Project Title:
Fetch POMDP 2.0

Project Partners:
James Hwang and Ari Beller

Abstract:

In their 2017 FETCH POMDP paper, Whitney, Rosen, et al present a multimodal interactive robotic system for approaching reference tasks, a critical problem in human robot interaction. Their model provides a strong groundwork for resolving references, integrating linguistic information, visual cues, and social feedback with a partially observable markov decision process. However, the original system utilizes a substantially simplified underlying language model. To more reliably and naturally accommodate the referential power of human language, we sought to improve upon that underlying model. We began by replicating the linguistic portions of the POMDP's functioning (language model and question asking). To push the capability of the system further, we augmented the language model with understanding of given prepositions so that the system could leverage locational information when determining which item the speaker wants. In the context of the FETCH POMDP, this is a powerful asset since all of the items are laid out in a straightforward horizontal setup. With this additional knowledge, the system was able to more quickly determine the user's desired object, resolving references that would have required further clarification in the original language model.

Introduction:

Reference tasks are a critical and difficult problem in human-robot linguistic interaction. They are ubiquitous throughout human language use, and they occur in diverse contexts at various levels of abstraction. We refer to single out everything from concrete objects in our environment like chairs to abstract conceptual notions like freedom. Though humans are generally able to resolve under-specified, context-sensitive references without difficulty, state of the art machine approaches have not yet hit that benchmark. In order for machines to interact naturally with humans using natural language, improvements to reference resolution are still necessary.

To approach this problem we decided to build on the FETCH-POMDP a multi-modal system developed in this lab by Whitney, Rosen, et al. The FETCH-POMDP integrates linguistic knowledge, visual cues, and social feedback in order to resolve references in a simple reference task. Incorporating this variety of input allows the original FETCH-POMDP to perform robustly in the given task, however its individual sub-models are somewhat simplistic. In particular we were interested in focusing on the language model, which essentially utilizes word frequency relative to a given item vocabulary in order to guess which item is most likely the desired object of the speaker. The language that participants use to refer to the items in the task, especially in ambiguous cases, is often more complex that what is accounted for by this simple model. In particular, given the spatial arrangement of items for the task, participants often make use of positional, relational language, identifying the desired object by means of its location relative to the other objects. We wanted to improve the FETCH-POMDP in order to accommodate these sorts of linguistic phenomena

The first part of our project was to replicate the linguistic components of the FETCH-POMDP. The Partially Observable Markov Decision Process is a framework for modeling decision making where an agent has a reward function that is dependent on the state of the world. The agent however does not have perfect knowledge of the state of the world, instead she must infer probabilities of different states based on what she observes. The agent chooses actions to optimize expected reward under this uncertainty. In the context of our limited replication of the linguistic components of the FETCH POMDP, the state of the world is the agent's desired object. Our system is trying to discern what that object is based on its observations, the user's input language. The system also incorporates social feedback when it is not confident that it knows the participant's desired object. The POMDP balances the rewards and costs of guessing a particular object vs asking a clarifying question to discern whether a particular object is the participants desired object.

In the second part of our project, we built on this model to incorporate more sophisticated language understanding. As described above, our interest was in incorporating positional and relational understanding so that the system could utilize prepositional relations when trying to determine the user's desired object. In particular, we implemented prepositional heuristics as a Bayesian filter on our naive implementation (the basic language model of the FETCH-POMDP). These heuristics allowed the system to interpret the meaning of terms like "right of" or "near" so that it could identify objects in otherwise ambiguous cases without having to resort to question asking clarifications.

We evaluated our updated model by comparing its ability to resolve references with and without prepositions to the original model replication. We found that it performed as well as the original model on cases without prepositions and that it could additionally resolve references with prepositions that we specified without having to ask for further clarification. To more completely evaluate the capabilities of the model, we would have liked to perform a user study with a variety of participants and object setups. Unfortunately within the scope of the class we did not have time.

