Incrementally Identifying Objects from Referring Expressions using Spatial Object Models

Gaurav Manek

December 15, 2015

1 Research Question

We want to parse referring expressions incrementally, identifying objects in physical scenes.

We assume that we have access to:

1. an object model that includes the position of each object, and
2. a language model that, when provided with a noun phrase, can return a probability distribution over objects that it may refer to.

We use this information to construct an incremental parser that takes referring expressions as input and produces a probability distribution of possible objects that the expression is identifying.

(This thesis project will build on work done from Fall 2014 to present. Some work was previously completed in collaboration with undergraduate student Zachary Loery.)

2 Significance

Incremental parsing of referring expressions is an important problem because referring expressions are commonly used in natural human language when identifying objects. Improving on the ability of a robot to understand these expressions and making this understanding incremental can facilitate more natural human-robot interaction, especially in tasks where the robot needs to identify one object from many.

Currently, referring expression parsing is done with the entire referring expression as input: the G^3 framework by Tellex et al. (2011), work done by Matuszek et al. (2012), and the parser developed by Artzi and Zettlemoyer (2013), all require a complete sentence. Since the robot is presented with the input one word at a time, batch mode requires waiting for the complete utterance before processing and providing output, which takes time to complete. For example, practical implementations of the G^3 system can take up to 30 seconds from the end of the input and the start of a response.

Our incremental parsing updates the distribution with each added word, substantially reducing the delay between input and response.

Incremental parsing also allows the robot to provide social feedback to the human. The robot could, for example, transition from a confused to a smiling face as more input words reduce the uncertainty of the referring expression.

Referring expression parsing is also closely related to the common robot task of pick-and-place, and could thus be a useful improvement for the Baxter robot and the H2R lab.

2.1 Examples

Given a *red apple*, *blue tape*, *red bowl*, and a *metal bowl* scattered on a table, we want to be able to convert referring expressions similar to these into probability distributions over the objects:

1. "The object between the apple and the tape."
2. "The red object."
3. "The red thing on the left of the bowl."
4. "The red thing on the left."
5. "The thing on the left of the bowl."

Parsing a referring expression and finding a distribution over possible objects has been accomplished by Tellex et al. (2011),Matuszek et al. (2012), and Artzi and Zettlemoyer (2013), among others.

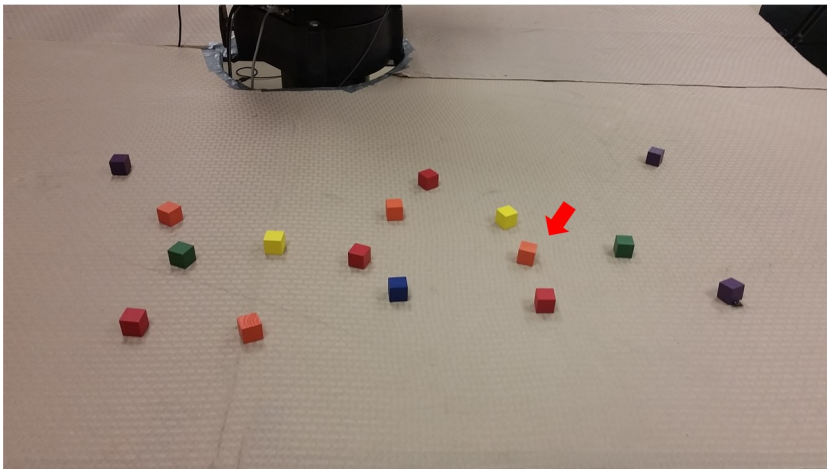


Figure 1: Arrangement of objects on a table.

We want to be able to parse these referring expressions incrementally, including each new word in our understanding of the referring expression as we receive it. Given the referring expression "The orange cube between the red and the yellow" to refer to the indicated object in Figure 1, our model should give better estimations of the correct object as we receive more words in the referring expression:

1. "The . . . "
2. "The orange cube . . . "
3. "The orange cube between the red and . . . "
4. "The orange cube between the red and the yellow."

Figure 2 shows how the distribution over all objects changes with more words of the referring expression.

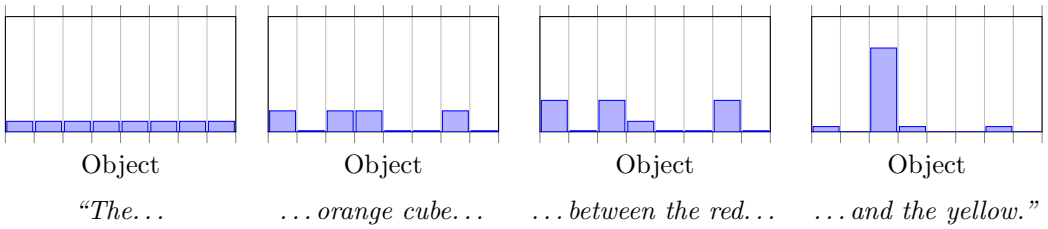


Figure 2: Probability distribution over all objects for incremental parts of "The orange cube between the red and the yellow."

