Learning to Follow Navigational Directions

Adam Vogel and **Dan Jurafsky**
Department of Computer Science
Stanford University
{acvogel,jurafsky}@stanford.edu

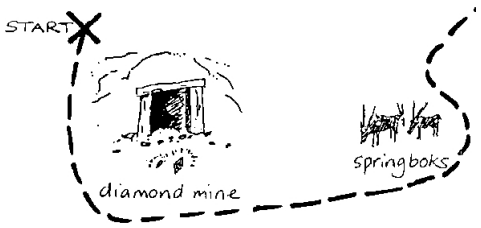
Abstract

We present a system that learns to follow navigational natural language directions. Where traditional models learn from linguistic annotation or word distributions, our approach is grounded in the world, learning by apprenticeship from routes through a map paired with English descriptions. Lacking an explicit alignment between the text and the reference path makes it difficult to determine what portions of the language describe which aspects of the route. We learn this correspondence with a reinforcement learning algorithm, using the deviation of the route we follow from the intended path as a reward signal. We demonstrate that our system successfully grounds the meaning of spatial terms like *above* and *south* into geometric properties of paths.

1 Introduction

Spatial language usage is a vital component for physically grounded language understanding systems. Spoken language interfaces to robotic assistants (Wei et al., 2009) and Geographic Information Systems (Wang et al., 2004) must cope with the inherent ambiguity in spatial descriptions.

The semantics of imperative and spatial language is heavily dependent on the physical setting it is situated in, motivating automated learning approaches to acquiring meaning. Traditional accounts of learning typically rely on linguistic annotation (Zettlemoyer and Collins, 2009) or word distributions (Curran, 2003). In contrast, we present an apprenticeship learning system which learns to imitate human instruction following, without linguistic annotation. Solved using a reinforcement learning algorithm, our system acquires the meaning of spatial words through grounded interaction with the world. This draws on the intuition that children learn to use spatial language through a mixture of observing adult language usage and situated interaction in the world, usually without explicit definitions (Tanz, 1980).



1. go vertically down until you're underneath eh diamond mine
2. then eh go right until you're
3. you're between springbok and highest viewpoint

Figure 1: A path appears on the instruction giver's map, who describes it to the instruction follower.

Our system learns to follow navigational directions in a route following task. We evaluate our approach on the HCRC Map Task corpus (Anderson et al., 1991), a collection of spoken dialogs describing paths to take through a map. In this setting, two participants, the *instruction giver* and *instruction follower*, each have a map composed of named landmarks. Furthermore, the *instruction giver* has a route drawn on her map, and it is her task to describe the path to the *instruction follower*, who cannot see the reference path. Our system learns to interpret these navigational directions, without access to explicit linguistic annotation.

We frame direction following as an apprenticeship learning problem and solve it with a reinforcement learning algorithm, extending previous work on interpreting instructions by Branavan et al. (2009). Our task is to learn a policy, or mapping

Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 806–814,
Uppsala, Sweden, 11-16 July 2010.

from world state to action, which most closely follows the reference route. Our state space combines world and linguistic features, representing both our current position on the map and the communicative content of the utterances we are interpreting. During training we have access to the reference path, which allows us to measure the utility, or reward, for each step of interpretation. Using this reward signal as a form of supervision, we learn a policy to maximize the expected reward on unseen examples.

2 Related Work

Levit and Roy (2007) developed a spatial semantics for the Map Task corpus. They represent instructions as Navigational Information Units, which decompose the meaning of an instruction into orthogonal constituents such as the reference object, the type of movement, and quantitative aspect. For example, they represent the meaning of "move two inches toward the house" as a reference object (the house), a path descriptor (towards), and a quantitative aspect (two inches). These representations are then combined to form a path through the map. However, they do not learn these representations from text, leaving natural language processing as an open problem. The semantics in our paper is simpler, eschewing quantitative aspects and path descriptors, and instead focusing on reference objects and frames of reference. This simplifies the learning task, without sacrificing the core of their representation.

