Automatic Answer Validation using COGEX

Marta Tatu, Brandon Iles and Dan Moldovan
Language Computer Corporation
Richardson, Texas, 75080
United States of America
marta,brandon.iles,moldovan@languagecomputer.com

Abstract

This paper reports on Language Computer Corporation's natural language logic prover's performance for the English and Spanish subtasks of the Answer Validation Exercise. COGEX takes as input a pair of plain English text snippets, it transforms them into highly semantic logic forms, automatically generates natural language axioms which will be used during the search for a proof and, determines the degree of entailment between the entailing text snippet and a possible relaxed version of the entailed text. The system labels an input pair as true entailment if its proof score is above the threshold learned during training. Our semantic logic-based approach achieves 43.93% F-measure for the English data and 60.63% on Spanish.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.1 Content Analysis and Indexing; H.3.3 Information Search and Retrieval; H.3.4 Systems and Software; H.3.7 Digital Libraries; H.2.3 [**Database Management**]: Languages—*Query Languages*; I.2 [**Artificial Intelligence**]: I.2.7 Natural Language Processing; I.2.4 Knowledge Representation Formalisms and Methods; I.2.3 Deduction and Theorem Proving

General Terms

Measurement, Performance, Experimentation

Keywords

Answer Validation, Question Answering, Recognizing Textual Entailment (RTE)

1 Introduction

While communicating, humans use different expressions to convey the same meaning. One of the central challenges for natural language understanding systems is to determine whether different text fragments have the same meaning or, more generally, if the meaning of one text can be derived from the meaning of another. A module that recognizes the *textual entailment* between two snippets can be employed by various NLP applications. For example, Question Answering (QA) systems should identify texts that entail the expected answers. Alternatively, a subsystem designed to identify semantic entailments can validate the correctness of the answers returned by a QA system. This latter idea was promoted as the Answer Validation Exercise (AVE)[1] subtask

[1] http://nlp.uned.es/QA/ave/index.php

of the 2006 QA[2] track at the Cross-Language Evaluation Forum (CLEF)[3]. Once an answer with its supporting snippet was returned by a QA system, a hypothesis was build by reformulating and transforming the question and its answer into a declarative statement. If the supportive text semantically entails this hypothesis, then the answer given by the QA system is expected to be correct.

In this paper, we describe a model to represent the knowledge encoded in text and a logical setting suitable to a recognizing semantic entailment system. We cast the textual inference problem as a *logic implication between meanings*. The entailing supportive text (T) semantically infers the entailed hypothesis (H) if its meaning logically implies the meaning of H. Thus, our system, first, transforms both text fragments into logic forms, captures their meaning by detecting the *semantic* relations that hold between the constituents of T and, subsequently, between the constituents of H and loads these rich logic representations into a natural language logic prover to decide if the entailment holds or not and to provide a justification for it. This approach [15] proved to be highly effective at the Second PASCAL Recognizing Textual Entailment Challenge[4] [2]. Figure 1 illustrates our answer validation system's architecture. We used this setup for the English dataset and for the automatic translation (in English) of the Spanish data[5]. LCC's machine translation system implements a phrase-based statistical translation method [8]. It was trained using the Europarl corpus [7]. The details of our translation solution is described in [14]. The following sections of the paper shall detail the logic proving methodology, our logical representation of text and the various types of axioms the prover uses.

