Language Grounding Via Reinforcement Learning

Matt Shaffer
UC Berkeley
mattshaffer@berkeley.edu

Abstract

We introduce an approach to learning grounded language representations from latent representations in an online setting. Our learning agent observes language through the lens of consequence and behavior, allowing it to correlate interactions and outcomes with instructive feedback from a human docent. Using a modified version of the simple path-planning environment Gridworld called *MiniGrid* (Chevalier-Boisvert, 2018), we use multimodal embeddings to teach our learner to complete tasks and test its ability to generalize in new environments.

1 Introduction

Understanding context within one's environment is a critical prerequisite for making intelligent decisions. As humans, we have the ability to make intelligent decisions in environments with imperfect information states by inferring how our actions will affect the state of the world. Most notably, we do this using a generalizable framework that attributes causality to actions.

We also have a unique advantage over other species in our ability to communicate concepts and strategies to each other through language, allowing us to navigate unknown environments more effectively than when in isolation. Although the field of computer science has made impressive progress in machine learning and surpassed benchmarks of human-level performance in many domains requiring complex decision-making, the question of how to teach conceptual understanding is still an open research question.

Natural Language Processing (NLP) in particular has made effective use of methodological approaches to data understanding that involve analytics, but lacks the degree of language grounding required for greater generalization. The effects of this can be observed particularly in problems with one-to-many mappings across discrete outputs, where the data can suggest many possible "correct" answers or predictions. For example, in translation tasks where a sequence in one language is trained to output a sequence in another language, there could be many different acceptable translations depending on the context.

Since context may also depend on non-textual information such as visual perception or social expectations, it may not be available to the model in a large majority of cases. In these scenarios the model may not have the capacity to actually solve the task it has been given, and we might need to consider a multimodal regime instead. Multimodal tasks like text and image captioning, audio transcription or speech generation add many layers of complexity to machine learning models, and often still have difficulty capturing the nuance of novel situations that aren't described in training examples. To excel in unseen environments, learning agents must be able to continually adapt and revise their models of the world beyond the static datasets we usually train them with.

To demonstrate an approach to this problem, we use a simple path-planning environment to develop self-guided learning of contextual association around language inputs when paired with a human mentor. Instead being directly supervised by target-label pairs in data, our learner is encouraged to explore an environment, and uses supervision in the form of natural language instructions that are encoded by a neural network. If a small-scale implementation of this method is successful, it might be expanded to more dynamic environments, such as Embodied Question Answering (EmbodiedQA) (Das et al., 2018) or robotic control tasks, where concepts can be represented by

image segmentation semantics instead of coarse-grained grid space.

2 Background

The idea of giving learning agents the freedom to explore their environment and make unsupervised decisions is paramount to the ambition of building intelligent machines. However, without the ability to correct their behavior in a meaningful way, we risk divergent objective functions between hard-coded algorithms and contextual use cases when deployed by a human operator. Even people who are skilled in the art of building algorithms usually surrender control of them once deployed, and the field is rife with examples of bias, overfitting, feedback loops and unintended consequences of releasing them into the real world.

Online learning environments, in particular, provide hope that we might build algorithms that can be deployed with dynamic, error-correcting behavior. Moreover, imbuing the ability to communicate effectively with human counterparts should make them more robust to serving common objectives.

Most commonly, online learning is associated with Reinforcement Learning (RL) methods using pixel space representations, but there is a growing interest in methods that incorporate natural language embeddings. We reference some of the most relevant works, but acknowledge that in such a broad field, there will be many contributions overlooked as an influence.

2.1 Sparse Rewards

Sparse rewards are a well-studied area of reinforcement learning, and there are many approaches to crafting intermediate rewards and subtasks, or overcoming sparsity with techniques such as *curriculum learning* (Bengio et al., 2009) and (Graves et al., 2017) or *hindsight experience replay* (Andrychowicz et al., 2017). In the former, training begins with easy tasks that gradually increase in difficulty or complexity over time. In the latter, an agent is designed to learn from missteps, and trained on objectives achieved – regardless of their initial intent.

Sparse feedback is one of the primary reasons present-day reinforcement algorithms suffer from sample inefficiency. Backpropagating a reward that is received many time steps into the future makes causal attribution difficult to determine, especially for models that lack the complex reasoning abilities of higher order primates. So, whereas humans learn tasks quickly with few examples, it can take a model millions of iterations to reach human level performance. Recent results using multi-task and meta-learning methods have closed the gap, and while the ideas they use are not necessarily new, they have benefited greatly from modern advances in deep learning. Model Agnostic Meta-Learning (MAML) (Finn et al., 2017a) showed that few-shot learning could be effectively deployed in online environments when a model is trained to optimize its parameters for performance across multiple tasks. Similar, "one-shot" approaches have also been demonstrated by researchers as in *One-Shot Imitation Learning* (Duan et al., 2017) and *One-Shot Visual Imitation Learning via Meta-Learning* (Finn et al., 2017b).