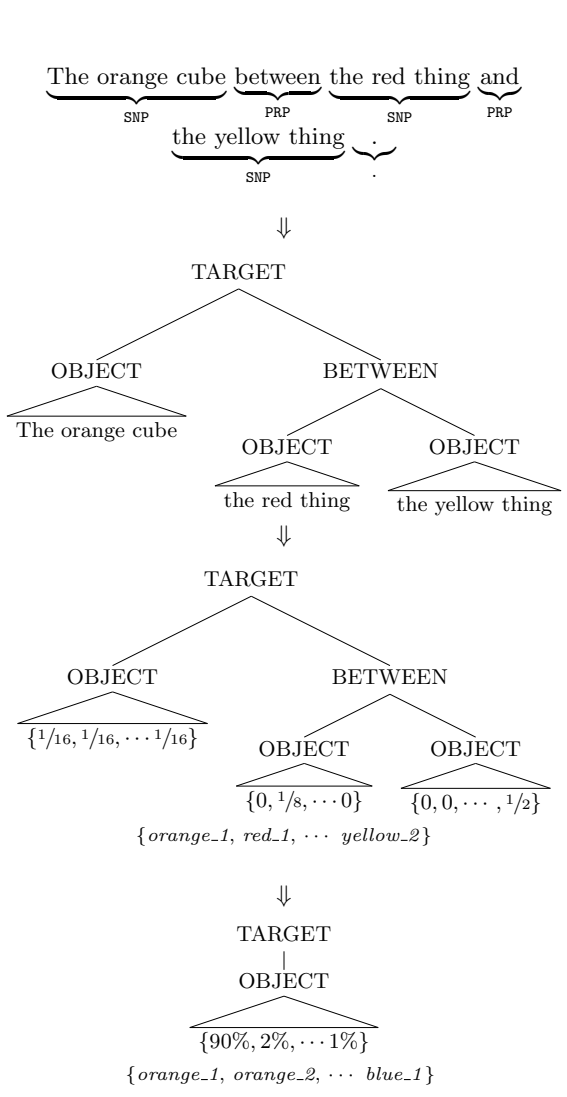
3 Technical Approach

Here we describe the general approach of our algorithm, how we parse referring expressions, and how we plan on making parsing both efficient and incremental.

We represent the referring expression as a tree, updating it each time we receive a new token. This representation allows us to perform computationally-intensive tasks once and cache intermediate results. This strategy allows us to produce intermediate results without having to recompute the entire expression so far.

More precisely, we construct a semantic tree such that each leaf node in the tree refers to some object. This tree is constructed by tagging the input using conditional random fields and then using deterministic transformations to turn that into a tree. We convert each of the leaf nodes into distributions over objects using a language model, and then finally evaluate this structure to obtain the final distribution. Here is a concrete example:

We attempt to locate the *orange cube* using the following referring expression: "The orange cube between the red thing and the yellow thing." A selection of *red*, *blue*, *orange*, *purple*, *green*, and *yellow* cubes are arranged on a table, as shown in Figure 1. The orange cube referred to in the above statement is pointed to by a red arrow in the figure. The process of estimating the object from this information is:



Tagging
First, we use Mallet, (McCallum 2002) a sequence tagging library, to tag the referring expression with a set of tags.
The grammar is a subset of English grammar, and is developed to suit our use case.

Semantic Tree
The tagged sequence is then converted to a semantic tree. Because of the simplified tag set in the previous step, this transformation is simple.

Language Model
We use a language model to convert each of the simple noun phrases into individual distributions over possible objects.

Bottom-up Evaluation
We evaluate the tree bottom-up, using spatial features and trained weights to convert each input distribution and the type of preposition into a set of output scores. The final result is the object with the highest score.

3.1 Tagging and Semantic Tree construction

We model the relationship between words and tags as a conditional random field, where the tag for any particular word depends on the neighboring words. We directly estimate the distribution of tags using the existing library Mallet, developed by McCallum (2002).

The transformation from the tagged sequence to the semantic tree is entirely deterministic, as the tags are tailored to the specific form of the queries in the corpus.

