Integrating Logical Representations with Probabilistic Information using Markov Logic

Dan Garrette
University of Texas at Austin
dhg@cs.utexas.edu

Katrin Erk
University of Texas at Austin
katrin.erk@mail.utexas.edu

Raymond Mooney
University of Texas at Austin
mooney@cs.utexas.edu

Abstract

First-order logic provides a powerful and flexible mechanism for representing natural language semantics. However, it is an open question of how best to integrate it with uncertain, probabilistic knowledge, for example regarding word meaning. This paper describes the first steps of an approach to recasting first-order semantics into the probabilistic models that are part of Statistical Relational AI. Specifically, we show how Discourse Representation Structures can be combined with distributional models for word meaning inside a Markov Logic Network and used to successfully perform inferences that take advantage of logical concepts such as factivity as well as probabilistic information on word meaning in context.

1 Introduction

Logic-based representations of natural language meaning have a long history. Representing the meaning of language in a first-order logical form is appealing because it provides a powerful and flexible way to express even complex propositions. However, systems built solely using first-order logical forms tend to be very brittle as they have no way of integrating uncertain knowledge. They, therefore, tend to have high precision at the cost of low recall (Bos and Markert, 2005).

Recent advances in computational linguistics have yielded robust methods that use weighted or probabilistic models. For example, distributional models of word meaning have been used successfully to judge paraphrase appropriateness. This has been done by representing the word meaning in context as a point in a high-dimensional semantics space (Erk and Padó, 2008; Thater et al., 2010; Erk and Padó, 2010). However, these models typically handle only individual phenomena instead of providing a meaning representation for complete sentences. It is a long-standing open question how best to integrate the weighted or probabilistic information coming from such modules with logic-based representations in a way that allows for reasoning over both. See, for example, Hobbs et al. (1993).

The goal of this work is to combine logic-based meaning representations with probabilities in a single unified framework. This will allow us to obtain the best of both situations: we will have the full expressivity of first-order logic and be able to reason with probabilities. We believe that this will allow for a more complete and robust approach to natural language understanding. In order to perform logical inference with probabilities, we draw from the large and active body of work related to Statistical Relational AI (Getoor and Taskar, 2007). Specifically, we make use of Markov Logic Networks (MLNs) (Richardson and Domingos, 2006) which employ weighted graphical models to represent first-order logical formulas. MLNs are appropriate for our approach because they provide an elegant method of assigning weights to first-order logical rules, combining a diverse set of inference rules, and performing inference in a probabilistic way.

While this is a large and complex task, this paper proposes a series of first steps toward our goal. In this paper, we focus on three natural language phenomena and their interaction: implicativity and factivity, word meaning, and coreference. Our framework parses natural language into a logical form, adds rule weights computed by external NLP modules, and performs inferences using an MLN. Our end-to-end approach integrates multiple existing tools. We use Boxer (Bos et al., 2004) to parse natural

language into a logical form. We use Alchemy (Kok et al., 2005) for MLN inference. Finally, we use the exemplar-based distributional model of Erk and Padó (2010) to produce rule weights.

2 Background

Logic-based semantics. Boxer (Bos et al., 2004) is a software package for wide-coverage semantic analysis that provides semantic representations in the form of Discourse Representation Structures (Kamp and Reyle, 1993). It builds on the C&C CCG parser (Clark and Curran, 2004). Bos and Markert (2005) describe a system for Recognizing Textual Entailment (RTE) that uses Boxer to convert both the premise and hypothesis of an RTE pair into first-order logical semantic representations and then uses a theorem prover to check for logical entailment.

Distributional models for lexical meaning. Distributional models describe the meaning of a word through the context in which it appears (Landauer and Dumais, 1997; Lund and Burgess, 1996), where contexts can be documents, other words, or snippets of syntactic structure. Distributional models are able to predict semantic similarity between words based on distributional similarity and they can be learned in an unsupervised fashion. Recently distributional models have been used to predict the applicability of paraphrases in context (Mitchell and Lapata, 2008; Erk and Padó, 2008; Thater et al., 2010; Erk and Padó, 2010). For example, in "The wine left a stain", "result in" is a better paraphrase for "leave" than is "entrust", while the opposite is true in "He left the children with the nurse". Usually, the distributional representation for a word mixes all its usages (senses). For the paraphrase appropriateness task, these representations are then reweighted, extended, or filtered to focus on contextually appropriate usages.

