Help, Anna! Visual Navigation with Natural Multimodal Assistance via Retrospective Curiosity-Encouraging Imitation Learning

Khanh Nguyen[⊙] and **Hal Daumé III**[⊙♡]
University of Maryland, College Park[⊙], Microsoft Research, New York[♡]
{kxnguyen,hal}@umiacs.umd.edu

Abstract

Mobile agents that can leverage help from humans can potentially accomplish more complex tasks than they could entirely on their own. We develop "Help, Anna!" (HANNA), an interactive photo-realistic simulator in which an agent fulfills object-finding tasks by requesting and interpreting natural language-and-vision assistance. An agent solving tasks in a HANNA environment can leverage simulated human assistants, called ANNA (Automatic Natural Navigation Assistants), which, upon request, provide natural language and visual instructions to direct the agent towards the goals. To address the HANNA problem, we develop a memory-augmented neural agent that hierarchically models multiple levels of decision-making, and an imitation learning algorithm that teaches the agent to avoid repeating past mistakes while simultaneously predicting its own chances of making future progress. Empirically, our approach is able to ask for help more effectively than competitive baselines and, thus, attains higher task success rate on both previously seen and previously unseen environments. We publicly release code and data at https://github.com/khanhptnk/hanna .

1 Introduction

The richness and generalizability of natural language makes it an effective medium for directing mobile agents in navigation tasks, even in environments they have never encountered before (Anderson et al., 2018b; Chen et al., 2019; Misra et al., 2018; de Vries et al., 2018; Qi et al., 2019). Nevertheless, even with language-based instructions, such tasks can be overly difficult for agents on their own, especially in unknown environments. To accomplish tasks that surpass their knowledge and skill levels, agents must be able to actively seek for and leverage assistance in the environment. Humans are rich external knowledge sources but, unfortunately, they may not be available all the time to provide guidance, or may be unwilling to help too frequently. To reduce the needed effort from human assistants, it is essential to design research platforms for teaching agents to request help mindfully.

In natural settings, human assistance is often:

◇ derived from *interpersonal* interaction (a lost tourist asks a local for directions);

◇ *reactive* to the situation of the receiver, based on the assistant's knowledge (the local may guide the tourist to the goal, or may redirect them to a different source of assistance);

◇ delivered via a *multimodal* communication channel (the local uses a combination of language, images, maps, gestures, etc.).

We introduce the "Help, Anna!" (HANNA) problem (§ 3), in which a mobile agent has to navigate (without a map) to an object by interpreting its first-person visual perception and requesting help from *Automatic Natural Navigation Assistants* (ANNA). HANNA models a setting in which a human is not always available to help, but rather that human assistants are scattered throughout the environment and provide help upon request (modeling the *interpersonal* aspect). The assistants are *not* omniscient: they are only familiar with certain regions of the environment and, upon request, provide *subtasks*, expressed in language and images (modeling the *multimodal* aspect), for getting *closer* to the goal, not necessarily for fully completing the task (modeling the *reactive* aspect).

In HANNA, when the agent gets lost and becomes unable to make progress, it has the option of requesting assistance from ANNA. At test time, the agent must decide where to go and whether to request help from ANNA without additional supervision. At training time, we leverage imitation learning to learn an effective agent, both in terms of navigation, and in terms of being able to decide

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 684–695,
Hong Kong, China, November 3–7, 2019. ©2019 Association for Computational Linguistics

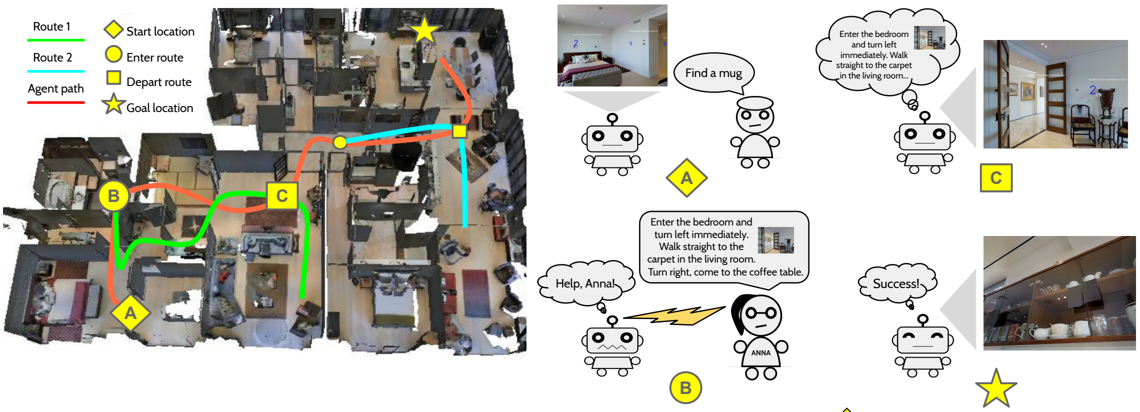
Route 1
Route 2
Agent path
Start location
Enter route
Depart route
Goal location
A
B
C
Find a mug
Enter the bedroom and turn left immediately. Walk straight to the carpet in the living room...
Enter the bedroom and turn left immediately. Walk straight to the carpet in the living room. Turn right, come to the coffee table.
Help, Anna!
ANNA
Success!
A
B
C

Figure 1: An example HANNA task. Initially, the agent stands in the bedroom at A and is requested by a human requester to "find a mug." The agent begins, but gets lost somewhere in the bathroom. It gets to the start location of route ~ (B) to request help from ANNA. Upon request, ANNA assigns the agent a navigation subtask described by a natural language instruction that guides the agent to a target location, and an image of the view at that location. The agents follows the language instruction and arrives at C, where it observes a match between the target image and the current view, thus decides to depart route ~. After that, it resumes the main task of finding a mug. From this point, the agent gets lost one more time and has to query ANNA for another subtask that helps it follow route ~ and enter the kitchen. The agent successfully fulfills the task it finally stops within ϵ meters of an instance of the requested object (☆). Here, the ANNA feedback is simulated using two pre-collected language-assisted routes (~ and ~).

when it is most worthwhile to request assistance.

