Abstract

This paper proposes a knowledge representation model and a logic proving setting with axioms on demand which proved to be very successful for the recognizing textual entailment task. LCC’s submission to the Second RTE Challenge exploits the logical entailment between deep semantics and syntax of $T$ and $H$ as well as a shallow lexical alignment of the two texts.

1 System Description

LCC’s submission to the Second Recognizing Textual Entailment Challenge was the result of a three system combination: our logic prover COGEX when it receives as input two different types of logic forms ($COGEX$ and $COGEX_d$) and a more lexical oriented system which tries to align the text $T$ and the hypothesis $H$ (LEXALIGN). For every $(T, H)$ pair, each system returns a score between 0 and 1 (higher values showing more probable entailment). A linear combination of these scores assigns the final label to each $(T, H)$ pair.

1.1 COGEX

COGEX (Moldovan et al., 2003) is a natural language prover originating from OTTER (McCune, 1994). The prover requires a list of clauses called the set of support which is used to initiate the search for inferences. COGEX loads the set of support with the negated form of the hypothesis ($\neg H$) as well as the predicates that make up the logic form of the text passage ($T$). A second list, called the usable list, contains the axioms used by COGEX to generate inferences. For the textual entailment task, we considered several types of axioms: eXtended WordNet, linguistic, semantic calculus and temporal axioms. Once the set of support and usable lists are created, the logic prover can begin searching for proofs. The clauses in the set of support list are weighted in the order in which they should be chosen to participate in the search. The negated hypothesis is assigned the largest weight to ensure that it is the last clause to participate in the search. The logic prover removes the clause with the smallest weight from the set of support, and searches the usable list for new inferences that can be made. Any produced inferences are assigned an appropriate weight depending on what axiom they were derived from and appended to the set of support list. The logic prover continues in this fashion until the set of support list is empty. If a refutation is found, then the proof is complete. If a refutation cannot be found, then predicate arguments are relaxed. If argument relaxation fails to produce a refutation, predicates are dropped from the negated hypothesis until a refutation is found. Once a proof by refutation is found, a score for that proof is calculated by starting with an initial perfect score and deducting points for axioms that are utilized in the proof, arguments that are relaxed, and predicates that are dropped.

1.1.1 Proof scoring

The computed score is a measure of the kinds of axioms used in the proof and the significance of the dropped arguments and predicates. If we assume
T: The Council of Europe has 45 member states. Three countries from ...

H: The Council of Europe is made up by 45 member states.

Table 1: The lexical alignment for pair T615

<table>
<thead>
<tr>
<th>T: The Council of Europe has 45 member states. Three countries from ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEL</td>
</tr>
</tbody>
</table>

that both text fragments are existential, then \( T \vdash H \) if and only if \( T \)'s entities are a subset of \( H \)'s entities (\( H \) is more general than \( T \)) and penalizing a pair whose \( H \) contains predicates that cannot be inferred is a correct way to ensure the entailment. But if both \( T \) and \( H \) are universally quantified, then the groups mentioned in \( H \) must be a subset of the ones from \( T \). Thus, the scoring module adds the points for the modifiers dropped from \( H \) and subtracts points for \( T \)'s modifiers not present in \( H \). When \( T \) is universally quantified and \( H \) is existential, COGEX adds the dropped points for \( H \)'s modifiers not present in \( T \). In the remaining case, the points of the modifiers not present in \( H \) are deducted from the proof's score.

Because \((T, H)\) pairs with longer sentences can potentially drop more predicates and receive a lower score, the system normalizes the scores by first calculating the maximum penalty that can be assessed to a pair by dropping all of predicates from \( H \). The penalty assessed by the logic prover is then divided by the maximum drop penalty to determine the pair's normalized score \((sCOGEX, sCOGEX_d)\).

