Intrinsic Curiosity in 3D Task-Oriented Language Grounding

Daniel Do
Department of Computer Science
Stanford University
dktd@stanford.edu

Chris Manning
Mentor

Abstract

Task-oriented grounding is the task of executing a natural language instruction in the context of an environment. This involves mapping a representation of the natural language command to visual elements in the scene, and ultimately actions to take. When starting from scratch with no existing linguistic or perceptual knowledge or pretraining in a reinforcement learning setting, this is naturally a sparse proposition. In order for the agent to receive rewards based on actions, we need to hope that the agent stumbles into executing the language command correctly, which isn't practical for difficult environments. To remedy this reward sparsity, we build on top of an existing language-grounding model based in VizDoom, pretraining the model using an intrinsic curiosity reward signal. This signal is based on the error in the agent's ability to predict the consequences of its own actions. Our results show that this has an interesting, although deleterious effect on the agents' actions.

1 Introduction

Consider a reinforcement learning scenario in which an autonomous agent in a 3D environment is given a first-person view of the environment and a natural language command to execute, and is expected to carry out the task indicated by the command with no prior perceptual or linguistic knowledge. In this task, language grounding systems must learn how to extract semantically relevant features from both the natural language command and raw pixel state of the first-person view, and combine these features in such a way that produces actions conducive to reaching the objective. In other words, the agent must learn how to recognize objects, explore the environment, ground the language of the instruction in vision and in actions, and navigate correctly to objects while avoiding incorrect objects. One approach to this task described in Chaplot[1] uses an end-to-end system that consists of both a state processing module, which combines the visual and linguistic modalities through an attention mechanism, and a policy learner that predicts the agent's most optimal action in the current timestep. They show that the trained attention weights in their model correspond to visual attributes of the objects described in the command, and they ultimately conclude that this attention layer was the most important feature in the model improving performance.

This task is naturally a sparse proposition. The agent is positively rewarded in the timestep when it reaches the correct object and is negatively rewarded for every other timestep. In order for the agent to learn the association between an object and its corresponding language expression, it needs to stumble into the correct object through random chance multiple times. This quickly becomes intractable in larger environments, and so there becomes a need for a more consistent reward signal from which the agent can surmise the dynamics of the environment. Pathak[2] introduces the concept of an intrinsic

[1] Devendra Singh Chaplot et al. "Gated-attention architectures for task-oriented language grounding". In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

[2] Deepak Pathak et al. "Curiosity-driven Exploration by Self-supervised Prediction". In: *arXiv preprint arXiv:1705.05363* (2017).

reward signal for sparse tasks such as language grounding. This signal is based on the concept of curiosity; Pathak[3] introduces a new intrinsic curiosity module, which is based on an inverse dynamics model. This module has two separate neural networks, one of which predicts the current image based on the previous image and action, and the other which predicts the previous action based on the previous image and current image. The outputs of these two networks are compared with the actual action and state of the current timestep using a loss function, and the resulting error is our intrinsic curiosity signal. A higher intrinsic signal signals that the agent is exploring novel situations, such as walking around in an unexplored part of the environment or processing a new object, which we would like to reward in our training paradigm.

In this work, we pretrain our language grounding model using this intrinsic curiosity reward signal, and then proceed to train the Chaplot[4] model with some major changes that we will detail in section 3. Our driving motivation in introducing curiosity was to encourage exploratory behavior in the agent through pretraining before training the main model. We find that the curiosity pretraining we performed marginally increased the exploratory behavior of our autonomous agent.

2 Related Work

Because language grounding has major applications in robotics, there has been plenty of work focusing on grounding objects and attributes in the real world. For example, Lemaignan et al.[5] attempt to ground concepts through interactions between robots and humans, and Guadrama et al.[6] work on grounding navigational instruction. Although the results from these papers are impressive, the main aim of this paper is to teach an agent language from no prior knowledge as an approach for generality, rather than implement a practical task executor.