This paper has two primary contributions:

1. Constructing the HANNA simulator by augmenting an indoor photo-realistic simulator with simulated human assistance, mimicking a scenario where a mobile agent finds objects by asking for directions along the way (§3).
2. An effective model and training algorithm for the HANNA problem, which includes a hierarchical memory-augmented recurrent architecture that models human assistance as *sub-goals* (§ 5), and introduces an imitation learning objective that enhances exploration of the environment and interpretability of the agent's help-request decisions. (§4).

We embed the HANNA problem in the photo-realistic Matterport3D environments (Chang et al., 2017) with *no* extra annotation cost by reusing the pre-existing Room-to-Room dataset (Anderson et al., 2018b). Empirical results (§ 7) show that our agent can effectively learn to request and interpret language and vision instructions, given a training set of 51 environments and less than 9,000 language instructions. Even in new environments, where the scenes and the language instructions are previously unseen, the agent successfully accomplishes 47% of its tasks. Our methods for training the navigation and help-request policies outperform competitive baselines by large margins.

2 Related work

Simulated environments provide an inexpensive platform for fast prototyping and evaluating new ideas before deploying them into the real world. Video-game and physics simulators are standard benchmarks in reinforcement learning (Todorov et al., 2012; Mnih et al., 2013; Kempka et al., 2016; Brockman et al., 2016; Vinyals et al., 2017). Nevertheless, these environments under-represent the complexity of the world. Realistic simulators play an important role in sim-to-real approaches, in which an agent is trained with arbitrarily many samples provided by the simulators, then transferred to real settings using sample-efficient transfer learning techniques (Kalashnikov et al., 2018; Andrychowicz et al., 2018; Karttunen et al., 2019). While modern techniques are capable of simulating images that can convince human perception (Karras et al., 2017, 2018), simulating language interaction remains challenging. There are efforts in building complex interactive text-based worlds (Côté et al., 2018; Urbanek et al., 2019) but the lack of a graphical component makes them not suitable for visually grounded learning. On the other hand, experimentation on real humans and robots, despite expensive and time-consuming, are important for understanding the true complexity of real-world scenarios (Chai et al., 2018, 2016; Rybski et al., 2007; Mohan and Laird, 2014; Liu et al., 2016; She et al., 2014).

Recent navigation tasks in photo-realistic simulators have accelerated research on teaching agents to execute human instructions. Nevertheless, modeling human assistance in these problems remains simplistic (Table 1): they either do not incorporate the ability to request additional help while executing tasks (Misra et al., 2014, 2017; Anderson et al., 2018b; Chen et al., 2019; Das et al., 2018; Misra et al., 2018; Wijmans et al., 2019; Qi et al., 2019), or mimic human verbal assistance with primitive, highly scripted language (Nguyen et al., 2019; Chevalier-Boisvert et al., 2019). HANNA improves the realisticity of the VNLA setup (Nguyen et al., 2019) by using fully natural language instructions.

Imitation learning algorithms are a great fit for training agents in simulated environments: access to ground-truth information about the environments allows optimal actions to be computed in many situations. The "teacher" in standard imitation learning algorithms (Daumé III et al., 2009; Ross et al., 2011; Ross and Bagnell, 2014; Chang et al., 2015; Sun et al., 2017; Sharaf and Daumé III, 2017) does not take into consideration the agent's capability and behavior. He et al. (2012) present a coaching method where the teacher gradually increases the complexity of its demonstrations over time. Welleck et al. (2019) propose an "unlikelihood" objective, which, similar to our curiosity-encouraging objective, penalizes likelihoods of candidate negative actions to avoid mistake repetition. Our approach takes into account the agent's past and future behavior to determine actions that are most and least beneficial to them, combining the advantages of both model-based and progress-estimating methods (Wang et al., 2018; Ma et al., 2019a,b).

3 The HANNA Simulator

Problem. HANNA simulates a scenario where a *human requester* asks a mobile *agent* via language to find an object in an indoor environment. The task request is only a high-level command ("find [object(s)]"), modeling the general case when the requester does not need know how to accomplish a task when requesting it. We assume the task is always feasible: there is at least an instance of the requested object in the environment.

Figure 1, to which references in this section will be made, illustrates an example where the agent is asked to "find a mug." The agent starts at a random location (A), is given a task request, and is allotted a budget of T time steps to complete the task. The agent succeeds the if its final location is within $\epsilon_{\text{success}}$ meters of the location of any instance of the requested object (☆). The agent is not given any sensors that help determine its location or the object's location and must navigate only with a monocular camera that captures its first-person view as an RGB image (e.g., image in the upper right of Figure 1).

Problem	Request assistance	Multimodal instructions	Simulated humans
VLN	✗	✗	✗
VNLA	✓	✗	✓
CVDN	✓	✗	✗
HANNA (this work)	✓	✓	✓

Table 1: Comparing HANNA with other photo-realistic navigation problems. VLN (Anderson et al., 2018b) does not allow agent to request help. VNLA (Nguyen et al., 2019) models an advisor who is always present to help but speaks simple, templated language. CVDN (Thomason et al., 2019b) contains natural conversations in which a human assistant aids another human in navigation tasks but offers limited language interaction simulation, as language assistance is not available when the agent deviates from the collected trajectories and tasks. HANNA simulates human assistants that provide language-and-vision instructions that adapt to the agent's current position and goal.

The only source of help the agent can leverage in the environment is *assistants*, who are present at both training and evaluation time. The assistants are not aware of the agent unless it enters their *zones of attention*, which include all locations within ϵ_{attn} meters of their locations. When the agent is in one of these zones, it has an option to request help from the corresponding assistant. The assistant helps the agent by giving a *subtask*, described by a natural language instruction that guides the agent to a specific location, and an image of the view at that location.

In our example, at Ⓑ, the assistant says *"Enter the bedroom and turn left immediately. Walk straight to the carpet in the living room. Turn right, come to the coffee table."* and provides an image of the destination in the living room. Executing the subtask may not fulfill the main task, but is guaranteed to get the agent to a location closer to a goal than where it was before (e.g., 🄲).

