Learning to Follow Directions in Street View

Karl Moritz Hermann [1] **Mateusz Malinowski** [2] **Piotr Mirowski** [2]
Andras Banki-Horvath [2] **Keith Anderson** [2] **Raia Hadsell** [2]

Abstract

Navigating and understanding the real world remains a key challenge in machine learning and inspires a great variety of research in areas such as language grounding, planning, navigation and computer vision. We propose an instruction-following task that requires all of the above, and which combines the practicality of simulated environments with the challenges of ambiguous, noisy real world data. StreetNav is built on top of Google Street View and provides visually accurate environments representing real places. Agents are given driving instructions which they must learn to interpret in order to successfully navigate in this environment. Since humans equipped with driving instructions can readily navigate in previously unseen cities, we set a high bar and test our trained agents for similar cognitive capabilities. Although deep reinforcement learning (RL) methods are frequently evaluated only on data that closely follow the training distribution, our dataset extends to multiple cities and has a clean train/test separation. This allows for thorough testing of generalisation ability. This paper presents the StreetNav environment and tasks, a set of novel models that establish strong baselines, and analysis of the task and the trained agents.

1. Introduction

How do you get to Carnegie Hall?
— Practice, practice, practice...

The joke hits home for musicians and performers, but the rest of us expect actual directions. For humans, asking for directions and subsequently following those directions to successfully negotiate a new and unfamiliar environment comes naturally. Unfortunately the same cannot be said of most current techniques in machine learning. Transferring the experience of artificial agents from known to unknown environments remains a key obstacle in deep reinforcement learning (RL). For humans, transfer is frequently achieved through the common medium of language. This is particularly noticeable in various navigational tasks, where access to textual directions can greatly simplify the challenge of traversing a new city. This is made possible by our ability to integrate visual information and language and to use this to inform our actions in an inherently ambiguous world.

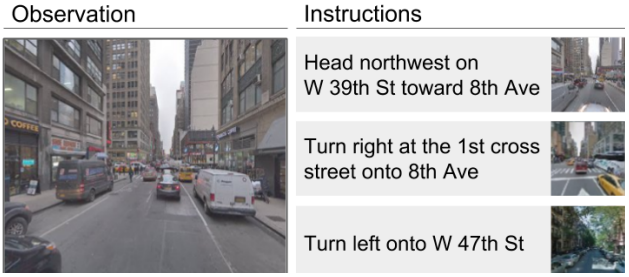

Figure 1. An example route with visual observation, written instructions and thumbnails. The agent must learn to interpret the text and thumbnails and navigate to the goal location.

Recent progress in the development of RL agents that can act in increasingly more sophisticated environments (Anderson et al., 2018b; Hill et al., 2017; Mirowski et al., 2018; 2017) supports the hope that such an understanding might be possible to develop within virtual agents in the near future. That said, the question of transfer and grounding requires more realistic circumstances in order to be fully investigated. While proofs of concept may be demonstrated in simulated, virtual environments, it is important to consider the transfer problem also on real-world data, where training and test environments may radically differ in many ways, such as visual appearance or topology of the map.

We propose a challenging new suite of RL environments termed StreetNav, based on Google Street View, which consists of natural images of real-world places with realistic connections between them. These environments are packaged together with Google Maps-based driving instructions to allow for a number of tasks that resemble the human experience of following directions to navigate a city and reach a destination (see Figure 1).

Successful agents need to integrate inputs coming from

[1]DeepMind, Berlin, Germany [2]DeepMind, London, UK.
Correspondence to: Karl Moritz Hermann <kmh@google.com>.

various sources that lead to optimal navigational decisions under realistic ambiguities, and then apply those behaviours when evaluated in previously unseen areas. StreetNav provides a natural separation between different neighbourhoods and cities so that such transfer can be evaluated under increasingly challenging circumstances. Concretely, we train agents in New York City and then evaluate them in an unseen region of the city, in an altogether new city (Pittsburgh), as well as on larger numbers of instructions than encountered during training. We describe a number of approaches that establish strong baselines on this problem, and demonstrate how RL-based agents can generalise better on this task than fully supervised sequence-to-sequence agents.

2. Related Work

Reinforcement Learning for Navigation End-to-end RL-based approaches to navigation (Mirowski et al., 2017; Jaderberg et al., 2017; Zhu et al., 2017; Chaplot et al., 2017) jointly learn a representation of the environment (mapping) together with a suitable sequence of actions (planning). These research efforts have utilised synthetic 3D environments such as VizDoom (Kempka et al., 2016), DeepMind Lab (Beattie et al., 2016), HoME (Brodeur et al., 2017), House 3D (Wu et al., 2018), Chalet (Yan et al., 2018), or AI2-THOR (Kolve et al., 2017). The challenge of generalisation to unseen test scenarios has been highlighted in Dhiman et al. (2018) and partially addressed by using differentiable memory that learns to store a map-like representation of the environment over a short episode (Wayne et al., 2018; Oh et al., 2016) or by learning to generate explicit maps of the environment (Parisotto et al., 2018; Zhang et al., 2017). More visually realistic environments such as Matterport Room-to-Room (Chang et al., 2017), AdobeIndoorNav (Mo et al., 2018), Stanford 2D-3D-S (Armeni et al., 2016), ScanNet (Dai et al., 2017), Gibson Env (Xia et al., 2018), and MINOS (Savva et al., 2017) have been recently introduced to represent indoor scenes, some augmented with navigational instructions. Notably, Anderson et al. (2018b) trained agents with supervised 'student / teacher forcing' (requiring privileged access to ground-truth actions at training time) to navigate in virtual houses. We implement this method as a baseline for comparison. Mirowski et al. (2018) have trained deep RL agents in large-scale environments based on Google Street View and studied the transfer of navigation skills by doing limited, modular retraining in new cities.

Our approach to enabling transfer is to train the agent to ground and follow instructions, relying on the common medium of language to act as a bridge to new environments. We have thus adopted some of the recommendations on the evaluation of embodied navigational agents proposed in Anderson et al. (2018a), namely using separated training, validation, and test environments and relying on geodesic distances. Unlike in Anderson et al. (2018a), we use a discrete action space that is more appropriate for our environment, we use a simple interpretable metric ('percentage of goals reached') rather than Success Path Length (SPL), and the agent does not emit the action 'DONE'.

