A Logic-based Semantic Approach to Recognizing Textual Entailment

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

Abstract

This paper proposes a knowledge representation model and a logic proving setting with axioms on demand successfully used for recognizing textual entailments. It also details a lexical inference system which boosts the performance of the deep semantic oriented approach on the RTE data. The linear combination of two slightly different logical systems with the third lexical inference system achieves 73.75% accuracy on the RTE 2006 data.

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 semantic entailment between two text snippets can be employed by many NLP applications. For example, Question Answering systems have to identify texts that entail expected answers. In Multi-document Summarization, the redundant information should be recognized and omitted from the summary.

Trying to boost research in textual inferences, the PASCAL Network proposed the *Recognizing Textual Entailment* (RTE) challenges (Dagan et al., 2005; Bar-Haim et al., 2006). For a pair of two text fragments, the task is to determine if the meaning of one text (the entailed hypothesis denoted by H) can be inferred from the meaning of the other text (the entailing text or T).

In this paper, we propose 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. Text T semantically entails H if its meaning logically implies the meaning of H. Thus, we, first, transform both text fragments into logic form, capture their meaning by detecting the *semantic* relations that hold between their constituents and load these rich logic representations into a natural language logic prover to decide if the entailment holds or not. Figure 1 illustrates our approach to RTE. The following sections of the paper shall detail the logic proving methodology, our logical representation of text and the various types of axioms that the prover uses.

To our knowledge, there are few logical approaches to RTE. (Bos and Markert, 2005) represents T and H into a first-order logic translation of the DRS language used in Discourse Representation Theory (Kamp and Reyle, 1993) 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. (de Salvo Braz et al., 2005) proposes a Description 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

Our system uses COGEX (Moldovan et al., 2003), a natural language prover originating from OTTER (McCune, 1994). Once its set of support is loaded with T and the negated hypothesis ($\neg H$) and its usable list with the axioms needed to gener-

Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions, pages 819–826,
Sydney, July 2006. ©2006 Association for Computational Linguistics