Photo-realistic Navigation Simulator. HANNA uses the Matterport3D simulator (Chang et al., 2017; Anderson et al., 2018b) to photo-realistically emulate a first-person view while navigating in indoor environments. HANNA features 68 Matterport3D environments, each of which is a residential building consisting of multiple rooms and floors. Navigation is modeled as traversing an undirected graph $G = (V, E)$, where each location corresponds to a node $v \in V$ with 3D-coordinates $\mathbf{x}_v$, and edges are weighted by their lengths (in meters). The state of the agent is fully determined by its pose $\tau = (v, \psi, \omega)$, where v is its location, $\psi \in (0, 2\pi]$ is its heading (horizontal camera angle), and $\omega \in \left[-\frac{\pi}{6}, \frac{\pi}{6}\right]$ is its elevation (vertical camera angle). The agent does not know v, and the angles are constrained to multiples of $\frac{\pi}{6}$. In each step, the agent can either stay at its current location, or it can rotate toward and go to a location adjacent to it in the graph[1]. Every time the agent moves (and thus changes pose), the simulator recalculates the image to reflect the new view.

Automatic Natural Navigation Assistants (ANNA). ANNA is a simulation of human assistants who do not necessarily know themselves how to optimally accomplish the agent's goal: they are only familiar with scenes along certain paths in the environment, and thus give advice to help the agent make partial progress. Specifically, the assistance from ANNA is modeled by a set of *language-assisted routes* $R = \{r_1, r_2, \ldots, r_{|R|}\}$. Each route $r = (\psi^r, \omega^r, p^r, l^r)$ is defined by initial camera angles (ψ^r, ω^r), a path p^r in the environment graph, and a natural language instruction l^r. A route becomes *enterable* when its start location is adjacent to and within ϵ_{attn} meters of the agent's location. When the agent enters a route, it first adjusts its camera angles to (ψ^r, ω^r), then attempts to interpret the language instructions l^r to traverse along p^r. At any time, the agent can *depart* the route by stopping following l^r. An example of a route in Figure 1 is the combination of the initial camera angles at Ⓑ, the path ⤴, and the language instruction *"Enter the bedroom and turn left immediately..."*

The set of all routes starting from a location simulates a *human assistant* who can recall scenes along these routes' paths. The *zone of attention* of the simulated human is the set of all locations from which the agent can enter one of the routes; when the agent is in this zone, it may ask the human for help. Upon receiving a help request, the human selects a route $r^\star$ for the agent to enter (e.g., ⤴), and a location v_d on the route where it wants the agent to depart (e.g., 🄲). It then replies the agent with a *multimedia message* $(l^{r^\star}, \mathcal{I}^{v_d})$, where $l^{r^\star}$ is the selected route's language instruction, and $\mathcal{I}^{v_d}$ is an image of the panoramic view at the departure location. The message describes a *subtask* which requires the agent to follow the direction described by $l^{r^\star}$ and to stop if it reaches the location referenced by $\mathcal{I}^{v_d}$. The route $r^\star$ and the departure node v_d are selected to get the agent as close to a goal location as possible. Concretely, let R_{curr} be the set of all routes associated with the requested human. The selected route minimizes the distance to the goal locations among all routes in R_{curr}:

$$r^\star = \operatorname*{argmin}_{r \in R_{\text{curr}}} \bar{d}\left(r, V_{\text{goal}}\right) \tag{1}$$

$$\text{where } \bar{d}\left(r, V_{\text{goal}}\right) \overset{\text{def}}{=} \min_{g \in V_{\text{goal}}, v \in p^r} d\left(g, v\right) \tag{2}$$

$d(.,.)$ returns the (shortest-path) distance between two locations, and V_{goal} is the set of all goal locations. The departure location minimizes the distance to the goal locations among all locations on the selected route:

$$v_d = \operatorname*{argmin}_{g \in V_{\text{goal}}, v \in p_{r^\star}} d\left(g, v\right) \overset{\text{def}}{=} \bar{d}\left(r^\star, V_{\text{goal}}\right) \tag{3}$$

When the agent chooses to depart the route (not necessarily at the departure node), the human further assists it by providing $\mathcal{I}^{g^\star}$, an image of the panoramic view at the goal location closest to the departure node:

$$g^\star = \operatorname*{argmin}_{g \in V_{\text{goal}}} d\left(g, v_d\right) \tag{4}$$

The way the agent leverages ANNA to accomplish tasks is analogous to how humans travel using public transportation systems (e.g., bus, subway). For example, passengers of a subway system utilize fractions of pre-constructed routes to make progress toward a destination. They execute travel plans consisting of multiple subtasks, each of which requires entering a start stop, following a route (typically described by its name and last stop), and exiting at a departure stop (e.g., *"Enter the Penn Station, hop on the Red line in the direction toward the South Ferry, get off at the World Trade Center"*). Occasionally, users walk short distances (at a lower speed) to switch routes. Our setup follows the same principle, but instead of having physical vehicles and railways, we employ

[1] We use the "panoramic action space" (Fried et al., 2018).

low-level language-and-vision instructions as the "high-speed means" to accelerate travel.

Constructing ANNA route system. Given a photo-realistic simulator, the primary cost for constructing the HANNA problem comes from crowd-sourcing the natural language instructions. Ideally, we want to collect sufficient instructions to simulate humans in any location in the environment. Let $N = |V|$ be the number of locations in the environment. Since each simulated human is familiar with at most N locations, in the worst case, we need to collect $O(N^2)$ instructions to connect all location pairs. However, we theoretically prove that, assuming the agent executes instructions perfectly, it is possible to guide the agent between any location pair by collecting only $\Theta(N \log N)$ instructions. The key idea is using $O(\log N)$ instead of a single instruction to connect each pair, and reusing an instruction for multiple routes.