Markov Logic. An MLN consists of a set of weighted first-order clauses. It provides a way of softening first-order logic by making situations in which not all clauses are satisfied less likely but not impossible (Richardson and Domingos, 2006). More formally, let X be the set of all propositions describing a world (i.e. the set of all ground atoms), $\mathcal{F}$ be the set of all clauses in the MLN, w_i be the weight associated with clause $f_i \in \mathcal{F}$, $\mathcal{G}_{f_i}$ be the set of all possible groundings of clause f_i, and $\mathcal{Z}$ be the normalization constant. Then the probability of a particular truth assignment $\mathbf{x}$ to the variables in X is defined as:

$$P(X = \mathbf{x}) = \frac{1}{\mathcal{Z}} \exp\left(\sum_{f_i \in \mathcal{F}} w_i \sum_{g \in \mathcal{G}_{f_i}} g(\mathbf{x})\right) = \frac{1}{\mathcal{Z}} \exp\left(\sum_{f_i \in \mathcal{F}} w_i n_i(\mathbf{x})\right) \quad (1)$$

where $g(\mathbf{x})$ is 1 if g is satisfied and 0 otherwise, and $n_i(\mathbf{x}) = \sum_{g \in \mathcal{G}_{f_i}} g(\mathbf{x})$ is the number of groundings of f_i that are satisfied given the current truth assignment to the variables in X. This means that the probability of a truth assignment rises exponentially with the number of groundings that are satisfied.

Markov Logic has been used previously in other NLP application (e.g. Poon and Domingos (2009)). However, this paper marks the first attempt at representing deep logical semantics in an MLN.

While it is possible learn rule weights in an MLN directly from training data, our approach at this time focuses on incorporating weights computed by external knowledge sources. Weights for word meaning rules are computed from the distributional model of lexical meaning and then injected into the MLN. Rules governing implicativity and coreference are given infinite weight (hard constraints).

3 Evaluation and phenomena

Textual entailment offers a good framework for testing whether a system performs correct analyses and thus draws the right inferences from a given text. For example, to test whether a system correctly handles implicative verbs, one can use the *premise* p along with the *hypothesis* h in (1) below. If the system analyses the two sentences correctly, it should infer that h holds. While the most prominent forum using textual entailment is the Recognizing Textual Entailment (RTE) challenge (Dagan et al., 2005), the RTE datasets do not test the phenomena in which we are interested. For example, in order to evaluate our system's ability to determine word meaning in context, the RTE pair would have to specifically test word

sense confusion by having a word's context in the hypothesis be different from the context of the premise. However, this simply does not occur in the RTE corpora. In order to properly test our phenomena, we construct hand-tailored premises and hypotheses based on real-world texts.

In this paper, we focus on three natural language phenomena and their interaction: implicativity and factivity, word meaning, and coreference. The first phenomenon, implicativity and factivity, is concerned with analyzing the truth conditions of nested propositions. For example, in the premise of the entailment pair shown in example (1), "arrange that" falls under the scope of "forget to" and "fail" is under the scope of "arrange that". Correctly recognizing nested propositions is necessary for preventing false inferences such as the one in example (2).

(1) p: Ed did not forget to arrange that Dave fail[1]
h: Dave failed

(2) p: The mayor hoped to build a new stadium[2]
h*: The mayor built a new stadium

For the second phenomenon, word meaning, we address paraphrasing and hypernymy. For example, in (3) "covering" is a good paraphrase for "sweeping" while "brushing" is not.

(3) p: A stadium craze is **sweeping** the country
h_1: A stadium craze is **covering** the country
h_2*: A stadium craze is **brushing** the country

The third phenomenon is coreference, as illustrated in (4). For this example, to correctly judge the hypothesis as entailed, it is necessary to recognize that "he" corefers with "Christopher" and "the new ballpark" corefers with "a replacement for Candlestick Park".

(4) p: George Christopher has been a critic of the plan to build a replacement for Candlestick Park. As a result, he won't endorse the new ballpark.
h: Christopher won't endorse a replacement for Candlestick Park.

Some natural language phenomena are most naturally treated as categorial, while others are more naturally treated using weights or probabilities. In this paper, we treat implicativity and coreference as categorial phenomena, while using a probabilistic approach to word meaning.

4 Transforming natural language text to logical form

In transforming natural language text to logical form, we build on the software package Boxer (Bos et al., 2004). We chose to use Boxer for two main reasons. First, Boxer is a wide-coverage system that can deal with arbitrary text. Second, the DRSs that Boxer produces are close to the standard first-order logical forms that are required for use by the MLN software package Alchemy. Our system transforms Boxer output into a format that Alchemy can read and augments it with additional information.

