FoMo rewards: Can we cast foundation models as reward functions?

Ekdeep Singh Lubana[1,2,3*] **Johann Brehmer**[1†] **Pim de Haan**[1†] **Taco Cohen**[1]
[1]Qualcomm AI Research[‡] [2]University of Michigan [3]CBS, Harvard University

Abstract

We explore the viability of casting foundation models as generic reward functions for reinforcement learning. To this end, we propose a simple pipeline that interfaces an off-the-shelf vision model with a large language model. Specifically, given a trajectory of observations, we infer the likelihood of an instruction describing the task that the user wants an agent to perform. We show that this generic likelihood function exhibits the characteristics ideally expected from a reward function: it associates high values with the desired behaviour and lower values for several similar, but incorrect policies. Overall, our work opens the possibility of designing open-ended agents for interactive tasks via foundation models.

1 Introduction

Recent advances in reinforcement learning (RL) algorithms have significantly simplified the design of data-driven, interactive agents [1–12]. Such algorithms generally rely on a reward function that captures a notion of "desirable behavior" to learn policies that output actions for engaging with the environment—which, itself, may [6, 7, 13–17] or may not be explicitly modeled [18–23]—in a manner that maximizes the reward. Though an integral component of the RL pipeline, the reward function is generally assumed to be provided by the user. However, designing a mathematically expressible reward function that can be optimized using RL algorithms is difficult for most real-world applications: e.g., even for the simple task of making a good cup of coffee, how does one mathematically express if a cup of coffee made by a robot is "good"? Heuristically designed reward functions based on domain expertise can often allow *hacking* in even fairly simple scenarios, i.e., they can enable learning of policies that maximize the reward, but do not engage in the desired behavior of interaction with the environment [24–27]. Research on *reward modeling* [28–35] and *imitation learning* [36–43] has tried to address this situation, using offline trajectories of desirable behavior to either learn a reward function or to imitate behavior of the agent that produced the trajectories. Such

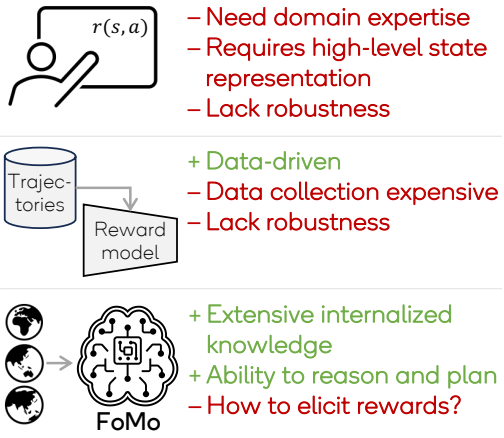


Figure 1: **Reward functions in RL** can be constructed explicitly by humans (top), learned from trajectory datasets (middle), or extracted from the world knowledge internalized by foundation models (bottom).

*Work done during an internship at Qualcomm AI Research. Email: eslubana@umich.edu.
†Order determined through *Iene miene mutte*.
‡Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

Foundation Models for Decision Making Workshop at NeurIPS 2023.

frameworks however inherit the usual issues witnessed in learning-based systems; e.g., existence of shortcut solutions that lead to lack of robustness under domain shifts [44–53]. Moreover, collecting offline trajectories makes these strategies expensive for several real-world applications; e.g., teaching a robot to not hurt a human by bumping into them is a difficult task to collect trajectories for.

Foundation models (FoMos), especially large language models (LLMs), pretrained on huge, web-crawled datasets have revolutionized several fields of machine learning [54–58, 58–62]. Demonstrating unprecedented capabilities for data-driven systems [63–66], the downstream adaptation of an off-the-shelf LLM has become the norm in most fields [67–75]. In fact, recent work has demonstrated LLMs can be integrated with other modalities, such as vision, by training of minimalistic *interface* components that transform the output of modality-specific models to a representation space "legible" to the LLM [76–80]. Used alongside pretrained large vision models (LVMs), this interfacing pipeline further enables the use of LLMs in extremely broad applications that they were never trained for, e.g., text based image editing [81, 82] or generation of 3D graphics [83, 84]. Given the large amount of internalized knowledge in an LLM [85–87], its ability to reason and plan (to an extent) [63, 88–94], and a capability to "perceive" the real world via interfacing with perception models such as LVMs [77, 79, 95–97], we argue foundation models are starting to elicit qualitative capabilities expected to be useful for design of generalist agents that can robustly function in open-ended scenarios.

