Determining Semantic Textual Similarity using Natural Deduction Proofs

Hitomi Yanaka[1]
hitomiyanaka@g.ecc.u-tokyo.ac.jp

Koji Mineshima[2]
mineshima.koji@ocha.ac.jp

Pascual Martínez-Gómez[3]
pascual.mg@aist.go.jp

Daisuke Bekki[2]
bekki@is.ocha.ac.jp

[1]The University of Tokyo
[2]Ochanomizu University
[3]Artificial Intelligence Research Center, AIST
Tokyo, Japan

Abstract

Determining semantic textual similarity is a core research subject in natural language processing. Since vector-based models for sentence representation often use shallow information, capturing accurate semantics is difficult. By contrast, logical semantic representations capture deeper levels of sentence semantics, but their symbolic nature does not offer graded notions of textual similarity. We propose a method for determining semantic textual similarity by combining shallow features with features extracted from natural deduction proofs of bidirectional entailment relations between sentence pairs. For the natural deduction proofs, we use ccg2lambda, a higher-order automatic inference system, which converts Combinatory Categorial Grammar (CCG) derivation trees into semantic representations and conducts natural deduction proofs. Experiments show that our system was able to outperform other logic-based systems and that features derived from the proofs are effective for learning textual similarity.

1 Introduction

Determining semantic textual similarity (STS) is one of the most critical tasks in information retrieval and natural language processing. Vector-based sentence representation models have been widely used to compare and rank words, phrases or sentences using various similarity and relatedness scores (Wong and Raghavan, 1984; Mitchell and Lapata, 2010; Le and Mikolov, 2014). Recently, neural network-based sentence representation models (Mueller and Thyagarajan, 2016; Hill et al., 2016) have been proposed for learning textual similarity. However, these vector-based models often use shallow information, such as words and characters, and whether they can account for phenomena such as negation and quantification is not clear. Consider the sentences: *Tom did not meet some of the players* and *Tom did not meet any of the players*. If functional words such as *some* or *any* are ignored or represented as the same vector, then these sentences are to be represented by identical vectors. However, the first sentence implies that there is a player who Tom did not meet, whereas the second sentence means that Tom did not meet anyone, so the sentences have different meanings.

Conversely, logic-based approaches have been successful in representing the meanings of complex sentences, having had a positive impact for applications such as recognizing textual entailment (Mineshima et al., 2015, 2016; Abzianidze, 2015, 2016). However, purely logic-based approaches only assess entailment or contradiction relations between sentences and do not offer graded notions of semantic similarity.

In this paper, we propose to leverage logic cues to learn textual similarity. Our hypothesis is that *observing proof processes when testing the semantic relations is predictive of textual similarity*. We show that our approach can be more effective than systems that ignore these logic cues.

2 Related Work

Vector-based models of semantic composition have been widely studied with regards to calculating STS. Mitchell and Lapata (2008, 2010) pro-

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 692–702
Copenhagen, Denmark, September 7–11, 2017.

posed a sentence vector model involving word vector addition or component-wise multiplication. Addition and multiplication are commutative and associative and thus ignore word order. Polajnar et al. (2015) proposed a discourse-based sentence vector model considering extra-intra sentential context. Also, a categorical compositional distributional semantic model has been developed for recognizing textual entailment and for calculating STS (Grefenstette and Sadrzadeh, 2011; Kartsaklis et al., 2014; Kartsaklis and Sadrzadeh, 2016). However, these previous studies are mostly concerned with the structures of basic phrases or sentences and do not address logical and functional words such as negations and connectives. Neural network-based models of semantic composition (Mueller and Thyagarajan, 2016; Hill et al., 2016) have also been proposed. Although these models achieve higher accuracy, their end-to-end nature introduces challenges in the diagnosis of the reasons that make two sentences to be similar or dissimilar to each other. These diagnosis capabilities may play an important role in making the system explainable and also to guide future system improvements in a more precise manner. Our approach presented in this paper is partially inspired by the latter two objectives.

Meanwhile, some previous studies have proposed logic systems for capturing the semantic relatedness of sentences. The Meaning Factory (Bjerva et al., 2014) uses both shallow and logic-based features for learning textual similarity. In this system, the overlap of predicates and entailment judgments are extracted as logic-based features. UTexas (Beltagy et al., 2014b) uses Probabilistic Soft Logic for learning textual similarity. In this system, each ground atom in the logical formulas has a probability based on distributional semantics of a word. The weights of the logical formulas are calculated from the probabilities of their ground atoms and are extracted as features. These previous studies improved the accuracy by using logic-based features derived from the entailment results of first-order theorem proving in addition to using shallow features such as sentence lengths.

In our study, we determine the semantic similarity of sentences based on the conception of proof-theoretic semantics (Bekki and Mineshima, 2017). The key idea is that not only the entailment results but also the *theorem proving process* can be considered as features for learning textual similarity. That is, by taking into account not only whether a theorem is proved but also *how* it is proved, we can capture the semantic relationships between sentence pairs in more depth.