Methods describing intrinsic motivation (Chentanez et al., 2005) and more recent hierarchical implementations describe overcoming sparse extrinsic rewards by exploring new behaviors with the objective of learning rather than problem solving (Kulkarni et al., 2016). Predicting uncertainty in future states rather than rewards has also shown promising results as demonstrated in *Curiosity-driven Exploration by Self-supervised Prediction* (Pathak et al., 2017) which describes one way to create an objective for curiosity.

2.2 Natural Language Inputs

As computational resources become more widely available, multimodal networks are becoming more commonplace, and interest in combining visual inputs with text or audio is growing. Using networks like these has enabled research around grounding representations of language by correlating language elements with objects and actions. In (Andreas et al., 2016), for example, semantic parsing is used to impose structure upon natural language inputs which enables their model to modularize language tokens to improve semantic understanding of visual inputs. We have taken this inspiration and employ a similar architecture to encode language embeddings that serve as instructional inputs.

In online environments, video games often serve as a proving ground for new approaches, and in addition to the many pixel-based RL achievements publicized by organizations like DeepMind

and OpenAI, there have also been successes that leverage NLP. One of these trained an agent to follow NLP instructions using template matched keyframes as subtasks that corresponded to natural language commands. Kaplan et. al showed in *Beating Atari with Natural Language Guided Reinforcement Learning* (Kaplan et al., 2017) that Atari games such as Breakout and Montezuma's Revenge could reach near state of the art performance using natural language commands to guide a policy where the sparsity of rewards thwarted other learners from making progress.

3 Methods

3.1 Data and Environment

Gridworld is a simulated path-planning environment often used to train autonomous agents to reach a goal. It comes in many different configurations, but usually consists of a variably-sized grid with a few sparse obstacles or rewards placed within grid squares along with a starting location and an ending location. The most simple configurations may be 3 x 4 grids with a single obstacle, while more complex forms might be multilevel mazes. There are many open sourced versions available on the web, some of which are highly customizable.

We use a version of Gridworld that has been created for use at the Montreal Institute for Learning Algorithms (MILA) to specifically train RL agents to follow natural language instructions. Their environment, *MiniGrid* (Chevalier-Boisvert and Montreal Institute for Learning Algorithms, 2018), uses OpenAI Gym (Brockman et al., 2016) to provide different learning environments in which to test models. Some are simple maze-like environments that require efficiently navigating to an exit, while others include human interpretable symbols or objects that require manipulation to successfully complete a task. Another challenge provided is a room question answering environment, that was designed with EmbodiedQA's *House3D* virtual 3D environment. The MiniGrid implementation of this challenge is a scaled-down version the requires a fraction of the processing power needed to process the rich graphical world created in House3D.

A successful demonstration of our approach on the Minigrid QA task, would warrant further testing in the full scale House3D simulation.