1.2 Lexical Alignment

Inspired by the positive development examples whose hypothesis is in a high degree lexically subsumed by the text, we developed a system which measures the level of subsumption by computing an edit distance between the text \( T \) and the hypothesis \( H \). First, we removed the punctuation and the stop words from both text fragments and then, using dynamic programming, we computed the edit distance between \( T \) and \( H \). For this task, we considered the cost of deleting a word from \( T \) \((w_T \rightarrow *)\) equal to 0, the cost of replacing a word from \( T \) with another from \( H \) \((w_T \rightarrow w_H)\) equal to \( \infty \) (we didn’t allow replace operations) and the cost of inserting a word from \( H \) \((* \rightarrow w_H)\) varied depending on the part-of-speech of the inserted word (if \( w_H \) is a noun, adjective or an adverb in WordNet, the insertion cost is 30, if it is a verb, 13 and in all other cases, 10).

When we compared the words in \( H \) with the ones in \( T \), we considered the edit distance between two synonyms equal to 0. We divide the edit distance by the length of the hypothesis (measured in words), which results in a number between 0 and 30 (low values indicate high lexical matching, higher values, low lexical matching). The lexical alignment score, \( s_{LexAlign} \), is the normalized average cost, a real valued number between 0 and 1. Table 1 shows the minimum cost alignment for pair T615.

The following sections of the paper shall detail our logical representation of text and the various types of axioms that COGEX uses. Section 6 describes the system combination process and the overall performance.

2 Knowledge Representation

For the textual entailment task, our logic prover uses a logical representation which consists of two layers that express the syntactic and semantic propositions encoded in \( T \) and \( H \).

2.1 Logic Form Transformation

In the first stage of our system, we convert the text and the hypothesis into logic forms (Moldovan and Rus, 2001). This first layer of the logic representation is derived from a full syntactic parse tree and acknowledges syntax-based relationships such as: syntactic subjects, syntactic objects, prepositional attachments, complex nominals, and adjectival/adverbial adjuncts. These syntactic relations provide signals to the detection of semantic relations in the second layer of our representation. For example, pair D439’s hypothesis *Gilda Flores was kidnapped on the 13th of January 1990.* has the following representation in logic form: *Gilda NN(x1) & Flores NN(x2) & nn NNC(x3,x1,x2) & human NE(x3) & kidnap VB(e1,x9,x3) & on IN(e1,x8)*

\(^1\)We precede each pair id with its source set (“D” for development or “T” for the test set).
Throughout this report, we’ll refer to the system which uses these “constituency” logic forms as COGEX.<sup>c</sup>

Using the preprocessed data provided by the challenge organizers, we created a second set of logic forms from the dependency parse trees of and.

This representation captures more accurately the syntactic dependencies between the concepts, but lacks the semantic information that our semantic parser extracts using the constituency parse trees. However, we imported the named entities identified by our NE module to the logic form created from the dependency parse of the data whenever we matched the tokens. Because we did not have any control over the tokenization of the preprocessed data, some entities were not labeled with their corresponding class. For example, the logic form of the above sample sentence (Gilda Flores NN(x2) human_NE(x2) & kidnap_VB(e1,x4,x2) & on_IN(e1,x3) & 13th_NN(x3) & of_IN(x3,x1) & January_1990_NN(x1)), contains only one of the two named entities identified. COGEX’s version which uses the logic forms derived from dependency trees is called COGEX<sub>d</sub>.

2.1.1 Negation

During the logic form transformation step, a predicate is created for each noun, verb, adjective and adverb. There are several exceptions to this rule including the not and never adverbs. In cases similar to pair D154’s text further details were not released, the system removes not_RB(x3,e1) and negates the verb’s predicate (¬release_VB(e1,x1,x2)). Similarly, for nouns whose determiner is no, for example, No case of indigenously acquired rabies infection has been confirmed (pair D601), the verb’s predicate is negated (case_NN(x1) & ¬confirm_VB(e2,x15,x1)).

2.2 Semantic Relations

The second layer of our logic representation adds the semantic relations, the underlying relationships between concepts. They provide the semantic background for the text, which allows for a denser connectivity between the concepts expressed in text. Our semantic parser takes free English text or parsed sentences and extracts a rich set of semantic relations between words or concepts in each sentence. It focuses not only on the verb and its arguments, but also on semantic relations encoded in syntactic patterns such as complex nominals, genitives, adjectival phrases, and adjectival clauses. Our system maps each semantic relation identified by the parser to a predicate whose arguments are the events and entities that participate in the relation and it adds these semantic predicates to the logic form of the text. If we consider the previous example (D439), its logic form is augmented with the THEME<sub>SR</sub>(x3,e1) & TEMPORAL<sub>SR</sub>(x8,e1) relations<sup>2</sup> (a theme relation between Gilda Flores and kidnap and a temporal relation between 13th of January 1990 and kidnap).