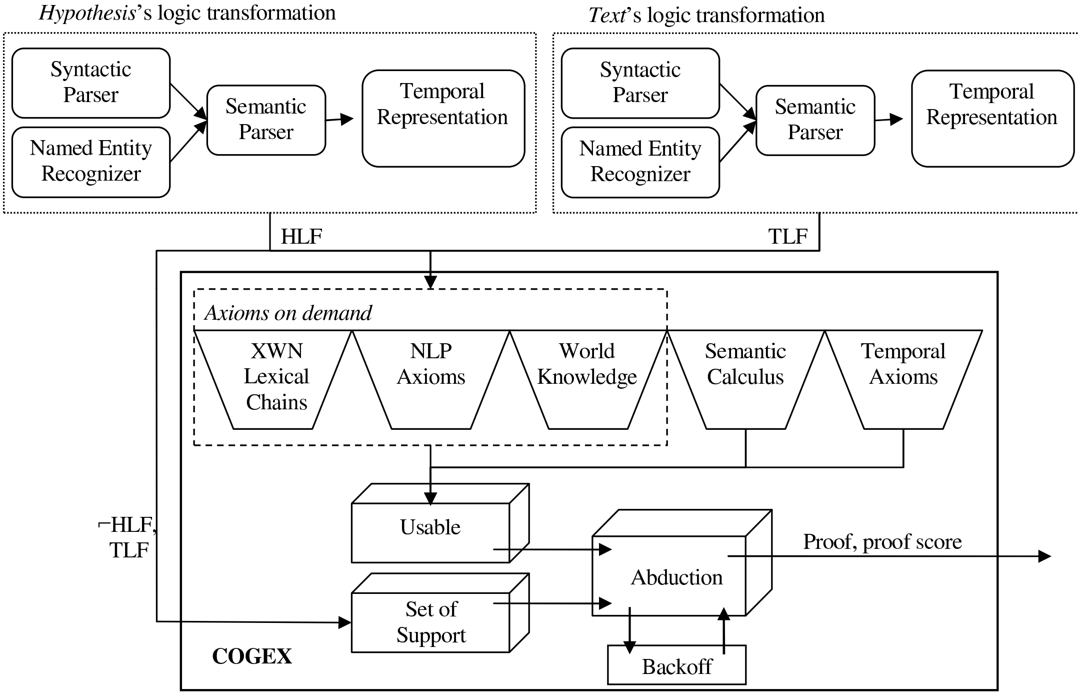

Figure 1: COGEX's Architecture

ate inferences, COGEX begins to search for proofs. To every inference, an appropriate weight is assigned depending on the axiom used for its derivation. If a refutation is found, the proof is complete; if a refutation cannot be found, then predicate arguments are relaxed. When argument relaxation fails to produce a refutation, entire predicates are dropped from the negated hypothesis until a refutation is found.

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* ⊢ *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* ⊬ *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* ⊢ *All smart people read* and *All smart people read* ⊬ *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.

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 (Moldovan and Rus, 2001). 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. Each named entity class of a noun has a corresponding predicate which shares its argument with the noun predicate it modifies. Predicates for

$(\forall_T,\exists_H)$	$(\exists_T,\forall_H)$
All people read ⊢ *Some smart people read* *All smart people read* ⊢ *Some people read*	*Some people read* ⊬ *All smart people read* *Some smart people read* ⊬ *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

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.

In order to objectively evaluate our representation, we derived it from two different sources: *constituency parse trees* (generated with our implementation of (Collins, 1997)) and *dependency parse trees* (created using Minipar (Lin, 1998))[1]. The two logic forms are slightly different. The dependency representation captures more accurately the syntactic dependencies between the concepts, but lacks the semantic information that our semantic parser extracts from the constituency parse trees. For instance, the sentence *Gilda Flores was kidnapped on the 13th of January 1990*[2] is "constituency" represented as `Gilda_NN(x1) & Flores_NN(x2) & nn_NNC(x3,x1,x2) & _human_NE(x3) & kidnap_VB(e1,x9,x3) & on_IN(e1,x8) & 13th_NN(x4) & of_NN(x5) & January_(x6) & 1990_NN(x7) & nn_NNC(x8,x4,x5,x6,x7) & _date_NE(x8)` and its "dependency" logic form is `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)`.

3.1.1 Negation

The exceptions to the one-predicate-per-open-class-word rule include the adverbs *not* and *never*. In cases similar to *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*, the verb's predicate is negated (`case_NN(x1) & -confirm_VB(e2,x15,x1)`).

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[3] 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 previous logic form is augmented with the `THEME_SR(x3,e1) & TIME_SR(x8,e1)` relations[4] (*Gilda Flores* is the *theme* of the *kidnap* event and *13th of January 1990* shows the *time* of the *kidnapping*).

3.3 Temporal Representation

In addition to the semantic predicates, we represent every *date/time* into a normalized form `time_TMP(BeginFn(event), year, month, date, hour, minute, second) & time_TMP(EndFn(event), year, month, date, hour, minute, second)`. Furthermore, temporal reasoning

[1] The experimental results described in this paper were performed using two systems: the logic prover when it receives as input the *constituency* logic representation (COGEX$_C$) and the *dependency* representation (COGEX$_D$).

[2] All examples shown in this paper are from the entailment corpus released as part of the Second RTE challenge (`www.pascal-network.org/Challenges/RTE2`). The RTE datasets will be described in Section 7.

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

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

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) (Moldovan et al., 2005). 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, *13th of January 1990* is normalized to the interval `time_TMP(BeginFn(e2), 1990, 1, 13, 0, 0, 0) & time_TMP(EndFn(e2), 1990, 1, 13, 23, 59, 59)` and `during_TMP(e1,e2)` is added to the logical representation to show when the *kidnapping* 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. The axioms on demand are lexical chains and world knowledge axioms. 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 (Moldovan and Novischi, 2002). In order to avoid errors introduced by a Word Sense Disambiguation system, we used the first k senses for each word[5] unless the source and the target of the chain are synonyms. If a chain exists[6], 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 ($Chicago \xrightarrow{is-a} city \xrightarrow{hyponymy} Detroit$). 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. Another restriction imposed on the lexical chains generated for entailment is not to start from or include too general concepts[7]. 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

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

[6] 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.