Motivated by the above, in this work, we aim to assess if the framework of RL can benefit from the use of foundation models as well. Specifically focusing on the task of reward modeling, we make the following contributions.

- **FoMo as Rewards: A framework for eliciting reward functions from foundation models (Sec. 3).** Focusing on scenarios where the task of interest can be described via natural language, we propose a framework for casting decoder-based LLMs, the workhorse of SOTA language models, into reward functions. The assumption underlying our framework is that the LLM's input embedding space is "approximately grounded" [76, 77, 98], i.e., a pretrained LVM's outputs can be accurately processed by the LLM by learning a relatively light-weight interface model. Accordingly, our framework involves computing representations of observations from a trajectory via an LVM, using a *learned interface* to morph these representations to the LLM's input embedding space, and evaluating the likelihood of the task's description using the LLM.
- **Extensive qualitative analysis of FoMo as rewards (Sec. 5).** We perform an extensive evaluation to assess whether the rewards elicited by foundation models are sensible. Specifically, we use an oracle policy that engages in desired behavior and several adversarially perturbed versions thereof that, to different extents, differ in behavior from the desired one. For example, in a simple pick-and-place task, instead of picking the correct object, the agent might pick a different object from the scene. As we show, such adversarial policies achieve worse rewards than the correct one, indicating the use of FoMo Rewards is in fact viable for practical RL scenarios.

Before proceeding, we emphasize that our work is currently focused on qualitative assessment of how to retrieve rewards from foundation models, i.e., we do not train RL policies from scratch and rely only on procedural environments to demonstrate that the use of foundation models for design of reward functions is in fact viable. We do believe evaluating RL policies trained using these rewards and demonstration of the pipeline on real world tasks, e.g., via use of an ego-centric dataset for interfacing LLMs and LVMs, is both feasible and important. We leave this analysis for future work.

2 Related Work

Several recent works have explored the use of foundation models for interactive tasks, generally focusing on learning policies based on behavior cloning [99–118], pretraining representations for improved sample complexity [102, 119–134], or for designing a world model for model-based RL [17, 135, 136]. A few works have considered the task of reward modeling for training RL policies using foundation models [137–140]; we discuss approaches most relevant to our work below.

Rewards from foundation models. Assuming access to task descriptions, prior work has considered the use of vision-language foundation models, e.g., CLIP [141], to infer the representational

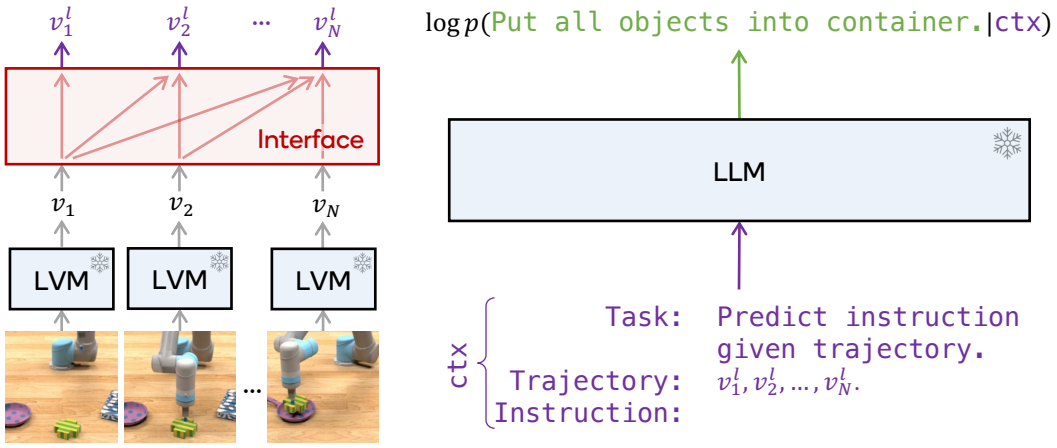


Figure 2: **Casting FoMo as Rewards.** Our proposed pipeline for casting pretrained foundation models involves learning an *interface* model that maps representations of visual observations ($\{v_i\}$) extracted via pretrained LVMs into a space "legible" to the LLM ($\{v_i^l\}$). This interface is a minimal, 1-layer Transformer in our experiments, allowing accommodation of temporal information. The remapped visual representations are added to a context (`ctx`) inspired by instruction induction tasks for fine-tuning LLMs on downstream settings [156]. Finally, the likelihood of an `instruction` describing the desired task is evaluated by inputting the context into a pretrained LLM via Eq. 1.