Related Work:

Our work builds on the system developed by Whitney and Rosen in their 2017 paper. To integrate the various sources of information that the system utilizes and determine the optimal action given an observation, Whitney and Rosen construct a partially observable markov decision process (POMDP). POMDPs provide a framework for modeling how agents should act under uncertainty given observations of the state of the world. The POMDP is defined in terms of a state space, action space, transition function, reward function, and observation function. The transition function defines how the model transitions from one state to another given a particular action. The reward function defines a utility for each state action pair. Lastly the observation function defines the probability of a particular observation given the underlying state and an action. The task of the agent is learn a policy, a function from observations to actions, that maximizes the expected reward. The details of the particular POMDP which Whitney and

Rosen defined and which we replicated for our project are included in “Technical Approach” section.

Technical Approach:

The transition function for the FETCH-POMDP is deterministic. The model assumes that the participant’s desired object does not change within a given trial. The state of the trial also reflects any objects that the system asks about in order to clarify whether a particular item is the user’s desired item. This begins with a null value and is updated deterministically based on any clarifying questions the system asks. We found the same reward function that Whitney and Rosen used in the original paper worked well in our replication even with the changes that we made to the structure of the POMDP. The reward function gives correct choices strong positive weight, incorrect choices strong negative weight, and clarifying questions middle negative weights to disincentivize unnecessary question asking.

The main sophistication of the model lies in the observation function. The model splits a linguistic input into two components, the base utterance and response utterance. The base utterance encompasses all words that are not part of the response to a clarifying question. The probability of the utterance is equal to the product of the probabilities of the words times a base utterance probability. There is a small probability assigned to the case where the base utterance is empty. The probability of words in the base utterance is determined by a unigram model. The unigram model is defined by the following equation:

$$P(w|i_d) = \frac{|i_d.vocab[w]| + \alpha}{|i_d.vocab| + |vocab|\alpha}$$

The probability of a particular word w given the true object the participant desires is object i_d, is equal to the number of times w appears in the specified vocabulary for that object plus a smoothing parameter alpha, divided by the full size of the vocabulary for that particular object plus the full vocab size (for all objects) multiplied by the smoothing parameter. In plain English the probability of a word is equal to the frequency that the word appears in the vocabulary of the given object plus a smoothing parameter. The smoothing parameter ensures that no word has zero probability. This is important because when we take the product of all the word probabilities to find the probability of the full utterance, if any of the probabilities were zero, the full utterance would have zero probability. All vocabularies for objects are pre-specified.

The response utterance is a set of either affirmative or negative words in the respective aff/neg vocabularies. The probability of each affirmative word is 0.99 if the chosen object matches the desired object, and 0.01 otherwise and vice versa. Given that there is no chosen object, the probability of any response word is 0.5, thus giving a uniform probability in the neutral state to avoid biasing the algorithm. The total response utterance probability is the product of the individual utterance probabilities.

To determine actions, we used a model independent belief sparse sampler. It simulates all possible actions multiple times given the last state, observation, and action, and for each of the result states using the transition function, we apply this sampling again, creating a simulated tree. For our purposes, we have a tree of depth 2. The complexity increases exponentially, so more than 2 is not feasible, and 2 is sufficient for FETCH. In those simulated states, an observation is sampled given the state and action. The model must provide the sample_obs function. For each of the end states, the reward is calculated based on the reward function provided by the model, and one step above in the tree, the maximum reward is propagated up, and the sampler chooses the action whose max expected reward is the highest.

After choosing an action, the robot gets a new observation from the user, and using the observation probability function in the model described above, a new belief array is calculated, where the belief represents the probability of each object being the desired object.

