The Promise and Perils of Reward Specification via Grounded Natural Language

Anonymous Author(s)

Abstract: Learning from visually grounded textual descriptions offers promise to achieve natural reward shaping for RL tasks. CLIP excels at grounding, yet comes with considerable large-scale data bias perils. We advocate modular and debiasing methods to limit the latter, show a factored model which learns a perceptual task given only goal text, and discuss a research roadmap towards general grounding.

Introduction Reinforcement learning offers one of the most appealing premises in the study of AI: from a reward signal alone, algorithms which learn optimal policies that maximize expected reward can learn to perform navigation, dexterous manipulation, and a host of other impactful tasks. Yet discovering, or specifying, a reward function for a given task is often a very challenging problem, especially when one is considering agents that can learn from un-instrumented environments, e.g., from raw image observations alone. We wish to have an agent that can learn purely from pixels, with no access to the underlying state of the environment at any point during learning or task execution. Achieving this goal without access to an instrumented reward function has been exceedingly challenging.

Previous efforts have explored image-based goal specification, with significant successes in visual navigation and manipulation tasks [1, 2, 3, 4]. Yet existing image-based goal specification paradigms are limited in that a goal is typically limited to a particular scene instance in an environment, rather than a semantic goal comprising multiple possible scene configurations. One can use image-based reward specification to cause a robot agent to navigate to a particular chair next to a specific tall plant, but that agent may not always succeed at the generic task of “go to a chair next to a tall flowering plant”: e.g., if the goal specification image shows a red chair next to a plant with a yellow flower the agent may navigate away from a scene with a blue chair next to a red flower, depending on the model’s underlying image representation. To be sure that the true goal is properly specified irrespective of the invariances of the model’s underlying perceptual representation, a user may have to provide a set of goal image examples that cover the variation of the target concept, potentially a very expensive undertaking to collect or generate.

We advocate semantic reward specification via grounded natural language, where a user describes a goal configuration in the world using a natural language description referring to entities in the world. This direction has long been a “holy grail” of AI research and a presumed capability of AI science fiction—the ability to instruct a robot with natural language—yet attempts have been limited by the state of the art in grounded language perception. It also falls under the general umbrella of leveraging large-scale passive data to bootstrap embodied learning, where we rely on language as a means to provide necessary reward shaping missing in visual data alone.

Several previous efforts train reward functions or policies that take natural language as input for goal description [5, 6, 7, 8, 9, 10, 11]. They all however rely on reward signals that have access to state of the system or demonstrations of the task distribution they are training on. There are works that use human videos to learn reward functions to train their agent with [11, 12, 13], but they require a curated dataset of humans performing the tasks.

Recently however, the advent of large-scale multimodal training data, together with large capacity language and vision deep learning models, has significantly advanced the landscape of the possible. A steady series of innovations have advanced grounded language modeling, from early work on multimodal translation and fusion models [14, 15, 16], to large-scale joint embedding models [17, 18],

Submitted to the 5th Conference on Robot Learning (CoRL 2021). Do not distribute.
to the plethora of multimodal transformer models currently under investigation [19, 20, 21, 22]. CLIP, in particular, demonstrated a transformative advance on zero-shot object recognition [18].

We can now consider specifying a goal state to a robot by simply offering a description of the goal configuration in natural language, and using the CLIP embedding dot product with an observed image to evaluate proximity to goal state. Surprisingly, this can work in simple cases—e.g., see the top example in Figure 3. But on closer inspection, one relies on naive CLIP-based reward specification at one’s peril: performance diminishes with complex captioned scenes involving spatial relationships and on domains that differ from web images as shown in [18] and the bottom example in that figure.

To overcome these limitations, we propose a factored grounding of ‘what’ vs ‘where/how’ aspects of goal state (Figure 1), and offer a novel spatial-saliency scheme for the former. We argue that existing (e.g., CLIP-like) models can be used to ground the what aspects of a goal quite effectively, including appropriate attribute and concept-level generalization, while a separate where/how model can ground spatial relationship aspects of goal configuration. Our first effort employs simple common-sense rule-based semantics for spatial relationships, which are incomplete in general but yet adequate for a set of robotic manipulation tasks.

We show below this approach can learn a task from only a textual goal, and uses only raw visual input for learning and execution with no access to state or demonstrations. We then discuss limitations of this approach and propose future “blue-sky” directions.