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

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)` → `acting_NN(e1)` (ACT), `act_VB(e1,x1,x2)` → `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) & francisco_NN(x1) & merino_NN(x2)` → `merino_NN(x3)` breaks down the noun compound *Francisco Merino* into *Francisco* and *Merino* and helps COGEX infer *Merino's home* from *Francisco Merino's home*.

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[8] 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(x1)` ↔ `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 383 hand-coded world knowledge axioms, where 153 have been manually designed based on the entire development set data, and 230 originate from previous projects. These axioms express knowledge that could not be derived from WordNet regarding employment[9], 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 (Tatu and Moldovan, 2005). 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 $R_i(w_1, w_2)$ ($w_1 \xrightarrow{R_i} w_2$) and $R_j(w_2, w_3)$ ($w_2 \xrightarrow{R_j} w_3$) 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[10]. 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 states explicitly the KINSHIP (KIN) relations between *Nicholas Cage* and *Alice Kim Cage* and between *Alice Kim Cage* and *Kal-el Coppola Cage*, the logic prover uses the `KIN_SR(x1,x2) & KIN_SR(x2,x3)` → `KIN_SR(x1,x3)` semantic axiom (the transitivity of the blood relation) and the symmetry of this relationship (`KIN_SR(x1,x2)`

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

[9] For example, the axiom `_country_NE(x1) & negotiator_NN(x2) & nn_NNC(x3,x1,x2)` → `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.*

[10] Harabagiu and Moldovan (1998) lists the exact number of possible combinations for several WordNet relations and part-of-speech classes.

→ KIN_SR(x2,x1)) to infer H's statement (KIN(*Kal-el Coppola Cage, Nicholas Cage*)). Another frequent axiom is LOCATION_SR(x1,x2) & PARTWHOLE_SR(x2,x3) → 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) → LOCATION_SR(x1,x3) only if the LOCATION relation shows inclusion (*John is in the car in the garage* → LOCATION_SR(John,garage). *John is near the car behind the garage* ↛ 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, *October 2000* entails the year *2000*. These axioms are automatically generated before the search for a 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 (Allen, 1991) primitives, and temporal functions. For example, *during* is a transitive Allen primitive: during_TMP(e1,e2) & during_TMP(e2,e3) → during_TMP(e1,e3).

7 Experiments and Results

The benchmark corpus for the RTE 2005 task consists of seven subsets with a 50%-50% split between the positive entailment examples and the negative ones. Each subgroup corresponds to a different NLP application: Information Retrival (IR), Comparable Documents (CD), Reading Comprehension (RC), Question Answering (QA), Information Extraction (IE), Machine Translation (MT), and Paraphrase Acquisition (PP). The RTE data set includes 1367 English (T, H) pairs from the news domain (political, economical, etc.). The RTE 2006 data covered only four NLP tasks (IE, IR, QA and Multi-document Summarization (SUM)) with an identical split between positive and negative examples. Table 2 presents the data statistics.

	Development set	Test set
RTE 2005	567	800
RTE 2006	800	800

Table 2: Datasets Statistics

7.1 COGEX's Results

Tables 3 and 4 summarize COGEX's performance on the RTE datasets, when it received as input the different-source logic forms[11].

On the RTE 2005 data, the overall performance on the test set is similar for both logic proving runs, COGEX$_C$ and COGEX$_D$. On the development set, the semantically enhanced logic forms helped the prover distinguish better the positive entailments (COGEX$_C$ has an overall higher precision than COGEX$_D$). If we analyze the performance on the test data, then COGEX$_C$ performs slightly better on MT, CD and PP and worse on the RC, IR and QA tasks. The major differences between the two logic forms are the semantic content (incomplete for the dependency-derived logic forms) and, because the text's tokenization is different, the number of predicates in H's logic forms is different which leads to completely different proof scores.

