Grounding Language to Entities and Dynamics
for Generalization in Reinforcement Learning

Austin W. Hanjie [1] **Victor Zhong** [2] **Karthik Narasimhan** [1]

Abstract

We investigate the use of natural language to drive the generalization of control policies and introduce the new multi-task environment MESSENGER with free-form text manuals describing the environment dynamics. Unlike previous work, MESSENGER does not assume prior knowledge connecting text and state observations — the control policy must simultaneously ground the game manual to entity symbols and dynamics in the environment. We develop a new model, EMMA (Entity Mapper with Multi-modal Attention) which uses an entity-conditioned attention module that allows for selective focus over relevant descriptions in the manual for each entity in the environment. EMMA is end-to-end differentiable and learns a latent grounding of entities and dynamics from text to observations using only environment rewards. EMMA achieves successful zero-shot generalization to unseen games with new dynamics, obtaining a 40% higher win rate compared to multiple baselines. However, win rate on the hardest stage of MESSENGER remains low (10%), demonstrating the need for additional work in this direction.

1. Introduction

Interactive game environments are useful for developing agents that learn grounded representations of language for autonomous decision making (Golland et al., 2010; Branavan et al., 2011; Andreas & Klein, 2015; Bahdanau et al., 2018). The key objective in these environments is learning to interpret language specifications by relating entities and dynamics of the environment (i.e. how entities behave) to their corresponding references in the text, in order to effectively and efficiently win new settings with previously unseen entities or dynamics (Narasimhan et al., 2018; Zhong et al., 2020). While existing methods demonstrate successful transfer to new settings, they assume a ground-truth mapping between individual entities and their textual references.

We introduce MESSENGER,[1] an environment which features multiple game variants with differing dynamics and accompanying text manuals in English for each. The manuals contain descriptions of the entities and world dynamics obtained through crowdsourced human writers. Crucially, while prior work assumes a ground truth mapping (e.g. the word 'knight' in the manual refers to the entity name 'knight' in the observation), MESSENGER does not contain prior signals that map between text and state observations (e.g. between the phrase 'mounted warrior is fleeing' and the symbol 🏇 moving away from the agent). To succeed in MESSENGER, an agent must relate entities and dynamics of the environment to their references in the natural language manual using only scalar reward signals from the environment. The overall game mechanics of MESSENGER involve obtaining a *message* and delivering it to a *goal*. For instance, in game 1 of Figure 1, the agent must read the manual to:

1. Identify the entity that holds the message. In this case, description 2 (d-2) reveals that it is with the `thief` but there is an identical entity that is an enemy (d-4).
2. Map d-2 and d-4 to the correct symbols in the observation (green-cloaked person).
3. Observe the movement patterns of the two entities ('heading closer' vs. 'rushing away') to disambiguate which of the two entities holds the message.
4. Pick up the message from the entity that holds it.
5. Identify the entity that is the goal. Here, d-3 and d-6 reference a goal. It must realize that there is no 'canine' that is 'running away' and so d-6 must be a distractor, and a `mage` must be the goal.
6. Follow a similar procedure to 3 to disambiguate which `mage` is the goal and which is the enemy (d-3 vs d-5).
7. Bring the message to the goal.

To ground entities and dynamics to their corresponding

---

[1]Available at: https://github.com/ahjwang/messenger-emma

GAME 1 MANUAL

1. at a particular locale, there exists a motionless mongrel that is a formidable adversary.
2. the top-secret paperwork is in the crook's possession, and he's heading closer and closer to where you are.
3. the crucial target is held by the wizard and the wizard is fleeing from you.
4. the mugger rushing away is the opposition posing a serious threat.
5. the thing that is not able to move is the mage who possesses the enemy that is deadly.
6. *the vital goal is found with the canine, but it is running away from you.*

Figure 1. Two games from our multi-task environment MESSENGER where the agent must obtain the message and deliver it to the goal (white dotted lines). Within a single game, the same entities (e.g. mage) with different roles (e.g. enemy, goal) must be disambiguated by their dynamics (e.g. immovable, fleeing). The same entities may have different roles in different games forcing the agent to consult the manual to succeed consistently. Note the extraneous description (*italics*) and multiple synonyms for entities and roles (e.g. mage, wizard; adversary, opposition). Unlike prior work, the mapping from words in the manual to game entities is not available and must be learned using only scalar game rewards.

references in the manual, we develop a new model called EMMA (Entity Mapper with Multi-modal Attention). EMMA simultaneously learns to select relevant sentences in the manual for each entity in the game as well as incorporate the corresponding text description into its control policy. This is done using a multi-modal attention mechanism which uses entity representations as queries to attend to specific tokens in the manual text. EMMA then generates a text-conditioned representation for each entity which is processed further by a deep neural network to generate a policy. We train the entire model in a multi-task fashion using reinforcement learning to maximize task returns.

Our experiments demonstrate EMMA outperforms multiple baselines (language-agnostic, attention-ablated, and Bayesian attention) and an existing state of the art model (Zhong et al., 2020) — on unseen games (i.e. a zero-shot test), EMMA achieves more than 40% higher win rates. However, while EMMA can effectively map text references to their corresponding entity symbols in observation

space, its ability to disambiguate descriptions by grounding language to entity movement dynamics is lacking, and win rates on the test games for the hardest stage of MESSENGER remains low for all models evaluated ($\leq 10\%$), demonstrating the challenging nature of grounding natural language to dynamics using only interactive (reward-based) feedback.

In summary, our paper makes two key contributions: (1) a multi-task environment with novel challenges including a) learning entity symbol grounding from scratch in a multi-task setup with b) realistic, crowd-sourced text and (2) an attention-based model that is able to learn such a grounding where prior approaches struggle. We hope MESSENGER and EMMA will further enable the development of new models and learning algorithms for language grounding.