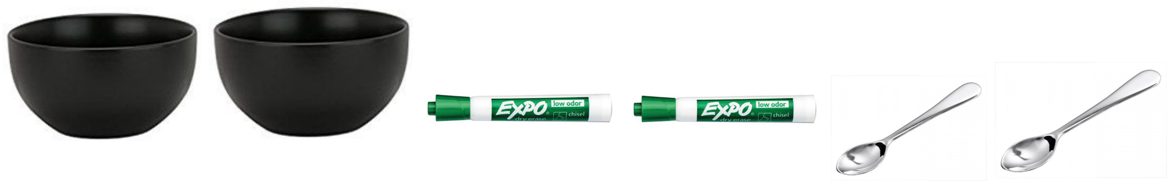
After calculating a belief update, we have a new belief array, and we apply a Bayesian filter to implement the heuristics for positional language. We separated the heuristics into two cases, two argument and one argument prepositional phrases. These arguments refer to the objects explicitly referenced. X “right of” Y would be a two argument phrase whereas X “on the right” would be a one argument phrase (note the one argument case does imply at least one other object of comparison, however that argument is implicit). For the two argument case, we build a matrix where the row represents the index of the object on the left side of the expression, and the column represents the index of the object on the right side of the expression. For each cell, the joint probability of those objects is computed by multiplying the unigram probabilities of each object, given the sentence is split into the left and right sides of the preposition. Because the two argument phrases anchor themselves on the object on the right side of the expression and find the other object relative to that, we loop through each column and apply a mask over the entire column. The mask would depend on the preposition. “Right of” would have all 0s up to and including the current column index, and then have a geometrically decreasing probability (we used 0.8 for the base of the exponent) as the object is further away from the current index. This encapsulates that we are not looking for objects on the left side, and that what we want is likely to be closer to the referenced object than not. “Left of” would have geometrically increasing probabilities from the lower indices (lower indices are toward the left on the table and higher indices on the right), up to and not including the current column index, and 0 afterwards. “Near” would have 0 probability for the current index and geometrically decreasing probability in both directions. This is extensible to a 2D or 3D arrangement of objects, as the number of matrix dimensions can simply be increased. The run time would still be around n^2 in the number of objects as we need to go through every object pair once.

For the single argument case, we gather up the objects with the highest probability (based on the belief array calculated in the update step) until the sum of probabilities passes a threshold (we used 0.6). This selected group forms a “context” of likely candidates for the heuristic to pick from. The objects in this context are arranged geometrically from left to right. Then, given the positional response word “left” or “right”, we increase the probabilities of the objects in the context based on their position. If the word was “left”, the further left an object was among the gathered ones, the more its probability would increase. If the word was “right” the opposite would be true. To increase the probability value of the relevant objects we add to the probability of all those objects that we have gathered. The amount that we add decreases exponentially as we move farther away from either the right or left (depending on the keyword). After we have added this buff to these most probable terms, we renormalize their values with the rest of the probabilities to return a valid probability distribution. We return this distribution as our belief.

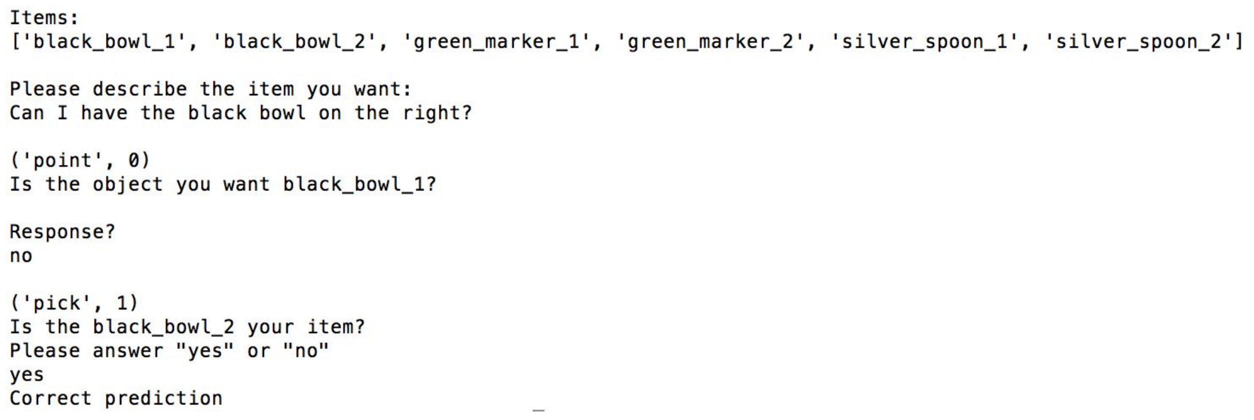
Evaluation:

To evaluate our new model, we compared the performance of our updated system with the performance of the model that we replicated for the midterm checkpoint. We found that indeed our updated model was able to utilize prepositional information to resolve references faster than the original model. We provide an example below showcasing our model's ability to resolve reference without additional clarification.