2.3 Temporal Representation

The temporal reasoning predicates are derived from both the detected semantic relations as well as from a module which utilizes a learning algorithm to detect temporally ordered events ((S,E₁,E₂), where S is the temporal signal linking the two events E₁ and E₂). From each triple, temporally related SUMO predicates are generated based on hand-coded rules for the signal classes. Also, every date is transformed into the normalized form time_TMP(BeginFn(event), year, month, date, hour, minute, second) & time_TMP(EndFn(event), year, month, date, hour, minute, second). In the above example (D439), time_TMP(BeginFn(e2), 1990, 1, 13, 0, 0, 0) & time_TMP(EndFn(e2), 1990, 1, 13, 23, 59, 59) & during_TMP(e1,e2) is added to the logic representation.

3 Axioms on Demand

In our system, the usable list consists of all the axioms that have been generated either automatically or by hand. COGEX generates axioms on demand for a given (T,H) pair whenever the semantic connectivity between two concepts needs to be estab-

<sup>2</sup>R(x,y) indicates that relation R holds between x and y.
lished in a proof. Axioms in our system are utilized to provide external world knowledge, knowledge of syntactic equivalence between logic form predicates, and lexical knowledge in the form of lexical chains.

3.1 eXtended WordNet lexical chains

For the semantic entailment task, the ability to recognize two semantically-related (distinct) words is an important requirement. Therefore, we, automatically, construct lexical chains of WordNet relations annotated between the synsets, from the words in T to H’s constituents (Moldovan and Novisch, 2002). For each relation in the best chain\(^3\), the system generates, on demand, an axiom with the predicates of the synsets in the WordNet relation. For example, given the ISA relation between the synsets of murder and kill, the system generates, when needed, the axiom \(\text{murder}_V B(e_1, x_1, x_2) \rightarrow \text{kill}_V B(e_1, x_1, x_2)\)\(^4\).

Because we added named entities to each noun concept from WordNet, as part of the eXtended WordNet project\(^5\), this year’s lexical chain axioms append the entity name of the target concept, whenever it exists. For example, for pair D548, COGEX uses the axiom \(\text{Nicaraguan}_J J(x_1, x_2) \rightarrow \text{Nicaragua}_N N(x_1) \land \text{country}_N E(x_1)\) when it tries to infer \text{electoral campaign} = \text{held in Nicaragua} from \text{Nicaraguan electoral campaign}.

We ensured the relevance of the lexical chains by limiting the path length to three relations and the set of WordNet relations used to create the chains by discarding the paths that contain certain relations in a particular order. For example, the automatic axiom generation module did not consider chains with an IS-A relation followed by a HYPONYMY link. Without removing these types of chains, our system inferred, for instance, \text{John lives in Detroit} from \text{John lives in Chicago} (\text{Chicago} \rightarrow \text{city} \rightarrow \text{Detroit}). Similarly, the system rejected chains with more than one HYPONYMY relations. Although this relation links semantically related concepts, the hypothesis should be more general than the text and too many HYPONYMY relations can lead to a too specific concept in \(H\). Another restriction imposed on the lexical chains that the system generates for the prover is not to include too general concepts. We assigned each noun and verb synset from WordNet a generality weight computed based on the position of the concept within its hierarchy. If \(n_i\) is the number of hyponyms of concept \(c_i\) and \(N\) represents the total number of concepts in \(c_i\)’s hierarchy, then

\[
generalityW(c_i) = \frac{\log(1+n_i)}{\log(1+N)}.
\]

In our experiments, COGEX discarded the chains with concepts whose generality weight exceeded 0.8 such as \text{object}_N N, \text{act}_V B, \text{be}_V B, \text{etc}.