Language, Perception, and Actions Humans acquire basic linguistic concepts through communication about, and interaction within their physical environment, e.g. by assigning words to visual observations (Gopnik & Meltzoff, 1984; Hoff, 2013). Linguistic concepts may also shape the world we perceive (Kay & Kempton, 1984; Boroditsky, 2001). The challenge of associating symbols to observations is often referred to as the symbol grounding problem (Harnad, 1990); it has been studied using symbolic approaches such as semantic parsers (Vogel & Jurafsky, 2010; Tellex et al., 2011; Krishnamurthy & Kollar, 2013; Matuszek et al., 2012; Malinowski & Fritz, 2014), and, more recently, deep learning (Kong et al., 2014; Ren et al., 2015; Malinowski et al., 2017; Rohrbach et al., 2017; Johnson et al., 2016; Park et al., 2018; Teney et al., 2017). Grounding has also been studied in the context of actions. For instance, Hermann et al. (2017), Hill et al. (2017), Chaplot et al. (2017); Shah et al. (2018), Yan et al. (2018), Das et al. (2018), and Oh et al. (2017) all investigate language-based or question answering-based navigation in synthetic, small-scale environments.

In terms of more realistic environments, similar to Anderson et al. (2018b), Fried et al. (2018) consider the problem of following textual instructions in Matterport 3D and focus on augmenting instructions. Using New York imagery, de Vries et al. (2018) use navigation instructions but rely on categorical annotation of nearby landmarks rather than visual observations and use a dataset of 500 panoramas only (ours is two orders of magnitude larger). Very recently, Cirik et al. (2018) and particularly Chen et al. (2018) have also proposed larger datasets of driving instructions grounded in Street View imagery. Our work shares a similar motivation to Chen et al. (2018), with key differences being that their agents observed both their heading and by how much to turn to reach the next street/edge in the graph, whereas ours need to learn what is a traversable direction purely from vision. We also focus on multiple instructions (by learning to parse a list), as well as our introduce models with hard attention.

3. The StreetNav Suite

We design navigation environments in the StreetNav suite by extending the dataset and environment available from StreetLearn[1] through the addition of driving instructions from Google Maps by randomly sampling start and goal positions. Admittedly, the language obtained is synthetic,

[1] An open-source environment built with Google Street View for navigation research: http://streetlearn.cc

but it is used by millions of people to see, hear, and follow instructions every day, which we feel justifies its inclusion in this 'real-world' problem setting.

We designate geographically separated training, validation and testing environments. Specifically, we reserve Lower Manhattan in New York City for training and use parts of Midtown for validation (hyperparameter tuning and model selection). Agents are evaluated both in-domain (a separate area of upper NYC), as well as out-of-domain (Pittsburgh). We make the described environments, data and tasks available for download at http://streetlearn.cc.

More precisely, each environment $\mathcal{R}$ is an undirected, connected graph $\mathcal{R} = \{\mathcal{V}, \mathcal{E}\}$, with a set of nodes $\mathcal{V}$ and connecting edges $\mathcal{E}$. Each node $v_i \in \mathcal{E}$ is a tuple containing a 360° panoramic image p_i and a location c_i (given by latitude and longitude). An edge $e_{ij} \in \mathcal{E}$ means that node v_i is accessible from v_j and vice versa. Each environment has a set of associated *routes* which are defined by a start and goal node, and a list of textual instructions and thumbnail images which represent directions to waypoints leading from start to goal. Success is determined by the agent reaching the destination described in those instructions within the allotted time frame. This problem requires from agents to understand the given instructions, determine which instruction is applicable at any point in time, and correctly follow it given the current context. Dataset statistics are in Table 1.

Environment	#routes	Ave: len	steps	instrs	$\lvert instr \rvert$
NYC train	580,415	1,194	128	4.0	7.1
NYC valid	10,000	1,184	125	3.7	8.1
NYC test	10,000	1,180	123	3.8	7.9
NYC larger	3,923	1,667	174	6.5	8.1
Pittsburgh test	8,474	998	106	3.8	6.6

Table 1. Dataset statistics: the number of routes, their average length in meters and in environment steps, the average number of instructions per route and average number of words per instruction.

3.1. Agent Interface

At each time step, the agent receives an observation and a list of instructions with matching thumbnails. Observations and thumbnails are RGB images taken from Google Street View, with the observation image representing the agent's field of view from the current location and pose. The instructions and thumbnails are drawn from the Google Directions API and describe a route between two points from within the given environment. n instructions are matched with $n + 1$ thumbnails, as both the initial start and the final goal are included in the list of locations represented by a thumbnail.

All RGB images (observations and thumbnails) are 60° crops from the panoramic image that are scaled to 84-by-84 pixels. The action space is composed of five discrete actions: move forward, slow rotation ($\pm 10^\circ$), and fast rotation ($\pm 30^\circ$). While moving forward the agent takes the graph edge that is within its viewing cone and most closely aligned to the agent's current orientation, while the lack of a suitable graph edge results in a NO-OP.

3.2. Variants of the StreetNav Task

To examine how an agent might learn to follow visually grounded navigation directions, we propose three task variants. They all share the same underlying structure, but differ in how the directions are presented (step-by-step, or all at once) and whether any feedback is given to the agents when reaching intermediate waypoints. The latter enables agents to use a signal to improve their policy, while without any signal agents have to deduce purely from observations where they are in the list of instructions.

3.2.1. Task 1: List + Goal Reward

The List + Goal Reward task mimics the human experience of following printed directions, with access to the complete set of instructions and referential images but without any incremental feedback as to whether one is still on the right track. Formally, at each step the agent is given as input a list of directions $\mathbf{d} = \langle d_1, d_2, \dots d_N \rangle$ where $d_i = \{\iota_i, t_i^s, t_i^e\}$ (instruction, start and end thumbnail). $\iota_i = \langle \iota_{i,1}, \iota_{i,2}, \dots \iota_{i,M} \rangle$ where $\iota_{i,j}$ is a single word token. t_i^s, t_i^e are thumbnails in $\mathbb{R}^{3\times 84\times 84}$. The number of directions N varies per route and the number of words M varies according to the instruction.

An agent begins each episode in an initial state $s_0 = \langle v_0, \theta_0 \rangle$ at the start node and heading associated with the given route, and is given an RGB image x_0 corresponding to that state. The goal is defined as the final node of the route, G. The agent must generate a sequence of actions $\langle s_0, a_0, s_1, a_1, \dots s_T \rangle$, with each action a_t leading to a modified state s_{t+1} and corresponding observation x_{t+1}.

The episode ends either when a maximal number of actions is reached $T > T_{\text{max}}$, or when the agent reaches G. The goal reward R_g is awarded if the agent reaches the final goal node G associated with the given route, i.e. $r_t = R_g$ if $v_t = G$.