Another difference between our study and previous logic systems is that we use higher-order predicate logic. Higher-order predicate logic is able to represent complex sentence semantics such as generalized quantifiers more precisely than first-order predicate logic. In addition, higher-order predicate logic makes the logical structure of a sentence more explicit than first-order predicate logic does, so it can simplify the process of proof search (Miller and Nadathur, 1986).

3 System Overview

Figure 1 shows an overview of the system which extracts features for learning textual similarity from logical proofs. To produce semantic representations of sentences and prove them automatically, we use ccg2lambda (Martínez-Gómez et al., 2016), which is a semantic parser combined with an inference system based on natural deduction.

First, sentences are parsed into syntactic trees based on Combinatory Categorial Grammar (CCG) (Steedman, 2000). CCG is a syntactic theory suitable for semantic composition from syntactic structures. Meaning representations are obtained based on semantic templates and combinatory rules for the CCG trees. Semantic templates are defined manually based on formal semantics. Combinatory rules specify the syntactic behaviors of words and compositional rules for the CCG trees. In ccg2lambda, two wide-coverage CCG parsers, C&C (Clark and Curran, 2007) and EasyCCG (Lewis and Steedman, 2014), are used for converting tokenized sentences into CCG trees robustly. According to a previous study (Martínez-Gómez et al., 2017), EasyCCG achieves higher accuracy. Thus, when the output of both C&C and EasyCCG can be proved, we use EasyCCG's output for creating features.

Second, the meanings of words are described using lambda terms. Semantic representations are obtained by combining lambda terms in accordance with the meaning composition rules specified in the CCG tree. The semantic representations are based on Neo-Davidsonian event semantics (Parsons, 1990; Mineshima et al., 2015), in which every verb is decomposed into a predicate over events and a set of functional expressions

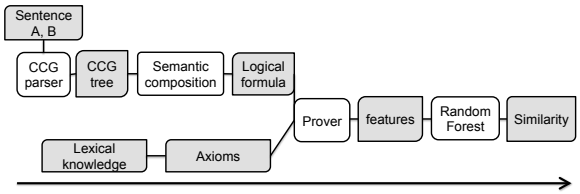

Figure 1: System overview.

relating the events. Adverbs and prepositions are also represented as predicates over events.

Third, we attempt to prove entailment relations between sentence pairs. For this purpose, we use Coq (Bertot and Castran, 2010), which can be used for efficient theorem-proving for natural language inference using both first-order and higher-order logic (Mineshima et al., 2015). Coq's proof calculus is based on natural deduction (Prawitz, 1965), a proof system based on inference rules called introduction and elimination rules for logical connectives. The inference system implemented in ccg2lambda using Coq achieves efficient automatic inference by feeding a set of predefined tactics and user-defined proof-search tactics to its interactive mode. The natural deduction system is particularly suitable for injecting external axioms during the theorem-proving process (Martínez-Gómez et al., 2017).

Finally, features for learning textual similarity are extracted from the proofs produced by ccg2lambda during the theorem-proving process. In this study, we experimented with logistic regression, support vector regression and random forest regression, finding that random forest regression was the most effective. We therefore chose random forest regression for learning textual similarity, with its hyperparameters being optimized by grid search. The mean squared error (MSE) was used to measure the prediction performance of our system.

4 Proof Strategy for Learning Textual Similarity

4.1 Overview of the proof strategy

Sentence similarity depends on complex elements, such as word overlaps and semantic relations. We capture the similarity between the sentence pair (A, B) as a function of the provability of bidirectional entailment relations for (A, B) and combine it with shallow features. After obtaining logical formulas A' and B' from A and B, we attempt to prove the bidirectional entailment relations, $A' \Rightarrow B'$ and $B' \Rightarrow A'$. If the initial natural deduction proofs fail, we re-run the proof, adding relevant external axioms or skipping unproved sub-goals until the proof is completed. After that, features for learning textual similarity are extracted by quantifying the provability of the bidirectional entailment relations.

The details of the procedure are as follows. First, we attempt a natural deduction proof without using external axioms, aiming to prove entailment relations, $A' \Rightarrow B'$ and $B' \Rightarrow A'$. If both fail, then we check whether A' contradicts B', which amounts to proving the negation of the original conclusion, namely $A' \Rightarrow \neg B'$ and $B' \Rightarrow \neg A'$.

The similarity of a sentence pair tends to be higher when the negation of the conclusion can be proved, compared with the case where neither the conclusion nor its negation can be proved. In the SICK (Sentences Involving Compositional Knowledge) dataset (Marelli et al., 2014) (see Section 6.1 for details), 70% of the sentence pairs annotated as contradictory are assigned a relatedness score in $[3, 5)$.

Next, if we fail to prove entailment or contradiction, that is, we cannot prove the conclusion or its negation, we identify an unproved sub-goal which is not matched by any predicate in the premise. We then attempt to prove $A' \Rightarrow B'$ and $B' \Rightarrow A'$ using axiom injection, following the method introduced in Martínez-Gómez et al. (2017). In axiom injection, unproved sub-goals are candidates to form axioms. We focus only on predicates that share at least one argument with both the premise and the conclusion. This means that an axiom can be generated only if there is a predicate p in the pool of premises and a predicate q in a sub-goal and p and q share a variable in an argument position, possibly with the same case (e.g., Subject or Object).