Learning to follow instructions by interacting with the world was recently introduced by Branavan et al. (2009), who developed a system which learns to follow Windows Help guides. Our reinforcement learning formulation follows closely from their work. Their approach can incorporate expert supervision into the reward function in a similar manner to this paper, but is also able to learn effectively from environment feedback alone. The Map Task corpus is free form conversational English, whereas the Windows instructions are written by a professional. In the Map Task corpus we only observe expert route following behavior, but are not told how portions of the text correspond to parts of the path, leading to a difficult learning problem.

The semantics of spatial language has been studied for some time in the linguistics literature. Talmy (1983) classifies the way spatial meaning is

Figure 2: The instruction giver and instruction follower face each other, and cannot see each others maps.

encoded syntactically, and Fillmore (1997) studies spatial terms as a subset of deictic language, which depends heavily on non-linguistic context. Levinson (2003) conducted a cross-linguistic semantic typology of spatial systems. Levinson categorizes the frames of reference, or spatial coordinate systems[1], into

1. *Egocentric*: Speaker/hearer centered frame of reference. Ex: "the ball to your left".

2. *Allocentric*: Speaker independent. Ex: "the road to the north of the house"

Levinson further classifies allocentric frames of reference into absolute, which includes the cardinal directions, and intrinsic, which refers to a featured side of an object, such as "the front of the car". Our spatial feature representation follows this egocentric/allocentric distinction. The intrinsic frame of reference occurs rarely in the Map Task corpus and is ignored, as speakers tend not to mention features of the landmarks beyond their names.

Regier (1996) studied the learning of spatial language from static 2-D diagrams, learning to distinguish between terms with a connectionist model. He focused on the meaning of individual terms, pairing a diagram with a given word. In contrast, we learn from whole texts paired with a

[1]Not all languages exhibit all frames of reference. Terms for 'up' and 'down' are exhibited in most all languages, while 'left' and 'right' are absent in some. Gravity breaks the symmetry between 'up' and 'down' but no such physical distinction exists for 'left' and 'right', which contributes to the difficulty children have learning them.

path, which requires learning the correspondence between text and world. We use similar geometric features as Regier, capturing the allocentric frame of reference.

Spatial semantics have also been explored in physically grounded systems. Kuipers (2000) developed the Spatial Semantic Hierarchy, a knowledge representation formalism for representing different levels of granularity in spatial knowledge. It combines sensory, metrical, and topological information in a single framework. Kuipers et al. demonstrate its effectiveness on a physical robot, but did not address the learning problem.

More generally, apprenticeship learning is well studied in the reinforcement learning literature, where the goal is to mimic the behavior of an expert in some decision making domain. Notable examples include (Abbeel and Ng, 2004), who train a helicopter controller from pilot demonstration.

3 The Map Task Corpus

The HCRC Map Task Corpus (Anderson et al., 1991) is a set of dialogs between an *instruction giver* and an *instruction follower*. Each participant has a map with small named landmarks. Additionally, the instruction giver has a path drawn on her map, and must communicate this path to the instruction follower in natural language. Figure 1 shows a portion of the instruction giver's map and a sample of the instruction giver language which describes part of the path.

The Map Task Corpus consists of 128 dialogs, together with 16 different maps. The speech has been transcribed and segmented into utterances, based on the length of pauses. We restrict our attention to just the utterances of the instruction giver, ignoring the instruction follower. This is to reduce redundancy and noise in the data - the instruction follower rarely introduces new information, instead asking for clarification or giving confirmation. The landmarks on the instruction follower map sometimes differ in location from the instruction giver's. We ignore this caveat, giving the system access to the instruction giver's landmarks, without the reference path.

Our task is to build an automated *instruction follower*. Whereas the original participants could speak freely, our system does not have the ability to query the instruction giver and must instead rely only on the previously recorded dialogs.

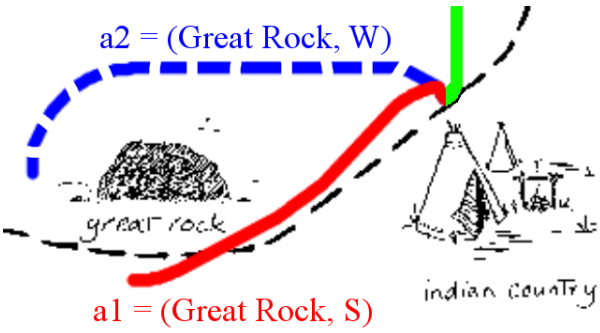


Figure 3: Sample state transition. Both actions get credit for visiting the great rock after the indian country. Action a1 also gets credit for passing the great rock on the correct side.