**CLIP Baseline Model**  Our base model is the most intuitive way to use the CLIP model to compute reward. Our base model simply computes reward by taking a dot product between the goal language input feature and image observation feature through CLIP’s language and image encoders respectively: 
\[ r_t = E_I(I_t) \cdot E_L(g) \]
A subset of tasks can be learned with this reward model. We visualize the limits of this model on two environments in Figure 3 and then use it as a baseline below.

**CLIP-Saliency for Phrase Grounding**  We have developed a simple method which factors phrase grounding and spatial-relation processing, which can be used to specify a reward function that operates only on image content using a natural language description. Our model leverages CLIP’s encoders in a very different way than the baseline model and is visualized in Figure 1. Our model first parses the goal language description \( g \) into object noun phrases \( g_{o_1} \ldots g_{o_k} \) and the desired object interaction. The object noun phrases are passed through CLIP’s language encoder and the current image observation is passed through CLIP’s Image encoder. We use the language encodings as class embeddings of each object noun phrase to perform a saliency analysis (using Grad-CAM [23]) on the image encoding of the observation on the last convolutional layer (specifically, the ReLU layer of CLIP’s ResNet-50 backbone). Saliency models such as Grad-CAM generally output a heatmap of the features that indicate a class exists in the current image input. The state extractor in Figure 1 computes the “state” of each object using the argmax of the saliency heatmap. See Figure 2.

<table>
<thead>
<tr>
<th>Spatial Relationship</th>
<th>Reward Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obj1 on the left of Obj2</td>
<td>( O_{x}^{1} &gt; O_{x}^{2} )</td>
</tr>
<tr>
<td>Obj1 on the right of Obj2</td>
<td>( O_{x}^{1} &gt; O_{x}^{2} )</td>
</tr>
<tr>
<td>Obj1 on top of Obj2</td>
<td>(</td>
</tr>
<tr>
<td>Obj1 below Obj2</td>
<td>(</td>
</tr>
<tr>
<td>Obj1 in between Obj2, Obj3</td>
<td>( \text{min}(O_{x}^{1}, O_{x}^{2}) &lt; O_{x} &lt; \text{max}(O_{x}^{1}, O_{x}^{2}) )</td>
</tr>
<tr>
<td>Obj1 in front of Obj2, Obj3</td>
<td>( O_{x}^{1} &gt; O_{x}^{2} )</td>
</tr>
<tr>
<td>Obj1 behind Obj2, Obj3</td>
<td>( O_{x}^{2} &gt; O_{x}^{1} )</td>
</tr>
<tr>
<td>Obj1 close to Obj2</td>
<td>(</td>
</tr>
<tr>
<td>Obj1 inside of Obj2</td>
<td>(</td>
</tr>
</tbody>
</table>

**Table 1: Spatial Grounding Heuristics**
and desired spatial relationship as input. Our spatial heuristics can be used to describe a broad set of manipulation tasks to push or place objects to semantic locations. A reward of 1 is given to the RL algorithm for every timestep that the conditions of the reward criteria are met for a given spatial relationship. The oracle reward uses true x,y,z spatial positions of the objects to determine if the desired spatial relationship is reached and also outputs 1 to the RL algorithm for every timestep the conditions of the task are met.

Our spatial heuristics are fully defined in Table 1. The first set (left of, right of, on top of, below, and in between) assume coordinates in a front camera view and the semantics are defined in that camera view with positive y in the upward direction and positive x in the right direction. The second set (in front of & behind) has access to a left camera view with coordinate x2 pointing towards the right which is towards the front in the first camera view and y2 pointing upward similar to the first camera view. The second camera is needed to know if the object is placed in the front or behind another object correctly. The third set (close to & inside of) require access to a front camera view with a 45 degree downward tilt towards the ground. This camera view is needed to see if the object is getting closer to another object in two orthogonal directions at once where as a left only or front view only allows you to determine one dimension of closeness. For “inside of” a 45 degree camera helps the agent see if the object is going inside another object without occlusion. The $\epsilon$ threshold for “inside of” is much smaller than for “close to” since the centroid can come much closer when an object goes inside a container object.

Baseline Results  Object detectors are one way to extract object states, however, they are usually not used off the shelf and need to be fine-tuned with in domain data. In Figure 2 we show pretrained Mask R-CNN [24] outputs on different camera views for the block stack environment. As you can see in the first two subfigures, the blocks are not proposed as objects with Mask R-CNN from both far and close camera views. This is a demonstration of the need for in-domain fine-tuning of object detectors to work in the environment you want to use. Our Grad-CAM output from CLIP however, highlights exactly the objects we are interested in from the object noun phrases that describes each object in the last two subfigures.