similarity between a task's description and the trajectory followed by an agent in the pursuit of performing that task [142–148]. Generally the vision component of these foundation models is trained on static images, which does not bode well for RL tasks where temporal information is important to account for. Hence, the works above retrain the vision pipeline from scratch on egocentric datasets such as Ego4D or on manually collected text-video pairs for the task of interest [121, 149]. A possible pitfall of such similarity based frameworks is that the training objective for CLIP-like pretraining has been shown by prior work to yield a "bag-of-words" model that is unable to capture concepts defined relatively between two objects; e.g., in an image of a person wearing a hat, the model enables recognizing the person and the hat, but not the fact that the hat is worn by the person [150–153]. As shown by Huang et al. [154], this bag-of-words pitfall leads to a lack of ability to encode temporal constraints in such vision-language models. For example, the representation for instructions `Do A then B` and `Do B then A`, where `A` and `B` are two specific tasks, will be extremely close. As expected, this can lead to undesirable behavior in the learned policies [154].

Success detection. Arguably most relevant to our work, Du et al. [155] investigate the use of multimodal foundation models based on a vision-interfaced decoder-only LLM, specifically Flamingo [79]. Flamingo models are trained via learning of a cross-attention interface on top of a PaLM model [61], yielding a highly performant baseline for multimodal tasks. Du et al. exploit Flamingo models and assess the successful completion of a task, i.e., provided a trajectory of visual observations from an agent executing a policy alongside the query "has the task finished yet?", does the model produce `True` or does it produce `False`? Such a binary success token can be considered a reward function akin to commonly used success based rewards in learning board game agents [6, 7]. However, success based rewards can be extremely sparse, limiting their usability in several tasks of interest. In contrast, our proposed pipeline utilizes likelihood of the task description based on the trajectory up to the current instant, enabling a dense reward.

3 Proposed framework: FoMo rewards

We next propose our framework for casting pretrained foundation models as reward functions (*FoMo Rewards*), which involves assessing the likelihood of an instruction describing the task, provided representations of visual observations encompassing a trajectory of behavior (see Fig. 2 for an overview). We exploit the emergent grounding of LLMs, whereby a minimal transformation of a visual input's (or in fact other modalities' inputs) representation can be processed via an LLM to

perform extremely involved tasks such as visual question answering [77, 98]. Broadly, we perform the following steps.

1. **Infer visual representation of trajectory.** Given a set of frames from the agent's interaction with the environment up to the current instant t, we use an `LVM` to transform the frames into visual embeddings $\{v_1, v_2, \ldots, v_t\}$.
2. **Process visual representations via a pretrained interface model.** Trained using a dataset of instruction-trajectory data (training protocols discussed in Sec. 3.1), the `interface` transforms the visual representations to a space processable by the LLM, denoted $\texttt{Traj}_\texttt{t} = \{v_1^l, v_2^l, \ldots, v_t^l\}$. We emphasize that prior works have trained interfaces as simple as a linear layer to enable language processing of visual scenes [77]; however, these works focus on static image tasks, such as Visual Question Answering. We find incorporating temporal information can be difficult for a mere linear interface and address this by using a 1-layer decoder-only transformer.
3. **Infer likelihood of task description given the trajectory's representation.** Finally, inspired by the "instruction induction" format of Honovich et al. [156], we embed the remapped visual representations into a broader context, denoted $\texttt{ctx}_\texttt{t}$ (see Fig. 2), which is defined by (i) the phrase "Task: Infer instruction given trajectory", (ii) the interfaced visual trajectory's representation ($\texttt{Traj}_\texttt{t}$), and (iii) a prompt "Instruction:". We then infer the log-likelihood of the instruction $\texttt{I} = \{i_1, i_2, \ldots i_N\}$, where i_n denotes the n^{th} token in the instruction, as follows.

$$\log \mathcal{P}(\texttt{I}|\texttt{ctx}_\texttt{t}) = \sum_n \log \mathcal{P}\left(\texttt{i}_n|\texttt{ctx}_\texttt{t}, i_1, i_2, \ldots, i_{n-1}\right). \tag{1}$$

The likelihood computed in Eq. 1 acts as our notion of reward. The intuition herein is that a grounded LLM can exploit its world knowledge to reason how likely is it that the agent is trying to engage in behavior described in the solve the task description provided by the user. For example, if the user says "pick up the cup" and the agent picks up a plate, ideally, the likelihood of the instruction should be lower in this scenario than the one in which the agent actually picks up the cup. This ideally implies if an agent were to engage in behavior described using the task description, the likelihood of the description would improve. Our experimental evaluation tests this intuition in extensive detail.