2. Related Work

Grounding for Instruction Following   Grounding natural language to policies has been explored in the context of instruction following in tasks like navigation (Chen & Mooney, 2011; Hermann et al., 2017; Fried et al., 2018; Wang et al., 2019; Daniele et al., 2017; Misra et al., 2017; Janner et al., 2018), games (Golland et al., 2010; Reckman et al., 2010; Andreas & Klein, 2015; Bahdanau et al., 2018; Küttler et al., 2020) or robotic control (Walter et al., 2013; Hemachandra et al., 2014; Blukis et al., 2019) (see Luketina et al. (2019) and Tellex et al. (2020) for more detailed surveys). Recent work has explored several methods for enabling generalization in instruction following, including environmental variations (Hill et al., 2020a), memory structures (Hill et al., 2020c) and pre-trained language models (Hill et al., 2020b). In a slightly different setting, Co-Reyes et al. (2019) use incremental guidance, where the text input is provided online, conditioned on the agent's progress in the environment. Andreas et al. (2017) developed an agent that can use sub-goal specifications to deal with sparse rewards. Oh et al. (2017) use sub-task instructions and hierarchical reinforcement learning to complete tasks with long action sequences.

In all these works, the text conveys the goal to the agent (e.g. 'move forward five steps'), thereby encouraging a direct connection between the instruction and the control policy. This tight coupling means that any grounding learned by the agent is likely to be tailored to the types of tasks seen in training, making generalization to a new distribution of dynamics or tasks challenging. In extreme cases, the agent may even function without acquiring an appropriate grounding between language and observations (Hu et al., 2019). In our setup, we assume that the text only provides high-level guidance without directly describing the correct actions for every game state.

Language Grounding by Reading Manuals A different line of work has explored the use of language as an auxiliary source of knowledge through text manuals. These manuals provide useful descriptions of the entities in the world and their dynamics (e.g. how they move or interact with other entities) that are optional for the agent to make use of and do not directly reveal the actions it has to take. Branavan et al. (2011) developed an agent to play the game of Civilization more effectively by reading the game manual. They make use of dependency parses and predicate labeling to construct feature-based representations of the text, which are then used to construct the action-value function used by the agent. Our method does not require such feature construction. Narasimhan et al. (2018) and Zhong et al. (2020) used text descriptions of game dynamics to learn policies that generalize to new environments, without requiring feature engineering. However, these works assume some form of initial grounding provided to the agent (e.g. a mapping between entity symbols and their descriptions, or the use of entity names in text as state observations). In contrast, MESSENGER requires that this fundamental mapping between entity symbols in observation space and their text references be learned entirely through interaction with the environment.

3. Preliminaries

Our objective is to demonstrate grounding of environment dynamics and entities for generalization to unseen environments. An `entity` is an object represented as a symbol in the observation. Dynamics refer to how entities behave in the environment including how they interact with the agent. Notably, movement dynamics are the frame-to-frame position changes exhibited by entities (e.g. fleeing).

Environment We model decision making in each environment as a Partially-Observable Markov Decision Process (POMDP) with the 8-tuple (S, A, O, P, R, E, Z, M). S and O are the set of all states and observations respectively where each $o \in O$ contains entities from the set of entities E. At each step t, the agent takes some action $a_t \in A$. $P(s_{t+1}|s_t, a_t)$ is the transition distribution over all possible next states s_{t+1} conditioned on the current state s_t and action a_t. $R(s_t, a_t, s_{t+1})$ is a function that provides the agent with a reward $r_t \in \mathbb{R}$ for action a_t and transition from s_t to s_{t+1}. Z is a set of text descriptions, with each $z \in Z$ providing information about an entity $e \in E$. M is the map $z_e \mapsto e$ which identifies the entity that each description describes. M, P, and R are not available to the agent. Note that there might not be a one-to-one mapping between Z and entities in the current state observation.

Reinforcement Learning (RL) The objective of the agent is to find a policy $\pi : O \rightarrow A$ to maximize its cu-

mulative reward in an episode. If π is parameterized by θ, standard deep RL approaches optimize θ to maximize the expected reward of following π_θ. In our setup, we want the agent to learn a policy $\pi_\theta(a|o, Z)$ that conditions its behavior on the provided text. However, in contrast to previous work (Narasimhan et al., 2018; Zhong et al., 2020), M is not available to our agent and must be learned.

Differentiating Entities, Roles, and Text References For ease of exposition, we use type face to differentiate between `entity symbols`, *roles*, and 'text references'. For example, `plane` refers to the entity ✈, where 'plane' and 'aircraft' are text references to `plane`. Additionally, `plane` can take on the role of an *enemy*.

4. MESSENGER

We require an environment where grounding text descriptions Z to dynamics and learning the mapping M for all entities in E is necessary to obtain a good reward. Moreover, there must be enough game instances of the environment to induce the mapping M.

With these requirements in mind, we devise a new multi-task environment MESSENGER using the Py-VGDL framework (Schaul, 2013). In MESSENGER, each entity can take on one of three roles: an *enemy*, *message*, or *goal*. The agent's objective is to bring the message to the goal while avoiding the enemies. If the agent encounters an enemy at any point in the game, or the goal without first obtaining the message, it loses the game and obtains a reward of -1. Rewards of 0.5 and 1 are provided for obtaining and delivering the message to the goal respectively.[2] There are twelve different entities and three possible movement types: stationary, chasing, or fleeing. Each set of entity-role assignments (henceforth referred to as a game) is initialized on a 10×10 grid. The agent can navigate via `up`, `down`, `left`, `right`, and `stay` actions and interacts with another entity when both occupy the same cell.

The same set of entities with the same movements may be assigned different roles in different games. Thus, two games may have identical observations but differ in the reward function R (which is not available to the agent) and the text manual Z (which is available). Thus, our agent must learn to extract information from Z to succeed consistently. Some game examples are presented in Figure 1.

Grounding Entities MESSENGER requires agents to learn M without priors connecting state observations O to descriptions Z. Aside from using independent entity symbols disjoint from the text vocabulary, the set of training

---

[2]We find that our approach also works with sparser terminal ± 1 rewards (Fig. 8, Appendix).
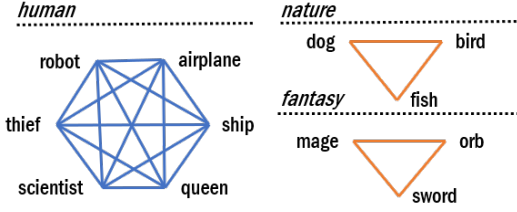
Figure 2. Entities and their subdivision into human, nature and fantasy sub-worlds. Each K_3 subgraph is a combination of entities that may appear during training.

games is designed such that simple co-occurrence statistics between entity and text do not completely reveal M.