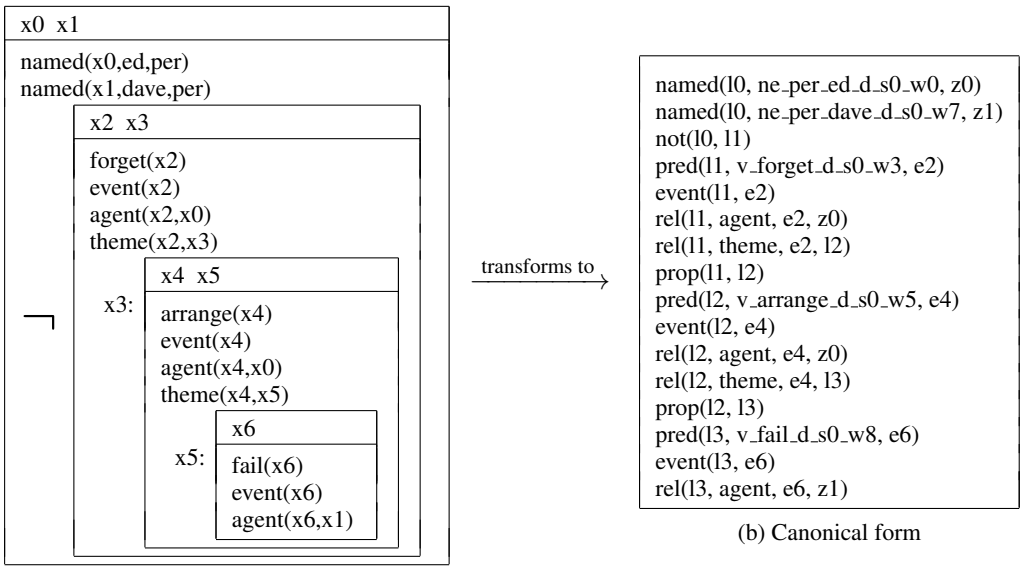
To demonstrate our transformation procedure, consider again the premise of example (1). When given to Boxer, the sentence produces the output given in Figure 1a. We then transform this output to the format given in Figure 1b.

Flat structure. In Boxer output, nested propositional statements are represented as nested sub-DRS structures. For example, in the premise of (1), the verbs "forget to" and "arrange that" both introduce nested propositions, as is shown in Figure 1a where DRS *x3* (the "arranging that") is the *theme* of "forget to" and DRS *x5* (the "failing") is the *theme* of "arrange that".

In order to write logical rules about the truth conditions of nested propositions, the structure has to be flattened. However, it is clearly not sufficient to just conjoin all propositions at the top level. Such an approach, applied to example (2), would yield $(hope(x_1) \wedge theme(x_1, x_2) \wedge build(x_2) \wedge \ldots)$, leading to the wrong inference that the stadium was built. Instead, we add a new argument to each predicate that

[1]Examples (1) and (16) and Figure 2 are based on examples by MacCartney and Manning (2009)

[2]Examples (2), (3), (4), and (18) are modified versions of sentences from document wsj_0126 from the Penn Treebank



(a) Output from Boxer

(b) Canonical form

Figure 1: Converting the premise of (1) from Boxer output to MLN input

names the DRS in which the predicate originally occurred. Assigning the label *l1* to the DRS containing the predicate *forget*, we add *l1* as the first argument to the atom *pred(l1, v_forget_d_s0_w3, e2)*.[3] Having flattened the structure, we need to re-introduce the information about relations between DRSs. For this we use predicates *not*, *imp*, and *or* whose arguments are DRS labels. For example, $not(l0, l1)$ states that $l1$ is inside $l0$ and negated. Additionally, an atom $prop(l0, l1)$ indicates that DRS $l0$ has a subordinate DRS labeled $l1$.