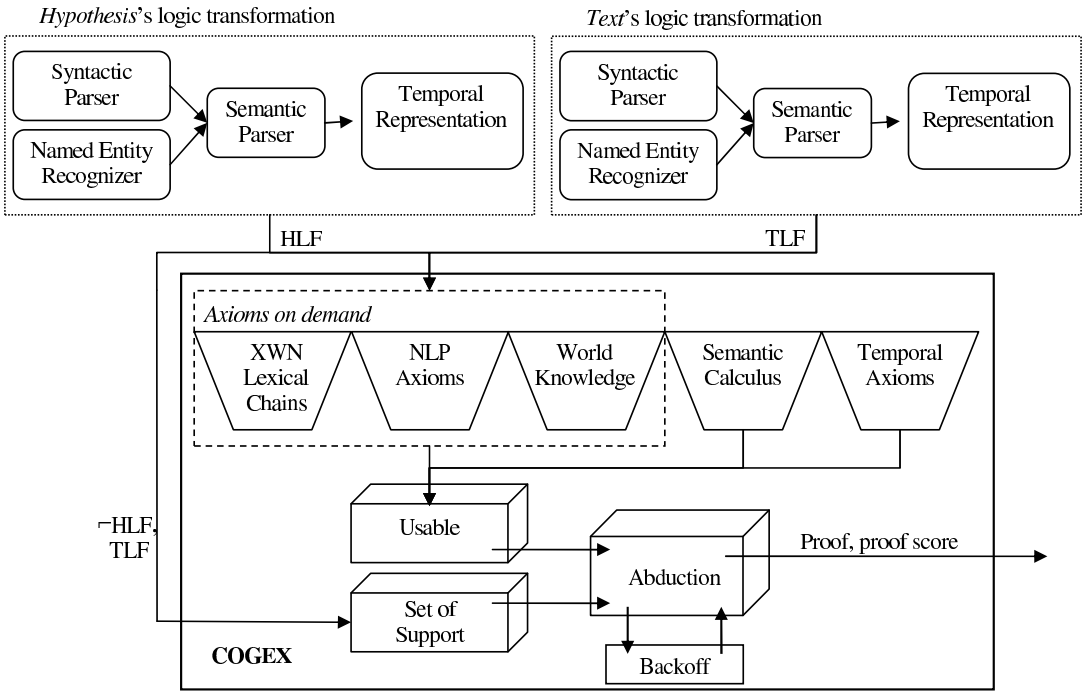

Figure 1: COGEX's Architecture

To our knowledge, there are few logical approaches to textual entailment. [3] represents T and H into a first-order logic translation of the DRS language used in Discourse Representation Theory [6] and uses a theorem prover and a model builder with some generic, lexical and geographical background knowledge to prove the entailment between the two texts. [4] proposes a Description

[2] http://clef-qa.itc.it/CLEF-2006.html
[3] http://www.clef-campaign.org
[4] http://www.pascal-network.org/Challenges/RTE2
[5] Any processing (automatic or manual) of the Spanish data was performed on its English translation.

Logic-based knowledge representation language used to induce the representations of T and H and uses an extended subsumption algorithm to check if any of T's representations obtained through equivalent transformations entails H.

2 Cogex - A Logic Prover for NLP

LCC's entailment system uses COGEX [11], a natural language prover originating from OTTER [9]. The prover requires a list of clauses called the *set of support* which is used to initiate the search for inferences. For this *recognizing textual entailment* task, the negated form of the hypothesis ($\neg H$) as well as the predicates that make up the logic representation of the supporting text passage (T) form the set of support. A second list, called the *usable* list, contains the axioms used to generate inferences. We consider several types of axioms: eXtended WordNet, linguistic, semantic and temporal axioms. Once the set of support and usable lists are created, the logic prover begins its search for a proof. 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. Any newly inferred clause is assigned an appropriate weight depending on its parent-clauses as well as on the axiom it was 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. The purpose of this iterative relaxation process is to determine the prover to output (partial) proofs when the background knowledge captured in the axioms is incomplete and to compensate for parsing errors (for example, incorrect prepositional-phrase attachments).

Due to the logic prover's relaxation techniques, it is always successful in producing a proof. The determination of whether entailment exists is made by examining the penalties assessed by the logic prover in the process of generating the proof. As more axioms are utilized and more predicates are dropped, it becomes much less likely that entailment exists between T and H. All normalized prover scores that fall below a specified threshold are considered false entailment and all scores that are above the threshold are considered true entailment. An appropriate threshold is calculated by examining the scoring output of the development data set to determine what threshold produces the highest F-measure for the positive entailment class.

2.1 Proof scoring algorithm

Once a proof by contradiction is found, its score is computed by starting with an initial perfect score and deducting points for each axiom utilized in the proof, every relaxed argument, and dropped predicate. 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 that both text fragments are existential, then $T \vdash H$ if and only if T's entities are a subset of H's entities (*Some smart people read* $\vdash$ *Some people read*) and penalizing a pair whose H contains predicates that cannot be inferred is a correct way to ensure entailment (*Some people read* $\nvdash$ *Some smart people read*). But, if both T and H are universally quantified, then the groups mentioned in H must be a subset of the ones from T (*All people read* $\vdash$ *All smart people read* and *All smart people read* $\nvdash$ *All people read*). Thus, the scoring module adds back the points for the modifiers dropped from H and subtracts points for T's modifiers not present in H. The remaining two cases are summarized in Table 1.