In generating axioms, the semantic relationships between the predicates in the premise and those in the conclusion are checked using lexical knowledge. In this study, we use WordNet (Miller, 1995) as the source of lexical knowledge. Linguistic relations between predicates are checked in the following order: inflections, derivationally related forms, synonyms, antonyms, hypernyms, similarities, and hyponyms. If any one of these relations is found in the lexical knowledge, an axiom can be generated. Again, if the proof fails, we attempt

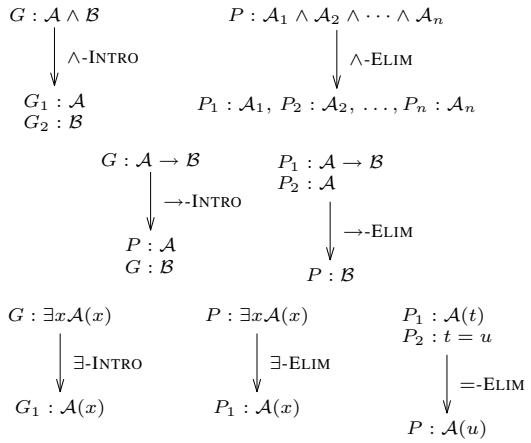

Figure 2: Example of the inference rules used in natural deduction. $P, P_1, \ldots P_n$ are formulas in the premise, while G, G_1, G_2 are formulas in the goal. The initial formulas are at the top, with the formulas obtained by applying the inference rules shown below.

to prove the negation of the conclusion using the axiom injection mechanism.

If the proof by axiom injection fails because of a lack of lexical knowledge, we obtain sentence similarity information from partial proofs by simply accepting the unproved sub-goals and forcibly completing the proof. After the proof is completed, information about the generated axioms and skipped sub-goals is used to create features.

4.2 Proving entailment relations

As an illustration of how our natural deduction proof works, consider the case of proving entailment for the following sentence pair:

A: A man is singing in a bar.

B: A man is singing.

The sentences A and B are mapped onto logical formulas A' and B' based on event semantics via CCG-based semantic composition, as follows.

$$A' : \exists e_1 x_1 x_2 (\mathbf{man}(x_1) \wedge \mathbf{sing}(e_1) \wedge (\mathbf{subj}(e_1) = x_1) \wedge \mathbf{bar}(x_2) \wedge \mathbf{in}(e_1, x_2))$$

$$B' : \exists e_1 x_1 (\mathbf{man}(x_1) \wedge \mathbf{sing}(e_1) \wedge (\mathbf{subj}(e_1) = x_1))$$

First, we attempt a natural deduction proof of $A' \Rightarrow B'$, setting A' as the premise and B' as the goal of the proof. Then A' and B' are decomposed according to the inference rules.

Figure 2 shows the major inference rules we use in the proofs. Inference rules in natural deduction are divided into two types: introduction rules and elimination rules. Introduction rules specify how

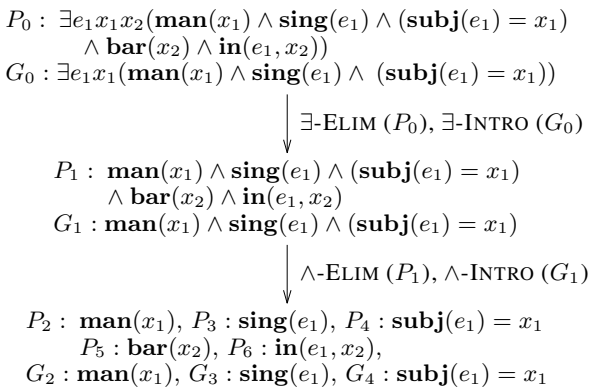

Figure 3: The proof process for the example entailment relation.

to prove a formula in the goal, decomposing a goal formula into smaller sub-goals. Elimination rules specify how to use a premise, decomposing a formula in the pool of premises into smaller ones.

The proof process for $A' \Rightarrow B'$ is shown in Figure 3. Here A' is initially set to the premise P_0 and B' to the goal G_0. P_0 and G_0 are then decomposed using elimination rules ($\wedge$-ELIM, $\exists$-ELIM) and introduction rules ($\wedge$-INTRO, $\exists$-INTRO). Then we obtain a set of premise formulas $\mathcal{P} = \{P_2, P_3, P_4, P_5, P_6\}$, and a set of sub-goals $\mathcal{G} = \{G_2, G_3, G_4\}$. The proof is performed by searching for a premise P_i whose predicate and arguments match those of a given sub-goal G_j. If such a logical premise is found, the sub-goal is removed. In this example, the sub-goals G_2, G_3, and G_4 match the premises P_2, P_3, and P_4, respectively. Thus, $A' \Rightarrow B'$ can be proved without introducing axioms.