Another important change that we introduced in our extension of WordNet is the refinement of the DERIVATION relation which links verbs with their corresponding nominalized nouns. Because the relation is ambiguous regarding the role of the noun, we split this relation in three: ACT-DERIVATION, AGENT-DERIVATION and THEME-DERIVATION. The role of the nominalization determines the argument given to the noun predicate. For instance, the axioms \(\text{employ}_V B(e_1, x_1, x_2) \rightarrow \text{employment}_N N(e_1)(\text{ACT}), \text{employ}_V B(e_1, x_1, x_2) \rightarrow \text{employer}_N N(x_1)(\text{AGENT}), \text{and employ}_V B(e_1, x_1, x_2) \rightarrow \text{employee}_N N(x_2)(\text{THEME})\) are created from the different types of derivation.

This type of axioms helps our logic prover to infer \(H\)’s concepts from \(T\)’s concepts when lexical chains are found between the two.

3.2 NLP Axioms

Our NLP axioms are linguistic rewriting rules that help break down complex logic structures and express syntactic equivalence. After analyzing the logic form and the parse trees of each text fragment, the system, automatically, generates NLP axioms to break down complex nominals and coordinating conjunctions into their constituents so that other axioms can be applied, individually, to the components. These axioms are made available only to the \((T, H)\) pair that generated them. For example, for pair D557, the noun compound \text{Francisco Merino} is broken down into \text{Francisco} and \text{Merino}. The axiom \(\text{m} \_ \text{NNC}(x_3, x_1, x_2) \land \text{francisco}_N N(x_1) \land\)

\(^3\)Shorter chains are better than longer ones. The relations are not equally important and their order in the chain influences its strength.

\(^4\)Axiom used to prove the entailment for pair D197.

\(^5\)http://xwn.hlt.utdallas.edu
merino_NN(x_2) \rightarrow merino_NN(x_3) \text{ helps the system infer Merino’s home from Francisco Merino’s home.}

3.3 World Knowledge Axioms

Because, sometimes, the lexical or the syntactic knowledge cannot solve an entailment pair, we exploit the WordNet glosses, an abundant source of world knowledge. We used the logic forms of the glosses provided by the eXtended WordNet project to, automatically, create our world knowledge axioms. For example, the first sense of noun Pope and its definition the head of the Roman Catholic Church introduces the axiom Pope_NN(x_1) \leftrightarrow head_NN(x_1) \& of_IN(x_1,x_2) \& Roman_Catholic_Church_NN(x_2) \& organization_NE(x_2) which is used by COGEX to prove the entailment of pair D398 (T: A place of sorrow, after Pope John Paul II died, became a place of celebration, as Roman Catholic faithful gathered in downtown Chicago to mark the installation of new Pope Benedict XVI; H: Pope Benedict XVI is the new leader of the Roman Catholic Church). We also incorporate in our system a small common-sense knowledge base of 383 hand-coded world knowledge axioms, where 73 have been manually designed based on the development set data, and 310 originate from previous projects.

4 Semantic Calculus

The Semantic Calculus axioms combine two semantic relations identified within a text fragment and increase the semantic connectivity of the text. This type of axioms enable inference of unstated meaning from the semantics detected in text. For example, for pair D641 whose T states explicitly the KINSHIP relations between Nicholas Cage and Alice Kim Cage and between Alice Kim Cage and Kal-el Coppola Cage, the logic prover uses the KINSHIP_SR(x_1,x_2) \& KINSHIP_SR(x_2,x_3) \rightarrow KINSHIP_SR(x_1,x_3) semantic axiom which expresses the transitivity of the blood relation and the symmetry of this relationship to infer H’s statement (KINSHIP(Kal-el Coppola Cage, Nicholas Cage)). Currently, COGEX uses 82 semantic combinations axioms manually identified and validated against large corpora. These axioms use relations such as PartWhole, Isa, Location, Attribute, or Agent. For example, the axiom ISA_SR(x_1,x_2) \& ATTRIBUTE_SR(x_2,x_3) \rightarrow ATTRIBUTE_SR(x_1,x_3) infers from the sentence Mike is a rich man that Mike is rich. Another frequently applied axiom is LOCATION_SR(x_1,x_2) \& PARTWHOLE_SR(x_2,x_3) \rightarrow LOCATION_SR(x_1,x_3). Given the text John lives in Dallas, Texas and using the axiom, the system inferences that John lives in Texas. The system applies the 82 axioms independent of the concepts involved in the semantic composition. There are rules that can be applied only if the concepts that participate satisfy a certain condition or if the relations are of a certain type. For example, LOCATION_SR(x_1,x_2) \& LOCATION_SR(x_2,x_3) \rightarrow LOCATION_SR(x_1,x_3) only if the LOCATION relation shows inclusion (John is in the car in the garage \rightarrow LOCATION_SR(John,garage). John is near the car behind the garage \not\rightarrow LOCATION_SR(John,garage)).