Consider when every possible combination of entities is observed during training. Then, for an entity e, its symbol in the observation (e.g. `plane`) is the only one that always appears together with its text references (e.g. 'aircraft'). This tight coupling provides an inherent bias towards the correct grounding without needing to act in the environment. We denote such a set of games where each entity can appear with every other entity as **multi-combination (MC)**.

The MC assumption may not always be realistic in practice — some entities are very unlikely to appear together (e.g. `plane`, `thief`, `sword`) while others may co-occur exclusively with each other (e.g. `mage`, `orb`, `sword`). We denote games in which the same entities always appear together as **single-combination (SC)**. For SC games, every text symbol in the manual (e.g. 'mage', 'enemy', 'the', etc.) co-occurs the same number of times with all entity symbols in the observation. For example, if the entity symbols ◯ and ■ always appear simultaneously with both text symbols 'mage' and 'sword', it is impossible to map 'mage' to ◯ without interacting with the entities. That is, co-occurrences between entity and text symbols provide no information about M and **the agent must ground these entities entirely via interaction**. To learn M for the entities in this example, the agent must interact with ◯ and if it obtains the message from it, it must infer from the description 'The mage has the message' that ◯ must be a 'mage'.

We divide the entities in MESSENGER into human, nature, and fantasy sub-worlds (Fig. 2) and exclude from training any games in which entities from different sub-world appear together. In particular, the nature and fantasy subworlds form SC and the human subworld forms the MC games.

Grounding Dynamics To force agents to distinguish varying movement dynamics, multiple copies of the same entity with different roles in MESSENGER may exhibit different movement patterns. For example, within the same game there may be descriptions: (1) 'the chasing mage is an enemy' and (2) 'the fleeing mage is the goal'. This

means that even after grounding words such as 'mage' to its corresponding entity symbol, the agent must additionally consider the position of e through a sequence of observations $o_{t-k}, ..., o_t$ in order to find the correct description z_e.

Text Descriptions We collected 5,316 unique free-form entity descriptions in English via Amazon Mechanical Turk (Buhrmester et al., 2016) by asking workers to paraphrase prompt sentences. To increase the diversity of responses, the prompts were themselves produced from 82 crowdsourced templates. When constructing the prompts, we inject multiple synonyms for each entity. Workers further paraphrased these synonyms, resulting in multiple ways to describe the same entity (e.g. 'airplane', 'jet', 'flying machine', 'aircraft', 'airliner'). Furthermore, we observe responses with multiple sentences per description, typos ('plane' vs 'plan') and the need to disambiguate similar words ('flying machine', 'winged creature'). Each training manual consists of a set of descriptions with an average total length of 30 - 60 words depending on the level. The total vocabulary size of the descriptions is 1,125. Besides lower-casing the worker responses, we do not do any preprocessing. Example descriptions can be found in Fig. 1. Further details regarding data collection can be found in appendix A.

Train-Evaluation Split We ensure that any assignment of an entity to the roles *message* or *goal* in the evaluation games never appears during training (e.g. if e is the *goal* in evaluation, no e is ever the *goal* in any training game). This forces models to make compositional entity-role generalizations to succeed on the evaluation games. In total we have 44 training, 32 validation, and 32 test games. We train on 2,863 of the text descriptions and reserve 1,227 and 1,226 for validation and testing respectively.

Comparison with Previous Environments We chose to realize MESSENGER in a grid-world as it allows us to (1) study generalization to rich sets of procedurally generated dynamics, (2) conduct controlled studies of co-occurrence statistics (SC, vs. MC) and (3) explicitly verify the learned groundings with well-defined, discrete entities (see Fig. 6).

Other grid-worlds used to study language grounding include RTFM (Zhong et al., 2020), BabyAI (Chevalier-Boisvert et al., 2019) and Narasimhan et al. (2018). An oracle is used in Narasimhan et al. (2018) to concatenate the text representation to its corresponding entity representation. Access to such an oracle is a strong assumption in the wild and eliminates the need to ground the entities altogether.

In RTFM, the observation is a grid of text in which entity names are lexically identical to their references in the manual (e.g. 'plane'). The key challenge unique to MESSENGER is learning to map between the observed entity symbol (e.g. ✈) and its natural language references in the

manual (e.g. 'aircraft'). Furthermore, RTFM is a **MC** environment which may simplify the grounding problem. Both Narasimhan et al. (2018) and Zhong et al. (2020) do not consider disambiguation by grounding movement dynamics, whereas agents in Messenger need to distinguish entities based on how they move (e.g. fleeing, chasing).

Unlike previous work on language grounding in grid environments (Zhong et al., 2020; Chevalier-Boisvert et al., 2019), we do not use templated or rule-generated text. RTFM uses a small number of rule-based templates to construct each manual, and each entity is referred to in a single way (e.g. `goblin` is always 'goblin'). In contrast, MESSENGER features thousands of hand-written descriptions and each entity may be referenced in multiple ways. For further comparisons of RTFM and MESSENGER, including why we do not simply extend RTFM, please see Appendix B.1.

5. The EMMA Model

As we saw in the previous section, an agent must learn to map entities to their corresponding references in the natural language manual in order to perform well in MESSENGER. To learn this mapping, we develop a new model, EMMA (Entity Mapper with Multi-modal Attention), which employs a soft-attention mechanism over the text descriptions. At a high level, for each entity description, EMMA first generates key and value vectors from their respective token embeddings obtained using a pretrained language model. Each entity attends to the descriptors via a symbol embedding that acts as the attention query. Then, instead of representing each entity with its embedding, we use the resulting attention-scaled values as a proxy for the entity. This approach helps our model learn a control policy that focuses on entity roles (e.g. *enemy, goal*) while using the entities' identity (e.g. `queen`, `mage`) to selectively read the text. We describe each component of EMMA below and in Figure 3.