3.1 Interfacing protocols

Unlike prior work that interfaces static images with an LLM to perform tasks such as Visual Question Answering, our goal is to interface a trajectory of observations that are temporally correlated. To this end, we use a 1-layer decoder-only Transformer model that involves sinusoidal positional encodings, a causal Attention module, and a 1-layer MLP to map representations from a VLM to an output with dimensions equal to that of the LLM's embeddings. In the benchmark we utilize in this paper (see Sec. 4), the complexity of a task is often correlated with the number of actions required to solve it; correspondingly, the length of the trajectory can be used as a shortcut solution to output some paraphrase of the ground-truth instruction that specifies the task. It is easy to imagine such an issue emerging with realistic data involving, e.g., egocentric videos, where pick-and-place snippets may be smaller than movement from one place to another. We hence propose and evaluate three different protocols for training our interface models, as specified below (see Fig. 3 for an overview).

Maximum likelihood: The simplest of our protocols, herein we train the `interface` to minimize the negative log-likelihood of instruction `I` given a trajectory of observations that successfully solves the task. The mapped representations are inputted to the LLM in a manner similar to Fig. 2, yielding the following objective.

$$\mathcal{L}_{\texttt{ML}} = -\log \mathcal{P}(\texttt{I}|\texttt{ctx}). \tag{2}$$

Contrastive training: Herein, we first define a set of `positive` and `negative` trajectories that, respectively, correspond to an accurate versus inaccurate demonstration of the task specified by instruction `I`. To define a `negative` trajectory, we (i) randomly choose and repeat an observation up to the length of the `positive` trajectory or (ii) reverse the trajectory, preserving its length. The negative log-likelihood of `instruction` is thus minimized for the `positive` trajectory, but maximized for the `negative` one. Since negative log-likelihood is unbounded, we clamp the loss on `negative` trajectories up to an upper bound (set to 0.5) in practice, yielding the following objective.

$$\mathcal{L}_{\texttt{Contrast}} = -\left(\log \mathcal{P}(\texttt{I}|\texttt{ctx}_\texttt{P}) - \texttt{clamp}\left(\log \mathcal{P}(\texttt{I}|\texttt{ctx}_\texttt{N}), 0.5\right)\right), \tag{3}$$

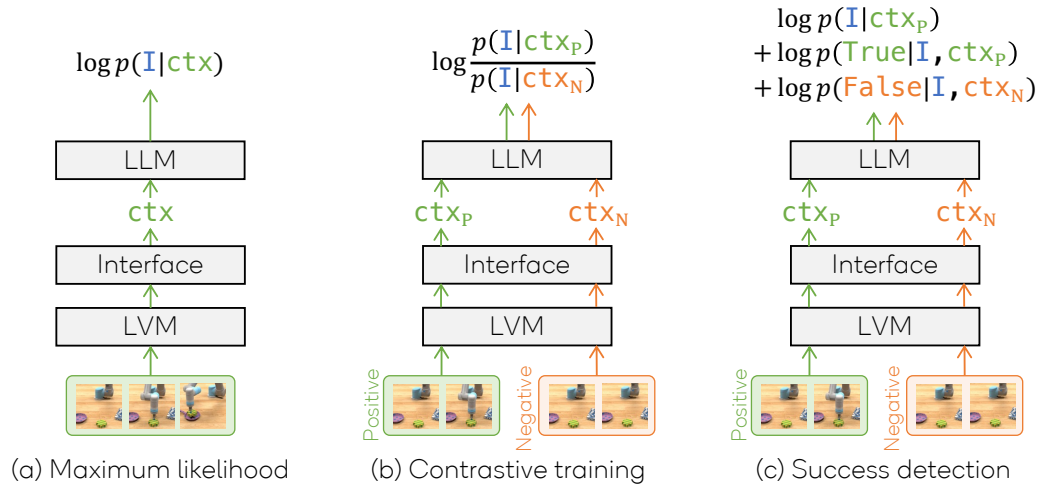


Figure 3: **Different protocols for training the interface models. (a) Maximum likelihood:** The interface is trained to map representations of the visual observations into a space such that the log-likelihood of the `instruction` , given the trajectory, is maximized; this is the standard next word prediction loss decoder-based LLMs are trained with. **(b) Contrastive training:** A pair of `positive` and `negative` trajectories are defined, wherein the `positive` trajectory does perform the desired task, while the `negative` trajectory does not. The loss focuses on maximizing the ratio of the log-likelihood of the `instruction` given the `positive` trajectory with respect to the likelihood given the `negative` trajectory. **(c) Success detection:** Similar to (b), `positive` / `negative` pairs are used; similar to (a), the likelihood of the `instruction` is maximized only for the positive trajectory. An auxiliary objective is added, which predicts the success (`True` / `False`) of the positive/negative trajectory in solving the task specified by the `instruction` .