Because (T, H) pairs with longer sentences can potentially drop more predicates and receive a lower score, COGEX normalizes the proof scores by dividing the assessed penalty by the maximum assessable penalty (all the predicates from H are dropped). If this final proof score is above a threshold learned on the development data, then the pair is labeled as positive entailment.

$(\forall_T, \exists_H)$	$(\exists_T, \forall_H)$
All people read $\vdash$ *Some smart people read*	*Some people read* $\nvdash$ *All smart people read*
All smart people read $\vdash$ *Some people read*	*Some smart people read* $\nvdash$ *All people read*
Add the dropped points for H's modifiers	Subtract points for modifiers not present in H

Table 1: The quantification of T and H influences the proof scoring algorithm

3 Knowledge Representation

For the textual entailment task, our logic prover uses a two-layered logical representation which captures the syntactic and semantic propositions encoded in a text fragment.

3.1 Logic Form Transformation

In the first stage of our representation process, COGEX converts T and H into logic forms [13]. More specifically, a predicate is created for each noun, verb, adjective and adverb. The nouns that form a noun compound are gathered under a `nn_NNC` predicate. Named entities are included in our representation with an `_NE` predicate which shares its argument with the noun it modifies. Predicates for prepositions and conjunctions are also added to link the text's constituents. This syntactic layer of the logic representation is automatically derived from a full parse tree and acknowledges syntax-based relationships such as: syntactic subjects, syntactic objects, prepositional attachments, complex nominals, and adjectival/adverbial adjuncts. These syntactic relations signal semantic relationships which will be captured by the second layer of our representation. If we consider, for instance, the hypothesis *Iraq invaded the country of Kuwait in 1990* [6], `iraq_NN(x1) & _country_NE(x1) & invade_VB(e1,x1,x2) & country_NN(x2) & of_IN(x2,x3) & kuwait_NN(x3) & _country_NE(x3) & in_IN(x3,x4) & 1990_NN(x4) & _date_NE(x4)` constitutes its first layer of logic representation.

3.1.1 Negation

Exceptions to the one-predicate-per-open-class-word rule include the adverbs *not* and *never*. In cases similar to *the provisions of the convention do not affect in any way the exercise of navigational rights and freedoms*, the system removes `not_RB(x3,e1)` and negates the verb's predicate (`-affect_VB(e1,x1,x2)`) unless the verb is modified by an adverb (*They don't admit it in public* is represented as `They_PRP(x8) & admit_VB(e1,x8,x9) & it_PRP(x9) & -in_public_RB(x11,e1)`). Similarly, for nouns whose determiner is *no*, for example, *no science data was lost*, the verb's predicate is negated (`science_NN(x1) & data_NN(x2) & nn_NNC(x3,x1,x2) & -lost_VB(e1,x4,x3)`).

3.2 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[7] 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 representation module 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. For example, the logic form of *Iraq invaded the country of Kuwait in 1990* is augmented with the `AGENT_SR(x1,e1) & THEME_SR(x2,e1) &`

[6] The examples shown in this paper were extracted from the English and the English translated version of the Spanish entailment corpora released as part of AVE 2006. The datasets will be described in Section 7.

[7] We consider relations such as `AGENT`, `THEME`, `TIME`, `LOCATION`, `MANNER`, `CAUSE`, `INSTRUMENT`, `POSSESSION`, `PURPOSE`, `MEASURE`, `KINSHIP`, `ATTRIBUTE`, etc.

`ISA_SR(x3,x2) & TIME_SR(x4,e1)` relations[8] (*Iraq* is the *agent* of the *invade* event, *the country of Kuwait* is its *theme* and *1990* shows the *time* of the *invasion*).

3.3 Temporal Representation

In addition to semantic predicates, we represent every *date/time* into a normalized form `time_TMP( BeginFn(date_event), beginning_year, beginning_month, beginning_day, beginning_hour, beginning_minute, beginning_second) & time_TMP(EndFn(date_event), ending_year, ending_month, ending_day, ending_hour, ending_minute, ending_second)`. Furthermore, 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, E_1, E_2), where S is the temporal signal linking two events E_1 and E_2) [10]. From each triple, temporally related SUMO predicates are generated based on hand-coded rules for the signal classes ((S sequence, E_1, E_2) $\equiv$ `earlier_TMP(e1,e2)`, (S contain, E_1, E_2) $\equiv$ `during_TMP(e1,e2)`, etc.). In the above example, *1990* is normalized to the interval `time_TMP(BeginFn(e2),1990,0,0,0,0,0) & time_TMP(EndFn(e2),1990,12,31,23,59,59)` and `during_TMP(e1,e2)` is added to the logical representation to show the period of time when the *invasion* occurred.