3.2 Language Model for Objects

The language model used is unigram language model, where each word w_i refers to object o_j with some joint score $Q(O = o_j, W = w_i)$ that we can estimate statistically. The distribution of simple noun phrase $S = (w_0, w_1, \cdots w_k)$ referring to object o_j is, therefore:

$$\begin{aligned}\Pr(O = o_i | S = (w_0, w_1, \cdots)) &= \frac{\Pr(O = o_i \cap S = (w_0, w_1, \cdots))}{\Pr(S = (w_0, w_1, \cdots))} \\ &\propto \Pr(O = o_i \cap S = (w_0, w_1, \cdots)) \\ &\approx \prod_{w_i \in (w_0, w_1, \cdots)} \left((1 - \alpha) * \frac{Q(O = o_j, W = W_i)}{\sum_o Q(O = o, W = w_i)} + \alpha * \frac{1}{k} \right)\end{aligned}$$

For smoothing, we assume that there is some small probability α that the object is uniformly at random chosen without regard to the word. We arbitrarily set $\alpha \approx 5\%$.

3.3 Bottom-up Evaluation

Once we have a semantic tree, we can recursively simplify the tree to obtain the final distribution. Figure 3 illustrates this process and the three possible cases we apply to convert the tree into a distribution over objects.

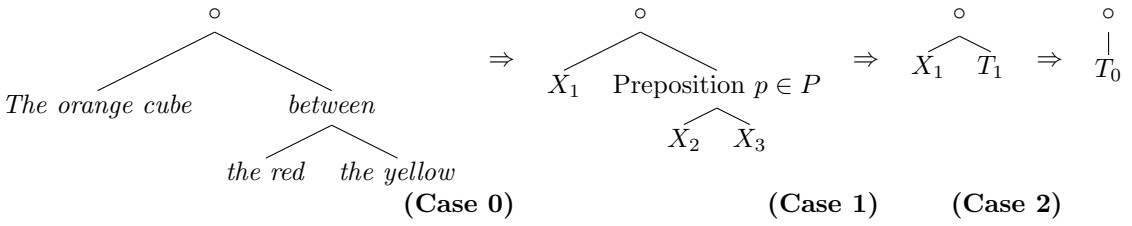


Figure 3: The three cases of bottom-up evaluation.

Case 0 Simplifying noun-phrases into a distribution.

Each grounding in the tree is modeled by a distribution that is obtained from the language model. We describe how this is done in Section 3.2. A simple implementation will be able to perform a lookup in $\mathcal{O}(|\boldsymbol{w}|)$ time, where $|\boldsymbol{w}|$ is the number of words in a single grounding.

Case 1 Simplifying a preposition and associated noun-phrases.

We have preposition $p \in P = \{\text{near, left, right, front, behind, between}\}$. Each grounding that the preposition relies on is modeled by the distributions $X_1, X_2, \cdots X_n$, obtained from the language model. T is the distribution of the object that the user is referring to.

In Figure 3, we illustrate how we simplify preposition p with groundings X_1 and X_2 into a single distribution T_1.

We find T with:

$$\Pr(T = o) = \Pr(T = o | X_1, X_2, \cdots X_n, P = p)$$

Assume that each grounding is independent and factor the expression:

$$\begin{aligned} &= \sum_{o_1, o_2, \cdots o_n \in O} \Pr(T = o \cap P = p \cap X_1 = o_1, X_2 = o_2, \cdots X_n = o_n) \\ &= \sum_{o_1, o_2, \cdots o_n \in O} \Pr(T = o \cap P = p | X_1 = o_1, X_2 = o_2, \cdots X_n = o_n) \cdot \Pr(X_1 = o_1, X_2 = o_2, \cdots X_n = o_n) \end{aligned}$$

Parametrize the probability with feature-vector function f, weights θ_p, logistic function S, and some normalization factor z:

$$= \sum_{o_1, o_2, \cdots o_n \in O} \mathrm{S}\left(\frac{f(o_1, o_2, \cdots, o_n) \cdot \theta_p}{z}\right) \cdot \prod_{i=1}^{n} \Pr(X_i = o_i)$$

We can estimate θ_p for each p using logistic regression, and select feature-vector function f separately.

We observe that the naïve implementation takes time to the order of $\mathcal{O}(|O|^{n+1})$, where $|O|$ is the number of objects and n is the number of groundings that each preposition has.

Case 2 Combining multiple distributions.

In Figure 3, we illustrate how we simplify groundings X_1 and independently derived distribution T_1 to get T_0, our estimated distribution.