Another major field of language grounding aims to map instructions directly to actions in the environment. Papers in this field include Artzi et al.[7] who use semantic parsing to perform this mapping, and Mei et al.[8] who extract bag-of-word features from the input to perform this mapping. Although this is closely related to our task, we also wish to ground the language of objects and their attributes specifically in the environment as an attempt at language learning, rather than mapping instructions directly to an action sequence.

Our paper belongs in the field that uses deep reinforcement learning on visual data. Lample et al.[9] use standard reinforcement learning techniques in order to play first-person-shooter games. Zhu et al.[10] focus on a navigation task similar to our task, but instead of providing a natural language instruction as the objective, they provide an image of the target object as the objective. Yu et al.[11] and Misra et al.[12] perform the same language grounding task as us, but in custom 2D environment, whereas we perform our task in 3D environment based on the Doom game engine. Yu provides a compositionality-based approach to the task, breaking various parts of the task such as reasoning and recognition into disparate submodules. The paper that we base our work is Chaplot et al. We

[3] Ibid.

[4] Chaplot et al., "Gated-attention architectures for task-oriented language grounding", op. cit.

[5] Séverin Lemaignan et al. "Grounding the Interaction: Anchoring Situated Discourse in Everyday Human-Robot Interaction". In: *International Journal of Social Robotics* (Apr. 2011), pp. 1–19. DOI: 10.1007/s12369-011-0123-x.

[6] Sergio Guadarrama et al. "Grounding spatial relations for human-robot interaction". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, pp. 1640–1647.

[7] Yoav Artzi and Luke Zettlemoyer. "Weakly supervised learning of semantic parsers for mapping instructions to actions". In: *Transactions of the Association for Computational Linguistics* 1 (2013), pp. 49–62.

[8] Teng Li et al. "Contextual bag-of-words for visual categorization". In: *IEEE Transactions on Circuits and Systems for Video Technology* 21.4 (2011), pp. 381–392.

[9] Guillaume Lample and Devendra Singh Chaplot. "Playing FPS games with deep reinforcement learning". In: *Thirty-First AAAI Conference on Artificial Intelligence*. 2017.

[10] Yuke Zhu et al. "Target-driven visual navigation in indoor scenes using deep reinforcement learning". In: *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2017, pp. 3357–3364.

[11] Haonan Yu, Haichao Zhang, and Wei Xu. "A deep compositional framework for human-like language acquisition in virtual environment". In: *arXiv preprint arXiv:1703.09831* (2017).

[12] Dipendra Misra, John Langford, and Yoav Artzi. "Mapping instructions and visual observations to actions with reinforcement learning". In: *arXiv preprint arXiv:1704.08795* (2017).

have described their architecture in the introduction already, and we will explore it in more detail in Section 3.

There has also been plenty of work in curiosity-based learning since the 1990s. The main two works we consulted were Pathak et al. and Burda et al.[13] Pathak et al. introduced the paradigm of an inverse dynamics model to quantify curiosity, and that is the module that we use in this work for pretraining. Burda et al. performed a large-scale study of curiosity training, exploring different featurizations that could be used with the inverse dynamics model.

3 Model

Our main approach is combining the intrinsic curiosity module developed in Pathak et al. with the combined processing and policy learning network developed by Chaplot et al. We will first describe these two modules in detail, and then describe our changes.

The intrinsic curiosity module developed by Pathak et al. functions by rewarding the agent in a reinforcement learning paradigm when it encounters a situation that it is unfamiliar with, such as an unexplored region or a new object. The specific architecture used is a self-supervised inverse dynamics model, consisting of a forward network and an inverse network. The inputs to the forward network consist of the first-person view of the environment at the last step, and the action taken at the last step. The forward network computes some feature representation of the first-person view and uses that representation concatenated with the action to predict the feature representation of the next state of the environment. The backward network takes in the images of the environment at the previous and current steps, computes the representation of each image, and then concatenates these representations to predict a probability distribution of the action taken at the previous step. We define the losses of both of these networks as the L_2 distance of their predictions from the actual feature representation of the current state and the previous action respectively. Formally, these are written as