4 Axioms on Demand

COGEX's usable list consists of all the axioms generated either automatically or by hand. The system generates axioms on demand for a given (T, H) pair whenever the semantic connectivity between two concepts needs to be established 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. We are keen on the idea of axioms on demand since it is not possible to derive apriori all axioms needed in an arbitrary proof. This brings a considerable level of robustness to our entailment system.

4.1 eXtended WordNet lexical chains

For the semantic entailment task, the ability to recognize two semantically-related words is an important requirement. Therefore, we automatically construct lexical chains of WordNet relations from T's constituents to H's [12]. In order to avoid errors introduced by a Word Sense Disambiguation system, we used the first k senses for each word[9] unless the source and the target of the chain are synonyms. If a chain exists[10], the system generates, on demand, an axiom with the predicates of the source (from T) and the target (from H). For example, given the ISA relation between *murder#1* and *kill#1*, the system generates, when needed, the axiom `murder_VB(e1,x1,x2)` $\rightarrow$ `kill_VB(e1,x1,x2)`. The remaining of this section details some of the requirements for creating accurate lexical chains.

Because our extended version of WordNet has attached named entities to each noun synset, the lexical chain axioms append the entity name of the target concept, whenever it exists. For example, the logic prover uses the axiom `Nicaraguan_JJ(x1,x2)` $\rightarrow$ `Nicaragua_NN(x1) & _country_NE(x1)` when it tries to infer *electoral campaign is held in Nicaragua* from *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 does not consider chains with an IS-A relation followed by a HYPONYMY link ($Chicago \xrightarrow{is-a} city \xrightarrow{hyponymy} Detroit$). Without removing these types of chains, our system inferred, for instance, *John lives*

[8] `R(x,y)` should be read as "`x` *is* `R` *of* `y`".

[9] Because WordNet senses are ranked based on their frequency, the correct sense is most likely among the first k. In our experiments, $k = 3$.

[10] Each lexical chain is assigned a weight based on its properties: shorter chains are better than longer ones, the relations are not equally important and their order in the chain influences its strength. If the weight of a chain is above a given threshold, the lexical chain is discarded.

in Detroit from *John lives in Chicago* ($Chicago \xrightarrow{is-a} city \xrightarrow{hyponymy} Detroit$). Similarly, the system rejected chains with more than one HYPONYMY relations. Although these relations link semantically related concepts, the type of semantic similarity they introduce is not suited for inferences (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 generated for entailment is not to start from or include too general concepts[11]. Therefore, we assigned to each noun and verb synset from WordNet a generality weight based on its relative position within its hierarchy and on its frequency in a large corpus. If d_i is the depth of concept c_i, D_{H_i} is the maximum depth in c_i's hierarchy H_i and $IC(c_i) = -log(p(c_i))$ is the information content of c_i measured on the British National Corpus, then

$$generalityW(c_i) = \frac{1}{\frac{d_i+1}{D_{H_i}} * IC(c_i)}.$$

In our experiments, we discarded the chains with concepts whose generality weight exceeded 0.8 such as *object_NN#1, act_VB#1, be_VB#1*, 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 `act_VB(e1,x1,x2)` $\rightarrow$ `acting_NN(e1)` (ACT), `act_VB(e1,x1,x2)` $\rightarrow$ `actor_NN(x1)` (AGENT) reflect different types of derivation.

4.2 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 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, the axiom `nn_NNC(x3,x1,x2) & Sigmund_NN(x1) & Freud_NN(x2)` $\rightarrow$ `Freud_NN(x3)` breaks down the noun compound *Sigmund Freud* into *Sigmund* and *Freud*.