Second, we attempt the proof in the opposite direction, $B' \Rightarrow A'$, by switching P_0 and G_0 in Figure 3. Again, by applying inference rules, we obtain the following sets of premises $\mathcal{P}$ and sub-goals $\mathcal{G}$:

$$\mathcal{P} = \{P_2 : \mathbf{man}(x_1),\ P_3 : \mathbf{sing}(e_1),\ P_4 : \mathbf{subj}(e_1) = x_1\}$$
$$\mathcal{G} = \{G_2 : \mathbf{man}(x_1),\ G_3 : \mathbf{sing}(e_1),\ G_4 : \mathbf{subj}(e_1) = x_1,\ G_5 : \mathbf{bar}(x_2), G_6 : \mathbf{in}(e_1, x_2))\}$$

Here, the two sub-goals G_5 and G_6 do not match any of the premises, so the attempted proof of $B' \Rightarrow A'$ fails. We therefore attempt to inject additional axioms, but in this case no predicate in $\mathcal{P}$ shares the argument x_2 of the predicates $\mathbf{bar}(x_2)$ and $\mathbf{in}(e_1, x_2)$ in $\mathcal{G}$. Thus, no axiom can be generated. To obtain information from a partial proof, we forcibly complete the proof of $B' \Rightarrow A'$ by skipping the unproved sub-goals $\mathbf{bar}(x)$ and $\mathbf{in}(e_1, x_2)$.

4.3 Proving the contradiction

The proof strategy illustrated here can be straightforwardly applied to proving the contradiction. In natural deduction, a negative formula of the form $\neg A$ can be defined as $A \rightarrow \mathbf{False}$ ("the formula A implies the contradiction"), by using a propositional constant $\mathbf{False}$ to encode the contradiction. Thus, the inference rules for negation can be taken as special cases of implication rules, as shown in Figure 4.

As an illustration, let us consider the following sentence pair:

A: No man is singing.

B: There is a man singing loudly.

Figure 5 shows the proof process. The sentences A and B are mapped to P_0 and P_1, respectively, via compositional semantics and the goal G_0 is set to $\mathbf{False}$. By decomposing P_1 using elimination rules and then by combining P_2, P_3, and P_4, we can obtain P_6. From P_0 and P_6 we can then derive the contradiction.

These proofs are performed by an automated prover implemented on Coq, using tactics for first-order theorem proving. When a proof is successful, Coq outputs the resulting proof (a proof term), from which we can extract detailed information such as the number of proof steps and the types of inference rules used. In addition to the entailment/contradiction result, information about the proof process is used to create features.

5 Description of the Features

To maximize accuracy when learning textual similarity, we adopt a hybrid approach that uses both logic-based features extracted from the natural deduction proof and other, non-logic-based features. All features are scaled to the $[0, 1]$ range.

5.1 Logic-based Features

We propose 15 features consisting of nine different types of logic-based features. Six of these feature types are derived from the bidirectional natural deduction proofs: six features are extracted from the direct proof ($A' \Rightarrow B'$) and another six from the reverse proof ($B' \Rightarrow A'$). The remaining three feature types are derived from semantic representations of the sentence pairs. The feature types are as follows.

Logical inference result. As stated in Section 4, we include features to distinguish the case where either the conclusion or its negation can be proved

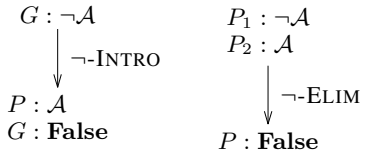

Figure 4: Inference rules of negation.

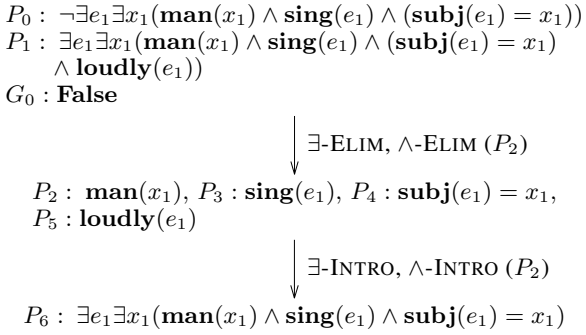

Figure 5: Proof process for the contradiction example.

from the one where neither can be proved. If the conclusion can be proved, the feature is set to 1.0. If the negation of the conclusion can be proved, the feature is set to 0.5. If neither can be proved, the feature is set to 0.0.

Axiom probabilities. The probability of an axiom and the number of axioms appearing in the proof are used to create features. The probability of an axiom is defined as the inverse of the length of the shortest path that connects the senses in the is-a (hypernym/hyponym) taxonomy in WordNet. When multiple axioms are used in the proof, the average of the probabilities of the axioms is extracted as a feature. If the proof can be completed without using axioms, the feature is set to 1.0.

Proved sub-goals. Given that proofs can be obtained either by proving all the sub-goals or skipping unproved sub-goals, we use the proportion of proved sub-goals as a feature. Our assumption is that if there are more unproved sub-goals then the sentence pair is less similar. When there are m logical formulas in the premise pool and n proved sub-goals, we set the feature to n/m. If the theorem can be proved without skipping any sub-goals, the feature is set to 1.0. It may be the case that the number of sub-goals is so large that some sub-goals remain unproved even after axiom injection. Since the proportion of unproved sub-goals is decreased by axiom injection, we use the proportion of unproved sub-goals both with and without axiom injection as features.