where $\texttt{ctx}_\texttt{P}$ / $\texttt{ctx}_\texttt{N}$ denote the instruction induction context (see Sec. 2) with representations of the `positive` / `negative` trajectory substituted within it.

Success detection: Finally, we evaluate the use of an auxiliary loss that promotes a binary, True/False distinction between the `positive` versus `negative` trajectory alongside the negative log-likelihood maximization of the instruction `I` given the `positive` trajectory. The primary benefit of this formalism is that it avoids training instabilities associated with maximizing the negative log-likelihood of the `negative` trajectories in Eq. 3. However, the use of a decoder LLM makes instantiation of this framework in a standard manner of train a classifier on some representation infeasible. We thus cast this auxiliary task in the instruction following format of Wei et al. [157] (see Fig. 4).

```
ctx { Task: Does the given
      trajectory solve
      its instruction?
 Trajectory: v_1^l, v_2^l, ..., v_N^l.
Instruction: Put all objects
      into container.
     Answer:
```

Figure 4: **Context used for success detection.** The LLM is given the context above and its likelihood of outputting the tokens `True` / `False` is used to instantiate the loss in Eq. 4.

$$\mathcal{L}_{\texttt{Success}} = -\left(\log \mathcal{P}(\texttt{I}|\texttt{ctx}_\texttt{P}) + \log \mathcal{P}(\texttt{True}|\texttt{I}, \texttt{ctx}_\texttt{P}) + \log \mathcal{P}(\texttt{False}|\texttt{I}, \texttt{ctx}_\texttt{N})\right). \tag{4}$$

4 Experimental protocols

We next define the experimental setup used in this paper to evaluate the use of FoMo as rewards. As noted in Sec. 1, our focus in this work is to perform a qualitative assessment of the question whether FoMos can serve as reward functions. Accordingly, we also describe several evaluation protocols aimed at systematically perturbing an oracle policy and assess the behavior of our pipeline under these systematic perturbations.

Evaluation Benchmark. Our evaluation focuses on the recently proposed multimodal, goal-directed RL benchmark *VIMABench* [100]. Specifically, VIMABench involves thirteen tabletop

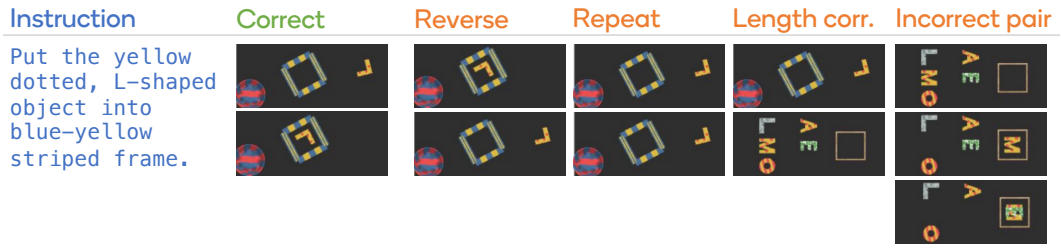


Figure 5: **Perturbed trajectories.** We perturb an oracle policy's trajectory of observations in systematic ways to assess if our pipeline elicits desirable behaviors expected from a reward function. We specifically evaluate likelihood of instruction given the: (i) *correct* trajectory; (ii) its *reverse*; (iii) an action-less trajectory where a randomly sampled observation from the correct trajectory is repeated; and (iv) an entirely incorrect trajectory corresponding to another task or instruction.

manipulation tasks that are specified via the use of a "multimodal prompt"; e.g., the prompt may specify a pick-and-place task that requires an `object` to be moved into a certain `container` in the scene or a manipulation task that requires it to be rotated by a specific `angle`. Herein, both the variables `object` and `container` are generally specified via an image. However, our proposed pipeline focuses on likelihood evaluation of the instruction that specifies a task in language, making VIMABench's multimodal prompts infeasible for our setup. To circumvent this, we write a preprocessing wrapper that transforms variables specified via images into text; e.g., an image of a blue-green polka-dotted flower will be replaced with the text `blue-green polka-dotted flower`. This yields us a unimodal version of the VIMABench benchmark. Note that only seven of VIMABench's tasks are feasible for easy conversion to such unimodal prompts (e.g., excluding when precise object coordinates are needed, which cannot be cheaply extracted without running an object detection module). For brevity, we often refer to this unimodal version of VIMA as `VIMA(Uni)`.