3.2.2. Task 2: List + Incremental Reward

This task uses the same presentation of instructions as the previous task (the full list is given at every step), however we mitigate the challenge of exploration by increasing the reward density. In addition to the goal reward, a smaller reward R_w is awarded, as the input for the agent, for the successful execution of individual directions, meaning that the agent is given positive feedback when reaching any of

the waypoints for the first time:

$$r_t = \langle R_w \text{ if } v_t \in V_w \wedge \forall_{i<t}\, v_t \neq v_i \rangle + \langle R_g \text{ if } v_t = G \rangle$$

where V_w denotes the set of all the waypoints in the given route. This formula enables agents to learn when to switch from one instruction to another.

3.2.3. Task 3: Step-by-Step

Finally, we consider a simpler version of the task where agents are provided with a single instruction at a time, automatically switching to the next instruction whenever a waypoint is reached, similar to the human experience of following directions given by a GPS (or NavSat). Thereby two challenges that the agents have to solve in the previous tasks—'when to switch' and 'what instruction to switch to'—are removed. As in the List + Incremental Reward task, smaller rewards are given as waypoints are reached.

3.3. Training and Evaluation

Reward shaping can be used to simplify exploration and to make learning more efficient. We use early rewards within a fixed radius at each waypoint and the goal, meaning that an agent will receive fractional rewards once it is within a certain distance ($50m$). Reward shaping is only used for training, and never during evaluation.

We report the percentage of goals reached by a trained agent in an environment (training, validation or test). Agents are evaluated for 1 million steps with a 1,000 step limit per route. This means that the performance is measured across at least 1,000 routes. We do not consider waypoint rewards as a partial success, and award a score of one only if the final goal of the route is reached, and zero otherwise.

4. Architectures

We approach the challenge of grounded navigation with a deep reinforcement learning framework and formalise the learning problem as a Markov Decision Process (MDP). Let us consider an environment $\mathcal{R} \in \mathbb{S}$ from the suite of the environments $\mathbb{S}$. Our MDP is a tuple consisting of states that belong to the environment, i.e. $\mathcal{S} \in \mathcal{R}$, together with the possible directions $\mathcal{D} = \{\mathbf{d}_l\}_l \in \mathcal{E}$, and possible actions $\mathcal{A}$. Each direction $\mathbf{d}$ is a sequence of instructions, and pairs of thumbnails, i.e. $\mathbf{d} = \langle d_1, \ldots, d_n \rangle$ where $d_i = \{\iota_i, t_i^s, t_i^e\}$ (instruction, starting thumbnail, ending thumbnail). The length of $\mathbf{d}$ varies between the episodes. Each state $s \in \mathcal{S}$ is associated with a real location, and has coordinates c and observation x which is an RGB image of the location. Transitions from state to state are limited to rotations and movements to new locations that are allowed by the edges in the connectivity graph. The reward function, $\mathcal{R} : \mathcal{S} \times \mathcal{D} \rightarrow \mathbb{R}$, depends on the current state and the final goal d_g.

Our objective, as typical in the RL setting, is to learn a policy π that maximises the expected reward $E[\mathcal{R}]$. In this work, we use a variant of the REINFORCE (Williams, 1992) advantage actor-critic algorithm $E_\pi\left[\sum_t \nabla_\theta \log \pi(a_t|s_t, \mathbf{d}; \theta)(\mathcal{R}_t - \mathcal{V}^\pi(s_t))\right]$, where $\mathcal{R}_t = \sum_{j=0}^{T-t} \gamma^j r_{t+j}$, $[r_t]_t$ is a binary vector with one at t where the final destination is achieved within the 1-sample Monte Carlo estimate of the return, γ is a discounting factor, and T is the episode length.

In the following, we describe methods that transform input signals into vector representations and are combined with a recurrent network to learn the optimal policy $\pi^* = \arg\max_\pi E_\pi[\mathcal{R}]$ via gradient descent. We also describe several baseline architectures: simple adaptations of existing RL agents as well as ablations of our proposed architectures.

4.1. Input-to-Vector Transformation

We use an LSTM (Hochreiter & Schmidhuber, 1997) to transform the text instructions into a vector. We also use CNNs (LeCun et al., 2012) to transform RGB images, both observations and thumbnails, into vectors. Using deep learning architectures to transform raw inputs into a unified multi-modal representation has been found to be very effective (Malinowski et al., 2017; Donahue et al., 2015; Malinowski et al., 2018; Teney et al., 2017; Hill et al., 2017; Ren et al., 2015; Zhu et al., 2015; Hudson & Manning, 2018). Since observations and thumbnails come from the same source, we share the weights of the CNNs. Let $\boldsymbol{\iota} = \text{LSTM}_{\theta_1}(\iota)$, $\mathbf{x} = \text{CNN}_{\theta_2}(x)$, $\mathbf{t}^s = \text{CNN}_{\theta_2}(t^s)$, and $\mathbf{t}^e = \text{CNN}_{\theta_2}(t^e)$ be vector representations of the instruction, observation, start-, and end-thumbnail respectively.

4.2. Multimodal Embedding

Combining signals from different modalities may require a deeper approach that can better align and 'mix' the signals. For this purpose, we use a three-layer multi-layer perceptron (MLP), with 256 units per layer, whose input is a concatenation of the signals, and whose output is their merged representation. In the following architectures, we use this module twice, once to embed instructions and the corresponding thumbnails $\mathbf{i} = \text{MLP}(\boldsymbol{\iota}_i, \mathbf{t}_i^s, \mathbf{t}_i^e)$ and another time to embed this output with the current observation, i.e. $\mathbf{p} = \text{MLP}(\mathbf{x}, \mathbf{i})$. The output of the second module is input to the policy network. This module is inspired by research on combining vision and language (Reed et al., 2016; Perez et al., 2018), and is shown to be important in our case.

