thus the word abduction system cannot predict *Yes*. More formally, the knowledge used in the readable subgoal *The helmet is red* is:

$$\forall x, y, e.(\mathsf{paint}(e) \wedge \mathsf{red}(x) \\ \wedge (\mathsf{Obj}(e) = x) \wedge (\mathsf{Dat}(e) = y) \rightarrow \mathsf{red}(y))$$

The other readable subgoal, *The man is a motorcyclist*, requires the following knowledge.

$$\forall e, x.(\mathsf{ride}(e) \wedge \mathsf{man}(x) \wedge \mathsf{motorcycle}(y) \\ \rightarrow \mathsf{motorcyclist}(x))$$

The pair in the second row is also an example where a phrasal knowledge *ride a wave* implies *surf*. We observe similar pairs of a premise and a readable subgoal, shown below.

- *A man is having lunch → A man is eating*.
- *The man is aiming a gun. → The man is handling a gun.*
- *A person is cutting an onion into pieces. → A person is dicing an onion.*

The pair in the third row shows a frequent pattern, where the readable subgoal is labeled as *Yes* but the correct label should be *Unknown*. This is because, strictly speaking, not all stages where someone makes a speech are podiums. However, it is questionable that this judgment is completely wrong; it may be reasonable enough to infer that a stage where someone is speaking is a podium. The definition of entailment in NLI is different from entailment in logic,[3] and this inference may be acceptable depending on the context. The flexibility of our method is that the human user can change the judgment according to the context. Similar examples are shown below.

- *A man is dropping a tree → A man is carrying a tree*
- *A man is thinking. → A man is not dancing.*
- *A kid is waiting wearing swim gear → A kid is sitting wearing swim gear*

From these analyses, we hypothesize that in case there is more phrasal knowledge required, our method gains more recall rate and accuracy compared to existing methods. SICK is a data set that is designed to reduce this kind of phrasal knowledge and is a good choice for the initial evaluation of NLI systems based on compositional semantics. Future

[3]The most well-accepted definition of entailment in NLI is that *A human reading the premise would infer that hypothesis is most probably true.* (Dagan et al., 2005; Dagan et al., 2010);

work should experiment on more realistic data sets with more complex inference phenomena such as common sense reasoning to verify if this conjecture is true or not.

6 Conclusion

We proposed a human-in-the-loop mechanism in which the user provides non-logical knowledge to the theorem prover. The interaction is done in natural language, making the method applicable to general users. We proposed to generate readable subgoals, which is a translation from the subgoal to a natural language text. The reaction to the readable subgoal is converted to a logical axiom, namely the temporary axiom. We experimented with our method by adding an interactive component to an existing NLI system, ccg2lambda. Experiments show the interaction improved the recall rate of ccg2lambda from 32.1% to 44.1%, and the accuracy from 70.4% to 75.1%.

Future works should consider whether the proposed approach is feasible in more complex settings. For example, inference which requires common sense or complex domain knowledge (e.g. law, medicine) are not covered in this study. We expect that since it is difficult to develop a complete and up to date knowledge base for these cases, human interaction may be more suitable than knowledge bases in these cases.

References

Lasha Abzianidze. 2017. LangPro: Natural language theorem prover. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 115–120, Copenhagen, Denmark, September. Association for Computational Linguistics.

Yves Bertot and Pierre Castéran. 2013. *Interactive Theorem Proving and Program Development: Coq'Art: The Calculus of Inductive Constructions*. Springer.

R. Bora-Kathariya and Y. Haribhakta. 2018. Natural language inference as an evaluation measure for abstractive summarization. In *2018 4th International Conference for Convergence in Technology (I2CT)*, pages 1–4.

Johan Bos, Stephen Clark, Mark Steedman, James R. Curran, and Julia Hockenmaier. 2004. Wide-coverage semantic representations from a CCG parser. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 1240–1246, Geneva, Switzerland, aug 23–aug 27. COLING.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In Joaquin Quiñonero Candela, Ido Dagan, Bernardo Magnini, and Florence d'Alché-Buc, editors, *Machine Learning Challenges, Evaluating Predictive Uncertainty, Visual Object Classification and Recognizing Textual Entailment, First PASCAL Machine Learning Challenges Workshop, MLCW 2005, Revised Selected Papers*, volume 3944 of *Lecture Notes in Computer Science*, pages 177–190. Springer.

Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. 2010. Recognizing textual entailment: Rational, evaluation and approaches - erratum. *Nat. Lang. Eng.*, 16(1):105.

Dorottya Demszky, Kelvin Guu, and Percy Liang. 2018. Transforming question answering datasets into natural language inference datasets. *CoRR*, abs/1809.02922.