Lemma 1. *(proof in Appendix A) To guide the agent between any two locations using $O(\log N)$ instructions, we need to collect instructions for $\Theta(N \log N)$ location pairs.*

In our experiments, we leverage the pre-existing Room-to-room dataset (Anderson et al., 2018b) to construct the route system. This dataset contains 21,567 natural language instructions crowd-sourced from humans and is originally intended to be used for the Vision-Language Navigation task (such as those in Figure 1), where an agent executes a language instruction to go to a location. We exclude instructions of the test split and their corresponding environments because ground-truth paths are not given. We use (on average) 211 routes to connect (on overage) 125 locations per environment. Even though the routes are selected randomly in the original dataset, our experiments show that they are sufficient for completing the tasks (assuming perfect assistance interpretation).

4 Retrospective Curiosity-Encouraging Imitation Learning

Agent Policies. Let s be a fully-observed state that contains ground-truth information about the environment and the agent (e.g., object locations, environment graph, agent parameters, etc.). Let o_s be the corresponding observation given to the agent, which only encodes the current view, the current task, and extra information that the agent keeps track of (e.g., time, action history, etc.). The agent maintains two stochastic policies: a navigation policy $\hat{\pi}_{\text{nav}}$ and a help-request policy $\hat{\pi}_{\text{ask}}$. Each policy maps an observation to a probability distribution over its action space. Navigation actions are tuples $(v, \Delta\psi, \Delta\omega)$, where v is a next location that is adjacent to the current location and $(\Delta\psi, \Delta\omega)$ is the camera angle change. A special `stop` action is added to the set of navigation actions to signal that the agent wants to terminate the main task or a subtask (by departing a route). The action space of the help-request policy contains two actions: `request_help` and `do_nothing`. The `request_help` action is only available when the agent is in a zone of attention. Alg 1 describes the effects of these actions during a task episode.

Algorithm 1 Task episode, given agent help-request policy $\hat{\pi}_{\text{ask}}$ and navigation policy $\hat{\pi}_{\text{nav}}$

```
agent receives task request $e$
initialize the agent mode: $m \leftarrow$ main_task
initialize the language instruction: $l_0 \leftarrow e$
initialize the target image: $I_0^{\text{tgt}} \leftarrow$ None
for $t = 1 \dots T$ do
    let $s_t$ be the current state, $o_t$ the current observation,
        and $\tau_t = (v_t, \psi_t, \omega_t)$ the current pose
    agent makes a help-request decision $\hat{a}_t^{\text{ask}} \sim \hat{\pi}_{\text{ask}}(o_t)$
    carry on task from the previous step:
        $l_t \leftarrow l_{t-1}, I_t^{\text{tgt}} = I_{t-1}^{\text{tgt}}$
    if $\hat{a}_t^{\text{ask}} =$ request_help then
        set mode: $m \leftarrow$ sub_task
        request help: $(r, I^{\text{depart}}, I^{\text{goal}}) \leftarrow$ ANNA$(s_t)$
        set the language instruction: $l_t \leftarrow l^r$
        set the target image: $I_t^{\text{tgt}} \leftarrow I^{\text{depart}}$
        set the navigation action:
            $\hat{a}_t^{\text{nav}} \leftarrow (p_0^r, \psi^r - \psi_t, \omega^r - \omega_t)$,
            where $p_0^r$ is the start location of route $r$
    else
        agent chooses navigation: $\hat{a}_t^{\text{nav}} \sim \hat{\pi}_{\text{nav}}(o_t)$
        if $\hat{a}_t^{\text{nav}} =$ stop then
            if $m =$ main_task then
                break
            else
                set mode: $m \leftarrow$ main_task
                set the language instruction: $l_t \leftarrow e$
                set the target image: $I_t^{\text{tgt}} \leftarrow I^{\text{goal}}$
                set navigation action: $\hat{a}_t^{\text{nav}} \leftarrow (v_t, 0, 0)$
            end if
        end if
    end if
    agent executes $\hat{a}_t^{\text{nav}}$ to go to the next location
end for
```

Imitation Learning Objective. The agent is trained with imitation learning to mimic behaviors suggested by a navigation teacher $\pi^{\star}_{\text{nav}}$ and a help-request teacher $\pi^{\star}_{\text{ask}}$, who have access to the fully-observed states. In general, imitation learning (Daumé III et al., 2009; Ross et al., 2011; Ross and Bagnell, 2014; Chang et al., 2015; Sun et al., 2017) finds a policy $\hat{\pi}$ that minimizes the expected imitation loss $\mathcal{L}$ with respect to a teacher policy $\pi^\star$ under the agent-induced state distribution $\mathcal{D}_{\hat{\pi}}$:

$$\min_{\hat{\pi}} \mathbb{E}_{s \sim \mathcal{D}_{\hat{\pi}}} \left[\mathcal{L}(s, \hat{\pi}, \pi^\star) \right] \tag{5}$$

We frame the HANNA problem as an instance of *Imitation Learning with Indirect Intervention* (I3L) (Nguyen et al., 2019). Under this framework, assistance is viewed as augmenting the current environment with new information. Interpreting the assistance is cast as finding the optimal acting policy in the augmented environment. Formally, I3L searches for policies that optimize:

$$\min_{\hat{\pi}_{\text{ask}}, \hat{\pi}_{\text{nav}}} \mathbb{E}_{s \sim \mathcal{D}^{\text{state}}_{\hat{\pi}_{\text{nav}}, \mathcal{E}}, \mathcal{E} \sim \mathcal{D}^{\text{env}}_{\hat{\pi}_{\text{ask}}}} \left[L(s) \right] \tag{6}$$

$$L(s) = \mathcal{L}_{\text{nav}}(s, \hat{\pi}_{\text{nav}}, \pi^\star_{\text{nav}}) + \mathcal{L}_{\text{ask}}(s, \hat{\pi}_{\text{ask}}, \pi^\star_{\text{ask}})$$

where $\mathcal{L}_{\text{nav}}$ and $\mathcal{L}_{\text{ask}}$ are the navigation and help-request loss functions, respectively, $\mathcal{D}^{\text{env}}_{\hat{\pi}_{\text{ask}}}$ is the environment distribution induced by $\hat{\pi}_{\text{ask}}$, and $\mathcal{D}^{\text{state}}_{\hat{\pi}_{\text{nav}}, \mathcal{E}}$ is the state distribution induced by $\hat{\pi}_{\text{nav}}$ in environment $\mathcal{E}$. A common choice for the loss functions is the agent-estimated negative log likelihood of the reference action:

$$\mathcal{L}_{\text{NL}}(s, \hat{\pi}, \pi^\star) = -\log \hat{\pi}(a^\star \mid o_s) \tag{7}$$

where $a^\star$ is the reference action suggested by $\pi^\star$. We introduce novel loss functions that enforce more complex behaviors than simply mimicking reference actions.