For the following setup (one of Whitney and Rosen's "ambiguous settings"):

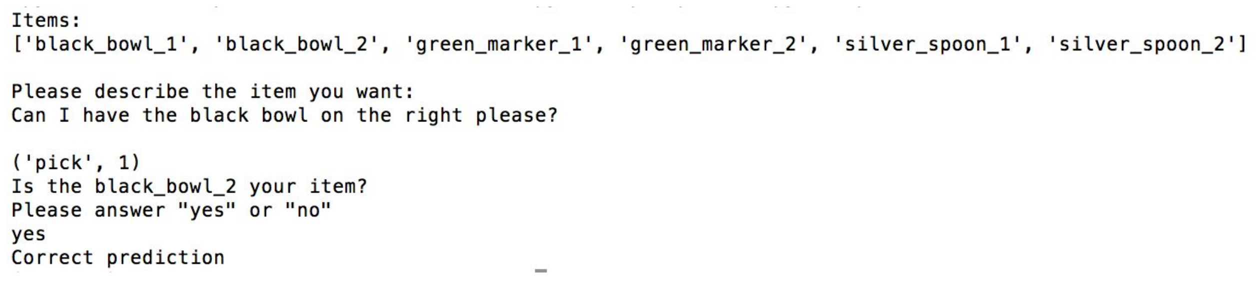


Old model:



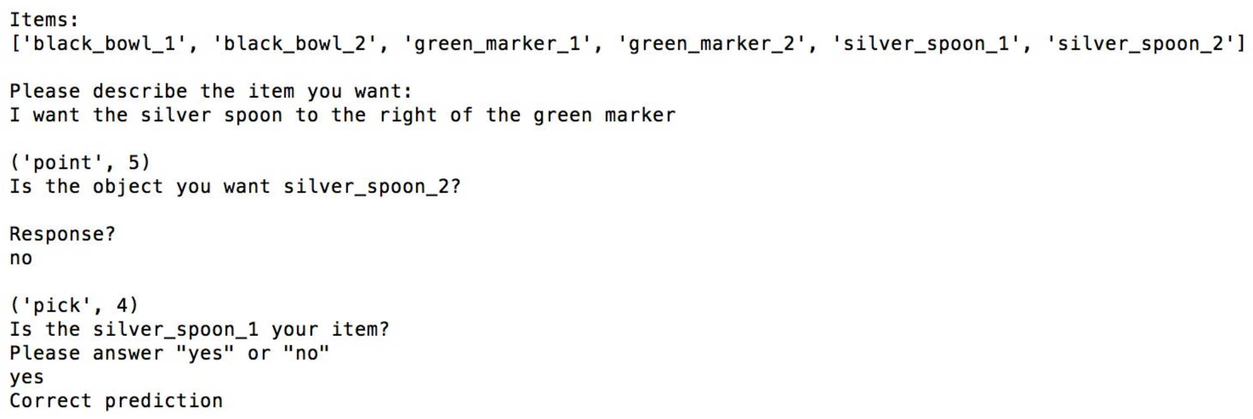
```
Items:
['black_bowl_1', 'black_bowl_2', 'green_marker_1', 'green_marker_2', 'silver_spoon_1', 'silver_spoon_2']

Please describe the item you want:
Can I have the black bowl on the right?

('point', 0)
Is the object you want black_bowl_1?

Response?
no

('pick', 1)
Is the black_bowl_2 your item?
Please answer "yes" or "no"
yes
Correct prediction
```

New model:



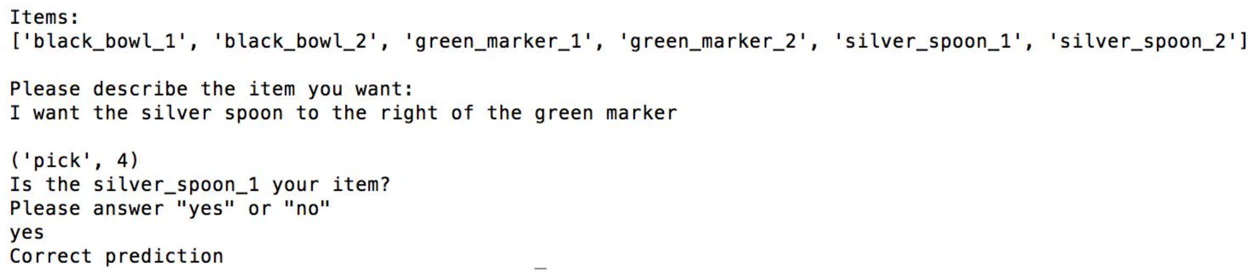
```
Items:
['black_bowl_1', 'black_bowl_2', 'green_marker_1', 'green_marker_2', 'silver_spoon_1', 'silver_spoon_2']

Please describe the item you want:
Can I have the black bowl on the right please?

('pick', 1)
Is the black_bowl_2 your item?
Please answer "yes" or "no"
yes
Correct prediction
```