4 Reinforcement Learning Formulation

We frame the direction following task as a sequential decision making problem. We interpret utterances in order, where our interpretation is expressed by moving on the map. Our goal is to construct a series of moves in the map which most closely matches the expert path.

We define intermediate steps in our interpretation as *states* in a set S, and interpretive steps as *actions* drawn from a set A. To measure the fidelity of our path with respect to the expert, we define a *reward* function $R : S \times A \rightarrow \mathbb{R}^+$ which measures the utility of choosing a particular action in a particular state. Executing action a in state s carries us to a new state s', and we denote this transition function by $s' = T(s, a)$. All transitions are deterministic in this paper.[2]

For training we are given a set of dialogs D. Each dialog $d \in D$ is segmented into utterances $(u_1, \ldots, u_m)$ and is paired with a map, which is composed of a set of named landmarks $(l_1, \ldots, l_n)$.

4.1 State

The states of our decision making problem combine both our position in the dialog d and the path we have taken so far on the map. A state $s \in S$ is composed of $s = (u_i, l, c)$, where l is the named landmark we are located next to and c is a cardinal direction drawn from $\{\text{North}, \text{South}, \text{East}, \text{West}\}$ which determines which side of l we are on. Lastly, u_i is the utterance in d we are currently interpreting.

[2]Our learning algorithm is not dependent on a deterministic transition function and can be applied to domains with stochastic transitions, such as robot locomotion.

4.2 Action

An action $a \in A$ is composed of a named landmark l, the *target* of the action, together with a cardinal direction c which determines which side to pass l on. Additionally, a can be the null action, with $l = l'$ and $c = c'$. In this case, we interpret an utterance without moving on the map. A target l together with a cardinal direction c determine a point on the map, which is a fixed distance from l in the direction of c.

We make the assumption that at most one instruction occurs in a given utterance. This does not always hold true - the instruction giver sometimes chains commands together in a single utterance.

4.3 Transition

Executing action $a = (l', c')$ in state $s = (u_i, l, c)$ leads us to a new state $s' = T(s, a)$. This transition moves us to the next utterance to interpret, and moves our location to the target of the action. If a is the null action, $s = (u_{i+1}, l, c)$, otherwise $s' = (u_{i+1}, l', c')$. Figure 3 displays the state transitions two different actions.

To form a path through the map, we connect these state waypoints with a path planner[3] based on A^*, where the landmarks are obstacles. In a physical system, this would be replaced with a robot motion planner.

4.4 Reward

We define a reward function $R(s, a)$ which measures the utility of executing action a in state s. We wish to construct a route which follows the expert path as closely as possible. We consider a proposed route P close to the expert path P_e if P visits landmarks in the same order as P_e, and also passes them on the correct side.

For a given transition $s = (u_i, l, c)$, $a = (l', c')$, we have a binary feature indicating if the expert path moves from l to l'. In Figure 3, both a_1 and a_2 visit the next landmark in the correct order.

To measure if an action is to the correct side of a landmark, we have another binary feature indicating if P_e passes l' on side c. In Figure 3, only a_1 passes l' on the correct side.

In addition, we have a feature which counts the number of words in u_i which also occur in the name of l'. This encourages us to choose policies which interpret language relevant to a given landmark.

Our reward function is a linear combination of these features.