Text Encoder Our input consists of a $h \times w$ grid observation $o \in O$ with a set of entity descriptions Z. We encode each description $z \in Z$ using a BERT-base model whose parameters are fixed throughout training (Devlin et al., 2019; Wolf et al., 2019). For a description z, let $t_1, ..., t_n$ be its token embeddings generated by our encoder. We obtain key and value vectors k_z, v_z, where σ is the softmax function:

$$k_z = \sum_{i=1}^{n} \alpha_i W_k t_i + b_k \qquad \alpha = \sigma\left((u_k \cdot t_j)_{j=1}^n\right) \quad (1)$$

$$v_z = \sum_{i=1}^{n} \beta_i W_v t_i + b_v \qquad \beta = \sigma\left((u_v \cdot t_j)_{j=1}^n\right) \quad (2)$$

The key and value vectors are simply linear combinations of $W_k t_i + b_k$ and $W_v t_i + b_v$ with weights α, β respectively, where W_k, W_v are matrices which transform each token
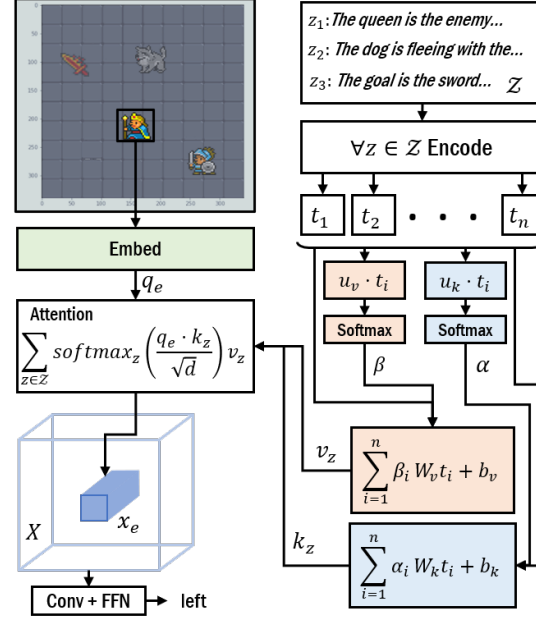

Figure 3. Schematic of our model EMMA, which creates a representation for entities using multi-modal attention over the observations and text manual. Mechanisms for the key, query, and value are shaded in blue, green, and red respectively.

to d dimensions and b_k, b_v are biases. The weights α, β are obtained by taking the softmax over the dot products $(u_k \cdot t_j)_{j=1}^n$ and $(u_v \cdot t_j)_{j=1}^n$ respectively. These weights imbue our model with the ability to focus on relevant tokens. All of $W_k, b_k, u_k, W_v, b_v, u_v$ are learned parameters.

Entity Representation Generator To get a representation for each entity e, we embed its symbol into a query vector q_e of dimension d to attend to the descriptions $z \in Z$ with their respective key and value vectors k_z, v_z. We use scaled dot-product attention (Vaswani et al., 2017) and denote the resulting representation for the entity e as x_e:

$$x_e = \sum_{i=1}^{m} \gamma_i v_{z_i} \qquad \gamma = \sigma\left(\left(\frac{q_e \cdot k_{z_j}}{\sqrt{d}}\right)_{j=1}^m\right) \quad (3)$$

where $m = |Z|$ is the number of descriptions in the manual. This mechanism allows EMMA to accomplish two forms of language grounding: the key and query select relevant descriptions for each object by matching entities to names (e.g. 'mage'), and the value extracts information relevant to the entities' behaviors in the world (e.g. *enemy, chasing*).

For each entity e in the observation, we place its representation x_e into a tensor $X \in \mathbb{R}^{h \times w \times d}$ at the same coordinates as the entity position in the observation o to maintain full spatial information. The representation for the agent is simply a learned embedding of dimension d.

Action Module To provide temporal information that assists with grounding movement dynamics, we concatenate the outputs of the representation generator from the three most recent observations to obtain a tensor $X' \in \mathbb{R}^{h \times w \times 3d}$. To get a distribution over the actions $\pi(a|o, Z)$, we run a 2D convolution on X' over the h, w dimensions. The flattened feature maps are passed through a fully-connected FFN terminating in a softmax over the possible actions.

$$y = \text{Flatten}\big(\text{Conv2D}(X')\big)$$
$$\pi(a|o, Z) = \sigma\big(\text{FFN}(y)\big) \tag{4}$$

In contrast to previous approaches that use global observation features to read the manual (Zhong et al., 2020), we build a text-conditioned representation for each entity (x_e). One advantage is that x_e can directly replace the entity embeddings typically used to embed the state observation in most models while still being completely end-to-end differentiable.

While designed for grid environments, our approach can be extended to more complex visual inputs by using CNN features as queries to extract relevant textual information for image regions, for example. By design, EMMA can also learn to attend to relevant descriptions even if they reference multiple other entities. Our current version of MESSENGER however, does not test for these challenges and we leave grounding entities across multiple descriptions with rich visual features to future work. Further details about EMMA and its design can be found in Appendix D.

6. Experimental Setup

6.1. Baselines

1) Mean-Bag of Sentences (Mean-BOS) This is a variant of EMMA with the attention mechanism ablated. We average the value vectors obtained from equation 2 for each descriptor to obtain $\bar{v}$ which is used by the action module.

$$\bar{v} = \frac{1}{|Z|} \sum_{z \in Z} v_z, \quad y = \text{Flatten}\big(\text{Conv2D}(\text{Emb}(o))\big)$$
$$\pi(a|o, Z) = \text{softmax}\big(\text{FFN}([y; \bar{v}])\big) \tag{5}$$

2) Game ID-Conditioned (G-ID) To assess the importance of language in our setup, we test a model with no language understanding on MESSENGER. We provide an auxillary vector I where each dimension corresponds to a role. I is then populated with the entity symbols that reveal the mapping between entities and roles (Fig. 9, Appendix). These symbols are embedded and concatenated to form the vector v_I which is used by the action module to generate a distribution over the next actions.