Old model:



```
Items:
['black_bowl_1', 'black_bowl_2', 'green_marker_1', 'green_marker_2', 'silver_spoon_1', 'silver_spoon_2']

Please describe the item you want:
I want the silver spoon to the right of the green marker

('point', 5)
Is the object you want silver_spoon_2?

Response?
no

('pick', 4)
Is the silver_spoon_1 your item?
Please answer "yes" or "no"
yes
Correct prediction
```

New model:

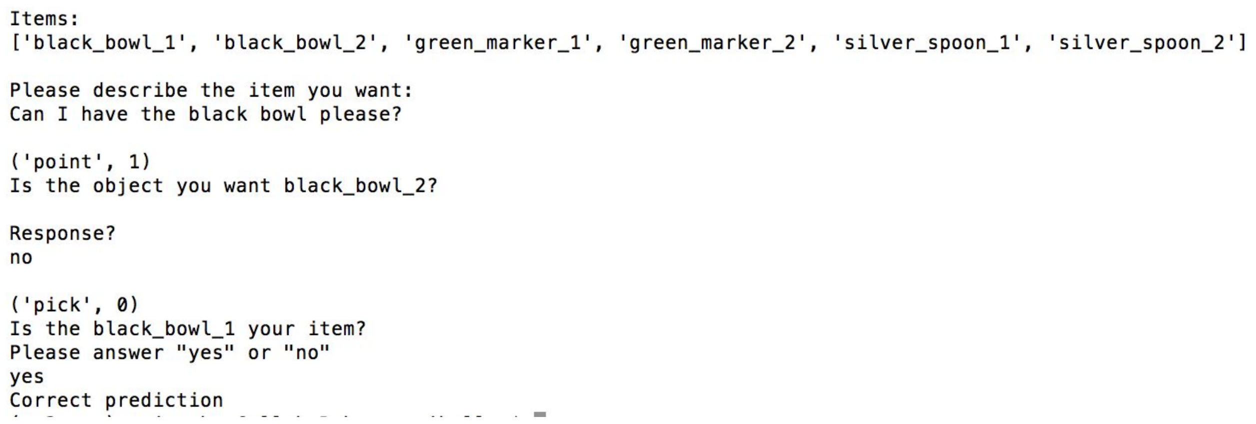


```
Items:
['black_bowl_1', 'black_bowl_2', 'green_marker_1', 'green_marker_2', 'silver_spoon_1', 'silver_spoon_2']

Please describe the item you want:
I want the silver spoon to the right of the green marker

('pick', 4)
Is the silver_spoon_1 your item?
Please answer "yes" or "no"
yes
Correct prediction
```

Each question preceded by a “point” tuple signifies a clarifying question where each question preceded by a “pick” tuple signifies the model’s choice for solving the reference task. Though both models are able to resolve the references, the new model can do so without asking for clarification. The old model needs to clarify with at least one question (in other cases it would ask for more).