4.3. Previous Reward and Action

Optionally, we input the previous action and previous obtained reward to the policy network. In that case the policy should formally be written as $\pi(a_t|s_t, a_{t-1}, \mathbf{d}; \theta)$ or

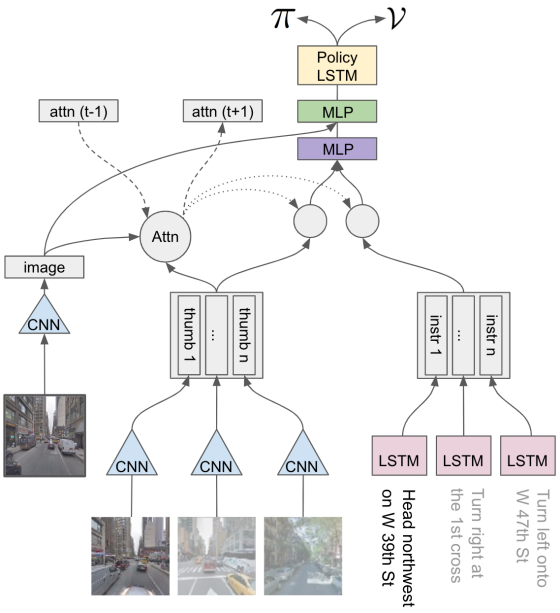

Figure 2. Architecture of an agent with attention. Feature representations are computed for the observation and for all directions, using a CNN and a LSTM. The attention module selects a thumbnail and passes it and the associated text instruction to the first multimodal MLP, whose output is concatenated with the image features and fed through the second MLP before being input to the policy LSTM. The policy LSTM outputs a policy π and a value function $\mathcal{V}$. Colours point to components that share weights.

$\pi(a_t|s_t, a_{t-1}, \mathcal{R}_{t-1}, \mathbf{d}; \theta)$. Note that adding the previous reward is only relevant when intermediate rewards are given for reaching waypoints, in which case the reward signal can be explicitly or implicitly used to determine which instruction to execute next, and that this architectural choice is unavailable in LIST + GOAL REWARD (Section 3.2.1).

4.4. Non-attentional Architectures

We introduce two architectures which are derived from the IMPALA agent and adapted to work in this setting (Espeholt et al., 2018). The agent does not use attention but rather observes the instructions and thumbnails at every time step. To accomplish this, we either concatenate the representations of all inputs (AllConcat) or first sum over all instructions and then concatenate this summed representation with the observation (AllSum), before passing the result of this operation as input to the multimodal embedding modules.

With this type of the architecture, the agent does not explicitly decide 'when to switch' or 'what to switch to', but rather relies on the policy network to learn a suitable strategy to use the instructions. When explicitly cued by the waypoint reward as an input signal, the decision when to move to the next instruction should be reasonably trivial to learn, while more problematic when trained or tested without that signal. Note that the concatenated model will not be able to transfer to larger numbers of instructions.

4.5. Attentional Architectures

We also consider architectures that use attention to select an instruction to follow as well as deciding whether to switch to a new instruction. We hypothesise that by factoring this out of the policy network, the agent will be capable of generalising to a larger number of instructions, while the smaller, specialised components could allow for easier training.

First, we design an agent that implements the switching logic with a hard attention mechanism, since selecting only one instruction at a time seems appropriate for the given task. Hard attention is often modelled as a discrete decision processes and as such difficult to incorporate with gradient-based optimisation (Bengio et al., 2013; Gülçehre et al., 2016; Jaderberg et al., 2015; Mnih et al., 2014; Xu et al., 2015). We side-step this issue by conditioning the hard attention choice on an observation($\mathbf{x}$) / thumbnail ($\mathbf{t}_i$) similarity metric, and then selecting the most suitable instruction via a generalisation of the max-pooling operator:

$$\mathbf{t}_{i*} = \text{argmax}_{\mathbf{t}_i} \left[\text{softmax}(-||\mathbf{t}_i - \mathbf{x}||_2)\right]$$

This results in a sub-differentiable model which can then be combined with other gradient-based components (Malinowski et al., 2018).

We implement the 'when to switch' logic as follows: when the environment signals the reaching of waypoints to the model, switching becomes a function of that. However, when this is not the case – as in LIST + GOAL REWARD– we use the thumbnail-observation representation similarity to determine whether to switch:

$$i_t = \begin{cases} i^*, & \text{if softmax}(-||\mathbf{t}_{i*} - \mathbf{x}||_2) > \tau \\ i_{t-1}, & \text{otherwise} \end{cases}$$

where the threshold parameter τ is tuned on the validation set. As this component is not trained explicitly, we can in practice train agents by manually switching at waypoint signals and only use the threshold-based switching architecture during evaluation.

Finally, we also use a 'soft' attention mechanism (Yang et al., 2016) that re-weights the representations of instructions instead of selecting just one instruction. We use $h_i = W_a \tanh\left(W_x \mathbf{x} + W_t \mathbf{t}_i\right)$ to compute the unnormalised attention weights. Here, $\mathbf{x} \in \mathbb{R}^{d_1}$ and $\mathbf{t}_i \in \mathbb{R}^{d_2}$ are image and i-th thumbnail representations respectively (although d_1 and d_2 are the same in our architecture, we give the general formulation here). Transformations $W_x \in \mathbb{R}^{k \times d_1}$ and $W_t \in \mathbb{R}^{k \times d_2}$ align both representations, and $W_a \in \mathbb{R}^{1 \times k}$ transforms a vector into a scalar. The normalised weights

$p_i = \text{softmax}(h_i)$ are used to weight the instructions and thumbnails, i.e. $\hat{\boldsymbol{\iota}} = \sum_i p_i \boldsymbol{\iota}_i$, $\hat{\mathbf{t}} = \sum_i p_i \mathbf{t}_i$, with the resultant representation being fed to the policy module.

4.6. Baselines and Ablations

No-Directions We train and evaluate an agent that only takes observations x as input and ignores the instructions, thus establishing a baseline agent that, presumably, will do little more than memorise the training data or perhaps discover exploitable regularities in the environment. We compare NoDir, which uses the waypoint reward signal when available, and NoSignal which does not. The former naturally has more strategies available to exploit the game.

No-Text and No-Thumbnails To establish the relative importance of text instructions and waypoint thumbnails we further consider two variants of the agent where one of these inputs is removed. The NoThumb agent is built on top of the no-attention architecture (Section 4.4), while the NoText version is based on the attentional architecture (Section 4.5).

Student and Teacher Forcing on Ground-truth Labels To obtain the full picture of the performance of our agents, we also consider a supervised baseline that replaces our RL-based framework. Here, we use multinomial regression of each predicted action from the agent's policy to the ground-truth action, similarly to Anderson et al. (2018b); Mei et al. (2016). For every waypoint, we compute the shortest path from the current agent location, and derive the optimal action sequence (turns and forward movements) from this. In Student forcing, the agent samples an action according to the learnt policy, whereas in Teacher forcing, the agent always executes the ground-truth action. Note that the forcing is only done during training, not evaluation. This baseline requires access to ground-truth actions at each time step, and it may overfit more easily.

5. Experiments and Results

Here we describe the training and evaluation of the proposed models on different tasks outlined in Section 3, followed by analysis of the results in Section 5.2.