4.5 Policy

We formally define an interpretive strategy as a *policy* $\pi : S \rightarrow A$, a mapping from states to actions. Our goal is to find a policy π which maximizes the expected reward $\mathbb{E}_\pi[R(s, \pi(s))]$. The expected reward of following policy π from state s is referred to as the *value* of s, expressed as

$$V^\pi(s) = \mathbb{E}_\pi[R(s, \pi(s))] \quad (1)$$

When comparing the utilities of executing an action a in a state s, it is useful to define a function

$$\begin{aligned} Q^\pi(s, a) &= R(s, a) + V^\pi(T(s, a)) \\ &= R(s, a) + Q^\pi(T(s, a), \pi(s)) \end{aligned} \quad (2)$$

which measures the utility of executing a, and following the policy π for the remainder. A given Q function implicitly defines a policy π by

$$\pi(s) = \max_a Q(s, a). \quad (3)$$

Basic reinforcement learning methods treat states as atomic entities, in essence estimating V^π as a table. However, at test time we are following new directions for a map we haven't previously seen. Thus, we represent state/action pairs with a feature vector $\phi(s, a) \in \mathbb{R}^K$. We then represent the Q function as a linear combination of the features,

$$Q(s, a) = \theta^{\mathrm{T}} \phi(s, a) \quad (4)$$

and learn weights θ which most closely approximate the true expected reward.

4.6 Features

Our features $\phi(s, a)$ are a mixture of world and linguistic information. The linguistic information in our feature representation includes the instruction giver utterance and the names of landmarks on the map. Additionally, we furnish our algorithm with a list of English spatial terms, shown in Table 1. Our feature set includes approximately 200 features. Learning exactly which words influence decision making is difficult; reinforcement learning algorithms have problems with the large, sparse feature vectors common in natural language processing.

For a given state $s = (u, l, c)$ and action $a = (l', c')$, our feature vector $\phi(s, a)$ is composed of the following:

[3] We used the Java Path Planning Library, available at `http://www.cs.cmu.edu/~ggordon/PathPlan/`.

above, below, under, underneath, over, bottom, top, up, down, left, right, north, south, east, west, on

Table 1: The list of given spatial terms.

- **Coherence**: The number of words $w \in u$ that occur in the name of l'
- **Landmark Locality**: Binary feature indicating if l' is the closest landmark to l
- **Direction Locality**: Binary feature indicating if cardinal direction c' is the side of l' closest to (l, c)
- **Null Action**: Binary feature indicating if $l' =$ NULL
- **Allocentric Spatial**: Binary feature which conjoins the side c we pass the landmark on with each spatial term $w \in u$. This allows us to capture that the word *above* tends to indicate passing to the north of the landmark.
- **Egocentric Spatial**: Binary feature which conjoins the cardinal direction we move in with each spatial term $w \in u$. For instance, if (l, c) is above (l', c'), the direction from our current position is south. We conjoin this direction with each spatial term, giving binary features such as "the word *down* appears in the utterance and we move to the south".

5 Approximate Dynamic Programming

Given this feature representation, our problem is to find a parameter vector $\theta \in \mathbb{R}^K$ for which $Q(s, a) = \theta^{\mathrm{T}}\phi(s, a)$ most closely approximates $\mathbb{E}[R(s, a)]$. To learn these weights θ we use SARSA (Sutton and Barto, 1998), an online learning algorithm similar to Q-learning (Watkins and Dayan, 1992).

Algorithm 1 details the learning algorithm, which we follow here. We iterate over training documents $d \in D$. In a given state s_t, we act according to a probabilistic policy defined in terms of the Q function. After every transition we update θ, which changes how we act in subsequent steps.

Exploration is a key issue in any RL algorithm. If we act greedily with respect to our current Q function, we might never visit states which are actually higher in value. We utilize Boltzmann exploration, for which

$$\Pr(a_t|s_t;\theta) = \frac{\exp(\frac{1}{\tau}\theta^{\mathrm{T}}\phi(s_t, a_t))}{\sum_{a'}\exp(\frac{1}{\tau}\theta^{\mathrm{T}}\phi(s_t, a'))} \quad (5)$$

The parameter τ is referred to as the *temperature*, with a higher temperature causing more exploration, and a lower temperature causing more exploitation. In our experiments $\tau = 2$.

Acting with this exploration policy, we iterate through the training dialogs, updating our feature weights θ as we go. The update step looks at two successive state transitions. Suppose we are in state s_t, execute action a_t, receive reward $r_t = R(s_t, a_t)$, transition to state s_{t+1}, and there choose action a_{t+1}. The variables of interest are $(s_t, a_t, r_t, s_{t+1}, a_{t+1})$, which motivates the name SARSA.