Reference Actions. The navigation teacher suggests a reference action $a^{\text{nav}\star}$ that takes the agent to the next location on the shortest path from its location to the target location. Here, the target location refers to the nearest goal location (if no target image is available), or the location referenced by the target image (provided by ANNA). If the agent is already at the target location, $a^{\text{nav}\star} =$ `stop`. To decide whether the agent should request help, the help-request teacher verifies the following conditions:

1. `lost`: the agent will not get (strictly) closer to the target location in the future;
2. `uncertain_wong`: the entropy[2] of the navigation action distribution is greater than or equal to a threshold γ, and the highest-probability predicted navigation action is *not* suggested by the navigation teacher;
3. `never_asked`: the agent previously never requested help at the current location;

If condition (1) or (2), and condition (3) are satisfied, we set $a^{\text{ask}\star} =$ `request_help`; otherwise, $a^{\text{ask}\star} =$ `do_nothing`.

Curiosity-Encouraging Navigation Teacher. In addition to a reference action, the navigation teacher returns $A^{\text{nav}\otimes}$, the set of all non-reference actions that the agent took at the current location while executing the same language instruction:

$$\begin{aligned} A^{\text{nav}\otimes}_t = \big\{ & a \in A^{\text{nav}} : \exists t' < t,\ \ v_t = v_{t'}, \\ & l_t = l_{t'},\ \ a = a^{\text{nav}}_{t'} \neq a^{\text{nav}\star}_{t'} \big\} \end{aligned} \tag{8}$$

where A^{nav} is the navigation action space.

We devise a *curiosity-encouraging* loss $\mathcal{L}_{\text{curious}}$, which minimizes the log likelihoods of actions in $A^{\text{nav}\otimes}$. This loss prevents the agent from repeating past mistakes and motivates it to explore untried actions. The navigation loss is:

$$\begin{aligned} \mathcal{L}_{\text{nav}}(s, \hat{\pi}_{\text{nav}}, \pi^\star_{\text{nav}}) = & \overbrace{-\log \hat{\pi}_{\text{nav}}(a^{\text{nav}\star} \mid o_s)}^{\mathcal{L}_{\text{NL}}(s, \hat{\pi}_{\text{nav}}, \pi^\star_{\text{nav}})} \\ & + \alpha \underbrace{\frac{1}{|A^{\text{nav}\otimes}|} \sum_{a \in A^{\text{nav}\otimes}} \log \hat{\pi}_{\text{nav}}(a \mid o_s)}_{\mathcal{L}_{\text{curious}}(s, \hat{\pi}_{\text{nav}}, \pi^\star_{\text{nav}})} \end{aligned} \tag{9}$$

where $\alpha \in [0, \infty)$ is a weight hyperparameter.

Retrospective Interpretable Help-Request Teacher. In deciding whether the agent should ask for help, the help-request teacher must consider the agent's future situations. Standard imitation learning algorithms (e.g., DAgger) employ an *online* mode of interaction which queries the teacher at every time step. This mode of interaction is not suitable for our problem: the teacher must be able to predict the agent's future actions if it is queried when the episode is not finished. To overcome this challenge, we introduce a more efficient *retrospective* mode of interaction, which waits until the agent completes an episode and queries the teacher for reference actions for *all* time steps at once. With this approach, because the future actions at each time step are now fully observed, they can be taken into consideration when computing the reference action. In fact, we prove that the retrospective teacher is optimal for teaching the agent to determine the `lost` condition, which is the only condition that requires knowing the agent's future.

Lemma 2. *(proof in Appendix B) At any time step, the retrospective help-request teacher suggests the*

[2] Precisely, we use *efficiency*, or entropy of base $|A^{\text{nav}}| = 37$, where A^{nav} is the navigation action space.

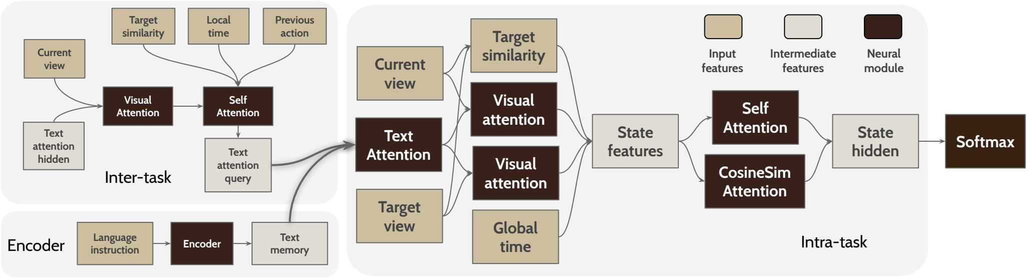

Figure 2: Our hierarchical recurrent model architecture (the navigation network). The help-request network is mostly similar except that the navigation action distribution is fed as an input to compute the "state features".

action that results in the agent getting closer to the target location in the future under its current navigation policy (if such an action exists).