5.1. Experimental Setup

Training, Validation, and Test In all experiments, four agents are trained for a maximum of 1 billion steps[2]. We use an asynchronous actor-critic framework with importance sampling weights (Espeholt et al., 2018). We choose the best performing agent through evaluation on the validation environment, and report mean and standard deviation over three independent runs on the test environments.

[2]We randomly sample learning rates ($1e{-}4 \leqslant \lambda \leqslant 2.5e{-}4$) and entropy ($5e{-}4 \leqslant \sigma \leqslant 5e{-}3$) for each training run.

Curriculum Training As the StreetNav suite is composed of tasks of increasing complexity, we can use these as a natural curriculum. We first train on the STEP-BY-STEP task, and fine-tune the same agent by continuing to train on the LIST + INCREMENTAL REWARD task. Agents that take waypoint reward signals as input are then evaluated on the LIST + INCREMENTAL REWARD task, while those that do not are evaluated on the LIST + GOAL REWARD task.

Visual and Language Features We train our visual encoder end-to-end together with the whole architecture using 2-layer CNNs. We choose the same visual architecture as Mirowski et al. (2018) for the sake of the comparison to prior work, and since this design decision is computationally efficient. In two alternative setups we use 2048 dimensional visual features from the second-to-last layer of a ResNet (He et al., 2016) pre-trained on ImageNet (Russakovsky et al., 2015) or on the Places dataset (Zhou et al., 2017). However, we do not observe improved results with the pre-trained features. In fact, the learnt ones consistently yield better results. We assume that owing to the end-to-end training and large amount of data provided, agents can learn visual representations better tailored to the task at hand than can be achieved with the pre-trained features. We report our results with learnt visual representations only.

We encode the text instructions using a word-level LSTM, restricting the vocabulary to all words encountered in the training data while removing proper nouns and large numbers. This choice of vocabulary reflects a strategic decision to help the agent attain stronger generalisation across multiple locations. By ignoring road names and learning only the most frequent terms, the agent will learn to understand instructions such as "Turn left at the 2nd intersection" but not "Turn left onto Lafayette Avenue" which would be of little use in a new city. We have experimented with learnt and pre-trained word embeddings and settled on Glove embeddings for convenience (Pennington et al., 2014).

Stochastic MDP To reduce the effect of memorisation, we inject stochasticity into the MDP during training. With a small probability $p = 0.001$ the agent cannot execute its 'move forward' action. This type of stochasticity on the action space resembles Dropout and it acts as a regulariser (Srivastava et al., 2014; Machado et al., 2017).

5.2. Results and Analysis

Table 2 presents the STEP-BY-STEP task, where 'what to switch to' and 'when to switch' are abstracted from the agents. Table 3 contains the LIST + INCREMENTAL REWARD results and contains the main ablation study of this work. Finally, Table 5 shows results on the most challenging

Model	Training	Valid.	Test	
			NYC	Pittsburgh
Random	0.8±0.2	0.8±0.1	0.8±0.2	1.3±0.4
Forward	0.7±0.3	0.2±0.2	0.9±0.1	0.6±0.1
NoSignal	3.5±0.2	1.9±0.6	3.5±0.3	5.2±0.6
NoDir	57.0±1.1	51.6±0.9	41.5±1.2	15.9±1.3
NoText	84.5±0.7	58.1±0.5	47.1±0.7	16.9±1.4
NoThumb	90.7±0.3	67.3±0.9	66.1±0.9	38.1±1.7
Student	94.8±0.9	4.6±1.4	5.5±0.9	0.9±0.2
Teacher	**95.0±0.6**	22.9±2.7	23.9±1.9	8.6±0.9
All-*$^+$	89.6±0.9	69.8±0.4	69.3±0.9	44.5±1.1
Hard-A$^+$	83.5±1.0	**74.8±0.6**	**72.7±0.5**	**46.6±0.8**
Soft-A$^+$	89.3±0.2	67.5±1.4	66.7±1.1	37.2±0.6

Table 2. STEP-BY-STEP (instructions are given one at a time as each waypoint is reached): Percentage of goals reached. Higher is better. ± denotes standard deviation over 3 independent runs of the agent. $^+$: Note that AllConcat and AllSum are equivalent in this setup. Further, the attention components of Soft-A and Hard-A are not used here, but results differ from the All-* agents due to the additional multi-modal projections used in those models.

Model	Training	Valid.	Test	
			NYC	Pittsburgh
NoText	53.5±1.2	43.5±0.5	32.6±2.1	15.8±1.7
NoThumb	69.7±0.7	58.7±2.1	52.4±0.4	**33.9±2.2**
AllConcat	64.5±0.6	**61.3±0.9**	**53.6±1.1**	**33.5±0.2**
AllSum	59.9±0.5	51.1±1.1	41.6±1.0	19.1±1.4
AllSum $_{\text{tuned}}$	**84.4±0.4**	57.7±0.8	48.3±0.9	22.1±0.9
Hard-A	55.4±1.8	51.1±1.1	42.6±0.7	24.0±0.5
Hard-A $_{\text{tuned}}$	62.5±1.1	57.9±0.5	42.9±0.6	22.5±2.0
Soft-A	74.8±0.4	52.2±1.0	43.2±2.2	23.0±0.9
Soft-A $_{\text{tuned}}$	82.7±0.1	57.9±2.1	44.1±1.8	26.6±0.5

Table 3. LIST + INCREMENTAL REWARD: Percentage of goals reached. Higher is better. ± denotes standard deviation over 3 independent runs of the agent. *tuned* denotes agents that were first trained on STEP-BY-STEP and subsequently directly on the LIST + INCREMENTAL REWARD task.

Model	Number of instructions						
	2	3	4	5	6	7	8
NoDir	72.3	59.4	49.2	44.1	31.5	28.0	11.9
NoSignal	5.4	1.4	2.2	0.7	0.1	0.1	0.0
AllSum	68.0	57.9	43.8	37.1	24.0	20.9	9.5
Hard-A	66.7	56.7	48.7	40.4	29.5	29.1	9.2
Hard-A $_{\text{tuned}}$	71.9	**62.9**	52.1	45.7	**32.8**	**33.8**	**15.1**
Soft-A	**75.4**	62.0	46.7	41.7	29.7	26.2	7.3
Soft-A $_{\text{tuned}}$	72.4	62.7	**54.8**	**46.1**	31.0	28.2	12.3

Table 4. Comparison of results on the LIST + INCREMENTAL REWARD task with a larger number of instructions than encountered during training. Number is percentage of goals reached.

LIST + GOAL REWARD task.