In cases where we have multiple distributions, we estimate the true distribution by assuming each of $X_1, X_2, \cdots X_n$ are independent and taking the joint probability:

$$\begin{aligned} \Pr(T = o) &= \Pr(T = o | X_1, X_2, \cdots X_n) \\ &= \Pr(X_1 = o, X_2 = o, \cdots X_n = o) \\ &= \prod_{i=1}^{n} \Pr(X_i = o) \end{aligned}$$

(We also assume that the prior distribution of T is uniform over all objects.)

The naïve implementation also takes time to the order of $\mathcal{O}(|O|^{n+1})$, where $|O|$ is the number of objects and n is the number of groundings for which the marginal distribution must be taken.

These two cases are sufficient to recursively reduce the tree into a single distribution.

3.4 Incremental Parsing

In the previous section we factored the simplification of the semantic tree into two separate cases. We cache the result of each simplifying step, as illustrated in Figure 4.

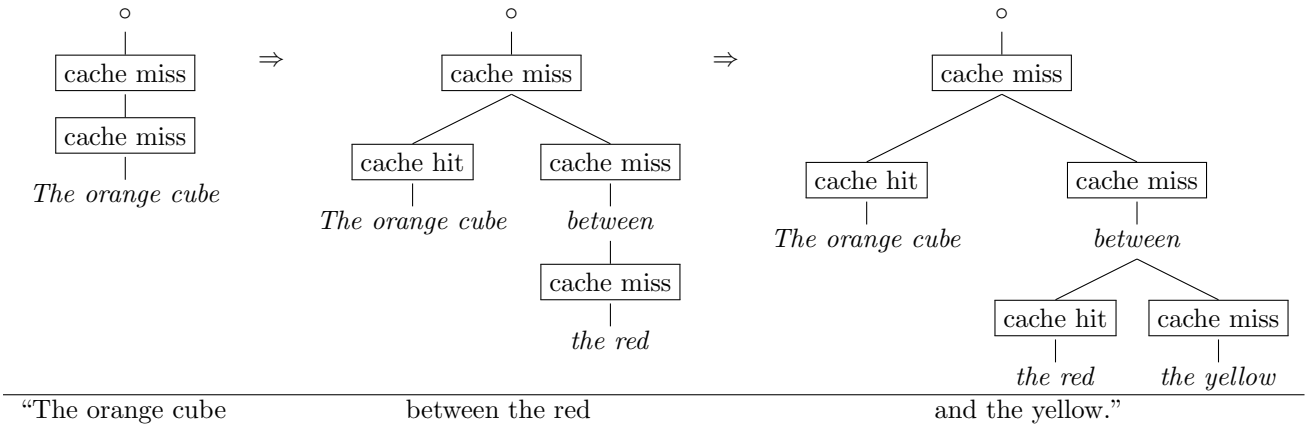


Figure 4: Cache behavior as words are added.

3.4.1 Runtime Analysis

We observe that this caching pattern guarantees that, in the worst case, the number of recursive simplifications we need to make with the addition of any one word is equal to one more than the maximum depth of the tree. In all our training data, the maximum tree depth observed is never more than two, so an upper bound of three simplifications per word input means that this algorithm can easily meet the runtime requirements of online algorithms.

More formally, given the runtimes and caching behavior discussed earlier, the worst-case time to update the distribution T_0 to include the next word from the user is $\mathcal{O}(|\boldsymbol{w}|)\prod_i^{k-1}\mathcal{O}(|O|^{n+1}) = \mathcal{O}(|\boldsymbol{w}| * |O|^{kn})$, where k is the number of layers in the tree.

When $k = 3$ and $n \leq 2$, as in real-world examples, this instead evaluates to the very manageable $\mathcal{O}(|\boldsymbol{w}|*|O|^4)$, where $|\boldsymbol{w}|$ is the number of words in the simple noun phrase, and $|O|$ is the number of objects in the scene.

3.4.2 Conditional Random Fields

The use of conditional random fields complicates the analysis somewhat because adding a word can affect the tags of words already processed, which causes the tree structure to change. In this case, recomputation is unavoidable, and we pay the penalty. Even if we were to use a simpler model with lower accuracy, such as a Hidden Markov Model for tagging with the Viterbi algorithm for decoding, it would likely also have this property.

We choose to use Conditional Random Fields in part because we can explicitly set the look-ahead limit, and so pick a tradeoff between tagging accuracy and potential worst-case performance. For our algorithm, we arbitrarily set it to 5.

The worst-case runtime scales linearly with the look-ahead limit. To characterize the typical performance penalty caused by our choice of look-ahead parameter, we gathered statistics on the likelihood of this happening in our training set.

We found that...