To help the agent better justify its help-request decisions, we train a *reason classifier* Φ to predict which conditions are satisfied. To train this classifier, the teacher provides a reason vector $\rho^\star \in \{0,1\}^3$, where $\rho^\star_i = 1$ indicates that the i-th condition is met. We formulate this prediction problem as multi-label binary classification and employ a binary logistic loss for each condition. Learning to predict the conditions helps the agent make more accurate and interpretable decisions. The help-request loss is:

$$\mathcal{L}_{\text{ask}}(s, \hat{\pi}_{\text{ask}}, \pi^\star_{\text{ask}}) = \overbrace{-\log \hat{\pi}_{\text{ask}}(a^{\text{ask}\star} \mid o_s)}^{\mathcal{L}^{\text{NL}}(s, \hat{\pi}_{\text{ask}}, \pi^\star_{\text{ask}})} \underbrace{- \frac{1}{3} \sum_{i=1}^{3} \left[\rho^\star_i \log \hat{\rho}_i + (1 - \rho^\star_i) \log(1 - \hat{\rho}_i)\right]}_{\mathcal{L}_{\text{reason}}(s, \hat{\pi}_{\text{ask}}, \pi^\star_{\text{ask}})} \tag{10}$$

where $\left(a^{\text{ask}\star}, \rho^\star\right) = \pi^\star_{\text{ask}}(s)$, and $\hat{\rho} = \Phi(o_s)$ is the agent-estimated likelihoods of the conditions.

Split	Environments	Tasks	ANNA Instructions
Train	51	82,484	8,586
Val SeenEnv	51	5,001	4,287
Val UnseenAll	7	5,017	2,103
Test SeenEnv	51	5,004	4,287
Test Unseen	10	5,012	2,331

Table 2: Data split.

5 Hierarchical Recurrent Architecture

We model the navigation policy and the help-request policy as two separate neural networks. The two networks have similar architectures, which consists of three main components: the *text-encoding* component, the *inter-task* component, and the *intra-task* component (Figure 2). We use self-attention instead of recurrent neural networks to better capture long-term dependency, and develop novel cosine-similarity attention and ResNet-based time-encoding. Detail on the computations in each module is in the Appendix.

The text-encoding component computes a text memory M^{text}, which stores the hidden representation of the current language instruction. The inter-task module computes a vector h^{inter}_t representing the state of the current task's execution. During the episode, every time the current task is altered (due to the agent requesting help or departing a route), the agent re-encodes the new language instruction to generate a new text memory and resets the inter-task state to a zero vector. The intra-task module computes a vector h^{intra}_t representing the state of the entire episode. To compute this state, we first calculate $\bar{h}^{\text{intra}}_t$, a tentative current state, and $\tilde{h}^{\text{intra}}_t$, a weighted combination of the past states at nearly identical situations. h^{intra}_t is computed as:

$$h^{\text{intra}}_t = \bar{h}^{\text{intra}}_t - \beta \cdot \tilde{h}^{\text{intra}}_t \tag{11}$$

$$\beta = \sigma(W_{\text{gate}} \cdot [\bar{h}^{\text{intra}}_t; \tilde{h}^{\text{intra}}_t]) \tag{12}$$

Eq 11 creates an context-sensitive dissimilarity between the current state and the past states at nearly identical situations. The scale vector β determines how large the dissimilarity is based on the inputs. This formulation incorporates past related information into the current state, thus enables the agent to optimize the curiosity-encouraging loss effectively. Finally, h^{intra}_t is passed through a softmax layer to produce an action distribution.

6 Experimental Setup

Dataset. We generate a dataset of object-finding tasks in the HANNA environments to train and evaluate our agent. Table 2 summarizes the dataset split. Our dataset features 289 object types; the

Agent	SeenEnv				UnseenAll			
	SR $\uparrow$ (%)	SPL $\uparrow$ (%)	Nav. $\downarrow$ Err. (m)	Requests/ task $\downarrow$	SR $\uparrow$ (%)	SPL $\uparrow$ (%)	Nav. $\downarrow$ Err. (m)	Requests/ task $\downarrow$
Non-learning agents								
RandomWalk	0.54	0.33	15.38	0.0	0.46	0.23	15.34	0.0
Forward10	5.98	4.19	14.61	0.0	6.36	4.78	13.81	0.0
Learning agents								
No assistance	17.21	13.76	11.48	0.0	8.10	4.23	13.22	0.0
Learn to interpret assistance (ours)	**88.37**	**63.92**	**1.33**	**2.9**	**47.45**	**25.50**	**7.67**	**5.8**
Skylines								
Shortest	100.00	100.00	0.00	0.0	100.00	100.00	0.00	0.0
Perfect assistance interpretation	90.99	68.87	0.91	2.5	83.56	56.88	1.83	3.2

Table 3: Results on test splits. The agent with "perfect assistance interpretation" uses the teacher navigation policy ($\pi^{\star}_{\text{nav}}$) to make decisions when executing a subtask from Anna. Results of our final system are in bold.

language instruction vocabulary contains 2,332 words. The numbers of locations on the shortest paths to the requested objects are restricted to be between 5 and 15. With an average edge length of 2.25 meters, the agent has to travel about 9 to 32 meters to reach its goals. We evaluate the agent in environments that are seen during training (SeenEnv), and in environments that are not seen (UnseenAll). Even in the case of SeenEnv, the tasks and the Anna language instructions given during evaluation were never given in the same environments during training.

Hyperparameters. See Appendix.

Baselines and Skylines. We compare our agent against the following *non-learning* agents: 1. Shortest: uses the navigation teacher policy to make decisions (this is a skyline); 2. RandomWalk: randomly chooses a navigation action at every time step; 3. Forward10: navigates to the next location closest to the center of the current view to advance for 10 time steps. We compare our learned help-request policy with the following *heuristics*: 1. NoAsk: does not request help; 2. RandomAsk: randomly chooses to request help with a probabilty of 0.2, which is the average help-request ratio of our learned agent; 3. AskEvery5: requests help as soon as walking at least 5 time steps.

Evaluation metrics. Our main metrics are: *success rate* (SR), the fraction of examples on which the agent successfully solves the task; *navigation error*, the average (shortest-path) distance between the agent's final location and the nearest goal from that location; and *SPL* (Anderson et al., 2018a), which weights task success rate by travel distance as follows:

$$\text{SPL} = \frac{1}{N}\sum_{i=1}^{N} S_i \frac{L_i}{\max(P_i, L_i)} \tag{13}$$

where N is the number of tasks, S_i indicates whether task i is successful, P_i is the agent's travel distance, and L_i is the shortest-path distance to the goal nearest to the agent's final location.