Our current estimate of the Q function is $Q(s, a) = \theta^{\mathrm{T}}\phi(s, a)$. By the Bellman equation, for the true Q function

$$Q(s_t, a_t) = R(s_t, a_t) + \max_{a'} Q(s_{t+1}, a') \quad (6)$$

After each action, we want to move θ to minimize the *temporal difference*,

$$R(s_t, a_t) + Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \quad (7)$$

Input: Dialog set D
Reward function R
Feature function ϕ
Transition function T
Learning rate α_t
Output: Feature weights θ

Initialize θ to small random values
until θ *converges* **do**
  foreach *Dialog* $d \in D$ **do**
    Initialize $s_0 = (l_1, u_1, \emptyset)$,
    $a_0 \sim \Pr(a_0|s_0;\theta)$
    for $t = 0$*;* s_t *non-terminal;* $t{+}{+}$ **do**
      Act: $s_{t+1} = T(s_t, a_t)$
      Decide: $a_{t+1} \sim \Pr(a_{t+1}|s_{t+1};\theta)$
      Update:
      $\Delta \leftarrow R(s_t, a_t) + \theta^{\mathrm{T}}\phi(s_{t+1}, a_{t+1})$
        $- \theta^{\mathrm{T}}\phi(s_t, a_t)$
      $\theta \leftarrow \theta + \alpha_t\phi(s_t, a_t)\Delta$
    end
  end
end
return θ

Algorithm 1: The SARSA learning algorithm.