Todo Characterize the penalty associated with CRF-retagging leading to recomputation. Some basic probing shows that it isn't much after the first three or four words, but I would like to characterize this better.

4 Data Collection

Training each model that we use requires a lot of data. We collect many examples to train and test our system on as well as run our test set against humans to benchmark the performance of our system against.

We have collected a corpus of human-generated data using Human Intelligence Tasks (HITs) on the Amazon Mechanical Turk platform and hand-generated scenes. We additionally use AMT to have humans evaluate our test set to obtain a baseline.

4.1 Referring Expression Generation

A set of HITs were created to elicit referring expressions. A total of 19 scenes were constructed, each with a set of about 12 to 15 objects scattered on a table and at least 6 identical orange cubes. For each orange cube in each scene, 9 different workers were told to ask a robot across the table for the indicated orange cube. Figure 5 provides an example of such a labeled scene.

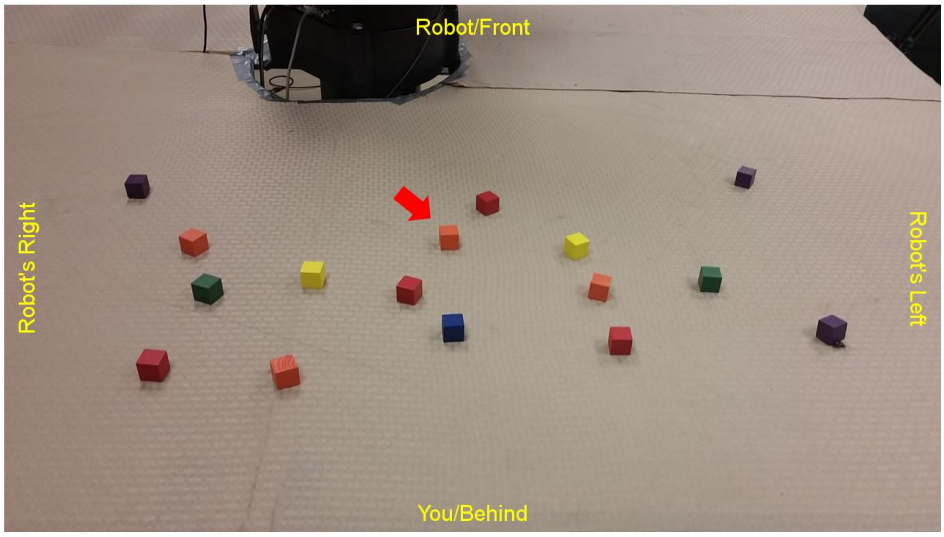


Figure 5: Example picture provided to HIT workers to elicit referring expressions.

We provide the following instructions to respondents:

> You are standing across the table from a robot. Write down what you would say to the robot if you wanted the indicated orange cube on the table.
>
> - Use phrases like "between", "near", "left of", "right of", "in front of" and "behind".
> - Use front/behind/left/right from the robot's perspective, as labeled in the image.
> - All orange cubes look the same.
> - You must ask for the item indicated by the red arrow
> - The instruction must be a single sentence.
> - Answer both questions.

4.2 Referring Expression Evaluation

For each referring expression in the test set, we get three separate raters to identify the target and provide feedback on the ease of understanding of the referring expression. Figure 6 shows an example scene.