$$L_F(\phi(s_t), \hat{\phi}(s_{t+1})) = \frac{1}{2}||\hat{\phi}(s_{t+1}) - \phi(s_{t+1})||_2^2 \tag{1}$$

$$L_I(\hat{a}_t, a_t) = \frac{1}{2}||\hat{a}_t - a_t||_2^2 \tag{2}$$

$\phi(s)$ is the computed feature representation of a state. $\hat{\phi(s)}$ is the predicted state representation produced by the forward network. $\hat{a_t}$ is the probability distribution of the predicted previous action produced by our backward network, where a_t is the one-hot vector containing a 1 at the actual index of the previous action taken. Now that we have defined the losses for both of the networks inside the module, we can jointly optimize over both of these networks:

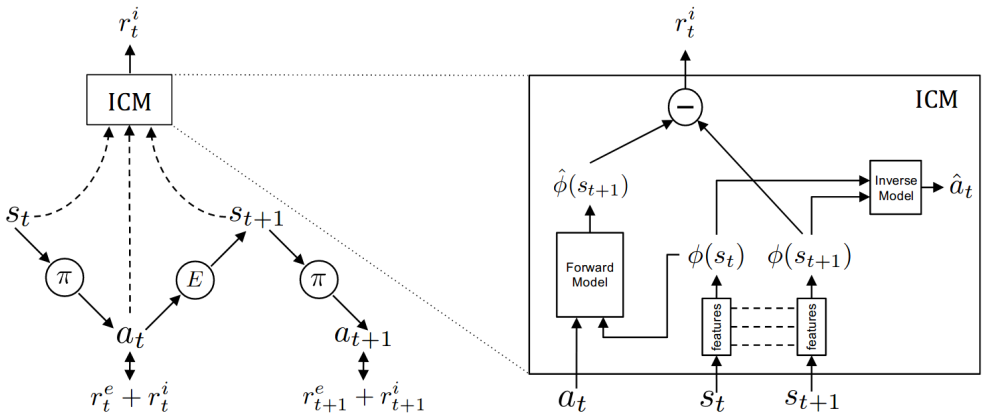
$$min \left[-\lambda \mathbb{E}_\pi(s_t; \theta_P) \left[\sum_t r_t \right] + (1-\beta)L_I + \beta L_F \right] \tag{3}$$

The first part of this equation denotes the standard expected rewards from a reinforcement learning task and $\lambda > 0$ denotes the importance of external rewards compared to our curiosity loss and β determins the importance of the backward model loss versus the forward model loss. Joint optimization over these losses trains the curious intrinsic reward signal, which we include in r_t and define to be

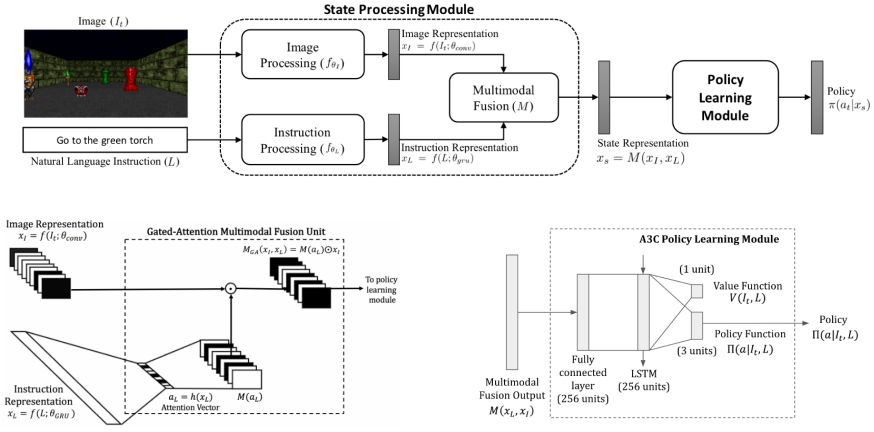
$$r_t^i = \frac{\eta}{2}||\hat{\phi}(s_{t+1}) - \phi(s_{t+1})||_2^2 \tag{4}$$

where $\eta > 0$ is a scaling factor for the reward. This is a picture for their model from their paper:

[13] Yuri Burda et al. "Large-scale study of curiosity-driven learning". In: *arXiv preprint arXiv:1808.04355* (2018).