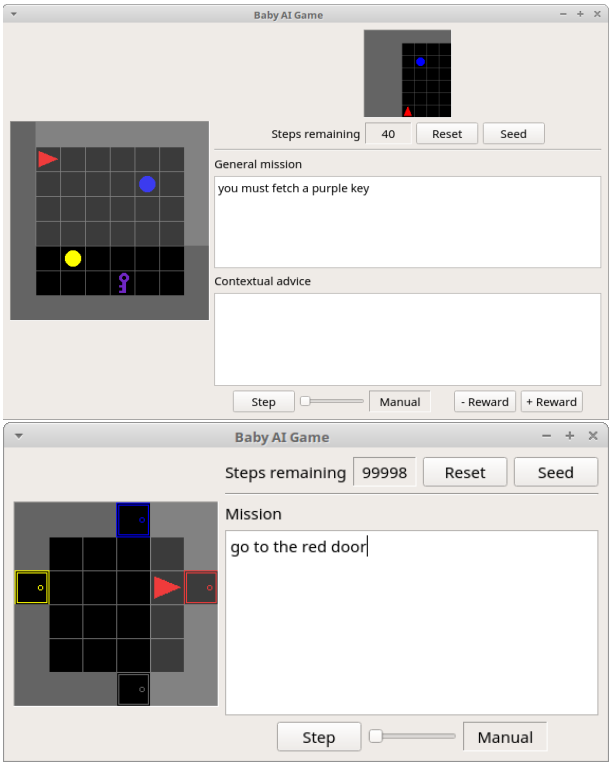


Figure 1: MiniGrid examples

3.2 Reinforcement Learning

There are several branches of Reinforcement learning from model-based approaches to model-free approaches using policy or value networks. Our network uses a version of policy gradients that is similar to the Intrinsic Curiosity Module described in Pathak et al. (2017), which encourages exploration driven by curiosity. Sensory inputs are taken in as raw pixels and transformed into a feature space by taking the current state (s_t) and the next state (s_{t+1}), trying to predict which action(a_t) would have caused the transition. Using the gradient between the predicted action and the real action, the model learns an embedding space that is then used to predict the next state space. The the prediction error for the next state acts as the intrinsic reward signal that corresponds to "curiosity".

Using this type of intrinsic motivation intuitively makes sense in our paradigm where we want our learner to understand how actions might be causally correlated to changes in the environment as well as to language inputs. Additionally, we want our agent to learn to use language inputs as subtasks that can help guide them to an eventual sparse reward. Since exploration is guided by a curiosity signal that is dependent on changes to the environment that are related to actions, it should

use structured language inputs to guide it toward tangible goal states.

3.3 Language Embeddings

Andreas et al. (2016) demonstrated that using parse trees to provide a structured embedding layer, and we use a similar idea here. Whereas the aforementioned work relies on pre-selection of categorical modules to construct layout candidates for parse trees, we choose a simpler approach that uses the Stanford dependency parser (Chen and Manning, 2014), but leaves out the additional structure that a more modularized layout might provide.

3.4 Results

Our models were evaluated on several environments within the *MiniGrid* collection, with a preference for those that would aid identification of subtasks described by natural language instructions. For example, the *Fetch* environment places the agent in a room with one or more exit doors, and requires that a specific task is completed before the door that leads to the goal state can be opened. The task might be to locate a key that corresponds to the door, and pick up that key. Once the key is picked up, the door can be opened, and the learning agent receives a reward.

The *GoToDoor* environment is likewise a simplified version of *Fetch* where no intermediate task exists, but the choice between different doors is differentiated by the descriptive properties of the door. So, if there are four doors along the perimeter of the room, the agent may obtain a reward by exiting through the green door. Given a text string of instructions "Go to the green door" would indicate which of the doors would lead to obtaining a reward.

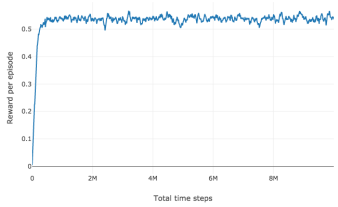
Both of the aforementioned environments offer increasing levels of complexity via grids of different scales, with the smallest grid size being *5x5* and the largest being *8x8*. The intended progression as part of this project was to train the agent on these self-contained environments, before implementing a model on one with more complexity. A project being built in conjunction with *MiniGrid*, called *Baby AI Game* uses the *Fetch* and *GoToDoor* tasks to create a larger-scale grid world amenable to curriculum learning tasks with different levels of difficulty. *Baby AI Game* would have served as the next step to progressively test the model's ability to generalize, but at the time of writing this paper, it was not ready for use and needed further refinement.

If sets of instructions are one-hot encoded as vectors, we would expect the agent to be able to learn a mapping from instructions to the intended goal, and maxmize the objective function that leads to this reward.

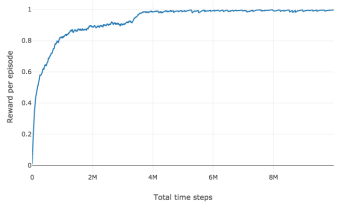
Evaluating models on *Fetch* and *GoToDoor*, we compare results to a model trained with the absence of language inputs as a baseline. The goal state in these baseline measurements remained the same as in the models with language, but the model was configured to exclude these from its inputs. This has the effect of allowing the model to learn that it must interact with objects to eventually reach a sparse reward, but without the explicit instructions that would identify which action and object it needed to interact with, and in what order.

In line with our expectations, we do see that leaving out the language encoding from the model results in a lower average reward than when encodings are used. We also find that this baseline reward also responds to the complexity of the environment, and as the state space increases with a larger grid, the average reward decreases. It should be noted here that episode termination occurs when the agent either reaches the goal state and receives a reward, or when a threshold of actions taken is reached. Since the complexity of the state space increases roughly at a rate of $\mathcal{S}^{\mathcal{A}}$, where S is the number of grid spaces and $\mathcal{A}$ is the possible number of actions that are possible at each state $a \in \mathbb{R}^n$, we should expect a precipitous decrease in accuracy unless the allowable actions scales proportionately.