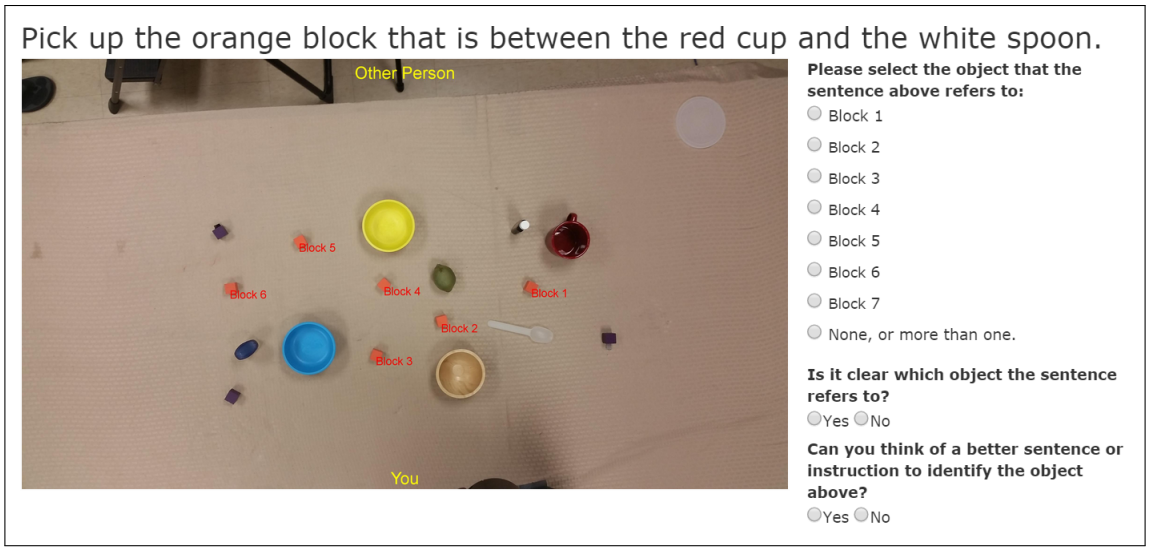


Figure 6: Example picture provided to HIT workers to evaluate referring expressions.

4.3 Data Gathered

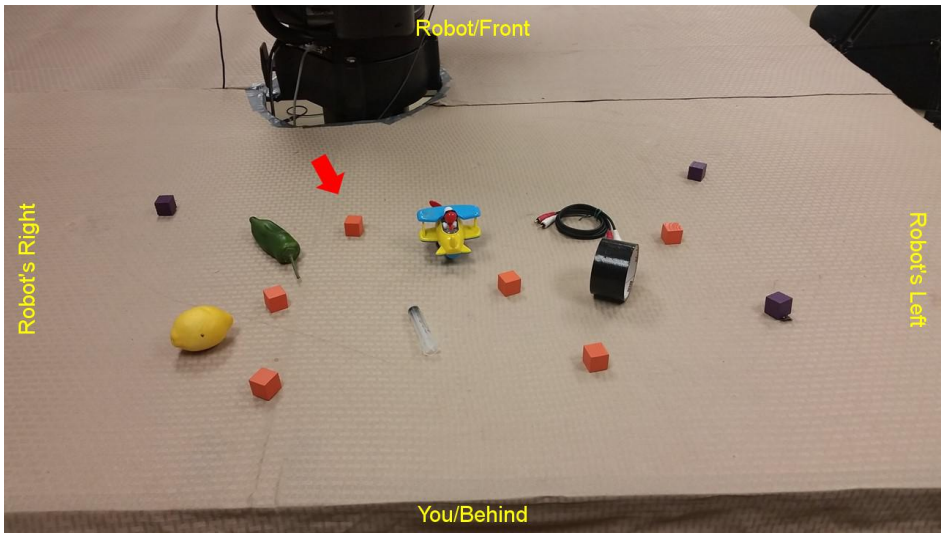


Figure 7: Example picture provided to HIT workers to elicit referring expressions.

We used Figure 7 to elicit referring expressions. Some referring expressions we received are:

- I want the orange cube in front of you, between the chili and the toy.
- hand me the orange cube that is in between jalapeño and airplane
- Directly in front of you next to the green pepper and airplane.
- Take the orange block between the toy airplane and green chili pepper
- bring the orange cube between the pepper and the plane.

5 Current Results

We trained the language model and spatial-prepositional model on a training set of 11 scenes with 417 input sentences total, each with an average of 14 objects. We trained the chunker on hand-annotated parses of these input sentences.

We then tested the incremental parser on a test set of 10 scenes with an average of 14 objects, with 381 input sentences. The results for the parser when run on the test set are as follows:

		Rate of correct identification, Test (%)				
		Baselines			Results	
Preposition		Human	Unigram	Random	Top-1	Top-3
26.8%	*between*	89.8	14.3	7.1	71.6	94.1
21.3%	*near*	76.2	17.2	7.3	7.4	50.6
14.2%	*behind*	68.9	15.7	6.9	25.9	75.9
11.0%	*in front of*	58.3	17.9	7.5	21.4	64.3
9.4%	*left of*	81.5	16.1	7.3	25.0	83.3
5.8%	*right of*	61.9	15.4	7.3	31.8	54.5
	Total	76.7	16.1	7.2	29.4	60.1

Figure 8: Performance on test set.

Figure 8 shows the correctness rate of our algorithm and of four baselines for comparison, when run on a reduced test set. The percentage next to each preposition is the fraction of sentences in the test set that contain this preposition, and so will not add up to 100%. The baselines are:

1. The *Human* baseline was established by having humans select the object best identified by the referring expression, and scoring them against our corpus.
2. The *Unigram* baseline is the expectation of selecting the correct object using a simple unigram object model across the entire input sentence.
3. The *Random* baseline is the expectation of selecting the correct object by selecting one object uniformly at random.