The level of difficulty between the three task variants is apparent from the relative scores. While agents reach the goal over 50% of the time in the STEP-BY-STEP task as well as in New York for the LIST + INCREMENTAL REWARD task, this number drops significantly when considering the LIST + GOAL REWARD task. Below we discuss key findings and patterns that warrant further analysis.

NoDir and Waypoint Signalling Even though the NoDir agent has no access to instructions, it performs surprisingly well; on some tasks even on par with other agents. Detailed analysis of the agent behaviour shows that it achieves this performance through a clever strategy: It changes the current direction based on the previous reward given to the agent (signalling a waypoint has been reached). Next, it circles around at the nearest intersection, determines the direction of traffic, and turns into the valid direction. Since many streets in New York are one-way, this strategy works surprisingly well. However, when trained and evaluated without the access to waypoint signal, it fails as expected (NoSignal in Table 2, NoDir in Table 5).

Non- vs. Attentional Architectures As expected, in the STEP-BY-STEP task performance of non-attentional agents is on par with those that have attention architectures (Table 2). However, in LIST + INCREMENTAL REWARD the AllConcat agent has the upper hand over other models (Table 3). Unlike Hard-A, this agent can simultaneously read all available instructions, but at the same time, we believe that the cost of 'mixing' representations in AllConcat is lower than in AllSum or Soft-A. Moreover, Hard-A and AllSum have roughly N times fewer parameters than AllConcat, where N is the number of instructions, which may also explain this gap.

We observe that the Soft-A agent quite closely mirrors the performance of the AllSum agent, and indeed the attention weights suggest that the agent pursues a similar strategy by mixing together all available instructions at a time. Therefore we drop this architecture from further consideration.

As an architectural choice, AllConcat is limited. Unlike the other models it cannot generalise to a larger number of instructions (Table 4). The same agent also fails on the hardest task, LIST + GOAL REWARD (Table 5).

On the LIST + GOAL REWARD, where reward is only available at the goal, Hard-A outperforms the other models, mirroring its superior performance when increasing the number

Model	Training	Valid.	Test	
			NYC	Pittsburgh
NoDir	3.5±0.2	1.9±0.6	3.5±0.3	5.2±0.6
AllConcat	**23.0**±**0.8**	7.4±0.2	11.3±1.8	9.3±0.8
AllSum $_{\text{step}}$	6.7±1.1	4.1±0.3	5.6±0.1	7.2±0.4
AllSum $_{\text{list}}$	13.2±2.0	3.2±0.3	6.7±1.1	5.3±1.2
Hard-A $_{\text{step}}$	18.5±1.3	13.8±1.3	**17.3**±**1.2**	**12.1**±**1.1**
Hard-A $_{\text{list}}$	21.9±2.8	**14.2**±**0.4**	16.9±1.2	10.0±1.3

Table 5. LIST + GOAL REWARD: Percentage of goals reached. Higher is better. ± denotes standard deviation over 3 independent runs of the agent. Switching thresholds for the attention agent are tuned on the validation data. *step* and *list* denote whether the agents were trained in the STEP-BY-STEP or LIST + INCREMENTAL REWARD setting.

of instructions. This underlines our motivation for this architecture in decoupling the instruction following from the instruction selection aspect of the problem.

Supervised vs RL agents While the majority of our agents use RL, Student and Teacher are trained with a dense signal of supervision (Section 4.6). The results in Table 2 show that the supervised agents can fit the training distribution, but end up generalising poorly. We attribute this lack of robustness to the highly supervised nature of their training, where the agents are never explicitly exposed to the consequences of their actions during training and hence never deviate from the gold path. Moreover, the signal of supervision for Student turns the agent whenever it makes a mistake, and in our setting each error is catastrophic.

Transfer We evaluate the transfer capabilities of our agents in a number of ways. First, by training, validating and testing the agents in different parts of Manhattan, as well as testing the agents in Pittsburgh. The RL agents all transfer reasonably well within Manhattan and to a lesser extent to Pittsburgh. The drop in performance there can mostly be attributed both to different visual features across the two cities and a more complex map (see Figure 3).

As discussed earlier, we also investigated the performance of our agents on a task with longer lists of directions than observed during training (Table 4). The declining numbers highlight the cost of error propagation as a function of the number of directions.

NoText and NoThumb As agents have access to both thumbnail images and written instructions to guide them to their goal, it is interesting to compare the performance of two agents that only use either one of these two inputs. The NoThumb agent consistently performs better than the NoText agent, which suggests that language is the key component of the instructions. Also note how NoThumb, which is based on the AllConcat architecture, effectively matches that agent's performance across all tasks, suggesting that thumbnails can largely be ignored for the success of the agents. That being said, NoText outperforms the directionless baseline (NoDir), meaning that the thumbnails by themselves also carry enough information to be of value.

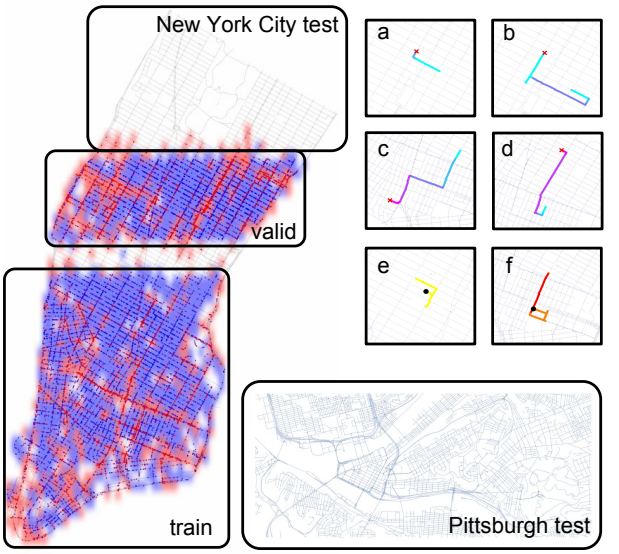

Figure 3. Left: Map of Manhattan with training, validation and test areas, overlaid with the heat map of goal locations reached (blue) or missed (red) on training and validation data, using a Hard-A agent trained on the List + Incremental Reward task. Bottom right: Pittsburgh area used for testing. Top right: trajectories with color-coded attention index predicted by a Hard-A agent with a learned switcher and trained on the List + Goal Reward task; we show successful trajectories on validation (a and b) and on training (c and d) data, as well as two trajectories with missed goal (e and f).

6. Conclusions

Generalisation poses a critical challenge to deep RL approaches to navigation, but we believe that the common medium of language can act as a bridge to enable strong transfer in novel environments. We have presented a new language grounding and navigation task that uses realistic images of real places, together with real (if not natural) language-based directions for this purpose.