5 Temporal Axioms

COGEX uses a SUMO knowledge base of temporal reasoning axioms that consists of axioms for a representation of time points and time intervals, Allen (Allen, 1991) primitives, and temporal functions. For example, during is a transitive Allen primitive: during_TMP(e_1,e_2) \& during_TMP(e_2,e_3) \rightarrow during_TMP(e_1,e_3).

6 Merging three systems and Results

Each system returns a score between 0 and 1, a number close to 0 indicating a probable negative example and a number close to 1 indicating a probable positive example. For each NLP task (IE, IR, QA, SUM), we built a classifier based on the linear combination of the three scores. Each task’s classifier labels pair i as positive if \lambda_{COGEX_i}sc_i + \lambda_{COGEX_i}sc_i + \lambda_{LexAlign_i}sc_i > 0.5, where the optimum values of the classifier’s real-valued parameters (\lambda_{C_i}, \lambda_{D_i}, \lambda_{LexA}) were de-
termined using a grid search on the development set. Given the different nature of each application, the λ parameters vary with each task. For example, the final score given to each Information Extraction (IE) pair is highly dependent on the score given by the logic prover when it received as input the logic forms created from the constituency parse trees with a small correction from COGEX with the dependency parse trees logic forms. For the IE task, the lexical alignment performs the worst among the three systems. On the other hand, for the Information Retrieval (IR) task, the score given by LEXALIGN is taken into account. Table 2 lists the λ values.

Table 2: The λ values of each task’s classifier

<table>
<thead>
<tr>
<th>Task</th>
<th>λ_{COGEX}</th>
<th>λ_{COGEX_d}</th>
<th>λ_{LEXALIGN}</th>
</tr>
</thead>
<tbody>
<tr>
<td>IE</td>
<td>1.100</td>
<td>0.350</td>
<td>-0.625</td>
</tr>
<tr>
<td>IR</td>
<td>0.325</td>
<td>0.125</td>
<td>0.675</td>
</tr>
<tr>
<td>QA</td>
<td>-0.225</td>
<td>0.575</td>
<td>0.350</td>
</tr>
<tr>
<td>SUM</td>
<td>0.200</td>
<td>0.325</td>
<td>0.550</td>
</tr>
</tbody>
</table>

Table 3 summarizes the performance of the three system combination. We perform significantly better on the (T, H) pairs in the multi-document summarization (SUM) task. For this task, all three systems performed the best because the T text of false pairs was not entailing the hypothesis even at a lexical level. For pair T682, T and H have very few words overlapping and there are no axioms that can be used to derive knowledge that supports the hypothesis. Contrarily, for the IE task, the systems were fooled by the high word overlap between T and H. For example, pair T678’s text contains the entire hypothesis in its if clause. For this task, we had the highest number of false positives, around double when compared to the other applications. Some of the pairs that the system, currently, cannot handle involve numeric calculus and human-oriented estimations. Consider, for example, pair D359 labeled as positive, for which the logic prover could not determine that 15 safety violations ⊨ numerous safety violations.

7 Conclusion

The Second RTE Challenge gave us the opportunity to test our latest research and engineering advancements for entailment. In this year’s challenge, our logic form representation correctly captures both negation and quantification and it includes special temporal predicates. We have made several changes to our eXtended WordNet lexical chains module which led to fewer unsound axioms. We plan to improve our system to detect false entailments even when the two texts have a high word overlap and expand our axiom set.

References