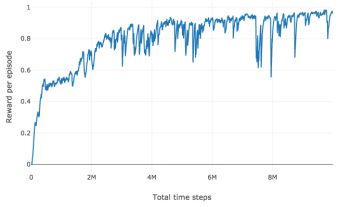
Even with natural language instructions, learning this sequence of subtasks in no trivial undertaking in the absence of a continuous reward signal. Since the model must learn that engaging with an object at a particular time step is related to a reward received at a later time step, there is a possibility of overlooking this relationship if the model does not have a sufficient memory state. We found that increasing the complexity of the environment decreased the stability of the model in the testing environment, and this could be related to the ability of the model to capture the dependencies between subtasks and rewards. Oftentimes, increasing the model capacity by adding layers can help a network in this regard, but we did not find this to be the case here, and hypothesize that the limita-



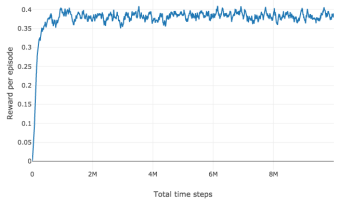
Fetch-5x5-B MiniGrid-Fetch-5x5 Baseline (note scale)



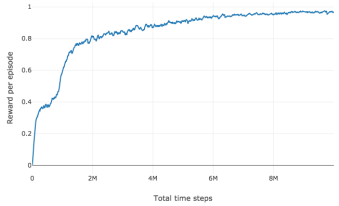
Fetch-5x5-C MiniGrid-Fetch-5x5 character-only embedding



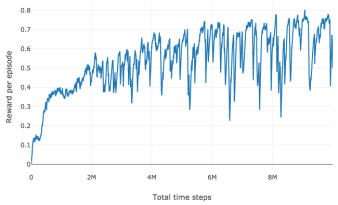
Fetch-5x5-L MiniGrid-Fetch-5x5 full language embedding



Fetch-8x8-B MiniGrid-Fetch-8x8 Baseline (note scale)



Fetch-8x8-C MiniGrid-Fetch-8x8 character-only embedding



Fetch-8x8-L MiniGrid-Fetch-5x5 full language embedding

tions are more indicative of the Actor-Critic model itself.

Interestingly, we also found that using a recurrent policy with the output of the fully connected layers prior to our reinforcement learner did not help the model performance in our tests. Perhaps, despite the recurrence over time steps, the model was unable to effectively disentangle past actions from the sparse reward regime. One way to transform a sparse reward signal into continuous signal in a similar setting was demonstrated by Kaplan et. al where they used a cross-entropy loss in pixel-space between images of the agent in the goal state and the state at the current timestep. This particular structuring of the loss objective has the additional advantage of avoiding the need to explicitly label rewards. Since cross-entropy loss is essentially unit-less, there is no need to set a reward value for reaching a particular endpoint. This is an intuitively appealing way to deal with the problem of sparse rewards, but leave this approach for the next iteration of our experiment, as it requires significant modifications of the environment to collect matrices that describe goal states.

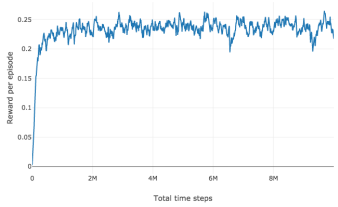
One final observation lies in the type of information captured in language instructions in relation to the environment. *GoToDoor* relies on explicit understanding of how a particular color is represented by a text string. Since the only task is to pass through the correctly identified door, the agent must decide which color door is being referred to. We find that this task is particularly difficult for the model, even with the language embeddings at the character level, which results in inconsistent performance. Using words to compose phrase-level embeddings from the SpaCy library (Honnibal and Johnson, 2015) seems like a plausible way to help provide the model with more context, but since the word embeddings are pre-trained, the meaning of different colors may not have as much differentiation as we would like. Training the language model end-to-end is probably a better approach.

In tasks requiring color differentiation, using 3-channel RGB inputs instead of image encodings seemed to give the largest boost in performance as we can see in *GoToDoor* task described above. With the only difference being the use of RGB inputs instead of the ”compact image encodings” offered as the observation space in *MiniGrid*, the avergae reward from an unstable return of approximately 0.09 to relatively stable 0.83 (see Figure 3.4).