$$y = \text{Flatten}\big(\text{Conv2D}(\text{Emb}(o))\big)$$
$$\pi(a|o, Z) = \text{softmax}\big(\text{FFN}([y; v_I])\big) \tag{6}$$

3) Bayesian Attention Module (BAM) To assess the extent that co-occurrence statistics can help models learn M, we train a naive Bayes classifier to learn M. This approach is similar to word alignment models used in machine translation such as the IBM Model 1 (Brown et al., 1993). Specifically, for some set of observed entities $E' \subseteq E$ in the current environment:

$$\text{BAM}(z, E') = \arg\max_{e \in E'} P(e|z)$$
$$P(z|e) = \prod_{t \in z} P(t|e), \quad P(t|e) = \frac{C(t, e)}{\sum_{t'} C(t', e)} \tag{7}$$

where $t \in z$ are tokens in z, t' is any token in the manual vocabulary and C refers to co-occurence counts. We let $x_e = v_z$ from equation 2 for the z that maps to e. By construction, M is random for BAM on SC games. Note that other models can still learn M using environment rewards on SC games.

4) Oracle-Map (O-Map) To get an *upper-bound* on performance, we consider a model that has access to the descriptor to entity map M, similar to Narasimhan et al. (2018). This is identical to EMMA except that the representation for each entity x_e is obtained as in equation 8.

$$x_e = \sum_{z \in Z} \mathbb{1}[M(z) = e]v_z \tag{8}$$

5) txt2π This method was introduced by Zhong et al. (2020) alongside RTFM and features successive layers of bidirecional feature-wise modulation (FILM^2) to model multi-hop reasoning. Unlike RTFM, MESSENGER has only one text (the manual), hence we replace txt2π's inter-text attention with self-attention. Moreover, txt2π does not have explicit state-tracking because it is able to identify the next correct action based on the current observation in RTFM. This is not possible in MESSENGER, hence we add a state-tracker LSTM to txt2π before the first FILM^2 layer. Unlike other baselines that embed each fact independently, txt2π does not explicitly distinguish between facts. Instead, it ingests the manual as a concatenated string of facts.

6.2. Curriculum

We introduce three stages of MESSENGER with progressive difficulty. On all stages, we train our models in a multi-task fashion by sampling a random game and appropriate manual at the start of each episode.

Stage 1 (S1) There are three entities corresponding to the *enemy*, *message* and *goal* with three corresponding descriptions. All entities begin two steps from the agent and are immovable. The agent either begins with or without the message and must interact with the correct entity. It is provided a reward of 1 if it does so, and -1 otherwise.
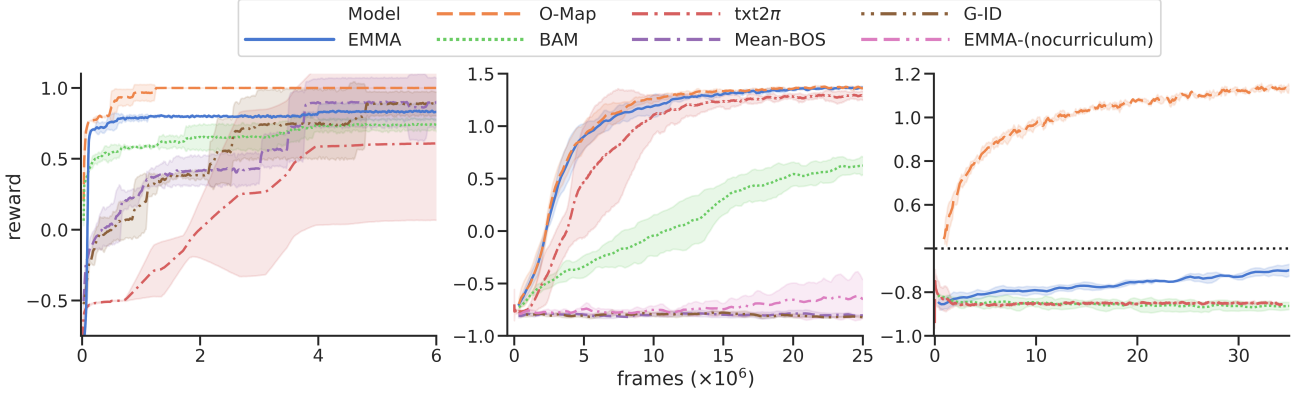
Figure 4. Average episodic rewards on S1 **(left)** S2 **(middle)** and S3 **(right)** on training games, as a function of training frames (x-axis). Note the discontinuous y-axis on S3. Reward is a combination of both single and multi-combination games. EMMA-(no curriculum) denotes EMMA trained directly on S2. Since the Mean-BOS and G-ID baselines were not able to fit to S2 at all, we do not transfer them to S3. All results are averaged over three seeds and shaded area indicates standard deviation. EMMA learns faster than other baselines and on S1 and S2 almost matches the performance of O-Map.

Stage 2 (S2) The same set of entities as stage 1 are present in stage 2, but entities are mobile and the agent always begins without the message. In each training game there is one chasing, one fleeing and one immovable entity. On test there may be any combination of movement types to force agents to adapt to unseen transition distributions $P(s_{t+1}|s_t, a_t)$.

Stage 3 (S3) In this stage there are 5 entities total with 6 descriptions, featuring one extraneous descriptor. On top of the *enemy*, *message* and *goal* entities present in stages 1 and 2, there are two additional copies of the *message* and *goal* entities, which are *enemies* and must be disambiguated by their different dynamics (e.g. 'the chasing mage is an enemy' and 'the fleeing mage is the goal.').

Human performance computed from expert playthroughs on S1, S2, and S3 are 98%, 98%, and 84% respectively (see Appendix B for details). Learning the entity groundings directly on stage 2 or 3 of MESSENGER proved to be too difficult for the models we consider. Thus, we introduce a three-stage curriculum to train our models (Bengio et al., 2009). Additional details regarding the training setup can be found in Appendix C.