7 Results

Main results. From Table 3, we see that our problem is challenging: simple heuristic-based baselines such as RandomWalk and Forward10 attain success rates less than 7%. An agent that learns to accomplish tasks without additional assistance from Anna succeeds only 17.21% of the time on Test SeenEnv, and 8.10% on Test UnseenAll. Leveraging help from Anna dramatically boosts the success rate by 71.16% on Test SeenEnv and by 39.35% on Test UnseenAll over not requesting help. Given the small size of our dataset (e.g., the agent has fewer than 9,000 subtask instructions to learn from), it is encouraging that our agent is successful in nearly half of its tasks. On average, the agent takes paths that are 1.38 and 1.86 times longer than the optimal paths on Test SeenEnv and Test UnseenAll, respectively. In unseen environments, it issues on average twice as many requests to as it does in seen environments. To understand how well the agent interprets the Anna instructions, we also provide results where our agent uses the optimal navigation policy to make decisions while executing subtasks. The large gaps on Test SeenEnv indicate there is still much room for improvement in the future, purely in learning to exe-

Assistance type	SEENENV	UNSEENALL
Target image only	84.95	31.88
+ Language instruction	**88.37**	**47.45**

Table 4: Success rates (%) of agents on test splits with different types of assistance.

	SEENENV		UNSEENALL	
$\hat{\pi}_{\text{ask}}$	SR ↑ (%)	Requests/ task ↓	SR ↑ (%)	Requests/ task ↓
NOASK	17.21	0.0	8.10	0.0
RANDOMASK	82.71	4.3	37.05	6.8
ASKEVERY5	87.39	3.4	34.42	7.1
Learned (ours)	**88.37**	**2.9**	**47.45**	**5.8**

Table 5: Success rates (%) of different help-request policies on test splits.

cute language instructions.

Does understanding language improve generalizability? Our agent is assisted with both language and visual instructions; similar to Thomason et al. (2019a), we disentangle the usefulness two these two modes of assistance. As seen in Table 4, the improvement from language on TEST UNSEENALL (+15.17%) is substantially more than that on TEST SEENENV (+3.42%), largely the agent can simply memorize the seen environments. This confirms that understanding language-based assistance effectively enhances the agent's capability of accomplishing tasks in novel environments.

Is learning to request help effective? Table 5 compares our learned help-request policies with baselines. We find that ASKEVERY5 provides a surprisingly strong baseline for this problem, leading to an improvement of +26.32% over not requesting help on TEST UNSEENALL. Nevertheless, our learned policy, with the ability to predict the future and access to the agent's uncertainty, outperforms all baselines by at least 10.40% in success rate on TEST UNSEENALL, while making less help requests. The small gap between the learned policy and ASKEVERY5 on TEST UNSEENALL is expected because, on this split, the performance is mostly determined by the model's memorizing capability and is mostly insensitive to the help-request strategy.

Is proposed model architecture effective? We implement an LSTM-based encoder-decoder model that is based on the architecture proposed by (Wang et al., 2019). To incorporate the target image, we add an attention layer that uses the image's vector set as the attention memory. We train this model with imitation learning using the standard negative log likelihood loss (Eq 7), without the curiosity-encouraging and reason-prediction losses. As seen in Table 6, our hierarchical recurrent model outperforms this model by a large margin on TEST UNSEENALL (+28.2%).

Model	SR ↑ (%)	Nav. mistake ↓ repeat (%)	Help-request ↓ repeat (%)
LSTM-ENCDEC	19.25	31.09	49.37
Our model ($\alpha = 0$)	43.12	25.00	40.17
Our model ($\alpha = 1$)	**47.45**	**17.85**	**21.10**

Table 6: Results on TEST UNSEENALL of our model, trained with and without curiosity-encouraging loss, and an LSTM-based encoder-decoder model (both models have about 15M parameters). "Navigation mistake repeat" is the fraction of time steps on which the agent repeats a non-optimal navigation action at a previously visited location while executing the same task. "Help-request repeat" is the fraction of help requests made at a previously visited location while executing the same task.

Does the proposed imitation learning algorithm achieve its goals? The curiosity-encouraging training objective is proposed to prevent the agent from making the same mistakes at previously encountered situations. Table 6 shows that training with the curiosity-encouraging objective reduces the chance of the agent looping and making the same decisions repeatedly. As a result, its success rate is greatly boosted (+4.33% on TEST UNSEENALL) over no curiosity-encouraging.

8 Conclusion

In this work, we present a photo-realistic simulator that mimics primary characteristics of real-life human assistance. We develop effective imitation learning techniques for learning to request and interpret the simulated assistance, coupled with a hierarchical neural network model for representing subtasks. Future work aims to provide more natural, linguistically realistic interaction between the agent and humans (e.g., providing the agent the ability ask a natural *question* rather than just signal for help), and to establish a theoretical framework for modeling human assistance. We are also exploring ways to deploy and evaluate our methods on real-world platforms.

References

Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. 2018a. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. 2018b. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3674–3683.

Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. 2018. Learning dexterous in-hand manipulation. *arXiv preprint arXiv:1808.00177*.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. Openai gym. *arXiv preprint arXiv:1606.01540*.

Joyce Y Chai, Rui Fang, Changsong Liu, and Lanbo She. 2016. Collaborative language grounding toward situated human-robot dialogue. *AI Magazine*, 37(4):32–45.

Joyce Y Chai, Qiaozi Gao, Lanbo She, Shaohua Yang, Sari Saba-Sadiya, and Guangyue Xu. 2018. Language to action: Towards interactive task learning with physical agents. In *International Joint Conference on Artificial Intelligence*.

Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. 2017. Matterport3D: Learning from RGB-D data in indoor environments. *International Conference on 3D Vision (3DV)*.

Kai-Wei Chang, Akshay Krishnamurthy, Alekh Agarwal, Hal Daume III, and John Langford. 2015. Learning to search better than your teacher. In *Proceedings of the International Conference of Machine Learning*.