Amy Felty and Dale Miller. 1988. Proof explanation and revision. Technical report, Technical Reports (CIS).

M. Ganesalingam and W. T. Gowers. 2017. A Fully Automatic Theorem Prover with Human-Style Output. *Journal of Automated Reasoning*, 58(2):253–291.

Atticus Geiger, Ignacio Cases, Lauri Karttunen, and Christopher Potts. 2018. Stress-testing neural models of natural language inference with multiply-quantified sentences. *CoRR*, abs/1810.13033.

Xiaorong Huang and A. Fiedler. 1996. Presenting machine-found proofs. In *CADE*.

Andrzej Indrzejczak. 2010. *Natural Deduction, Hybrid Systems and Modal Logics*, volume 30. Springer.

Omer Levy and Ido Dagan. 2016. Annotating relation inference in context via question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 249–255, Berlin, Germany, August. Association for Computational Linguistics.

Omer Levy, Torsten Zesch, Ido Dagan, and Iryna Gurevych. 2013. Recognizing partial textual entailment. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 451–455, Sofia, Bulgaria, August. Association for Computational Linguistics.

Claudia Maienborn. 2011. Event semantics. In Claudia Maienborn, Klaus von Heusinger, and Paul Portner, editors, *Semantics. An International Handbook of Natural Language Meaning*, volume 1, pages 802–829. Mouton de Gruyter.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zampar-

elli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland, May. European Language Resources Association (ELRA).

Pascual Martínez-Gómez, Koji Mineshima, Yusuke Miyao, and Daisuke Bekki. 2016. ccg2lambda: A compositional semantics system. In *Proceedings of ACL-2016 System Demonstrations*, pages 85–90, Berlin, Germany, August. Association for Computational Linguistics.

Pascual Martínez-Gómez, Koji Mineshima, Yusuke Miyao, and Daisuke Bekki. 2017. On-demand injection of lexical knowledge for recognising textual entailment. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 710–720, Valencia, Spain, April. Association for Computational Linguistics.

Koji Mineshima, Pascual Martínez-Gómez, Yusuke Miyao, and Daisuke Bekki. 2015. Higher-order logical inference with compositional semantics. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2055–2061, Lisbon, Portugal, September. Association for Computational Linguistics.

Terence Parsons. 1990. *Events in the Semantics of English*. MIT Press.

Adam Poliak, Yonatan Belinkov, James Glass, and Benjamin Van Durme. 2018. On the evaluation of semantic phenomena in neural machine translation using natural language inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 513–523, New Orleans, Louisiana, June. Association for Computational Linguistics.

Martin Schmitt and Hinrich Schütze. 2019. SherLIiC: A typed event-focused lexical inference benchmark for evaluating natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 902–914, Florence, Italy, July. Association for Computational Linguistics.

James Thorne and Andreas Vlachos. 2018. Automated fact checking: Task formulations, methods and future directions. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3346–3359, Santa Fe, New Mexico, USA, August. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium, November. Association for Computational Linguistics.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2020. Superglue: A stickier benchmark for general-purpose language understanding systems.

A. S. White, P. Rastogi, K. Duh, and B. Van Durme. 2017. Inference is everything: Recasting semantic resources into a unified evaluation framework. *Proceedings of the The 8th International Joint Conference on Natural Language Processing (IJCNLP'17)*, pages 996–1005.

Hitomi Yanaka, Koji Mineshima, Pascual Martínez-Gómez, and Daisuke Bekki. 2018. Acquisition of phrase correspondences using natural deduction proofs. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 756–766, New Orleans, Louisiana, June. Association for Computational Linguistics.

Hitomi Yanaka, Koji Mineshima, Daisuke Bekki, Kentaro Inui, Satoshi Sekine, Lasha Abzianidze, and Johan Bos. 2019. Can neural networks understand monotonicity reasoning? In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 31–40, Florence, Italy, August. Association for Computational Linguistics.

Hitomi Yanaka, Koji Mineshima, and Kentaro Inui. 2021. Exploring transitivity in neural NLI models through veridicality. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 920–934, Online, April. Association for Computational Linguistics.

Masashi Yoshikawa, Hiroshi Noji, and Yuji Matsumoto. 2017. A* CCG parsing with a supertag and dependency factored model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 277–287, Vancouver, Canada, July. Association for Computational Linguistics.

Masashi Yoshikawa, Hiroshi Noji, Koji Mineshima, and Daisuke Bekki. 2019. Automatic generation of high quality CCGbanks for parser domain adaptation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 129–139, Florence, Italy, July. Association for Computational Linguistics.