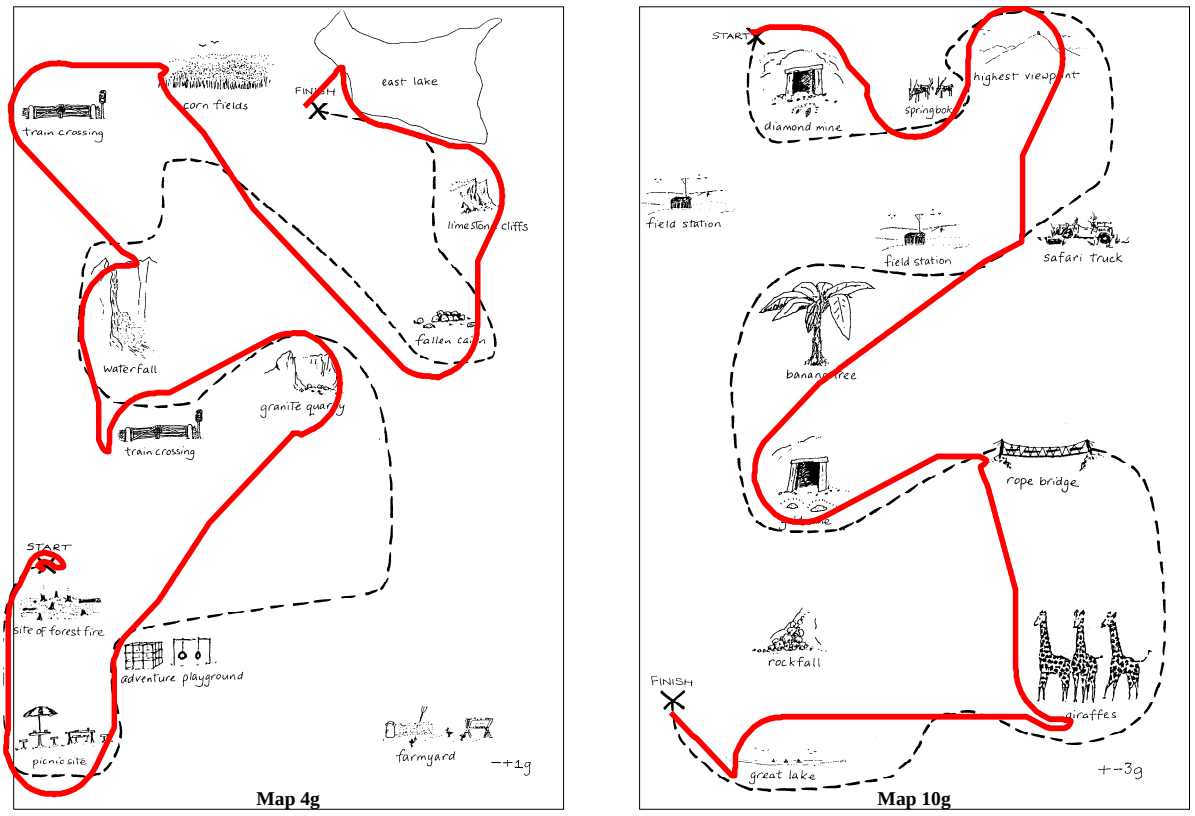


Figure 4: Sample output from the SARSA policy. The dashed black line is the reference path and the solid red line is the path the system follows.

For each feature $\phi_i(s_t, a_t)$, we change θ_i proportional to this temporal difference, tempered by a learning rate α_t. We update θ according to

$$\theta = \theta + \alpha_t \phi(s_t, a_t)(R(s_t, a_t) + \theta^{\mathrm{T}} \phi(s_{t+1}, a_{t+1}) - \theta^{\mathrm{T}} \phi(s_t, a_t)) \quad (8)$$

Here α_t is the learning rate, which decays over time[4]. In our case, $\alpha_t = \frac{10}{10+t}$, which was tuned on the training set. We determine convergence of the algorithm by examining the magnitude of updates to θ. We stop the algorithm when

$$||\theta_{t+1} - \theta_t||_\infty < \epsilon \quad (9)$$

6 Experimental Design

We evaluate our system on the Map Task corpus, splitting the corpus into 96 training dialogs and 32 test dialogs. The whole corpus consists of approximately 105,000 word tokens. The maps seen at test time do not occur in the training set, but some of the human participants are present in both.

[4]To guarantee convergence, we require $\sum_t \alpha_t = \infty$ and $\sum_t \alpha_t^2 < \infty$. Intuitively, the sum diverging guarantees we can still learn arbitrarily far into the future, and the sum of squares converging guarantees that our updates will converge at some point.

6.1 Evaluation

We evaluate how closely the path P generated by our system follows the expert path P_e. We measure this with respect to two metrics: the order in which we visit landmarks and the side we pass them on.

To determine the order P_e visits landmarks we compute the minimum distance from P_e to each landmark, and threshold it at a fixed value.

To score path P, we compare the order it visits landmarks to the expert path. A transition $l \rightarrow l'$ which occurs in P counts as correct if the same transition occurs in P_e. Let $|P|$ be the number of landmark transitions in a path P, and N the number of correct transitions in P. We define the *order precision* as $N/|P|$, and the *order recall* as $N/|P_e|$.

We also evaluate how well we are at passing landmarks on the correct side. We calculate the distance of P_e to each side of the landmark, considering the path to visit a side of the landmark if the distance is below a threshold. This means that a path might be considered to visit multiple sides of a landmark, although in practice it is usu-