Cases in unproved sub-goals. Subject or object

words can affect the similarity of sentence pairs. Therefore, the number of each case in unproved sub-goals, like $\mathbf{subj}(e_1)$ in Figures 3 and 5, is used as a feature. Here, we count subjective, objective, and dative cases.

Proof steps. In general, complex theorems are difficult to prove and in such cases the sentence pairs are considered to be less similar. We therefore use the number of Coq's proof steps, namely the number of inference rule applications in a given proof, as a feature.

Inference rules. The complexity of a natural deduction proof can be measured in terms of the inference rules used for each proof step. We therefore extract the relative frequency with which each inference rule is used in the proof as a feature. We check seven inference rules for natural deduction using Coq (cf. Figure 2): introduction and elimination rules for conjunction ($\wedge$-INTRO, $\wedge$-ELIM), implication ($\rightarrow$-INTRO, $\rightarrow$-ELIM), and existential quantification ($\exists$-INTRO, $\exists$-ELIM), and the elimination rule for equality ($=$-ELIM).

Predicate overlap. Intuitively, the more predicates overlap between the premise and the conclusion, the more likely it is that the inference can be proved. We therefore use the proportion of predicates that overlap between the premise and the conclusion as a feature.

Semantic type overlap. Each semantic representation in higher-order logic has a semantic type, such as Entity for entities and Prop for propositions. As with predicates, we use the degree of semantic type overlap between the premise and the conclusion as a feature.

Existence of negative clauses. Whether or not the premise or conclusion contain negative clauses is an effective measure of similarity. In semantic representations, negative clauses are represented by the negation operator $\neg$, so we check for negation operators in the premise and the conclusion and set this feature to 1.0 if either contains one.

5.2 Non-logic-based Features

We also use the following eight non-logic-based features.

Noun/verb overlap. We extract and lemmatize all nouns and verbs from the sentence pairs and use the degrees of overlap of the noun and verb lemmas as features.

Part-of-speech overlap. We obtain part-of-speech (POS) tags for all words in the sentence pairs by first tokenizing them with the Penn Treebank Project tokenizer[1] and then POS tagging them with C&C POS tagger (Curran and Clark, 2003). The degree of overlap between the sentences' POS tags is used as a feature.

Synset overlap. For each sentence in the pair, we obtain the set containing all the synonym lemmas (the synset) for the words in the sentence. The degree of overlap between the sentences' synsets is used as a feature.

Synset distance. For each word in the first sentence, we compute the maximum path similarity between its synset and the synset of any other word in the second sentence. Then, we use the average of maximum path similarities as a feature.

Sentence length. If the conclusion sentence is long, there will possibly be many sub-goals in the proof. We therefore use the average of the sentence lengths and the difference in length between the premise and the conclusion sentences as features.

String similarity. We use the similarity of the sequence of characters within the sentence pairs as a feature. The Python *Difflib*[2] function returns the similarity between two sequences as a floating-point value in $[0, 1]$. This measure is given by $2.0 * M/T$, where T is the total number of elements in both sequences and M is the number of matches. This feature is 1.0 if the sequences are identical and 0.0 if they have nothing in common.

Sentence similarity from vector space models. We calculate sentence similarity by using three major vector space models, TF-IDF, latent semantic analysis (LSA) (Deerwester et al., 1990), and latent Dirichlet allocation (LDA) (Blei et al., 2003). We use these cosine similarities as features.

Existence of passive clauses. Passive clauses have an influence on similarity. In CCG trees, passive clauses are represented using the syntactic category $S_{pss}\backslash NP$. We check for the occurrence of passive clauses in the premise and conclusion, and if either of them contains a passive clause then the feature is set to 1.0.

[1] ftp://ftp.cis.upenn.edu/pub/treebank/public_html/tokenization.html

[2] https://docs.python.org/3.5/library/difflib.html

ID	Sentence1	Sentence2	Entailment	Score
23	There is no biker jumping in the air.	A lone biker is jumping in the air	*no*	4.2
1412	Men are sawing logs.	Men are cutting wood.	*yes*	4.5
9963	The animal is grazing on the grass.	The cop is sitting on a police bike.	*unknown*	1

Table 1: Examples in the SICK dataset with different entailment labels and similarity scores.

	γ	ρ	MSE
Mueller et al. (2016)	0.882	0.835	0.229
Our system	0.838	0.796	0.561
SemEval2014 Best Score	0.828	0.769	0.325
The Meaning Factory	0.827	0.772	0.322
UTexas	0.714	0.674	0.499
Baseline	0.653	0.745	0.808

Table 2: Results on the test split of SICK dataset.

6 Experiments and Evaluation

6.1 Experimental Conditions

We evaluated our system[3] using two datasets: the SemEval-2014 version of the SICK dataset (Marelli et al., 2014) and the SemEval-2012 version of the MSR-paraphrase video corpus dataset (MSR-vid) (Agirre et al., 2012). The experimental conditions were as follows.