This type of axioms proved to be the most frequently used for both the English and Spanish datasets. More specifically, the apposition axioms which show the equivalence between an entity and its explanatory equivalent alone solved 33.22% of the Spanish entailments. For example, COGEX detects the entailment between the English supporting text *The space shuttle Atlantis blasted through low lying clouds yesterday* and the hypothesis *Atlantis is space shuttle* by using the correspondence axioms `space_shuttle_NN(x9) & atlantis_NN(x10)` $\rightarrow$ `space_shuttle_NN(x10)` and `space_shuttle_NN(x9) & atlantis_NN(x10)` $\rightarrow$ `atlantis_NN(x9)`.

Newly introduced types of NLP axioms include implications from named entity classes to nouns that describe the class (for example, the axioms `kuwait_NN(x1) & _country_NE(x1)` $\rightarrow$ `country_NN(x1)` and `kuwait_NN(x1) & _country_NE(x1)` $\rightarrow$ `country_NN(x2) & ISA_SR(x1,x2)` help COGEX infer hypothesis's *country of Kuwait* from texts that mention only *Kuwait*) and logical equivalences between name variations, for instance, *Mikhail Gorbachev*, *Michail Gorbatchov*, and *Mikhail S. Gorbachev*. Most of these correspondences cannot be found in WordNet even if the named entity has its appropriate synset.

4.3 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 eXtended WordNet[12] to, automatically, create our world knowl-

[11] There are no restrictions on the target concept.

[12] `http://xwn.hlt.utdallas.edu`

edge axioms. For example, the first sense of noun *Pope* and its definition *the head of the Roman Catholic Church* introduces the axiom `Pope_NN(x1)` $\leftrightarrow$ `head_NN(x1) & of_IN(x1,x2) & Roman_Catholic_Church_NN(x2)` which is used by prover to show the entailment between 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.* and 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 457 hand-coded world knowledge axioms, 74 where have been manually designed based on the entire development set data, and 383 originate from previous projects. These axioms express knowledge that could not be derived from WordNet regarding employment (the axiom `_country_NE(x1) & negotiator_NN(x2) & nn_NNC(x3,x1,x2)` $\rightarrow$ `work_VB(e1,x2,x4) & for_IN(e1,x1)` helps the prover infer that *Christopher Hill works for the US* from *top US negotiator, Christopher Hill*), family relations, awards, etc.

5 Semantic Calculus

The Semantic Calculus axioms combine two semantic relations identified within a text fragment and increase the semantic connectivity of the text [16]. A semantic axiom which combines two relations, R_i and R_j, is devised by observing the semantic connection between the w_1 and w_3 words for which there exists at least one other word, w_2, such that $R_i(w_1, w_2)$ ($w_1 \stackrel{R_i}{\rightarrow} w_2$) and $R_j(w_2, w_3)$ ($w_2 \stackrel{R_j}{\rightarrow} w_3$) hold true. We note that not any two semantic relations can be combined: R_i and R_j have to be compatible with respect to the part-of-speech of the common argument. Depending on their properties, there are up to 8 combinations between any two semantic relations and their inverses, not counting the combinations between a semantic relation and itself[13]. Many combinations are not semantically significant, for example, `KINSHIP_SR(x1,x2) & TEMPORAL_SR(x2,e1)` is unlikely to be found in text. Trying to solve the semantic combinations one comes upon in text corpora, we analyzed the RTE development corpora and devised rules for some of the $R_i \circ R_j$ combinations encountered. We validated these axioms by checking all the (w_1, w_3) pairs from the LA Times text collection such that $(R_i \circ R_j)(w_1, w_3)$ holds. We have identified 82 semantic axioms that show how semantic relations can be combined. These axioms enable inference of unstated meaning from the semantics detected in text. For example, if T (*even further to include external factors such as weather conditions*) states explicitly the THEME (THM) relation between *include* and *external factors* and the ISA relation between *factors* and *weather conditions*, the logic prover uses the `ISA_SR(x1,x2) & THM_SR(x2,e1)` $\rightarrow$ `THM_SR(x1,e1)` semantic axiom (the dominance of the THEME relation over ISA) to infer the THM(*weather conditions, include*). Another frequent axiom is `LOCATION_SR(x1,x2) & PARTWHOLE_SR(x2,x3)` $\rightarrow$ `LOCATION_SR(x1,x3)`. Given the text *John lives in Dallas, Texas* and using the axiom, the system infers 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(x1,x2) & LOCATION_SR(x2,x3)` $\rightarrow$ `LOCATION_SR(x1,x3)` 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)`).