One important consequence of our flat structure is that the truth conditions of our representation no longer coincide with the truth conditions of the underlying DRS being represented. For example, we do not directly express the fact that the "forgetting" is actually negated, since the negation is only expressed as a relation between DRS labels. To access the information encoded in relations between DRS labels, we add predicates that capture the truth conditions of the underlying DRS. We use the predicates $true(label)$ and $false(label)$ that state whether the DRS referenced by $label$ is *true* or *false*. We also add rules that govern how the predicates for logical operators interact with these truth values. For example, the rules in (5) state that if a DRS is *true*, then any negated subordinate must be *false* and vice versa.

$$\forall\, p\, n.[not(p, n) \rightarrow (true(p) \leftrightarrow false(n)) \wedge (false(p) \leftrightarrow true(n))] \tag{5}$$

Injecting additional information into the logical form. We want to augment Boxer output with additional information, for example gold coreference annotation for sentences that we subsequently analyze with Boxer. In order to do so, we need to be able to tie predicates in the Boxer output back to words in the original sentence. Fortunately, the optional "Prolog" output format from Boxer provides the sentence and word indices from the original sentence. When parsing the Boxer output, we extract these indices and concatenate them to the word lemma to specific the exact occurrence of the lemma that is under discussion. For example, the atom *pred(l1, v_forget_d_s0_w3, e2)* indicates that event *e2* refers to the lemma "forget" that appears in the 0^{th} sentence of discourse *d* at word index 3.

Atomic formulas. We represent the words from the sentence as arguments instead of predicates in order to simplify the set of inference rules we need to specify. Because our flattened structure requires that the inference mechanism be reimplemented as a set of logical rules, it is desirable for us to be able to write general rules that govern the interaction of atoms. With the representation we have chosen, we can quantify over all predicates or all relations. For example, the rule in (6) states that a predicate is accessible if it is found in an out-scoping DRS.

[3]The extension to the word, such as *d_s0_w3* for "forget", is an index providing the location of the original word that triggered this atom; this is addressed in more detail shortly.

	signature	example
managed to	+/-	he managed to escape ⊨ he escaped he did not manage to escape ⊨ he did not escape
refused to	-/o	he refused to fight ⊨ he did not fight he did not refuse to fight ⊭ {he fought, he did not fight}

Figure 2: Implication Signatures

$$\forall\, l_1\; l_2.[outscopes(l_1, l_2) \rightarrow \forall\, p\; x.[pred(l_1, p, x) \rightarrow pred(l_2, p, x)]] \tag{6}$$

We use three different predicate symbols to distinguish three types of atomic concepts: predicates, named entities, and relations. Predicates and named entities represent words that appear in the text. For example, *named(l0, ne_per_ed_d_s0_w0, z0)* indicates that variable *z0* is a person named "Ed" while *pred(l1, v_forget_d_s0_w3, e2)* says that *e2* is a "forgetting to" event. Relations capture the relationships between words. For example, *rel(l1, agent, e2, z0)* indicates that *z0*, "Ed", is the "agent" of the "forgetting to" event *e2*.

5 Handling the phenomena

Implicatives and factives

Nairn et al. (2006) presented an approach to the treatment of inferences involving implicatives and factives. Their approach identifies an "implication signature" for every implicative or factive verb that determines the truth conditions for the verb's nested proposition, whether in a positive or negative environment. Implication signatures take the form "x/y" where x represents the implicativity in the the positive environment and y represents the implicativity in the negative environment. Both x and y have three possible values: "+" for positive entailment, meaning the nested proposition is entailed, "-" for negative entailment, meaning the negation of the proposition is entailed, and "o" for "null" entailment, meaning that neither the proposition nor its negation is entailed. Figure 2 gives concrete examples.

We use these implication signatures to automatically generate rules that license specific entailments in the MLN. Since "forget to" has implication signature "-/+", we generate the two rules in (7).

(7) (a) $\forall\, l_1\; l_2\; e.[(pred(l_1, \text{“}forget\text{”}, e) \wedge true(l_1) \wedge rel(l_1, \text{“}theme\text{”}, e, l_2) \wedge prop(l_1, l_2)) \rightarrow false(l_2)]]$[4]

(b) $\forall\, l_1\; l_2\; e.[(pred(l_1, \text{“}forget\text{”}, e) \wedge false(l_1) \wedge rel(l_1, \text{“}theme\text{”}, e, l_2) \wedge prop(l_1, l_2)) \rightarrow true(l_2)]$

To understand these rules, consider (7a). The rule says that if the atom for the verb "forget to" appears in a DRS that has been determined to be *true*, then the DRS representing any "theme" proposition of that verb should be considered *false*. Likewise, (7b) says that if the occurrence of "forget to" appears in a DRS determined to be *false*, then the theme DRS should be considered *true*.

Note that when the implication signature indicates a "null" entailment, no rule is generated for that case. This prevents the MLN from licensing entailments related directly to the nested proposition, but still allows for entailments that include the factive verb. So *he wanted to fly* entails neither *he flew* nor *he did not fly*, but it does still license *he wanted to fly*.