Howard Chen, Alane Shur, Dipendra Misra, Noah Snavely, Ian Artzi, Yoav, Stephen Gould, and Anton van den Hengel. 2019. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahlou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. 2019. Babyai: A platform to study the sample efficiency of grounded language learning. In *Proceedings of the International Conference on Learning Representations*.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. 2018. Textworld: A learning environment for text-based games. In *Computer Games Workshop at ICML/IJCAI*.

Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. 2018. Embodied question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Hal Daumé III, John Langford, and Daniel Marcu. 2009. Search-based structured prediction. In *Machine learning*, volume 75, pages 297–325. Springer.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. 2018. Speaker-follower models for vision-and-language navigation. In *Proceedings of Advances in Neural Information Processing Systems*.

He He, Jason Eisner, and Hal Daumé III. 2012. Imitation learning by coaching. In *Proceedings of Advances in Neural Information Processing Systems*, pages 3149–3157.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. 2018. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. In *Proceedings of the Conference on Robot Learning*.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. 2017. Progressive growing of gans for improved quality, stability, and variation. In *Proceedings of the International Conference on Learning Representations*.

Tero Karras, Samuli Laine, and Timo Aila. 2018. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Janne Karttunen, Anssi Kanervisto, Ville Hautamäki, and Ville Kyrki. 2019. From video game to real robot: The transfer between action spaces. *arXiv preprint arXiv:1905.00741*.

Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. 2016. Vizdoom: A doom-based ai research platform for visual reinforcement learning. In *Computational Intelligence and Games (CIG), 2016 IEEE Conference on*, pages 1–8. IEEE.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Changsong Liu, Shaohua Yang, Sari Saba-Sadiya, Nishant Shukla, Yunzhong He, Song-Chun Zhu, and Joyce Chai. 2016. Jointly learning grounded task structures from language instruction and visual demonstration. In *Proceedings of Emperical Methods in Natural Language Processing*, pages 1482–1492.

Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. 2019a. Self-monitoring navigation agent via auxiliary progress estimation. In *Proceedings of the International Conference on Learning Representations*.

Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. 2019b. The regretful agent: Heuristic-aided navigation through progress estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6732–6740.

Dipendra Misra, Andrew Bennett, Valts Blukis, Eyvind Niklasson, Max Shatkhin, and Yoav Artzi. 2018. Mapping instructions to actions in 3d environments with visual goal prediction. In *Proceedings of Emperical Methods in Natural Language Processing*, pages 2667–2678. Association for Computational Linguistics.

Dipendra Misra, John Langford, and Yoav Artzi. 2017. Mapping instructions and visual observations to actions with reinforcement learning. *Proceedings of Emperical Methods in Natural Language Processing*.

Dipendra K Misra, Jaeyong Sung, Kevin Lee, and Ashutosh Saxena. 2014. Tell me dave: Contextsensitive grounding of natural language to mobile manipulation instructions. In *Robotics: Science and Systems*.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. In *NIPS Deep Learning Workshop*.

Shiwali Mohan and John Laird. 2014. Learning goal-oriented hierarchical tasks from situated interactive instruction. In *Association for the Advancement of Artificial Intelligence*.

Khanh Nguyen, Debadeepta Dey, Chris Brockett, and Bill Dolan. 2019. Vision-based navigation with language-based assistance via imitation learning with indirect intervention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12527–12537.

Yuankai Qi, Qi Wu, Peter Anderson, Marco Liu, Chunhua Shen, and Anton van den Hengel. 2019. Rerere: Remote embodied referring expressions in real indoor environments. *arXiv preprint arXiv:1904.10151*.

Stephane Ross and J Andrew Bagnell. 2014. Reinforcement and imitation learning via interactive no-regret learning. *arXiv preprint arXiv:1406.5979*.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of Artificial Intelligence and Statistics*, pages 627–635.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252.

Paul E Rybski, Kevin Yoon, Jeremy Stolarz, and Manuela M Veloso. 2007. Interactive robot task training through dialog and demonstration. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 49–56. ACM.

Amr Sharaf and Hal Daumé III. 2017. Structured prediction via learning to search under bandit feedback. In *Proceedings of the 2nd Workshop on Structured Prediction for Natural Language Processing*, pages 17–26.

Lanbo She, Shaohua Yang, Yu Cheng, Yunyi Jia, Joyce Chai, and Ning Xi. 2014. Back to the blocks world: Learning new actions through situated human-robot dialogue. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 89–97.

Wen Sun, Arun Venkatraman, Geoffrey J Gordon, Byron Boots, and J Andrew Bagnell. 2017. Deeply aggrevated: Differentiable imitation learning for sequential prediction. In *Proceedings of the International Conference of Machine Learning*.

Jesse Thomason, Daniel Gordan, and Yonatan Bisk. 2019a. Shifting the baseline: Single modality performance on visual navigation & qa. In *Conference of the North American Chapter of the Association for Computational Linguistics*.

Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. 2019b. Vision-and-dialog navigation. In *Proceedings of the Conference on Robot Learning*.

Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE.

Jack Urbanek, Angela Fan, Siddharth Karamcheti, Saachi Jain, Samuel Humeau, Emily Dinan, Tim Rocktäschel, Douwe Kiela, Arthur Szlam, and Jason Weston. 2019. Learning to speak and act in a fantasy text adventure game. *arXiv preprint arXiv:1903.03094*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, et al. 2017. Starcraft ii: A new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782*.

Harm de Vries, Kurt Shuster, Dhruv Batra, Devi Parikh, Jason Weston, and Douwe Kiela. 2018. Talk the walk: Navigating new york city through grounded dialogue. *arXiv preprint arXiv:1807.03367*.

Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. 2019. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Xin Wang, Wenhan Xiong, Hongmin Wang, and William Yang Wang. 2018. Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation. In *Proceedings of the European Conference on Computer Vision*.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019. Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319*.

Erik Wijmans, Samyak Datta, Oleksandr Maksymets, Abhishek Das, Georgia Gkioxari, Stefan Lee, Irfan Essa, Devi Parikh, and Dhruv Batra. 2019. Embodied question answering in photorealistic environments with point cloud perception. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6659–6668.