The *Results* column lists the rate of correct identification using the entire sentence as input, the *Top-1* column lists the rate at which the correct object is rated the most likely by the algorithm, and the *Top-3* column lists the rate at which the correct object is in the top 3 items.

For evaluation, the rate of correct identification is the number of trials in which the algorithm assigns a higher probability to the correct item than to any other item. Should there be a tie, the rate is divided by the number of items of equal probability.

The percentage on the left of each preposition is the fraction of the test set that the preposition makes up.

5.1 Incremental Performance

To evaluate incremental performance, we will be reporting the performance on the test set as a fraction of each sentence provided to the algorithm. Figure 9 reports the evaluation of the test set when our algorithm is run on it word-by-word. Note that, due to the variation in lengths of sentences, the charts are drawn by interpolating the correctness of each sentence into bins from 0 to 1.

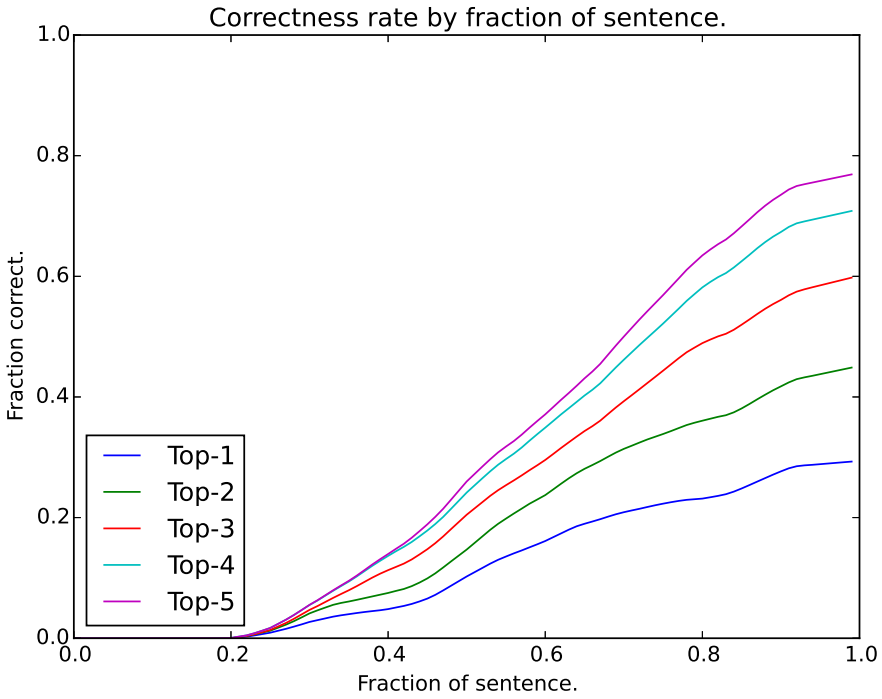


Figure 9: Performance on test set.

Five separate lines are drawn to show the distribution of the rank of the correct option. Each Top-k line includes an example if the target object has at least as much probability as the k^{th}-highest probability in the distribution. Should there be a tie, the rate is divided by the number of items of equal probability.

5.2 Future Work

The primary future task is to integrate this with the social feedback framework on Baxter in the H2R lab and conduct user studies to investigate if this provides a measurable change to user interaction.

6 Related Work

Prepositional phrases have not been subject to as much computational analysis and study as noun- and verb-phrases. However, there are still a number of papers related to the topic.

Tellex et al. (2011), Matuszek et al. (2012), and Artzi and Zettlemoyer (2013) all present modern models to process referring expressions.

Tellex et al. (2011) present the G^3 framework. We use several key ideas from this paper, in particular we implicitly assume the binary correspondence variable that they maximize the distribution of. Our algorithm is inspired in part by the algorithm they present.

Matuszek et al. (2012) presents a state-of-the-art process to learn a models for a semantic parser and word-classifier alignment. Our approach is substantially different from Matuszek et. al. since we do not separate perceptual features from the language model of each object. (i.e. we do not use any visual input.) Also, in the learning phase of their algorithm, Matuszek et. al. calculate the marginal probability of a particular grounding and a particular word by performing a beam search over all possible parses. We assume instead

that each node in the parse is independent of its sibling nodes, which allows us to use dynamic programming to incrementally build the distribution.

Artzi and Zettlemoyer (2013) trains Combinatory Categorial Grammars (CCG) with ambiguous validation functions to parse instructions, including spatial relations. While this approach is more flexible and likely performs better on entirely novel sentences, we (as we did with Matuszek et. al.) deliberately choose a simpler model that lends itself to a dynamic programming approach.