Ambiguity in word meaning

In order for our system to be able to make correct natural language inference, it must be able to handle paraphrasing and deal with hypernymy. For example, in order to license the entailment pair in (8), the system must recognize that "owns" is a valid paraphrase for "has", and that "car" is a hypernym of "convertible".

(8) *p:* Ed has a convertible
h: Ed owns a car

[4]Occurrence-indexing on the predicate "forget" has been left out for brevity.

In this section we discuss our probabilistic approach to paraphrasing. In the next section we discuss how this approach is extended to cover hypernymy. A central problem to solve in the context of paraphrases is that they are context-dependent. Consider again example (3) and its two hypotheses. The first hypothesis replaces the word "sweeping" with a paraphrase that is valid in the given context, while the second uses an incorrect paraphrase.

To incorporate paraphrasing information into our system, we first generate rules stating all paraphrase relationships that may *possibly* apply to a given predicate/hypothesis pair, using WordNet (Miller, 2009) as a resource. Then we associate those rules with weights to signal contextual adequacy. For any two occurrence-indexed words w_1, w_2 occurring anywhere in the premise or hypothesis, we check whether they co-occur in a WordNet synset. If w_1, w_2 have a common synset, we generate rules of the form $\forall\ l\ x.[pred(l, w_1, x)\ \leftrightarrow\ pred(l, w_2, x)]$ to connect them. For named entities, we perform a similar routine: for each pair of matching named entities found in the premise and hypothesis, we generate a rule $\forall\ l\ x.[named(l, w_1, x) \leftrightarrow named(l, w_2, x)]$.

We then use the distributional model of Erk and Padó (2010) to compute paraphrase appropriateness. In the case of (3) this means measuring the cosine similarity between the vectors for "sweep" and "cover" (and between "sweep" and "brush") in the sentential context of the premise. MLN formula weights are expected to be log-odds (i.e., $\log(P/(1-P))$ for some probability P), so we rank all possible paraphrases of a given word w by their cosine similarity to w and then give them probabilities that decrease by rank according to a Zipfian distribution. So, the k^{th} closest paraphrase by cosine similarity will have probability P_k given by (9):

$$P_k \sim 1/k \tag{9}$$

The generated rules are given in (10) with the actual weights calculated for example (3). Note that the valid paraphrase "cover" is given a higher weight than the incorrect paraphrase "brush", which allows the MLN inference procedure to judge h_1 as a more likely entailment than h_2.[5] This same result would not be achieved if we did not take context into consideration because, without context, "brush" is a more likely paraphrase of "sweep" than "cover".

(10) (a) -2.602 $\forall\ l\ x.[pred(l, \text{“}v\_sweep\_p\_s0\_w4\text{”}, x) \leftrightarrow pred(l, \text{“}v\_cover\_h\_s0\_w4\text{”}, x)]$
(b) -3.842 $\forall\ l\ x.[pred(l, \text{“}v\_sweep\_p\_s0\_w4\text{”}, x) \leftrightarrow pred(l, \text{“}v\_brush\_h\_s0\_w4\text{”}, x)]$

Since Alchemy outputs a probability of entailment and not a binary judgment, it is necessary to specify a probability threshold indicating entailment. An appropriate threshold between "entailment" and "non-entailment" will be one that separates the probability of an inference with the valid rule from the probability of an inference with the invalid rule. While we plan to automatically induce a threshold in the future, our current implementation uses a value set manually.

Hypernymy

Like paraphrasehood, hypernymy is context-dependent: In "A bat flew out of the cave", "animal" is an appropriate hypernym for "bat", but "artifact" is not. So we again use distributional similarity to determine contextual appropriateness. However, we do not directly compute cosine similarities between a word and its potential hypernym. We can hardly assume "baseball bat" and "artifact" to occur in similar distributional contexts. So instead of checking for similarity of "bat" and "artifact" in a given context, we check "bat" and "club". That is, we pick a synonym or close hypernym of the word in question ("bat") that is also a WordNet hyponym of the hypernym to check ("artifact").

A second problem to take into account is the interaction of hypernymy and polarity. While (8) is a valid pair, (11) is not, because "have a convertible" is under negation. So, we create weighted rules of the form $hypernym(w, h)$, along with inference rules to guide their interaction with polarity. We create

[5] Because weights are calculated according to the equation $\log(P/(1-P))$, any paraphrase that has a probability of less than 0.5 will have a negative weight. Since most paraphrases will have probabilities less than 0.5, most will yield negative rule weights. However, the inferences are still handled properly in the MLN because the inference is dependent on the *relative* weights.

these rules for all pairs of words w, h in premise and hypothesis such that h is a hypernym of w, again using WordNet to determine potential hypernymy.