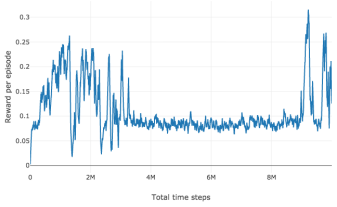
4 Discussion

Reinforcement learning is an active area of research, and there are many strategies to consider when building a model. Here we tried a number of different combinations of model configurations based on intuitions of what might improve performance, but acknowledge that there are many left to try in follow-up work.

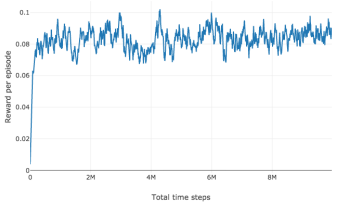
Training a language model end-to-end seems to be a necessary component of ensuring that word



GoToDoor-6x6-B MiniGrid-GoToDoor-6x6 Baseline (note scale)



GoToDoor-6x6-L MiniGrid-GoToDoor-6x6 full language embedding embedding



GoToDoor-6x6-RGB MiniGrid-GoToDoor-6x6 using RGB inputs

embeddings capture meaningful features of an environment.

Testing the model's ability to generalize across more complex and rich environments is also an obvious next step that could help provide evidence for using this approach to develop better methods of language grounding. Using environments currently being developed as part of *Baby AI Game* is a near term goal, while long-term implementations might involve *EmbodiedQA* to serve as a 3D surrogate for the real world before testing on real robots. Across all of these domains, adding a greater diversity of text inputs should also be a high priority, as the *MiniGrid* data generator currently has significant limitations.

5 Conclusion

Interacting with machines as naturally as we do humans is important not only for making them more universally accessible and interpretable, but also for instilling more generalizable intelligence than can learn without millions of training epochs. There is little doubt that if humans can communicate knowledge representations to machines effectively they will be able to learn faster, and if machines can communicate ideas to humans, their motives can be more interpretable to human operators.

If machines understood what their objectives were in a way that could be communicated to human counterparts, the resulting dialogue could allow more predictable cooperation between the two and ensure that tasks are completed efficiently, following a desirable strategy. The ultimate goal then is to make machines more teachable, by giving them a similar ability to communicate as humans do with each other.

References

J. Andreas, M. Rohrbach, T. Darrell, and D. Klein. 2016. Learning to Compose Neural Networks for Question Answering. *ArXiv e-prints*.

Jacob Andreas, Dan Klein, and Sergey Levine. 2016. Modular multitask reinforcement learning with policy sketches. *CoRR*, abs/1611.01796.

M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba. 2017. Hindsight Experience Replay. *ArXiv e-prints*.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 41–48, New York, NY, USA. ACM.

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

Danqi Chen and Christopher D. Manning. 2014. A fast and accurate dependency parser using neural networks. In *EMNLP*.

Nuttapong Chentanez, Andrew G. Barto, and Satinder P. Singh. 2005. Intrinsically motivated reinforcement learning. In L. K. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1281–1288. MIT Press.

Maxime Chevalier-Boisvert. 2018. Minimalistic gridworld environment for openai gym. https://github.com/maximecb/gym-minigrid.

Maxime Chevalier-Boisvert and Montreal Institute for Learning Algorithms. 2018. Minimalistic gridworld environment (minigrid). https://github.com/maximecb/gym-minigrid_2018.

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

Y. Duan, M. Andrychowicz, B. C. Stadie, J. Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba. 2017. One-Shot Imitation Learning. *ArXiv e-prints*.

C. Finn, P. Abbeel, and S. Levine. 2017a. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *ArXiv e-prints*.

C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. 2017b. One-Shot Visual Imitation Learning via Meta-Learning. *ArXiv e-prints*.

Alex Graves, Marc G. Bellemare, Jacob Menick, Rémi Munos, and Koray Kavukcuoglu. 2017. Automated curriculum learning for neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1311–1320, International Convention Centre, Sydney, Australia. PMLR.

Matthew Honnibal and Mark Johnson. 2015. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1373–1378, Lisbon, Portugal. Association for Computational Linguistics.

R. Kaplan, C. Sauer, and A. Sosa. 2017. Beating Atari with Natural Language Guided Reinforcement Learning. *ArXiv e-prints*.

T. D. Kulkarni, K. R. Narasimhan, A. Saeedi, and J. B. Tenenbaum. 2016. Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation. *ArXiv e-prints*.

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. 2017. Curiosity-driven exploration by self-supervised prediction. In *ICML*.