Chaplot et al. designed a compositional model to jointly process language and vision for language grounding in a 3D environment similar to the game Doom. Their main paradigm is an A3C model that rewards the agent if it successfully executes the given natural language instruction, and punishes the agent for every other timestep. Asynchronous advantageous actor-critic (A3C) is a stable reinforcement learning paradigm that allows multiple agents to perform asynchronous gradient descent on the same model parameters, while surpassing previous state of the art performance in deep Q-learning models (Mnih et al.[14]). In Chaplot's model, the image of the environment is processed through a basic CNN, and the natural language instruction is processed through a basic GRU. They expand the GRU output so that it matches the dimensionality of the CNN output, and they perform a multiplicative attention on the CNN output with the expanded GRU output. They name this the multimodal fusion unit. Next, their policy learning module for A3C consists of a fully connected layer, which processes the multimodal fusion output, connected to an LSTM, whose output is used as the input to two different fully connected layers. One is used to predict the value function for policy learning, and the other outputs a distribution over the action space (which only consists of three actions: turn left, turn right, and forward). The distribution is sampled from to produce the action that the agent takes. Chaplot et al. believe that the most distinguishing factor of their network is the multimodal fusion unit, showing graphs that demonstrate marked advantages over simple concatenation of the image processing module and the language processing module. Along with standard A3C, Chaplot et al. also utilized general advantage estimation to reduce the variance of policy gradient updates (Schulman et al.[15]). This technique is useful for high-dimensional continuous control, so it is appropriate for our current task. These are diagrams of Chaplot's model from their paper:



[14] Volodymyr Mnih et al. "Asynchronous methods for deep reinforcement learning". In: *International conference on machine learning*. 2016, pp. 1928–1937.

[15] John Schulman et al. "High-dimensional continuous control using generalized advantage estimation". In: *arXiv preprint arXiv:1506.02438* (2015).

Our model is a combination of these two approaches. Building on top of Chaplot's model, we implement the intrinsic curiosity module using the CNN processing unit of Chaplot's model to produce our image features for both the forward and backward curiosity networks. Both Pathak et al. and Chaplot et al. use action repeat for training the models (repeating an action for n amount of frames to avoid excess computation), but Pathak accounts for this by inputting the last 4 frames of their model to model temporal dependencies, whereas Chaplot only inputs the previous frame of the environment into their model. In our model, we adopted Pathak's approach and fed the past 5 frames into our model rather than just the last frame. Another major change we made to Chaplot's model was to the multimodal fusion unit. Instead of just inputting the result of the multiplicative attention between the output of the CNN and the GRU into the policy learning module, we also concatenated both the output of the CNN and the output of the GRU to the attention output before feeding this result into the policy learning module in an attempt to information lost in the multiplicative attention. The last major change we performed was introducing an action embedding that we utilized in the inverse dynamics model.

4 Experiments

4.1 Data

Chaplot et al. developed their environment on top of VizDoom and we decided to develop on top of that to ensure consistency and fairness of results. We also use their natural language command dataset, which consists of 70 manually generated instructions, each containing some combination of (action, attribute, object). Every command is a navigation command, such as "Go to the green torch". We include 55 of these instructions in our training set, and hold out 15 of these instructions for our test set, which consist of unseen attribute tuples. Chaplot et al. have three different difficulty methods for the task, and we choose the hardest difficulty out of the three. In this setting, the task takes place in a small room. 5 objects are spawned randomly in the room; 4 are incorrect and 1 is the objective. The agent is also spawned randomly in the room and is given the natural language instruction to process. The objects may be out of the agent's initial field of view, so the agent is required to explore the area to navigate to the correct object.