Aside from the StreetNav environment and related analyses, we have proposed a number of models that can act as strong baselines on the task. The hard-attention mechanism that we have employed is just one instantiation of a more general idea which can be further explored. Given the gap between reported and desired agent performance here, we acknowledge that much work is still left to be done.

References

Anderson, P., Chang, A., Chaplot, D. S., Dosovitskiy, A., Gupta, S., Koltun, V., Kosecka, J., Malik, J., Mottaghi, R., Savva, M., et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018a.

Anderson, P., Wu, Q., Teney, D., Bruce, J., Johnson, M., Sünderhauf, N., Reid, I., Gould, S., and van den Hengel, A. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2018b.

Armeni, I., Sener, O., Zamir, A. R., Jiang, H., Brilakis, I., Fischer, M., and Savarese, S. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Beattie, C., Leibo, J. Z., Teplyashin, D., Ward, T., Wainwright, M., Küttler, H., Lefrancq, A., Green, S., Valdés, V., Sadik, A., Schrittwieser, J., Anderson, K., York, S., Cant, M., Cain, A., Bolton, A., Gaffney, S., King, H., Hassabis, D., Legg, S., and Petersen, S. Deepmind Lab. *arXiv preprint arXiv:1612.03801*, 2016.

Bengio, Y., Léonard, N., and Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Boroditsky, L. Does language shape thought?: Mandarin and english speakers' conceptions of time. *Cognitive Psychology*, 43(1):1–22, 2001.

Brodeur, S., Perez, E., Anand, A., Golemo, F., Celotti, L., Strub, F., Rouat, J., Larochelle, H., and Courville, A. HoME: A household multimodal environment. *arXiv preprint arXiv:1711.11017*, 2017.

Chang, A., Dai, A., Funkhouser, T., Halber, M., Nießner, M., Savva, M., Song, S., Zeng, A., and Zhang, Y. Matterport3d: Learning from RGB-d data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017.

Chaplot, D. S., Sathyendra, K. M., Pasumarthi, R. K., Rajagopal, D., and Salakhutdinov, R. Gated-attention architectures for task-oriented language grounding. 2017.

Chen, H., Shur, A., Misra, D., Snavely, N., and Artzi, Y. Touchdown: Natural language navigation and spatial reasoning in visual street environments. *arXiv preprint arXiv:1811.12354*, 2018.

Cirik, V., Zhang, Y., and Baldridge, J. Following formulaic map instructions in a street simulation environment. *Visually Grounded Interaction and Language (ViGIL) Workshop, NeurIPS*, 2018.

Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T. A., and Nießner, M. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pp. 10, 2017.

Das, A., Datta, S., Gkioxari, G., Lee, S., Parikh, D., and Batra, D. Embodied question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 5, pp. 6, 2018.

de Vries, H., Shuster, K., Batra, D., Parikh, D., Weston, J., and Kiela, D. Talk the walk: Navigating New York City through grounded dialogue. *arXiv preprint arXiv:1807.03367*, 2018.

Dhiman, V., Banerjee, S., Griffin, B., Siskind, J. M., and Corso, J. J. A critical investigation of deep reinforcement learning for navigation. *arXiv preprint arXiv:1802.02274*, 2018.

Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2625–2634, 2015.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., Legg, S., and Kavukcuoglu, K. IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. *arXiv preprint arXiv:1802.01561*, 2018.

Fried, D., Hu, R., Cirik, V., Rohrbach, A., Andreas, J., Morency, L.-P., Berg-Kirkpatrick, T., Saenko, K., Klein, D., and Darrell, T. Speaker-follower models for vision-and-language navigation. *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Gopnik, A. and Meltzoff, A. N. Semantic and cognitive development in 15-to 21-month-old children. *Journal of child language*, 11(3):495–513, 1984.

Gülçehre, Ç., Chandar, S., Cho, K., and Bengio, Y. Dynamic neural Turing machine with soft and hard addressing schemes. *arXiv preprint arXiv:1607.00036*, 2016.

Harnad, S. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1-3):335–346, 1990.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Hermann, K. M., Hill, F., Green, S., Wang, F., Faulkner, R., Soyer, H., Szepesvari, D., Czarnecki, W. M., Jaderberg, M., Teplyashin, D., Wainwright, M., Apps, C., Hassabis, D., and Blunsom, P. Grounded language learning in a simulated 3d world. *arXiv preprint arXiv:1706.06551*, 2017.

Hill, F., Hermann, K. M., Blunsom, P., and Clark, S. Understanding grounded language learning agents. *arXiv preprint arXiv:1710.09867*, 2017.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

Hoff, E. *Language development*. Cengage Learning, 2013.

Hudson, D. A. and Manning, C. D. Compositional attention networks for machine reasoning. *arXiv preprint arXiv:1803.03067*, 2018.

Jaderberg, M., Simonyan, K., Zisserman, A., and Kavukcuoglu, K. Spatial transformer networks. In *Advances in Neural Information Processing systems (NIPS)*, pp. 2017–2025, 2015.

Jaderberg, M., Mnih, V., Czarnecki, W. M., Schaul, T., Leibo, J. Z., Silver, D., and Kavukcuoglu, K. Reinforcement learning with unsupervised auxiliary tasks. *International Conference on Learning Representations*, 2017.

Johnson, J., Karpathy, A., and Fei-Fei, L. Densecap: Fully convolutional localization networks for dense captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4565–4574, 2016.

Kay, P. and Kempton, W. What is the Sapir-Whorf hypothesis? *American Anthropologist*, 86(1):65–79, 1984.

Kempka, M., Wydmuch, M., Runc, G., Toczek, J., and Jaśkowski, W. VizDoom: A Doom-based AI research platform for visual reinforcement learning. In *IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 1–8. IEEE, 2016.

Kolve, E., Mottaghi, R., Gordon, D., Zhu, Y., Gupta, A., and Farhadi, A. AI2-THOR: An interactive 3d environment for visual AI. *arXiv preprint arXiv:1712.05474*, 2017.

Kong, C., Lin, D., Bansal, M., Urtasun, R., and Fidler, S. What are you talking about? Text-to-image coreference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3558–3565, 2014.

Krishnamurthy, J. and Kollar, T. Jointly learning to parse and perceive: Connecting natural language to the physical world. *Transactions of the Association for Computational Linguistics*, 1:193–206, 2013.

LeCun, Y. A., Bottou, L., Orr, G. B., and Müller, K.-R. Efficient backprop. In *Neural networks: Tricks of the trade*, pp. 9–48. Springer, 2012.