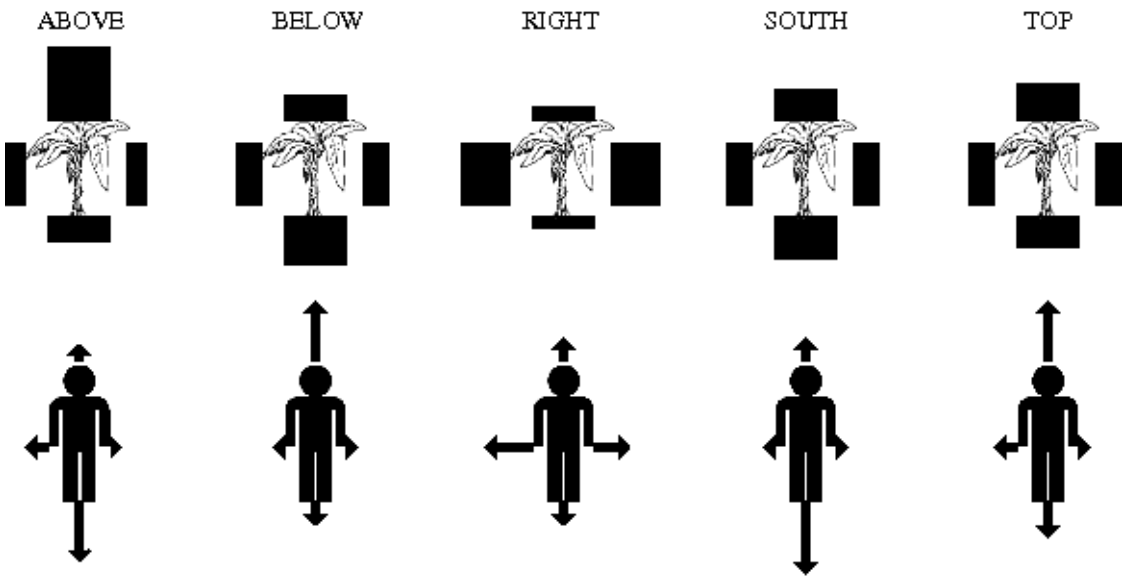


Figure 5: This figure shows the relative weights of spatial features organized by spatial word. The top row shows the weights of allocentric (landmark-centered) features. For example, the top left figure shows that when the word *above* occurs, our policy prefers to go to the north of the target landmark. The bottom row shows the weights of egocentric (absolute) spatial features. The bottom left figure shows that given the word *above*, our policy prefers to move in a southerly cardinal direction.

ally one. If C is the number of landmarks we pass on the correct side, define the *side precision* as $C/|P|$, and the *side recall* as $C/|P_e|$.

6.2 Comparison Systems

The baseline policy simply visits the closest landmark at each step, taking the side of the landmark which is closest. It pays no attention to the direction language.

We also compare against the policy gradient learning algorithm of Branavan et al. (2009). They parametrize a probabilistic policy $\Pr(s|a;\theta)$ as a log-linear model, in a similar fashion to our exploration policy. During training, the learning algorithm adjusts the weights θ according to the gradient of the value function defined by this distribution.

Reinforcement learning algorithms can be classified into *value* based and *policy* based. Value methods estimate a value function V for each state, then act greedily with respect to it. Policy learning algorithms directly search through the space of policies. SARSA is a value based method, and the policy gradient algorithm is policy based.

	Visit Order			Side		
	P	R	F_1	P	R	F_1
Baseline	28.4	37.2	32.2	46.1	60.3	52.2
PG	31.1	43.9	36.4	49.5	**69.9**	57.9
SARSA	**45.7**	**51.0**	**48.2**	**58.0**	64.7	**61.2**

Table 2: Experimental results. Visit order shows how well we follow the order in which the answer path visits landmarks. 'Side' shows how successfully we pass on the correct side of landmarks.

7 Results

Table 2 details the quantitative performance of the different algorithms. Both SARSA and the policy gradient method outperform the baseline, but still fall significantly short of expert performance. The baseline policy performs surprisingly well, especially at selecting the correct side to visit a landmark.

The disparity between learning approaches and gold standard performance can be attributed to several factors. The language in this corpus is conversational, frequently ungrammatical, and contains troublesome aspects of dialog such as conversational repairs and repetition. Secondly, our action and feature space are relatively primitive, and don't capture the full range of spatial expression. Path descriptors, such as the difference between *around* and *past* are absent, and our feature

representation is relatively simple.

The SARSA learning algorithm accrues more reward than the policy gradient algorithm. Like most gradient based optimization methods, policy gradient algorithms oftentimes get stuck in local maxima, and are sensitive to the initial conditions. Furthermore, as the size of the feature vector K increases, the space becomes even more difficult to search. There are no guarantees that SARSA has reached the best policy under our feature space, and this is difficult to determine empirically. Thus, some accuracy might be gained by considering different RL algorithms.

8 Discussion

Examining the feature weights θ sheds some light on our performance. Figure 5 shows the relative strength of weights for several spatial terms. Recall that the two main classes of spatial features in ϕ are egocentric (what direction we move in) and allocentric (on which side we pass a landmark), combined with each spatial word.

Allocentric terms such as *above* and *below* tend to be interpreted as going to the north and south of landmarks, respectively. Interestingly, our system tends to move in the opposite cardinal direction, i.e. the agent moves south in the egocentric frame of reference. This suggests that people use *above* when we are already above a landmark. *South* slightly favors passing on the south side of landmarks, and has a heavy tendency to move in a southerly direction. This suggests that south is used more frequently in an egocentric reference frame.