4.2 Evaluation Method

We planned to use Chaplot's metric of simple accuracy. In a trial, if an agent completes the natural language command within a certain time frame, then that counts as a success; if a trial doesn't satisfy that condition, then that the agent has failed the trial. We also wanted to graph the value loss and the average reward from the policy learning module over time. However, we ran into some issues while training our model that we will detail in the discussion section, and are only able to provide a qualitative examination of our model's performance.

4.3 Experimental Details

For the Chaplot model, we use similar parameters as described in Chaplot et al. For the CNN processing unit, we used a CNN that takes in 5 RGB images of size 3x300x168. The layers are as follows: 128 filters sized 8x8 and stride 4, 64 filters sized 4x4 with stride 2, and 64 filters of 4x4 size with stride 2. We pass the language instruction and pass it through a GRU with hidden unit size of 256. We then expand the final GRU hidden state to match the dimensionality of the CNN output, perform the multiplicative attention, and then feed (attention output, CNN output, GRU output) into our fully connected layer, which outputs a vector of size 384. We feed this into our LSTM layer of size 256. The final LSTM output is fed into a fully connected layer with one neuron to predict the value function, and is also fed into another fully connected layer which outputs a vector of size 3, which is a distribution over the action space. The overall A3C model is trained using SGD with a learning rate of 0.001, and a discount factor of 0.99 for calculating rewards. An action repeat of 5 frames was used, as detailed earlier. All of the convolutional layers and fully connected layers are followed by a ReLU activation function.

For the forward curiosity module, we first featurized the previous image by passing it through the CNN processing unit described above and embedded the previous action using an action embedding of size 8. We then concatenated these two outputs and fed these into a fully connected layer, which

outputted a vector of the same size of the output of the CNN processing unit. For the backward curiosity module, we featurized both the previous and current images by passing them through the CNN processing unit described above, and then passed them through a fully connected layer which outputted a vector of size 3. We then took the softmax of this vector and returned that as our predicted probability distribution of the previous action. In equation (3), we set λ to .2 and in equation (4), we see η to 1.

We pretrained our curiosity module for 6 hours. We gave our agent +2 reward whenever it touched any object and the reward based on the curiosity signal. We reset the scene whenever the agent touched an object. We then trained our model for 30 hours without curiosity, resetting our reward function back to the standard reward function which gave the agent +2 for touching the correct object and -.1 for every other timestep.

4.4 Results

We trained both our custom model and the baseline on the hardest difficulty setting for 36 hours on the specified command training set using a DS15_v2 machine, and the following video demonstrates our results: `https://youtu.be/IB_e10_cekM`. The video from the start to 1:10 is our custom model, 1:10 - 1:55 is our model after pretraining using the intrinsic curiosity module, and the rest of the video is the baseline model. The resulting accuracy for our custom model reached a maximum of .2 while training, while the baseline model reached .6 accuracy by the end of training. As a result, it wouldn't be fruitful to compare the training results of these models.

5 Discussion

The most immediate observation from our inference video is that our model spends too much time looking around for objects, and doesn't allocate enough time to move toward the objective, whereas the baseline develops strategies for looking around as well as moving to the desired object. Interestingly enough, the baseline model developed a much more effective way of observing the environment than our model, which was turning in a 360 degree circle to view the objects in the scene.

The most likely explanation of model's behavior is that the pretrained model's behavior looks exactly the same as the model trained after 30 more hours without curiosity. Our hypothesis is that after pretraining, the model focused too many of its actions on looking around rather than moving, meaning that it could never stumble into the correct object in the main reinforcement learning task and receive a reinforcing positive reward. As a result, training our model for another 30 hours did practically nothing. In an attempt to fix this problem, we introduced a small reward for walking forward and then trained our pretrained model for about 10 hours. However, we found that this failed because the model got caught in a local maxima of walking straightforward regardless of the surrounding environment. As a result, we find the pretraining with curiosity for 6 hours before the main task isn't sufficient for task-oriented language grounding.