6.1.1 The SICK dataset

The SICK dataset is a dataset for studying STS as well as for recognizing textual entailment (RTE). It was originally developed for evaluating compositional distributional semantics, so it contains logically challenging expressions such as quantifiers, negations, conjunctions and disjunctions. The dataset contains 9927 sentence pairs with a 5000/4927 training/test split. These sentence pairs are manually annotated with three types of labels *yes* (entailment), *no* (contradiction), or *unknown* (neutral) as well as a semantic relatedness scores in $[1, 5]$ (see Table 1 for a sample).

In this dataset, sentence pairs whose gold entailment labels are *no* tend to be scored a little more highly than the average, whereas those whose labels are *unknown* have a wide range of scores. Thus, we set the baseline of the relatedness score to 5 when the gold entailment label was *yes* and to 3 when the label was *no* or *unknown*.

We compared our system with the following systems: the state-of-the-art neural network-based system (Mueller and Thyagarajan, 2016); the best system (Zhao et al., 2014) from SemEval-2014; and two of the logic-based systems stated in Section 2: namely The Meaning Factory (Bjerva et al., 2014) and UTexas (Beltagy et al., 2014b). The Pearson correlation coefficient γ, Spearman's rank correlation coefficient ρ, and the MSE were used as the evaluation metrics.

6.1.2 The MSR-vid dataset

The MSR-vid dataset is our second dataset for the STS task and contains 1500 sentence pairs with a 750/750 training/test split. All sentence pairs are annotated with semantic relatedness scores in the range [0, 5]. We used this dataset to compare our system with the best system from SemEval-2012 (Bär et al., 2012) and the logic-based UTexas system (Beltagy et al., 2014a). We used the Pearson correlation coefficient γ as the evaluation metric.

6.2 Results

Table 2 shows the results of our experiments with the SICK dataset. Although the state-of-the-art neural network-based system yielded the best results overall, our system achieved higher scores than SemEval-2014 submissions, including the two logic-based systems (The Meaning Factory and UTexas), in terms of Pearson correlation and Spearman's correlation.

The main reason for our system's lower performance in terms of MSE is that some theorems could not be proved because of a lack of lexical knowledge. In the current work, we only consider word-level knowledge (word-for-word paraphrasing); we may expand the knowledge base in the future by using more external resources.

As we mentioned above, the sentence pairs annotated as *unknown* produced a wide range of scores. The Pearson correlation of the *unknown* portion of the SICK dataset was 0.766, which suggests that our logic-based system can also be applied to neutral sentence pairs.

Table 3 shows the results of our experiments with the MSR-vid dataset. These results also indicate that our logic-based system achieved higher accuracy than the other logic-based systems.

[3] Available at https://github.com/mynlp/ccg2lambda.

	γ
SemEval2012 Best Score	0.873
Our system	0.853
Beltagy et al. (2014)	0.830

Table 3: Results on the test split of MSR-vid.

	γ	ρ	MSE
Predicate overlap	**0.691**	0.609	**0.734**
Inference rules	0.632	**0.619**	0.794
Probability of axioms	0.543	0.540	0.865
Proof steps	0.458	0.494	0.915
Proved sub-goals	0.432	0.443	0.926
Logical inference result	0.386	0.399	0.939
Unproved sub-goals' case	0.301	0.307	0.973
Semantic type overlap	0.245	0.219	0.987
Negative clauses	0.163	0.323	1.004
Noun/verb overlap	0.661	0.554	0.763
Vector space model	0.594	0.510	0.857
String similarity	0.414	0.418	0.977
Synset overlap	0.382	0.341	0.978
Synset distance	0.352	0.330	0.999
Part-of-speech overlap	0.349	0.346	0.954
Sentence length	0.231	0.240	0.993
Passive clauses	0.023	0.046	1.017
Only logic-based	0.798	0.760	0.613
Only non logic-based	0.793	0.732	0.621
All	**0.838**	**0.796**	**0.561**

Table 4: Results when training our regressor with each feature group in isolation.

Table 4 shows evaluation results for each feature group in isolation, showing that inference rules and predicate overlaps are the most effective features. Compared with the non-logic-based features, the logic-based features achieved a slightly higher accuracy, a point that will be analyzed in more detail in the next section. Overall, our results show that combining logic-based features with non logic-based ones is an effective method for determining textual similarity.

6.3 Positive examples and error analysis

Table 5 shows some examples for which the prediction score was better when using logic-based features than when using non-logic-based ones.

For IDs 642 and 1360, one sentence contains a passive clause while the other sentence does not. In such cases, the sentence pairs are not superficially similar. By using logical formulas based on event semantics we were able to interpret the sentence containing the passive clause correctly and judge that the passive and non-passive sentences are similar to each other.

In ID 891, one sentence contains a negative clause while the other does not. Using shallow features, the word overlap is small and the prediction score was much lower than the correct score. Our logic-based method, however, interpreted the first sentence as a negative existential formula of the form $\neg\exists x\mathcal{P}(x)$ and the second sentence as an existential formula $\exists x\mathcal{P}'(x)$. Thus, it could easily handle the semantic difference between the positive and negative sentences.

In ID 1158, by contrast, the proportion of word overlap is so high that the prediction score with non-logic-based features was much higher than the correct score. Our method, however, was able to prove the contradiction using an antonym axiom of the form $\forall x(\mathbf{remove}(x) \rightarrow \neg\mathbf{add}(x))$ from WordNet and thus predict the score correctly.