On the RTE 2006 test data, the system which uses the dependency logic forms outperforms COGEX$_C$. COGEX$_D$ performs better on almost all tasks (except SUM) and brings a significant improvement over COGEX$_C$ on the IR task. Some of the positive examples that the systems did not label correctly require world knowledge that we do not have encoded in our axiom set. One example for which both systems returned the wrong answer is pair 353 (test 2006) where, from *China's decade-long practice of keeping its currency valued at around 8.28 yuan to the dollar*, the system should recognize the relation between the *yuan* and *China's currency* and infer that *the currency used in China is the yuan* because a *country's currency* ⊢ *currency used in the country*. Some of the pairs that the prover, currently, cannot handle involve numeric calculus and human-oriented estimations. Consider, for example, pair 359 (dev set, RTE 2006) labeled as positive, for which the logic prover could not determine that *15 safety violations* ⊢ *numerous safety violations*.

The deeper analysis of the systems' output

[11] For the RTE 2005 data, we list the *confidence-weighted score* (*cws*) (Dagan et al., 2005) and, for the RTE 2006 data, the *average precision* (*ap*) measure (Bar-Haim et al., 2006).

Task	COGEX$_C$			COGEX$_D$			LEXALIGN			COMBINATION		
	acc	*cws*	*f*	*acc*	*cws*	*f*	*acc*	*cws*	*f*	*acc*	*cws*	*f*
IE	**58.33**	60.90	60.31	57.50	57.03	51.42	56.66	53.41	59.99	62.50	67.63	57.14
IR	52.22	62.41	15.68	**53.33**	59.67	27.58	50.00	55.92	0.00	68.88	75.77	64.10
CD	**82.00**	88.90	79.69	79.33	87.15	74.38	**82.00**	88.04	80.57	84.66	91.73	82.70
QA	50.00	56.27	0.00	51.53	42.37	64.80	**53.07**	43.76	63.90	60.76	55.05	63.82
RC	53.57	56.38	38.09	**57.14**	59.32	58.33	57.85	60.26	49.57	60.00	62.89	50.00
MT	**55.83**	55.83	53.91	52.50	58.17	27.84	51.66	45.94	67.04	64.16	63.80	66.66
PP	**56.00**	63.11	26.66	54.00	58.15	30.30	50.00	47.03	0.00	68.00	75.27	63.63
TEST	59.37	63.09	48.00	59.12	57.17	54.52	59.12	55.74	59.17	**67.25**	**67.64**	**64.69**
DEV	63.66	63.44	64.48	61.19	63.63	57.52	62.08	59.94	60.83	70.37	71.89	66.66

Table 3: RTE 2005 data results (*accuracy, confidence-weighted score*, and *f-measure* for the true class)

Task	COGEX$_C$			COGEX$_D$			LEXALIGN			COMBINATION		
	acc	*ap*	*f*	*acc*	*ap*	*f*	*acc*	*ap*	*f*	*acc*	*ap*	*f*
IE	58.00	49.71	57.57	**59.00**	59.74	63.71	54.00	49.70	67.14	71.50	62.99	71.36
IR	62.50	65.91	56.14	**73.50**	72.50	73.89	64.50	69.45	65.02	74.00	74.30	72.92
QA	62.00	67.30	48.64	**64.00**	68.16	57.64	58.50	55.78	57.86	70.50	75.10	66.67
SUM	**74.50**	77.60	74.62	74.00	79.68	73.73	70.50	76.82	73.05	79.00	80.33	78.13
TEST	64.25	66.31	60.16	67.62	70.69	67.50	61.87	57.64	66.07	**73.75**	**71.33**	**72.37**
DEV	64.50	64.05	66.19	69.00	70.92	69.31	62.25	62.66	62.72	75.12	76.28	76.83

Table 4: RTE 2006 data results (*accurracy, average precision*, and *f-measure* for the true class)

showed that while WordNet lexical chains and NLP axioms are the most frequently used axioms throughout the proofs, the semantic and temporal axioms bring the highest improvement in accuracy, for the RTE data.