7. Results

7.1. Multi-Task Performance

Figure 4 shows rewards on training games as a function of training frames. The advantage of textual understanding is clear; on both S1 and S2, EMMA and O-Map converge to good policies much faster than the other baselines. However, all models except O-Map were not able to fit to S3. While EMMA can map the correct subset of descriptions to each entity, it struggles to disambiguate the descriptions based

on movement dynamics. Doing so requires the challenge of mapping movement descriptions to observations of entity positions relative to the agent's own through multiple frames. Furthermore, EMMA cannot fit onto S2 without pretraining on S1 (Fig. 4) due to longer episode lengths. These challenges demonstrate the need for further work on grounding text (1) to movement dynamics and (2) with long trajectories and sparse rewards.

Table 1 details win rates on the training games, with a breakdown over single (SC) and multi combination (MC) games. All models were able to fit to S1, but on S2 and S3, some models exhibited win rates close to random. We observe that on MC games, the naive Bayes classifier can achieve competitive win rates by assigning over 99% of training descriptors correctly. However, on SC games which require interactive entity grounding, win rates are up to 60% lower. This result highlights the importance of distinguishing entity groundings induced from co-occurrence statistics, and those learned from environment interactions.

Our model (EMMA) can consistently win on both MC and SC games in S1 and S2, demonstrating EMMA's ability to ground entities without co-occurrences statistics between entity and text symbols to guide its grounding. While txt2π is able to fit to the S1 training games, it requires an order of magnitude more steps to do so compared to EMMA. This is likely because txt2π must learn to distinguish between facts observed as a concatenated string, while lacking an explicit entity-manual grounding module.

7.2. Generalization

Test Games Results on test games are presented in Table 2. The G-ID, Mean-BOS and txt2π baselines fail to general-

Table 1. Win rates ($\pm$ stddev.) over three seeds on train. *All*, *MC*, and *SC* denote overall, multi and single-combination games respectively. The performance of random agent subject to the same step limit on S1, S2, S3 is 7.8%, 2.1% and 1.6% respectively.

	S1-ALL	S1-MC	S1-SC	S2-ALL	S2-MC	S2-SC	S3-ALL	S3-MC	S3-SC
G-ID	89 ± 3.8	90 ± 5.5	89 ± 3.7	3.6 ± 0.6	3.4 ± 0.7	3.9 ± 1.5	–	–	–
MEAN-BOS	90 ± 7.2	91 ± 6.5	90 ± 6.8	2.1 ± 0.5	2.9 ± 1.4	2.4 ± 0.6	–	–	–
BAM	84 ± 1.3	97 ± 0.9	51 ± 1.6	69 ± 1.1	85 ± 0.9	22 ± 4.8	1.4 ± 0.3	1.6 ± 0.5	1.6 ± 0.8
TXT2π	$\mathbf{98 \pm 2.1}$	$\mathbf{98 \pm 2.9}$	$\mathbf{99 \pm 1.7}$	94 ± 3.5	95 ± 2.1	94 ± 4.0	3.0 ± 0.6	2.9 ± 0.5	2.8 ± 0.3
EMMA	88 ± 2.3	88 ± 2.4	87 ± 1.6	$\mathbf{95 \pm 0.4}$	$\mathbf{96 \pm 0.2}$	$\mathbf{95 \pm 0.5}$	22 ± 3.8	$\mathbf{21 \pm 3.6}$	19 ± 2.9
O-MAP	97 ± 0.8	97 ± 0.3	96 ± 0.6	96 ± 0.8	96 ± 0.4	94 ± 0.4	85 ± 1.5	86 ± 0.7	85 ± 0.7

Table 2. Win rates ($\pm$ stddev.) on test games over three seeds. EMMA achieves win rates on S1 and S2 test games competitive with O-Map, but performance on S3 is significantly lower.

	S1-TEST	S2-TEST	S3-TEST
G-ID	18 ± 8.2	5.2 ± 0.2	–
MEAN-BOS	6.7 ± 2.8	4.7 ± 0.5	–
BAM	66 ± 1.5	41 ± 1.7	2.7 ± 0.9
TXT2π	2.5 ± 1.7	0.3 ± 0.08	2.6 ± 0.3
EMMA	$\mathbf{85 \pm 1.4}$	$\mathbf{85 \pm 0.6}$	$\mathbf{10 \pm 0.8}$
O-MAP	97 ± 0.3	87 ± 1.8	80 ± 1.5

Figure 5. Transfer performance of EMMA on S2 games with novel entities and reward mechanics not found in MESSENGER. EMMA trained on MESSENGER (transfer) learns the new games much faster than a model trained from scratch (scratch).

Table 3. Win rates ($\pm$ stddev.) over three seeds for EMMA on negation (Neg) and neutral (Neu) cases on S1 and S2.

	S1-NEU	S2-NEU	S1-NEG	S2-NEG
TRAIN	92 ± 1.0	95 ± 0.4	87 ± 3.8	88 ± 8.8
TEST	88 ± 0.7	78 ± 2.5	67 ± 29	59 ± 33

ize in all cases. Although the models have complete access to distinguishing information necessary to succeed, they overfit to entity-role assignments observed during training. BAM demonstrates some ability to generalize to test games, but performance on games with single-combination entities are considerably lower, bringing the average down.

In contrast, EMMA wins 85% of test games on S1 and S2, almost matching the performance of the O-Map model. By extracting information from the relevant descriptor for each entity, EMMA is able to considerably simplify each task — it simply needs to learn a policy for how to interact with *enemy*, *messenger* and *goal* archetypes instead of memorizing a policy for each combination of entities. This abstraction facilitates knowledge sharing between games, and generalization to unseen games. However, test performance on S3 for all models except O-Map does not exceed 10%.

New Entities To assess EMMA's ability to pick up novel game mechanics not specified in the manual, we introduce two new stationary collectibles into MESSENGER— a trap and gold which provide additional rewards of -1 and 1 respectively. An optimal agent in this new scenario will obtain the message and also collect the gold before reaching the goal, while avoiding the enemy and the trap. We transfer EMMA trained up to S2 onto 32 unseen games with these new entities. EMMA learns the new dynamics while accomplishing the original objectives in MESSENGER (Figure 5). Compared to training from scratch, EMMA pretrained on S2 achieves a higher reward in this modified setting in the same number of steps, exceeding the previous maximum reward
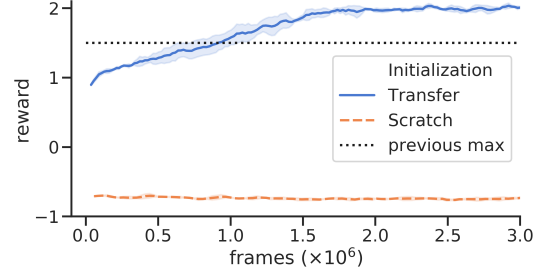
in S2 in 1×10^6 steps.