The CLIP model has been trained on a much larger dataset than Mask R-CNN and has a language component that allows us to request for the location of the object noun phrase of interest. With Mask R-CNN however, even if the object proposals were good, it would not necessarily classify objects it has not been trained for and therefore not output a filtered set of object proposals. In other words, we would not know which object is which if the detector hasn’t been trained with the label of objects we care about.

In Figure 3 we show what our base reward model outputs on two goal descriptions: 1. inverted pendulum 2. yellow object close to a blue object. For the first goal description we observe that the dot product increases as the pendulum becomes more inverted from either side as desired. For the second goal description we observe that as the blue object gets closer to the yellow object the dot product increases except for the closest image where it dips which results in an undesired output. We observed through these two goal descriptions and many others that the base reward model is not sufficient for recognizing object spatial relationships. The encoders are good at identifying what objects are in the image however, which is what we leverage for our full reward model.
Figures 4: We showcase the BaseRewardModel in the Double Inverted Pendulum doing slightly better than oracle reward and showcase our FullRewardModel performing almost as well as oracle reward on SawyerSimRobot-Pushing (pushing a blue puck close to a yellow puck), FetchSimRobot-Stacking (stacking a yellow block on top of a red block), and FetchSimRobot-Placing (placing a yellow block on the right of a red block).

Factored Model Results In Figure 4 we show how our full reward model performs on three manipulation tasks. The oracle reward function for Double-Inverted-Pendulum is alive bonus minus distance penalty minus velocity penalty. The oracle for SawyerSimRobot-Pushing is a sparse reward that outputs one when the centroid distance between two pucks are below a threshold. The oracle reward for FetchSimRobot-Stacking is a sparse reward that outputs one when a yellow block is within a horizontal and vertical threshold distance of a red block. The oracle reward for FetchSimRobot-Placing is a sparse reward that outputs one when a yellow block is correctly placed on the right of a red block. See fig 2 & 3 for image observation examples of FetchSimRobot and SawyerSimRobot.

Curiosity [25, 26] is used as our baseline since it only has access to images like our method for computing reward, but has only been successful for videogames such as Atari and Mario or Locomotion where exploring new states leads to progressing through the task (going further in levels of game for example). It is less privileged however, in that it doesn’t use language input for task specification. Curiosity learns the pushing pucks close together task but then starts learning separation of the pucks which reemphasizes that curiosity is only useful for tasks where exploring new dynamics leads to going farther in the task. For Double inverted pendulum our base reward model does better than oracle by chance because the oracle reward was originally designed for state input and was not tuned for learning image to reward mapping. Our base reward model fails for pushing, stacking, and placing which take “an image of a yellow block on top of a red block”, “an image of a yellow object close to a blue object”, and “an image of a yellow block on the right of a red block” as language input for those tasks respectively. For pushing, stacking, and placing our full reward model performs almost as well as oracle sparse reward which has privileged access to state.

Future Directions Our “blue-sky” research direction includes significant future work, including learning spatial relationships from data, leveraging separate datasets for spatial relationships and employing a type of modular network learning [27], or other multi-path integration framework. Pioneering work along these lines was proposed in [28, 29], who showed that spatial relationships could be grounded in a learning-based system. We are pursing the integration of these approaches to the grounding of spatial prepositions with the noun-phrase grounding capability we have demonstrated above based on our CLIP-salience method; we are encouraged by the success we have obtained so far with simple rule-based semantics, and expect that learned spatial relationships will outperform simple rules. Our current method does not address object pose, a key aspect of “how” specifications, but we will also incorporate that in future work.

We also ultimately advocate for a multi-modal goal specification paradigm, inspired in part by prior work on multi-modal search and/or “pointing” [30]. It is well known that users prefer to be able to refer to goals using a mixed modality paradigm when that is more efficient than a verbal description, i.e., to specify a target using both words and pixels, “move this plant (image of a strange and hard to verbally describe plant) by grabbing the base and place it next to the red chair”. While there is much work to be done to bring our overall vision to complete implementation, we believe the key elements of the approach are in hand and significant progress will be achieved in the near future.

Finally, in our “blue-sky” view we must address the ethical perils inherent in our leverage of models trained on large-scale vision and language datasets. Such datasets are well known to suffer from dataset bias that can cause failure or unintended harm [31]. While the near-term risks appear to be limited with the robotic applications presently envisioned, practitioners should continuously monitor systems for bias against underrepresented groups and ensure that robotic systems work across all socioeconomic domains. Techniques for bias assessment and debiasing should be employed whenever possible to ensure this remains the case [32, 33, 34, 35].
References