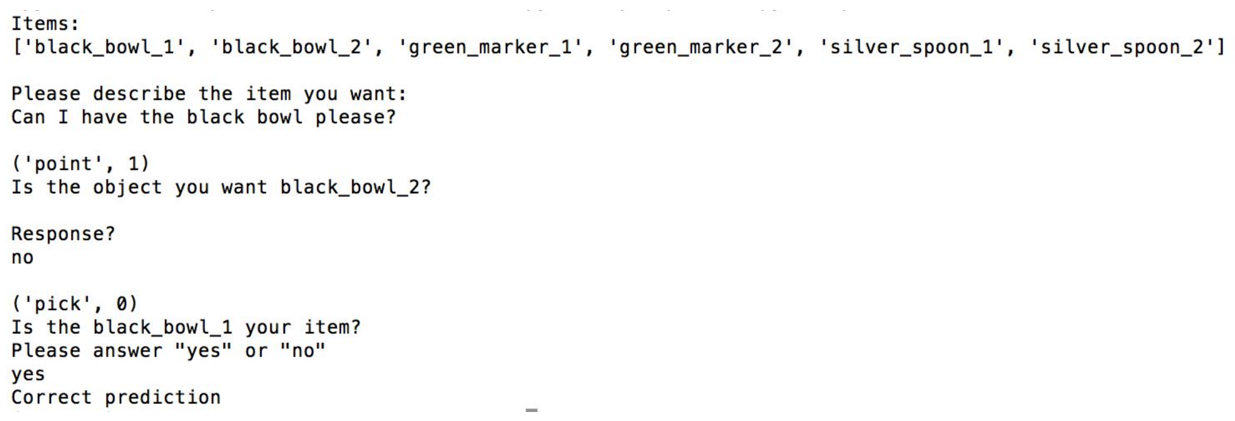
Additionally, the old model preserves all the capabilities of the new model as a baseline. We can see for the standard reference task without prepositional clarifiers they have the same behavior:

Old Model:



```
Items:
['black_bowl_1', 'black_bowl_2', 'green_marker_1', 'green_marker_2', 'silver_spoon_1', 'silver_spoon_2']

Please describe the item you want:
Can I have the black bowl please?

('point', 1)
Is the object you want black_bowl_2?

Response?
no

('pick', 0)
Is the black_bowl_1 your item?
Please answer "yes" or "no"
yes
Correct prediction
```

New Model:



```
Items:
['black_bowl_1', 'black_bowl_2', 'green_marker_1', 'green_marker_2', 'silver_spoon_1', 'silver_spoon_2']

Please describe the item you want:
Can I have the black bowl please?

('point', 1)
Is the object you want black_bowl_2?

Response?
no

('pick', 0)
Is the black_bowl_1 your item?
Please answer "yes" or "no"
yes
Correct prediction
```

The new features add inferential power without taking away from the system's capacities in any way that we are aware of.

Conclusion:

The FETCH-POMDP explores a critical question in computer human communication, how can we use a variety of tools, linguistic, social, and visual to help us resolve references. In our project we sought to build on the linguistic components of that framework. After replicating the linguistic portions of the original model, we developed a heuristic approach to improve the model's inferential abilities, allowing it to interpret positional information provided in natural language with prepositions. We found that our heuristics improved the model's capabilities without detracting from them. However further evaluation would be valuable. Because natural language is so flexible there are always unforseen descriptions that act as edge cases. Testing the model with a multi-person user study would be the next ideal step for assessing its capabilities.

In addition there are many possible avenues for further developing the capacities of the system. The FETCH-POMDP reference task, though interesting is somewhat restricted. The objects are presented to the user in an organized line. There is a lot we could do just in terms of experimenting with the setup of the problem. Adding to the number of items or changing their orientation could change the way that people choose to describe those items, or change the way the statistical model accommodates the data. If objects are arranged vertically rather than horizontally, people likely won't use left and right to describe locations. Adding objects and words to our vocabulary will change the normalizer for our unigram model in different circumstances. Each of these changes could alter expected performance, and it would be interesting to experiment with these different settings and see how we might be able to design general principles that accommodate many different types of environments.

One interesting possibility in that vein is thinking about how we manage hyper-parameter tuning. The success of our model is notably dependent on the choice of particular hyper-parameters (the rewards for different POMDP scenarios, the unigram smoothing parameter, the Bayesian filter on particular values) that may need to take different values to succeed in different contexts. One interesting avenue for exploration along this line would be to see if there was some automated way to tune hyper-parameters to different task setups so that the model could perform flexibly without requiring manual re-tuning for each context.

One of our original interests in pursuing this project was to find ways to integrate this framework with probabilistic programming. Though unfortunately we did not have time to fully explore that space, we did find fruitful ground for a potential wedding of probabilistic programming languages and statistical methods for solving POMDPs. Another potentially interesting avenue going forward would be to further explore the ways that PPLs could help us better address the difficult problem of solving POMDPs. PPLs offer an expressive and powerful language for framing statistical inference with generative models. To solve the FETCH POMDP, we used a fairly simple statistical method, but PPLs could give us more sophisticated tools to address these computationally intensive problems.