6 Temporal Axioms

One of the types of temporal axioms that we load in our logic prover links specific dates to more general time intervals. For example, *August 1990* entails the year *1990.* This rule is used to prove that *Iraq invaded the country of Kuwait in 1990* from *Trade sanctions imposed on Iraq after it invaded Kuwait in August 1990.* These axioms are automatically generated before the search for a

[13] Harabagiu and Moldovan [5] lists the exact number of possible combinations for several WordNet relations and part-of-speech classes.

proof starts. Additionally, the prover uses a SUMO knowledge base of temporal reasoning axioms that consists of axioms for a representation of time points and time intervals, Allen [1] primitives, and temporal functions. For example, *during* is a transitive Allen primitive: `during_TMP(e1,e2) & during_TMP(e2,e3)` $\rightarrow$ `during_TMP(e1,e3)`.

7 Experiments and Results

The AVE corpus consists of text-hypothesis pairs generated from the responses to the QA task. The text snippet returned by a QA system as supportive for its answer became the T and the question transformation into a declarative sentence which includes the answer became the hypothesis H. Table 2 present statistics about the datasets provided by the AVE organizers.

	Development (%)			**Test (%)**			
	True	**False**	**Total**	**True**	**False**	**Unknown**	**Total**
English	436 (15.19)	2434 (84.80)	2870	215 (10.29)	1144 (54.78)	729 (34.91)	2088
Spanish	638 (21.96)	2267 (78.03)	2905	671 (28.32)	1615 (68.17)	83 (3.50)	2369

Table 2: Datasets Statistics

The test pairs tagged as *unknown* (pairs with inexact answers or not judged by humans) were ignored in the performance evaluation of the system.

7.1 COGEX's Results

Table 3 summarize COGEX's performance on the English and Spanish datasets. Because our knowledge representation method relies on a full syntactic parse tree of the input and, in the AVE corpus, a large number of hypotheses were incorrect from a syntactic point of view, the logic representations were compromised. This created the largest source of errors for our logic-based system. We performed a deep analysis of the system's results in order to pinpoint the sources of its mistakes and we mainly focused on the negative pairs that COGEX labeled as true entailments which led to a low precision.

	English	**Spanish**
Development data (%)		
Precision for YES class	34.49	53.58
Recall for YES class	72.93	78.52
F-measure for YES class	46.83	63.69
Accuracy	74.84	80.34
Test data (%)		
Precision for YES class	31.87	52.70
Recall for YES class	70.70	71.39
F-measure for YES class	43.93	60.63
Accuracy	71.45	72.79

Table 3: System's Performance

Because there were pairs for which we felt that the system's response which contradicts the human annotated label of the pair was correct (for example, English pair 5183 with the text *that before D-day Hitler's army had ravaged Europe for five years, and many feared Germany might still win the war. Gen. Dwight D. Eisenhower called D-day "a great crusade," but privately, the day before D-day, Eisenhower wrote a message in case things went wrong: "Our landings ...* and the hypothesis *EISENHOWER called D-day "a great crusade"*), one of the authors of the paper manually labeled the test pairs which were annotated by judges with either YES or NO (we discarded the UNKNOWN pairs) and we computed the inter-annotator agreement for this data. The

values presented in Table 4 show substantial agreement for the English pairs and almost perfect agreement for the Spanish data. These findings combined with the system's higher results for the Spanish when compared to English suggests that the English data was more difficult (even for humans) or that pre-processing errors (some of which we listed below) made the English pairs more problematic.

	Proportional agreement (%)	**Kappa agreement (%)**
English	92.05	72.16
Spanish	95.05	88.24

Table 4: Inter-annotator agreement on the YES/NO test pairs

Some of the system's difficulties were posed by the semi-automatically generated English hypotheses which contained the entire supporting text on the answer's position (English test pair 6989, for example, has as *T David Shearer wrote about Hodgkin, Bellany and Vermeer but failed to mention David Hosie who has surely influenced his stolid, iconic figures most of all* and as *H Vermeer was David Shearer wrote about Hodgkin, Bellany and Vermeer but failed to mention David Hosie who has surely influenced his stolid, iconic figures most of all*). With so much overlap between T and H, the system assigned to this type of pairs high scores which lead to *true* assignments in 66.66% of false entailment cases while only 18.57% of the pairs with the entire T inserted in the hypothesis are true entailments. On the other hand, the system effortlessly labelled as *false* the 34 negative pairs from the English test set with empty Ts. For the Spanish data, 34 out of the 35 pairs whose H includes the whole T are false entailments and COGEX incorrectly labelled 41.17% of them.