7.3. Robustness

Train-Time We test EMMA's ability to learn entity groundings with added neutral entities and negated descriptions on S2 (Table 3). Due to poor performance of all models on S3, we conduct these studies on S1 and S2 only.

Neutral entities. At the start of each episode, we randomly select one of five neutral entities and insert it into the observation. The neutral entities are not described by the text, do not interact with the agent and provide no reward signal. The neutral entities are distinct from the entities in figure 2.

Negation. On each training episode with probability 0.25 we select one description, negate it, and change the role. (e.g. 'the mage is an enemy' becomes 'the mage is not the message'). This case forces the model to consider the roles of the other two entities to deduce the role of the entity with the negated description. While EMMA can ground entities

Table 4. Win rates (± stddev.) on S2 test games over three seeds for Append, Delete and Synonym cases for EMMA and BAM.

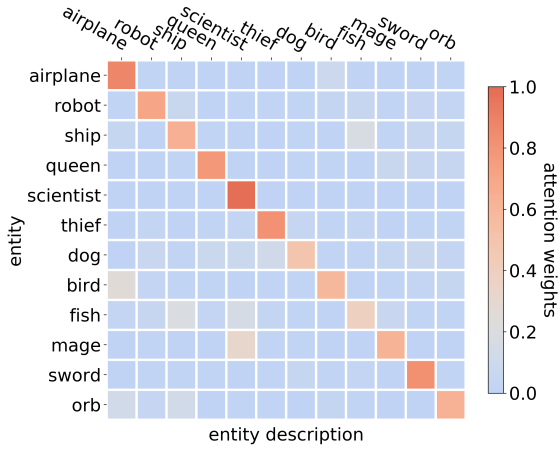|	APPEND	DELETE	SYNONYMS
BAM	36 ± 1.7	17 ± 2.0	8.5 ± 1.0
EMMA	$\mathbf{78 \pm 3.9}$	$\mathbf{33 \pm 2.3}$	$\mathbf{75 \pm 3.4}$

Figure 6. Attention weights for EMMA computed from equation 3. Each row is the attention weights for one entity over 12 random descriptors (one for each entity indicated by the column label). EMMA learns to map each description to the entity it references.

and performs well with neutral entities, it sometimes fails to ground the entities correctly with negated descriptions, affecting its performance on test games.

Test-Time We assess the robustness of trained BAM and EMMA models against text variations on S2 test games in table 4. We test each model's ability to: (1) handle an extra descriptor for an entity not found in the game (Append), (2) reason about the role of objects without a descriptor by deleting a sentence from the input at random (Delete) and (3) generalize to unseen synonyms (Synonyms). For the last case, we use (unseen) templated descriptions filled in with entity synonyms not seen during training.

Both models can retain their performance when presented with an extraneous description and suffer considerably when a description is deleted. However, EMMA generalize to unseen entity synonyms winning 75% of games compared to 8.5% by the BAM model in this setting.

7.4. Analysis of Grounding

We visualize the attention weights for EMMA in Figure 6. To assess the overall latent mapping learned by our model, we evaluate the attention weights over 12 descriptions, one for every entity. EMMA places most weight for entity e onto its descriptor z_e. In particular, EMMA learns a grounding

for dog, bird, fish, mage, sword and orb — entities for which co-occurrence statistics provide no meaningful alignment information, demonstrating that our model can learn groundings for these entities via interaction alone.

8. Conclusion

In this paper, we introduce a new environment MESSENGER which does not provide prior knowledge connecting text and state observations — the control policy must simultaneous learn to ground a natural language manual to symbols and dynamics in the environment. We develop a new model, EMMA (Entity Mapper with Multi-modal Attention) that leverages text descriptions for generalization of control policies to new environments. EMMA employs a multi-modal entity-conditioned attention module and learns a latent grounding of entities and dynamics using only environment rewards. Our empirical results on MESSENGER demonstrate that EMMA shows strong generalization performance and robust grounding of entities. However, the hardest stage of MESSENGER which requires grounding language to subtle differences in movement patterns remains difficult for EMMA and other state of the art models. We hope our work will lead to further research on generalization for RL using natural language.

Acknowledgements

References

Andreas, J. and Klein, D. Alignment-based compositional semantics for instruction following. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1165–1174, Lisbon, Portugal, 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1138. URL https://www.aclweb.org/anthology/D15-1138.

Andreas, J., Klein, D., and Levine, S. Modular multitask reinforcement learning with policy sketches. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 166–175. PMLR, 2017. URL http://proceedings.mlr.press/v70/andreas17a.html.

Bahdanau, D., Hill, F., Leike, J., Hughes, E., Kohli, P., and Grefenstette, E. Learning to follow language instructions with adversarial reward induction. *arXiv preprint arXiv:1806.01946*, 2018.

Bengio, Y., Louradour, J., Collobert, R., and Weston, J. Curriculum learning. In Danyluk, A. P., Bottou, L., and Littman, M. L. (eds.), *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pp. 41–48. ACM, 2009. doi: 10.1145/1553374.1553380. URL https://doi.org/10.1145/1553374.1553380.

Blukis, V., Terme, Y., Niklasson, E., Knepper, R. A., and Artzi, Y. Learning to map natural language instructions to physical quadcopter control using simulated flight. *arXiv preprint arXiv:1910.09664*, 2019.

Branavan, S., Silver, D., and Barzilay, R. Learning to win by reading manuals in a Monte-Carlo framework. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 268–277, Portland, Oregon, USA, 2011. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P11-1028.

Brown, P. F., Della Pietra, S. A., Della Pietra, V. J., and Mercer, R. L. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993. URL https://www.aclweb.org/anthology/J93-2003.