Our system has difficulty learning the meaning of *right*. *Right* is often used as a conversational filler, and also for dialog alignment, such as

> "right okay right go vertically up then between the springboks and the highest viewpoint."

Furthermore, *right* can be used in both an egocentric or allocentric reference frame. Compare

> "go to the uh right of the mine"

which utilizes an allocentric frame, with

> "right then go eh uh to your right horizontally"

which uses an egocentric frame of reference. It is difficult to distinguish between these meanings without syntactic features.

9 Conclusion

We presented a reinforcement learning system which learns to interpret natural language directions. Critically, our approach uses no semantic annotation, instead learning directly from human demonstration. It successfully acquires a subset of spatial semantics, using reinforcement learning to derive the correspondence between instruction language and features of paths. While our results are still preliminary, we believe our model represents a significant advance in learning natural language meaning, drawing its supervision from human demonstration rather than word distributions or hand-labeled semantic tags. Framing language acquisition as apprenticeship learning is a fruitful research direction which has the potential to connect the symbolic, linguistic domain to the non-symbolic, sensory aspects of cognition.

Acknowledgments

This research was partially supported by the National Science Foundation via a Graduate Research Fellowship to the first author and award IIS-0811974 to the second author and by the Air Force Research Laboratory (AFRL), under prime contract no. FA8750-09-C-0181. Thanks to Michael Levit and Deb Roy for providing digital representations of the maps and a subset of the corpus annotated with their spatial representation.

References

Pieter Abbeel and Andrew Y. Ng. 2004. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-first International Conference on Machine Learning*. ACM Press.

A. Anderson, M. Bader, E. Bard, E. Boyle, G. Doherty, S. Garrod, S. Isard, J. Kowtko, J. Mcallister, J. Miller, C. Sotillo, H. Thompson, and R. Weinert. 1991. The HCRC map task corpus. *Language and Speech, 34*, pages 351–366.

S.R.K. Branavan, Harr Chen, Luke Zettlemoyer, and Regina Barzilay. 2009. Reinforcement learning for mapping instructions to actions. In *ACL-IJCNLP '09*.

James Richard Curran. 2003. *From Distributional to Semantic Similarity*. Ph.D. thesis, University of Edinburgh.

Charles Fillmore. 1997. *Lectures on Deixis*. Stanford: CSLI Publications.

Benjamin Kuipers. 2000. The spatial semantic hierarchy. *Artificial Intelligence*, 119(1-2):191–233.

Stephen Levinson. 2003. *Space In Language And Cognition: Explorations In Cognitive Diversity*. Cambridge University Press.

Michael Levit and Deb Roy. 2007. Interpretation of spatial language in a map navigation task. In *IEEE Transactions on Systems, Man, and Cybernetics, Part B, 37(3)*, pages 667–679.

Terry Regier. 1996. *The Human Semantic Potential: Spatial Language and Constrained Connectionism*. The MIT Press.

Richard S. Sutton and Andrew G. Barto. 1998. *Reinforcement Learning: An Introduction*. MIT Press.

Leonard Talmy. 1983. How language structures space. In *Spatial Orientation: Theory, Research, and Application*.

Christine Tanz. 1980. *Studies in the acquisition of deictic terms*. Cambridge University Press.

Hongmei Wang, Alan M. Maceachren, and Guoray Cai. 2004. Design of human-GIS dialogue for communication of vague spatial concepts. In *GIScience*.

C. J. C. H. Watkins and P. Dayan. 1992. Q-learning. *Machine Learning*, pages 8:279–292.

Yuan Wei, Emma Brunskill, Thomas Kollar, and Nicholas Roy. 2009. Where to go: interpreting natural directions using global inference. In *ICRA'09: Proceedings of the 2009 IEEE international conference on Robotics and Automation*, pages 3761–3767, Piscataway, NJ, USA. IEEE Press.

Luke S. Zettlemoyer and Michael Collins. 2009. Learning context-dependent mappings from sentences to logical form. In *ACL-IJCNLP '09*, pages 976–984.