7.2 Lexical Alignment

Inspired by the positive examples whose H is in a high degree lexically subsumed by T, we developed a shallow system which measures their overlap by computing an edit distance between the text and the hypothesis. The cost of *deleting* a word from T ($w_T \rightarrow *$) is equal to 0, the cost of *replacing* a word from T with another from H ($w_T \rightarrow w_H$, where $w_T \neq w_H$ and w_T and w_H are not synonyms in WordNet) equal to ∞ (we do not allow replace operations) and the cost of *inserting* a word from H ($* \rightarrow w_H$) varies with the part-of-speech of the inserted word (higher values for WordNet nouns, adjectives or adverbs, lower for verbs and a minimum value for everything else). Table 5 shows a minimum cost alignment.

The performance of this lexical method (LEXALIGN) is shown in Tables 3 and 4. The alignment technique performs significantly better on the (T, H) pairs in the CD (RTE 2005) and SUM (RTE 2006) tasks. For these tasks, all three systems performed the best because the text of false pairs is not entailing the hypothesis even at the lexical level. For pair 682 (test set, RTE 2006), 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 678's text (test set, RTE 2006) 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. LEXALIGN works surprisingly well on the RTE data. It outperforms the semantic systems on the 2005 QA test data, but it has its limitations. The logic representations are generated from parse trees which are not always accurate (~86% accuracy). Once syntactic and semantic parsers are perfected, the logical semantic approach shall prove its potential.

7.3 Merging three systems

Because the two logical representations and the lexical method are very different and perform better on different sets of tasks, we combined the scores returned by each system[12] to see if a mixed approach performs better than each individual method. For each NLP task, we built a classifier based on the linear combination of the three scores. Each task's classifier labels pair i as positive if $\lambda_{cogex_C} score_C(i) + \lambda_{cogex_D} score_D(i) +$

[12] 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. Each (T, H) pair's lexical alignment score, $score_{LexAlign}$, is the normalized average edit distance cost.

T:	The Council of Europe	has	*	45 member states.	Three countries from ...
		DEL	INS		DEL
H:	The Council of Europe	*	is made up by	45 member states.	*

Table 5: The lexical alignment for RTE 2006 pair 615 (test set)

$\lambda_{LexAlign} score_{LexAlign}(i) > 0.5$, where the optimum values of the classifier's real-valued parameters ($\lambda_{cogex_C}, \lambda_{cogex_D}, \lambda_{LexAlign}$) were determined using a grid search on each development set. Given the different nature of each application, the λ parameters vary with each task. For example, the final score given to each IE 2006 pair is highly dependent on the score given by COGEX when it received as input the logic forms created from the constituency parse trees with a small correction from the dependency parse trees logic form system[13]. For the IE task, the lexical alignment performs the worst among the three systems. On the other hand, for the IR task, the score given by LEXALIGN is taken into account[14]. Tables 3 and 4 summarize the performance of the three system combination. This hybrid approach performs better than all other systems for all measures on all tasks. It displays the same behavior as its dependents: high accuracy on the CD and SUM tasks and many false positives for the IE task.

8 Conclusion

In this paper, we present 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 WordNet lexical chains module which lead to fewer unsound 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 expand our axiom set.

References

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

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

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

M. Collins. 1997. Three Generative, Lexicalized Models for Statistical Parsing. In *Proceedings of the ACL-97*.

I. Dagan, O. Glickman, and B. Magnini. 2005. The PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the PASCAL Challenges Workshop*, Southampton, U.K., April.

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

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

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

D. Lin. 1998. Dependency-based Evaluation of MINIPAR. In *Workshop on the Evaluation of Parsing Systems*, Granada, Spain, May.

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

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

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

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

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

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

[13] $\lambda_{cogex_C} = 1.1, \lambda_{cogex_D} = 0.3, \lambda_{LexAlign} = -0.6$

[14] $\lambda_{cogex_C} = 0.3, \lambda_{cogex_D} = 0.1, \lambda_{LexAlign} = 0.6$