(11) *p:* Ed does not have a convertible
h: Ed does not own a car

Our inference rules governing the interaction of hypernymy and polarity are given in (12). The rule in (12a) states that in a positive environment, the hyponym entails the hypernym while the rule in (12b) states that in a negative environment, the opposite is true: the hypernym entails the hyponym.

(12) (a) $\forall\, l\ p_1\ p_2\ x.[(hypernym(p_1, p_2) \wedge true(l) \wedge pred(l, p_1, x)) \rightarrow pred(l, p_2, x)]]$
(b) $\forall\, l\ p_1\ p_2\ x.[(hypernym(p_1, p_2) \wedge false(l) \wedge pred(l, p_2, x)) \rightarrow pred(l, p_1, x)]]$

Making use of coreference information

As a test case for incorporating additional resources into Boxer's logical form, we used the coreference data in OntoNotes (Hovy et al., 2006). However, the same mechanism would allow us to transfer information into Boxer output from arbitrary additional NLP tools such as automatic coreference analysis tools or semantic role labelers. Our system uses coreference information into two distinct ways.

The first way we make use of coreference data is to copy atoms describing a particular variable to those variables that corefer. Consider again example (4) which has a two-sentence premise. This inference requires recognizing that the "he" in the second sentence of the premise refers to "George Christopher" from the first sentence. Boxer alone is unable to make this connection, but if we receive this information as input, either from gold-labeled data or a third-party coreference tool, we are able to incorporate it. Since Boxer is able to identify the index of the word that generated a particular predicate, we can tie each predicate to any related coreference chains. Then, for each atom on the chain, we can inject copies of all of the coreferring atoms, replacing the variables to match. For example, the word "he" generates an atom *pred(l0, male, z5)*[6] and "Christopher" generates atom *named(l0, christopher, x0)*. So, we can create a new atom by taking the atom for "christopher" and replacing the label and variable with that of the atom for "he", generating *named(l0, christopher, x5)*.

As a more complex example, the coreference information will inform us that "the new ballpark" corefers with "a replacement for Candlestick Park". However, our system is currently unable to handle this coreference correctly at this time because, unlike the previous example, the expression "a replacement for Candlestick Park" results in a complex three-atom conjunct with two separate variables: *pred(l2, replacement, x6)*, *rel(l2, for, x6, x7)*, and *named(l2, candlestick_park, x7)*. Now, unifying with the atom for "a ballpark", *pred(l0, ballpark, x3)*, is not as simple as replacing the variable because there are two variables to choose from. Note that it would *not* be correct to replace both variables since this would result in a unification of "ballpark" with "candlestick_park" which is wrong. Instead we must determine that *x6* should be the one to unify with *x3* while *x7* is replaced with a fresh variable. The way that we can accomplish this is to look at the dependency parse of the sentence that is produced by the C&C parser is a precursor to running Boxer. By looking up both "replacement" and "Candlestick Park" in the parse, we can determine that "replacement" is the head of the phrase, and thus is the atom whose variable should be unified. So, we would create new atoms, *pred(l0, replacement, x3)*, *rel(l0, for, x3, z0)*, and *named(l0, candlestick_park, z0)*, where *z0* is a fresh variable.

The second way that we make use of coreference information is to extend the sentential contexts used for calculating the appropriateness of paraphrases in the distributional model. In the simplest case, the sentential context of a word would simply be the other words in the sentence. However, consider the context of the word "lost" in the second sentence of (13).

(13) p_1: In [the final game of the season]$_1$, [the team]$_2$ held on to their lead until overtime
p_2: But despite that, [they]$_2$ eventually **lost** [it all]$_1$

[6] Atoms simplified for brevity

Here we would like to disambiguate "lost", but its immediate context, words like "despite" and "eventually", gives no indication as to its correct sense. Our procedure extends the context of the sentence by incorporating all of the words from all of the phrases that corefer with a word in the immediate context. Since coreference chains 1 and 2 have words in p_2, the context of "lost" ends up including "final", "game", "season", and "team" which give a strong indication of the sense of "lost". Note that using coreference data is stronger than simply expanding the window because coreferences can cover arbitrarily long distances.