Another subset of negative pairs that our system mistakenly tagged as positive include pairs whose T contains the correct answer to the question being asked but this answer was not correctly identified and other phrase mentioned in the text was inserted in the hypothesis, for instance, the text *Jordan signed an accord with Israel on July 28 ending the state of war between them and pledging to conclude with a peace treaty* is not entailing the hypothesis *Jordan and Israel signed a peace treaty on state of war* created based on the question *On which day did Jordan and Israel sign a peace treaty?*, but it contains the correct answer. The very high lexical overlap between T and H coupled with COGEX's argument relaxation technique (the system relaxes the *on*-link between *signed* and *state of war*) led to high proof scores.

8 Conclusion

In this paper, we present as part of the Language Computer's system participating in the Answer Validation Exercise, a logic form representation of knowledge which captures syntactic dependencies as well as semantic relations between concepts and includes special temporal predicates. We implemented several changes to our eXtended WordNet lexical chains module which lead to fewer unsound axioms, refined our set of natural language axioms and incorporated in our logic prover semantic and temporal axioms which decrease its dependence on world knowledge. We plan to improve our logic prover to detect false entailments even when the two texts have a high word overlap and further expand our axiom set.

References

[1] J. Allen. Time and Time Again: The Many Ways to Represent Time. *Internatinal Journal of Intelligent Systems*, 4(6):341–355, 1991.

[2] R. Bar-Haim, I. Dagan, B. Dolan, L. Ferro, D. Giampiccolo, B. Magnini, and I. Szpektor. The Second PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the Second PASCAL Challenges Workshop*, 2006.

[3] J. Bos and K. Markert. Recognizing Textual Entailment with Logical Inference. In *Proceedings of HLT/EMNLP 2005*, Vancouver, Canada, October 2005.

[4] R. de Salvo Braz, R. Girju, V. Punyakanok, D. Roth, and M. Sammons. An Inference Model for Semantic Entailment in Natural Language. In *Proceedings of AAAI-2005*, 2005.

[5] S. Harabagiu and D. Moldovan. Knowledge Processing on Extended WordNet. In Christiane Fellbaum, editor, *WordNet: an Electronic Lexical Database and Some of its Applications*, pages 379–405. MIT Press, 1998.

[6] H. Kamp and U. Reyle. *From Discourse to Logic: Introduction to Model-theoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory.* Kluwer Academic Publishers, 1993.

[7] P. Koehn. Europarl: A Parallel Corpus for Statistical Machine Translation. In *MT Summit*, 2005.

[8] P. Koehn, F.J. Och, and D. Marcu. Statistical Phrase-based Translation. In *Proceedings of HLT/NAACL*, Edmonton, Canada, 2005.

[9] William W. McCune. *OTTER 3.0 Reference Manual and Guide*, 1994.

[10] D. Moldovan, C. Clark, and S. Harabagiu. Temporal Context Representation and Reasoning. In *Proceedings of IJCAI*, Edinburgh, Scotland, 2005.

[11] D. Moldovan, C. Clark, S. Harabagiu, and S. Maiorano. COGEX A Logic Prover for Question Answering. In *Proceedings of the HLT/NAACL*, 2003.

[12] D. Moldovan and A. Novischi. Lexical chains for Question Answering. In *Proceedings of COLING*, Taipei, Taiwan, August 2002.

[13] D. Moldovan and V. Rus. Logic Form Transformation of WordNet and its Applicability to Question Answering. In *Proceedings of ACL*, France, 2001.

[14] M. Olteanu, P. Suriyentrakorn, and D. Moldovan. Language Models and Reranking for Machine Translation. In *Proceedings of the NAACL Workshop On Statistical Machine Translation*, 2006.

[15] M. Tatu, B. Iles, J. Slavick, A. Novischi, and D. Moldovan. COGEX at the Second Recognizing Textual Entailment Challenge. In *Proceedings of the Second PASCAL Challenges Workshop*, 2006.

[16] M. Tatu and D. Moldovan. A Semantic Approach to Recognizing Textual Entailment. In *Proceedings of HLT/EMNLP*, 2005.