In ID 59, the proportion of word overlap is low, so the prediction score with non-logic-based features was lower than the correct score. Our method, however, was able to prove the partial entailment relations for the sentence pair and thus predict the score correctly. Here the logic-based method captured the common meaning of the sentence pair: both sentences talk about the kids playing in the leaves.

Finally, in ID 71, the prediction score with non-logic-based features was much higher than the correct score. There are two reasons for this phenomenon: negations tend to be omitted in non-logic-based features such as TF-IDF and the proportion of word overlap is high. However, as logical formulas and proofs can handle negative clauses correctly, our method was able to predict the score correctly.

Table 6 shows examples where using only logic-based features produced erroneous results. In ID 3974, the probability of axiom $\forall x(\mathbf{awaken}(x) \rightarrow \mathbf{up}(x))$ was low (0.25) and thus the prediction score was lower than the correct score. Likewise, in ID 4833, the probability of axiom $\forall x(\mathbf{file}(x) \rightarrow \mathbf{do}(x))$ was very low (0.09) and thus the prediction score was negatively affected. In these cases, we need to consider phrase-level axioms such as $\forall x(\mathbf{awaken}(x) \rightarrow \mathbf{wake\_up}(x))$ and $\forall x(\mathbf{file\_nail}(x) \rightarrow \mathbf{do\_manicure}(x))$ using a paraphrase database. This, however, is an issue for future study. In ID 1941, the system wrongly proved the bidirectional entailment relations by

ID	Sentence Pair	Gold	Pred +logic	Pred -logic	Entailment
642	A person is climbing a rock with a rope, which is pink. A rock is being climbed by a person with a rope, which is pink.	5.0	4.9	4.1	Yes
1360	The machine is shaving the end of a pencil. A pencil is being shaved by the machine.	4.7	4.6	3.8	Yes
891	There is no one on the shore. A bunch of people is on the shore.	3.6	3.7	2.6	No
1158	A woman is removing ingredients from a bowl. A woman is adding ingredients to a bowl.	3.3	3.5	4.1	No
59	Kids in red shirts are playing in the leaves. Three kids are jumping in the leaves.	3.9	3.8	3.1	Unknown
71	There is no child lying in the snow and making snow angels. Two people in snowsuits are lying in the snow and making snow angels.	3.3	3.3	4.1	Unknown

Table 5: Examples for which our regressor trained only with logic-based features performs better than when using non-logic features. "Gold": correct score, "Pred+logic": prediction score only with logic-based features, "Pred-logic": prediction score only with non-logic-based features.

ID	Sentence Pair	Gold	System	Axiom
3974	A girl is awakening. A girl is waking up.	4.9	3.6	$\forall x(\mathbf{awaken}(x) \rightarrow \mathbf{wake}(x))$ $\forall x(\mathbf{awaken}(x) \rightarrow \mathbf{up}(x))$
4833	A girl is filing her nails. A girl is doing a manicure.	4.2	1.8	$\forall x(\mathbf{nail}(x) \rightarrow \mathbf{manicure}(x))$ $\forall x(\mathbf{file}(x) \rightarrow \mathbf{do}(x))$
1941	A woman is putting the baby into a trash can. A person is putting meat into a skillet.	1.0	3.3	$\forall x(\mathbf{woman}(x) \rightarrow \mathbf{person}(x))$ $\forall x(\mathbf{trash}(x) \rightarrow \mathbf{skillet}(x))$ $\forall x(\mathbf{baby}(x) \rightarrow \mathbf{meat}(x))$

Table 6: Error examples when training the regressor only with logic-based features.

adding external axioms, so the prediction score was much higher than the correct score. Setting the threshold for the probability of an axiom may be an effective way of improving our axiom-injection method.

7 Conclusion

We have developed a hybrid method for learning textual similarity by combining features based on logical proofs of bidirectional entailment relations with non-logic-based features. The results of our experiments on two datasets show that our system was able to outperform other logic-based systems. In addition, the results show that information about the natural deduction proof process can be used to create effective features for learning textual similarity. Since these logic-based features provide accuracy improvements that are largely additive with those provided by non-logic-based features, neural network-based systems may also benefit from using them.

In future work, we will refine our system so that it can be applied to other tasks such as question answering. Compared with neural network-based systems, our natural deduction-based system can not only assess how similar sentence pairs are, but also explain what the sources of similarity/dissimilarity are by referring to information about sub-goals in the proof. Given this interpretative ability, we believe that our logic-based system may also be of benefit to other natural language processing tasks, such as question answering and text summarization.

Acknowledgments

We thank the three anonymous reviewers for their detailed comments. This work was supported by JST CREST Grant Number JPMJCR1301, Japan.

References

Lasha Abzianidze. 2015. A tableau prover for natural logic and language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP-15)*, pages 2492–2502, Lisbon, Portugal. Association for Computational Linguistics.