6 Evaluation

As a preliminary evaluation of our system, we constructed a set of demonstrative examples to test our ability to handle the previously discussed phenomena and their interactions and ran each example with both a theorem prover and Alchemy. Note that when running an example in the theorem prover, weights are not possible, so any rule that would be weighted in an MLN is simply treated as a "hard clause" following Bos and Markert (2005).

Checking the logical form. We constructed a list of 72 simple examples that exhaustively cover cases of implicativity (positive, negative, null entailments in both positive and negative environments), hypernymy, quantification, and the interaction between implicativity and hypernymy. The purpose of these simple tests is to ensure that our flattened logical form and truth condition rules correctly maintain the semantics of the underlying DRSs. Examples are given in (14).

(14) (a) The mayor did not manage to build a stadium $\nvDash$ The mayor built a stadium
(b) Fido is a dog and every dog walks $\vDash$ A dog walks

Examples in previous sections. Examples (1), (2), (3), (8), and (11) all come out as expected. Each of these examples demonstrates one of the phenomena in isolation. However, example (4) returns "not entailed", the incorrect answer. As discussed previously, this failure is a result of our system's inability to correctly incorporate the complex coreferring expression "a replacement for Candlestick Park". However, the system *is* able to correctly incorporate the coreference of "he" in the second sentence to "Christopher" in the first.

Implicativity and word sense. For example (15), "fail to" is a negatively entailing implicative in a positive environment. So, p correctly entails h_{good} in both the theorem prover and Alchemy. However, the theorem prover incorrectly licenses the entailment of h_{bad} while Alchemy does not. The probabilistic approach performs better in this situation because the categorial approach does not distinguish between a good paraphrase and a bad one. This example also demonstrates the advantage of using a context-sensitive distributional model to calculate the probabilities of paraphrases because "reward" is an *a priori* better paraphrase than "observe" according to WordNet since it appears in a higher ranked synset.

(15) p: The U.S. is watching closely as South Korea fails to honor U.S. patents[7]
h_{good}: South Korea does not **observe** U.S. patents
h_{bad}: South Korea does not **reward** U.S. patents

Implicativity and hypernymy. MacCartney and Manning (2009) extended the work by Nairn et al. (2006) in order to correctly treat inference involving monotonicity and exclusion. Our approaches to implicatives and factivity and hyper/hyponymy combine naturally to address these issues because of the structure of our logical representations and rules. For example, no additional work is required to license the entailments in (16).

(16) (a) John refused to dance $\vDash$ John didn't tango
(b) John did not forget to tango $\vDash$ John danced

[7]Example (15) is adapted from Penn Treebank document wsj_0020 while (17) is adapted from document wsj_2358

Example (17) demonstrates how our system combines categorial implicativity with a probabilistic approach to hypernymy. The verb "anticipate that" is positively entailing in the negative environment. The verb "moderate" can mean "chair" as in "chair a discussion" or "curb" as in "curb spending". Since "restrain" is a hypernym of "curb", it receives a weight based on the applicability of the word "curb" in the context. Similarly, "talk" receives a weight based on its hyponym "chair". Since our model predicts "curb" to be a more probable paraphrase of "moderate" in this context than "chair" (even though the priors according to WordNet are reversed), the system is able to infer h_{good} while rejecting h_{bad}.

(17) p: He did not anticipate that inflation would moderate this year
h_{good}: Inflation **restrained** this year
h_{bad}: Inflation **talked** this year

Word sense, coreference, and hypernymy. Example (18) demonstrates the interaction between paraphrase, hypernymy, and coreference incorporated into a single entailment. The relevant coreference chains are marked explicitly in the example. The correct inference relies on recognizing that "he" in the hypothesis refers to "Joe Robbie" and "it" to "coliseum", which is a hyponym of "stadium". Further, our model recognizes that "sizable" is a better paraphrase for "healthy" than "intelligent" even though WordNet has the reverse order.

(18) p: [Joe Robbie]$_{53}$ couldn't persuade the mayor , so [he]$_{53}$ built [[his]$_{53}$ own coliseum]$_{54}$.
[He]$_{53}$ has used [it]$_{54}$ to turn a healthy profit.[8]
h_{good}: *Joe Robbie* used *a stadium* to turn a **sizable** profit
h_{bad-1}: *Joe Robbie* used *a stadium* to turn an **intelligent** profit
h_{bad-2}: *The mayor* used *a stadium* to turn a healthy profit

7 Future work

The next step is to execute a full-scale evaluation of our approach using more varied phenomena and naturally occurring sentences. However, the memory requirements of Alchemy are a limitation that prevents us from currently executing larger and more complex examples. The problem arises because Alchemy considers every possible grounding of every atom, even when a more focused subset of atoms and inference rules would suffice. There is on-going work to modify Alchemy so that only the required groundings are incorporated into the network, reducing the size of the model and thus making it possible to handle more complex inferences. We will be able to begin using this new version of Alchemy very soon and our task will provide an excellent test case for the modification.