Although this experiment didn't produce desirable results, there a number of ways we could extend this work and possibly reintegrate curiosity into this task. Due to time constraints, we were not able to pretrain for extended periods of time; However, the most viable method seems to be pretraining for long enough for our model to be interested in walking into objects. In our pretraining, we had a positive reward for walking into any object in an attempt to encourage helpful behavior for the grounding task, but we also found nearly all of our rewards from pretraining came from the curiosity signal rather than external rewards. We predict that pretraining on a larger time scale would allow our agent to access these external rewards that weren't accessible within the timeframe available to us. Another way of extending this work is to introduce reward shaping. This is the process of giving intermediate rewards to the agent proportional to its distance to the correct objective during the main training task. We suspect that this would dramatically incentivize movement toward the correct objective in our model, which would work synergistically with our pretrained exploratory tendencies. We did not implement this in our model because this could be considered a form of supervision based on the environment and we wanted our agent to learn to complete the task without any explicit supervision besides rewarding the agent for reaching the correct object. However, this would be a practical recourse for encouraging movement toward objects.

6 Contributions

We upgraded the baseline Chaplot model from Pytorch 0.3.1 to Pytorch 1.0, and from Python2 to Python3.6, which required replacing deprecated features and method calls throughout the code. We also coded an custom implementation of the intrinsic curiosity module that we integrated into the Chaplot PyTorch module. We then introduced the changes that we detailed in the last paragraph of section 3. Finally, we coded a pretraining paradigm separate from the main language-grounding task.

References

Artzi, Yoav and Luke Zettlemoyer. "Weakly supervised learning of semantic parsers for mapping instructions to actions". In: *Transactions of the Association for Computational Linguistics* 1 (2013), pp. 49–62.

Burda, Yuri et al. "Large-scale study of curiosity-driven learning". In: *arXiv preprint arXiv:1808.04355* (2018).

Chaplot, Devendra Singh et al. "Gated-attention architectures for task-oriented language grounding". In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

Guadarrama, Sergio et al. "Grounding spatial relations for human-robot interaction". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, pp. 1640–1647.

Lample, Guillaume and Devendra Singh Chaplot. "Playing FPS games with deep reinforcement learning". In: *Thirty-First AAAI Conference on Artificial Intelligence*. 2017.

Lemaignan, Séverin et al. "Grounding the Interaction: Anchoring Situated Discourse in Everyday Human-Robot Interaction". In: *International Journal of Social Robotics* (Apr. 2011), pp. 1–19. DOI: 10.1007/s12369-011-0123-x.

Li, Teng et al. "Contextual bag-of-words for visual categorization". In: *IEEE Transactions on Circuits and Systems for Video Technology* 21.4 (2011), pp. 381–392.

Misra, Dipendra, John Langford, and Yoav Artzi. "Mapping instructions and visual observations to actions with reinforcement learning". In: *arXiv preprint arXiv:1704.08795* (2017).

Mnih, Volodymyr et al. "Asynchronous methods for deep reinforcement learning". In: *International conference on machine learning*. 2016, pp. 1928–1937.

Pathak, Deepak et al. "Curiosity-driven Exploration by Self-supervised Prediction". In: *arXiv preprint arXiv:1705.05363* (2017).

Schulman, John et al. "High-dimensional continuous control using generalized advantage estimation". In: *arXiv preprint arXiv:1506.02438* (2015).

Yu, Haonan, Haichao Zhang, and Wei Xu. "A deep compositional framework for human-like language acquisition in virtual environment". In: *arXiv preprint arXiv:1703.09831* (2017).

Zhu, Yuke et al. "Target-driven visual navigation in indoor scenes using deep reinforcement learning". In: *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2017, pp. 3357–3364.