Lasha Abzianidze. 2016. Natural solution to FraCaS entailment problems. In *Proceedings of the 5th Joint Conference on Lexical and Computational Semantics*, pages 64–74, Berlin, Germany. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 Task 6: A

pilot on semantic textual similarity. In *Proceedings of the 6th International Workshop on Semantic Evaluation (SemEval-2012)*, pages 385–393, Montréal, Canada. Association for Computational Linguistics.

Daniel Bär, Chris Biemann, Iryna Gurevych, and Torsten Zesch. 2012. UKP: Computing semantic textual similarity by combining multiple content similarity measures. In *Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval-2012)*, pages 435–440, Montréal, Canada. Association for Computational Linguistics.

Daisuke Bekki and Koji Mineshima. 2017. Context-passing and underspecification in dependent type semantics. In Stergios Chatzikyriakidis and Zhaohui Luo, editors, *Modern Perspectives in Type Theoretical Semantics*, Studies of Linguistics and Philosophy, pages 11–41. Springer.

Islam Beltagy, Katrin Erk, and Raymond Mooney. 2014a. Probabilistic soft logic for semantic textual similarity. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL-2014)*, pages 1210–1219, Baltimore, Maryland. Association for Computational Linguistics.

Islam Beltagy, Stephen Roller, Gemma Boleda, Katrin Erk, and Raymond Mooney. 2014b. UTexas: Natural language semantics using distributional semantics and probabilistic logic. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval-2014)*, pages 796–801, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.

Yves Bertot and Pierre Castran. 2010. *Interactive Theorem Proving and Program Development: Coq'Art The Calculus of Inductive Constructions*. Springer Publishing Company, Incorporated, New York, USA.

Johannes Bjerva, Johan Bos, Rob van der Goot, and Malvina Nissim. 2014. The Meaning Factory: Formal semantics for recognizing textual entailment and determining semantic similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval-2014)*, pages 642–646, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning*, 3:993–1022.

Stephen Clark and James R. Curran. 2007. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4):493–552.

James R Curran and Stephen Clark. 2003. Investigating GIS and smoothing for maximum entropy taggers. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 1*, pages 91–98. Association for Computational Linguistics.

Scott Deerwester, Susan T. Dumais, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Edward Grefenstette and Mehrnoosh Sadrzadeh. 2011. Experimental support for a categorical compositional distributional model of meaning. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP-2011)*, pages 1394–1404, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1367–1377, San Diego, California. Association for Computational Linguistics.

Dimitri Kartsaklis, Nal Kalchbrenner, and Mehrnoosh Sadrzadeh. 2014. Resolving lexical ambiguity in tensor regression models of meaning. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL-2014)*, pages 212–217, Baltimore, Maryland. Association for Computational Linguistics.

Dimitri Kartsaklis and Mehrnoosh Sadrzadeh. 2016. Distributional inclusion hypothesis for tensor-based composition. In *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers (COLING-2016)*, pages 2849–2860, Osaka, Japan. The COLING 2016 Organizing Committee.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31th International Conference on Machine Learning, (ICML-2014)*, pages 1188–1196, Beijing, China.

Mike Lewis and Mark Steedman. 2014. A* CCG parsing with a supertag-factored model. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP-2014)*, pages 990–1000, Doha, Qatar. Association for Computational Linguistics.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC-2014)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association.

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. 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 (EACL-2017)*, pages 710–720, Valencia, Spain. Association for Computational Linguistics.

Dale A. Miller and Gopalan Nadathur. 1986. Some uses of higher-order logic in computational linguistics. In *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*, pages 247–256, New York, New York, USA. Association for Computational Linguistics.

George A. Miller. 1995. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41.

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 (EMNLP-2015)*, pages 2055–2061, Lisbon, Portugal. Association for Computational Linguistics.

Koji Mineshima, Ribeka Tanaka, Pascual Martínez-Gómez, Yusuke Miyao, and Daisuke Bekki. 2016. Building compositional semantics and higher-order inference system for a wide-coverage Japanese CCG parser. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2236–2242, Austin, Texas. Association for Computational Linguistics.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL-08)*, pages 236–244, Columbus, Ohio. Association for Computational Linguistics.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429.

Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI-2016)*, pages 2786–2792, Arizona, USA. Association for the Advancement of Artificial Intelligence.

Terence Parsons. 1990. *Events in The Semantics of English: a Study in Subatomic Semantics*. MIT Press, Cambridge, USA.

Tamara Polajnar, Laura Rimell, and Stephen Clark. 2015. An exploration of discourse-based sentence spaces for compositional distributional semantics. In *Proceedings of the 1st Workshop on Linking Computational Models of Lexical, Sentential and Discourse-level Semantics*, pages 1–11, Lisbon, Portugal. Association for Computational Linguistics.

Dag Prawitz. 1965. *Natural Deduction – A Proof-Theoretical Study*. Almqvist & Wiksell, Stockholm, Sweden.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press, Cambridge, USA.

S. K. M. Wong and Vijay V. Raghavan. 1984. Vector space model of information retrieval: A reevaluation. In *Proceedings of the 7th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 167–185.

Jiang Zhao, Tiantian Zhu, and Man Lan. 2014. ECNU: One stone two birds: Ensemble of heterogenous measures for semantic relatedness and textual entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval-2014)*, pages 271–277, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.