Since Alchemy outputs a probability of entailment, it is necessary to fix a threshold that separates entailment from nonentailment. We plan to use machine learning techniques to compute an appropriate threshold automatically from a calibration dataset such as a corpus of valid and invalid paraphrases.

8 Conclusion

In this paper, we have introduced a system that implements a first step towards integrating logical semantic representations with probabilistic weights using methods from Statistical Relational AI, particularly Markov Logic. We have focused on three phenomena and their interaction: implicatives, coreference, and word meaning. Taking implicatives and coreference as categorial and word meaning as probabilistic, we have used a distributional model to generate paraphrase appropriateness ratings, which we then transformed into weights on first order formulas. The resulting MLN approach is able to correctly solve a number of difficult textual entailment problems that require handling complex combinations of these important semantic phenomena.

[8] Only relevent coreferences have been marked

References

Bos, J., S. Clark, M. Steedman, J. R. Curran, and J. Hockenmaier (2004). Wide-coverage semantic representations from a CCG parser. In *Proceedings of COLING 2004*, Geneva, Switzerland, pp. 1240–1246.

Bos, J. and K. Markert (2005). Recognising textual entailment with logical inference. In *Proceedings of EMNLP 2005*, pp. 628–635.

Clark, S. and J. R. Curran (2004). Parsing the WSJ using CCG and log-linear models. In *Proceedings of ACL 2004*, Barcelona, Spain, pp. 104–111.

Dagan, I., O. Glickman, and B. Magnini (2005). The pascal recognising textual entailment challenge. In *In Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*.

Erk, K. and S. Padó (2008). A structured vector space model for word meaning in context. In *Proceedings of EMNLP 2008*, Honolulu, HI, pp. 897–906.

Erk, K. and S. Padó (2010). Exemplar-based models for word meaning in context. In *Proceedings of ACL 2010*, Uppsala, Sweden, pp. 92–97.

Getoor, L. and B. Taskar (Eds.) (2007). *Introduction to Statistical Relational Learning*. Cambridge, MA: MIT Press.

Hobbs, J. R., M. Stickel, D. Appelt, and P. Martin (1993). Interpretation as abduction. *Artificial Intelligence 63*(1–2), 69–142.

Hovy, E., M. Marcus, M. Palmer, L. Ramshaw, and R. Weischedel (2006). Ontonotes: The 90% solution. In *Proceedings of HLT/NAACL 2006*, pp. 57–60.

Kamp, H. and U. Reyle (1993). *From Discourse to Logic; An Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and DRT*. Dordrecht: Kluwer.

Kok, S., P. Singla, M. Richardson, and P. Domingos (2005). The Alchemy system for statistical relational AI. Technical report, Department of Computer Science and Engineering, University of Washington. `http://www.cs.washington.edu/ai/alchemy`.

Landauer, T. and S. Dumais (1997). A solution to Platos problem: the latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review 104*(2), 211–240.

Lund, K. and C. Burgess (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments, and Computers 28*, 203—208.

MacCartney, B. and C. D. Manning (2009). An extended model of natural logic. In *Proceedings of the Eighth International Conference on Computational Semantics (IWCS-8)*, pp. 140–156.

Miller, G. A. (2009). Wordnet - about us. `http://wordnet.princeton.edu`.

Mitchell, J. and M. Lapata (2008). Vector-based models of semantic composition. In *Proceedings of ACL*, pp. 236–244.

Nairn, R., C. Condoravdi, and L. Karttunen (2006). Computing relative polarity for textual inference. In *Proceedings of Inference in Computational Semantics (ICoS-5)*, Buxton, UK.

Poon, H. and P. Domingos (2009). Unsupervised semantic parsing. In *Proceedings of EMNLP 2009*, pp. 1–10.

Richardson, M. and P. Domingos (2006). Markov logic networks. *Machine Learning 62*, 107–136.

Thater, S., H. Fürstenau, and M. Pinkal (2010). Contextualizing semantic representations using syntactically enriched vector models. In *Proceedings of ACL 2010*, Uppsala, Sweden, pp. 948–957.