Buhrmester, M., Kwang, T., and Gosling, S. D. Amazon's mechanical turk: A new source of inexpensive, yet high-quality data? 2016.

Chen, D. L. and Mooney, R. J. Learning to interpret natural language navigation instructions from observations. In Burgard, W. and Roth, D. (eds.), *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, USA, August 7-11, 2011*. AAAI Press, 2011. URL http://www.aaai.org/ocs/index.php/AAAI/AAAI11/paper/view/3701.

Chevalier-Boisvert, M., Bahdanau, D., Lahlou, S., Willems, L., Saharia, C., Nguyen, T. H., and Bengio, Y. BabyAI: First steps towards grounded language learning with a human in the loop. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJeXCo0cYX.

Co-Reyes, J. D., Gupta, A., Sanjeev, S., Altieri, N., Andreas, J., DeNero, J., Abbeel, P., and Levine, S. Guiding policies with language via meta-learning. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=HkgSEnA5KQ.

Daniele, A. F., Bansal, M., and Walter, M. R. Navigational instruction generation as inverse reinforcement learning with neural machine translation. In *2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI*, pp. 109–118. IEEE, 2017.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423.

Fried, D., Andreas, J., and Klein, D. Unified pragmatic models for generating and following instructions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1951–1963, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1177. URL https://www.aclweb.org/anthology/N18-1177.

Golland, D., Liang, P., and Klein, D. A game-theoretic approach to generating spatial descriptions. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 410–419, Cambridge, MA, October 2010. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D10-1040.

Hemachandra, S., Walter, M. R., Tellex, S., and Teller, S. Learning spatial-semantic representations from natural language descriptions and scene classifications. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pp. 2623–2630. IEEE, 2014.

Hermann, K. M., Hill, F., Green, S., Wang, F., Faulkner, R., Soyer, H., Szepesvari, D., Czarnecki, W. M., Jaderberg, M., Teplyashin, D., et al. Grounded language learning in a simulated 3d world. *arXiv preprint arXiv:1706.06551*, 2017.

Hill, F., Lampinen, A. K., Schneider, R., Clark, S., Botvinick, M., McClelland, J. L., and Santoro, A. Environmental drivers of systematicity and generalization in a situated agent. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia,*

April 26-30, 2020. OpenReview.net, 2020a. URL https://openreview.net/forum?id=SklGryBtwr.

Hill, F., Mokra, S., Wong, N., and Harley, T. Human instruction-following with deep reinforcement learning via transfer-learning from text. *arXiv preprint arXiv:2005.09382*, 2020b.

Hill, F., Tieleman, O., von Glehn, T., Wong, N., Merzic, H., and Clark, S. Grounded language learning fast and slow. *arXiv preprint arXiv:2009.01719*, 2020c.

Hu, R., Fried, D., Rohrbach, A., Klein, D., Darrell, T., and Saenko, K. Are you looking? grounding to multiple modalities in vision-and-language navigation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 6551–6557, Florence, Italy, 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1655. URL https://www.aclweb.org/anthology/P19-1655.

Janner, M., Narasimhan, K., and Barzilay, R. Representation learning for grounded spatial reasoning. *Transactions of the Association for Computational Linguistics*, 6:49–61, 2018. doi: 10.1162/tacl_a_00004. URL https://www.aclweb.org/anthology/Q18-1004.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y. (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

Küttler, H., Nardelli, N., Miller, A. H., Raileanu, R., Selvatici, M., Grefenstette, E., and Rocktäschel, T. The nethack learning environment. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/569ff987c643b4bedf504efda8f786c2-Abstract.html.

Luketina, J., Nardelli, N., Farquhar, G., Foerster, J. N., Andreas, J., Grefenstette, E., Whiteson, S., and Rocktäschel, T. A survey of reinforcement learning informed by natural language. In Kraus, S. (ed.), *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pp. 6309–6317. ijcai.org, 2019. doi: 10.24963/ijcai.2019/880. URL https://doi.org/10.24963/ijcai.2019/880.

Misra, D., Langford, J., and Artzi, Y. Mapping instructions and visual observations to actions with reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1004–1015, Copenhagen, Denmark, 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1106. URL https://www.aclweb.org/anthology/D17-1106.

Narasimhan, K., Barzilay, R., and Jaakkola, T. Grounding language for transfer in deep reinforcement learning. *Journal of Artificial Intelligence Research*, 63:849–874, 2018.

Oh, J., Singh, S. P., Lee, H., and Kohli, P. Zero-shot task generalization with multi-task deep reinforcement learning. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2661–2670. PMLR, 2017. URL http://proceedings.mlr.press/v70/oh17a.html.

Reckman, H., Orkin, J., and Roy, D. Learning meanings of words and constructions, grounded in a virtual game. *Semantic Approaches in Natural Language Processing*, pp. 67, 2010.

Schaul, T. A video game description language for model-based or interactive learning. In *Computational Intelligence in Games (CIG), 2013 IEEE Conference on*, pp. 1–8. IEEE, 2013.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Tellex, S., Gopalan, N., Kress-Gazit, H., and Matuszek, C. Robots that use language. *Annual Review of Control, Robotics, and Autonomous Systems*, 3:25–55, 2020.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 5998–6008, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Walter, M. R., Hemachandra, S., Homberg, B., Tellex, S., and Teller, S. Learning semantic maps from natural language descriptions. Robotics: Science and Systems, 2013.

Wang, X., Huang, Q., Çelikyilmaz, A., Gao, J., Shen, D., Wang, Y., Wang, W. Y., and Zhang, L. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 6629–6638. Computer Vision Foundation / IEEE, 2019. doi: 10.1109/CVPR.2019.00679. URL http://openaccess.thecvf.com/content_CVPR_2019/html/Wang_Reinforced_Cross-Modal_Matching_and_Self-Supervised_Imitation_Learning_for_Vision-Language_Navigation_CVPR_2019_paper.html.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., and Brew, J. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.

Zhong, V., Rocktäschel, T., and Grefenstette, E. RTFM: generalising to new environment dynamics via reading. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=SJgob6NKvH.