In addition to these, we rely on previous work about prepositional phrases:

Collins and Brooks (1995) present a model for disambiguating prepositional phrase attachments. They deal with Noun-Phrases and Verb-Phrases, but their statistical technique may be useful. This concept is also explored by Ratnaparkhi (1998), and Brill and Resnik (1994), each of whom suggest alternative models. Additionally, Merlo, Crocker, and Berthouzoz (1997) explore disambiguating multiple prepositional phrases, rather than a single phrase between multiple targets, both of which are relevant to future work. However, our current model uses sentences focused around a single noun phrase, unlike the general English corpus used in this paper. Additionally, the incremental nature of our parsing requires an alternative framework, and as such not all techniques suggested in these papers can be used.

Another issue we face is the challenge of identifying objects and mapping them to probability distributions via language model. To deal with this, Barbu, Narayanaswamy, and Siskind (2013) present a model for mapping language to object models in video data and Matuszek et al. (2012) describe an approach to the problem of simultaneously observing and extracting representations of a perceived world. These approaches can be adapted to help design features, choose objects, and select language models for prepositional phrase training, rather than using pre-defined objects and locations as we currently do, similar to the work of Tellex et al. (2011) which presents a method to dynamically generate a probabilistic model of a natural language input and perform inferences relating to semantic meaning. This is similar to the work we will preform in semantic parsing, though we will do so incrementally rather than with an entire sentence and thus our work will need to modify these models.

There are still a number of improvements to be made to our machine learning techniques. Rudzicz and Mokhov (2003) give us a framework for parsing and understanding prepositional phrases. Similarly, Liang, Jordan, and Klein (2013) describe a method to create a semantic parser using question-answer pairs as data, rather than requiring annotated sentences, solving the same issue we approach of transforming natural language into semantic meaning. However, while these may be useful for generating a semantic model over entire sentences our technique will be different as we are currently only working with noun phrases, and apply additional information from the spatial model.

7 Conclusion

This proposal is for an incremental referring expression parser that can process prepositional phrases. The incremental nature of the parser is the key contribution: the state-of-the-art parsers all operate on complete referring expressions. The incremental nature of this parser allows for social feedback and lower-latency interpretations of referring expressions.

References

Artzi, Yoav, and Luke Zettlemoyer. 2013. "Weakly supervised learning of semantic parsers for mapping instructions to actions." *Transactions of the Association for Computational Linguistics* 1:49–62.

Barbu, Andrei, Siddharth Narayanaswamy, and Jeffrey Mark Siskind. 2013. "Saying What You're Looking For: Linguistics Meets Video Search." *CoRR* abs/1309.5174. `http://arxiv.org/abs/1309.5174`.

Brill, E., and P. Resnik. 1994. "A Rule-Based Approach To Prepositional Phrase Attachment Disambiguation." In *eprint arXiv:cmp-lg/9410026,* 10026. October.

Collins, Michael, and James Brooks. 1995. "Prepositional Phrase Attachment through a Backed-Off Model." *CoRR* abs/cmp-lg/9506021. `http://arxiv.org/abs/cmp-lg/9506021`.

Liang, Percy, Michael I Jordan, and Dan Klein. 2013. "Learning dependency-based compositional semantics." *Computational Linguistics* 39 (2): 389–446.

Matuszek, Cynthia, Nicholas FitzGerald, Luke Zettlemoyer, Liefeng Bo, and Dieter Fox. 2012. "A Joint Model of Language and Perception for Grounded Attribute Learning." In *Proc. of the 2012 International Conference on Machine Learning.* Edinburgh, Scotland, June.

McCallum, Andrew Kachites. 2002. "MALLET: A Machine Learning for Language Toolkit." Http://mallet.cs.umass.edu.

Merlo, Paola, Matthew W. Crocker, and Cathy Berthouzoz. 1997. "Attaching Multiple Prepositional Phrases: Generalized Backed-off Estimation." *CoRR* cmp-lg/9710005. `http://arxiv.org/abs/cmp-lg/9710005`.

Ratnaparkhi, Adwait. 1998. "Statistical Models for Unsupervised Prepositional Phrase Attachment." *CoRR* cmp-lg/9807011. `http://arxiv.org/abs/cmp-lg/9807011`.

Rudzicz, Frank, and Serguei A. Mokhov. 2003. "Towards a Heuristic Categorization of Prepositional Phrases in English with WordNet." *CoRR* abs/1002.1095. `http://arxiv.org/abs/1002.1095`.

Tellex, Stefanie, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth J Teller, and Nicholas Roy. 2011. "Understanding Natural Language Commands for Robotic Navigation and Mobile Manipulation." In *AAAI.*