Machado, M. C., Bellemare, M. G., Talvitie, E., Veness, J., Hausknecht, M., and Bowling, M. Revisiting the Arcade Learning Environment: Evaluation protocols and open problems for general agents. *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, (IJCAI)*, 2017.

Malinowski, M. and Fritz, M. A multi-world approach to question answering about real-world scenes based on uncertain input. In *Advances in Neural Information Processing systems (NIPS)*, pp. 1682–1690, 2014.

Malinowski, M., Rohrbach, M., and Fritz, M. Ask your neurons: A deep learning approach to visual question answering. *International Journal of Computer Vision*, 125(1-3):110–135, 2017.

Malinowski, M., Doersch, C., Santoro, A., and Battaglia, P. Learning visual question answering by bootstrapping hard attention. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 3–20, 2018.

Matuszek, C., FitzGerald, N., Zettlemoyer, L., Bo, L., and Fox, D. A joint model of language and perception for grounded attribute learning. *Proceedings of the International Conference on Machine Learning (ICML)*, 2012.

Mei, H., Bansal, M., and Walter, M. R. Listen, attend, and walk: Neural mapping of navigational instructions to action sequences. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

Mirowski, P., Pascanu, R., Viola, F., Soyer, H., Ballard, A. J., Banino, A., Denil, M., Goroshin, R., Sifre, L., Kavukcuoglu, K., Kumaran, D., and Hadsell, R. Learning to navigate in complex environments. *International Conference on Learning Representations (ICLR)*, 2017.

Mirowski, P., Grimes, M. K., Malinowski, M., Hermann, K. M., Anderson, K., Teplyashin, D., Simonyan, K., Kavukcuoglu, K., Zisserman, A., and Hadsell, R. Learning to navigate in cities without a map. *Advances In Neural Information Processing Systems (NeurIPS)*, 2018.

Mnih, V., Heess, N., Graves, A., and Kavukcuoglu, K. Recurrent models of visual attention. In *Advances in Neural Information Processing systems (NIPS)*, pp. 2204–2212, 2014.

Mo, K., Li, H., Lin, Z., and Lee, J.-Y. The AdobeIndoorNav dataset: Towards deep reinforcement learning based real-world indoor robot visual navigation. *arXiv preprint arXiv:1802.08824*, 2018.

Oh, J., Chockalingam, V., Singh, S., and Lee, H. Control of memory, active perception, and action in Minecraft. *International Conference on Machine Learning (ICML)*, 2016.

Oh, J., Singh, S., Lee, H., and Kohli, P. Zero-shot task generalization with multi-task deep reinforcement learning. *International Conference on Machine Learning (ICML)*, 2017.

Parisotto, E., Chaplot, D. S., Zhang, J., and Salakhutdinov, R. Global pose estimation with an attention-based recurrent network. *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops*, 2018.

Park, D. H., Hendricks, L. A., Akata, Z., Rohrbach, A., Schiele, B., Darrell, T., and Rohrbach, M. Multimodal explanations: Justifying decisions and pointing to the evidence. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Pennington, J., Socher, R., and Manning, C. D. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.

Perez, E., Strub, F., De Vries, H., Dumoulin, V., and Courville, A. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

Reed, S., Akata, Z., Schiele, B., and Lee, H. Learning deep representations of fine-grained visual descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Ren, M., Kiros, R., and Zemel, R. Exploring models and data for image question answering. In *Advances in Neural Information Processing systems (NIPS)*, pp. 2953–2961, 2015.

Rohrbach, A., Rohrbach, M., Tang, S., Joon Oh, S., and Schiele, B. Generating descriptions with grounded and co-referenced people. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4979–4989, 2017.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. S., Berg, A. C., and Li, F. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

Savva, M., Chang, A. X., Dosovitskiy, A., Funkhouser, T., and Koltun, V. MINOS: Multimodal indoor simulator for navigation in complex environments. *arXiv preprint arXiv:1712.03931*, 2017.

Shah, P., Fiser, M., Faust, A., Kew, J. C., and Hakkani-Tur, D. FollowNet: Robot navigation by following natural language directions with deep reinforcement learning. *arXiv preprint arXiv:1805.06150*, 2018.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research (JMLR)*, 15(1):1929–1958, 2014.

Tellex, S., Kollar, T., Dickerson, S., Walter, M. R., Banerjee, A. G., Teller, S., and Roy, N. Understanding natural language commands for robotic navigation and mobile manipulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 1507–1514. AAAI Press, 2011.

Teney, D., Anderson, P., He, X., and Hengel, A. v. d. Tips and tricks for visual question answering: Learnings from the 2017 challenge. *arXiv preprint arXiv:1708.02711*, 2017.

Vogel, A. and Jurafsky, D. Learning to follow navigational directions. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 806–814. Association for Computational Linguistics, 2010.

Wayne, G., Hung, C., Amos, D., Mirza, M., Ahuja, A., Grabska-Barwinska, A., Rae, J. W., Mirowski, P., Leibo, J. Z., Santoro, A., Gemici, M., Reynolds, M., Harley, T., Abramson, J., Mohamed, S., Rezende, D. J., Saxton, D., Cain, A., Hillier, C., Silver, D., Kavukcuoglu, K., Botvinick, M., Hassabis, D., and Lillicrap, T. P. Unsupervised predictive memory in a goal-directed agent. *arXiv preprint arXiv:1803.10760*, 2018.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Wu, Y., Wu, Y., Gkioxari, G., and Tian, Y. Building generalizable agents with a realistic and rich 3d environment. *arXiv preprint arXiv:1801.02209*, 2018.

Xia, F., Zamir, A. R., He, Z., Sax, A., Malik, J., and Savarese, S. Gibson Env: Real-world perception for embodied agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9068–9079, 2018.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., and Bengio, Y. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning (ICML)*, pp. 2048–2057, 2015.

Yan, C., Misra, D., Bennnett, A., Walsman, A., Bisk, Y., and Artzi, Y. CHALET: Cornell house agent learning environment. *arXiv preprint arXiv:1801.07357*, 2018.

Yang, Z., He, X., Gao, J., Deng, L., and Smola, A. Stacked attention networks for image question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21–29, 2016.

Zhang, J., Tai, L., Boedecker, J., Burgard, W., and Liu, M. Neural SLAM. *arXiv preprint arXiv:1706.09520*, 2017.

Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., and Torralba, A. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., and Fidler, S. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 19–27, 2015.

Zhu, Y., Mottaghi, R., Kolve, E., Lim, J. J., Gupta, A., Fei-Fei, L., and Farhadi, A. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pp. 3357–3364. IEEE, 2017.