In `VIMA(Uni)`, objects (including containers) are denoted via a tuple of (`chars`, `identifier`). The variable `identifer` takes values from a list of predefined object and container shapes, while the variable `chars` takes values from a list of predefined characteristics that includes colors, textures, and orientations. An example prompt and start/end frame from a top-view camera of a VIMA agent performing the task specified by the prompt are shown in Fig. 5. The `VIMA(Uni)` tasks with their variable prompts are listed below. Here subscripts indicate different objects in the scene, while the variable `direction` is one of the ordinal directions, `angle` is a multiple of 30 degrees, and `constraint` is an obstacle object with which interaction is to be avoided.

- **Task 1: Same profile.** Put all objects with the same attributes as `chars container` into it.
- **Task 2: Manipulate old neighbor.** First put `chars` $_1$ object into `container` $_1$, then put the object that was previously at its `direction` into the same container.
- **Task 3: Scene Understanding.** Put the `chars` $_1$ object in scene into the `chars` $_2$ object.
- **Task 4: Pick in order then restore.** Put `chars` $_1$ object into the `chars` $_2$ container and then the `chars` $_3$ container. Finally restore it into its original container.
- **Task 5: Simple Manipulation.** Put the `chars` $_1$ object into the `chars` $_2$ container.
- **Task 6: Rotate.** Rotate `chars object` by `angle` degrees.
- **Task 7: Sweep without exceeding.** Sweep `chars` $_1$ `object` into the `chars` $_2$ `container` without exceeding `constraint`.

4.1 Systematic perturbations for evaluation

To evaluate if the recasting of FoMo as rewards is sensible, we evaluate the likelihood of several systematically perturbed setups. Specifically, we evaluate the following two scenarios.

Perturbed Trajectories. In this evaluation, we alter the trajectory taken by an oracle policy to perform the task specified by the `instruction` in the following different manners (see Fig. 5 for an overview). We evaluate the likelihood of the `instruction` given these perturbed trajectories.

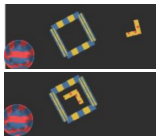
Trajectory	Correct	Objects	Color	Texture	Combined
	Put the yellow dotted, L-shaped object into blue-yellow striped frame.	Put the yellow dotted, L-shaped object into blue-yellow striped bowl.	Put the red-blue dotted, L-shaped object into yellow striped frame.	Put the yellow striped, L-shaped object into blue-yellow dotted frame.	Put the red-blue striped, L-shaped object into yellow dotted bowl.

Figure 6: **Perturbed Instructions.** We perturb the instruction corresponding to an oracle trajectory in systematic ways to assess if our pipeline elicits desirable behaviors expected from a reward function. Given the trajectory, we specifically evaluate likelihood of the following perturbed instructions: (i) *Object*: objects/containers are altered to other objects/containers in the scene; (ii) *Color*: the color specification of the target objects/containers is altered to other colors present in the scene; (iii) *Texture*: the texture specification of the target objects/containers is altered to other textures present in the scene; and (iv) *Combined*: all alteration are combined into a single, altered instructions.

1. $\texttt{PT}_{\texttt{Rev}}$: Reverse the oracle trajectory by rendering it from end to start.
2. $\texttt{PT}_{\texttt{Rep}}$: Randomly choose an observation (a frame) from the oracle trajectory and repeat it to match the oracle trajectory in length.
3. $\texttt{PT}_{\texttt{Len}}$: Randomly choose an observation from the oracle trajectory and replace it with an observation from another trajectory that may or may not correspond to the same task.
4. $\texttt{PT}_{\texttt{Inc}}$: Use an entirely different oracle trajectory that may or may not correspond to the same task.

Intuitively, the perturbations above are designed to assess if use of FoMo as rewards elicits desirable behaviors expected from a reward function. For example, $\texttt{PT}_{\texttt{Rev}}$ helps us assess *whether our pipeline assigns higher reward to appropriate temporal order of correct actions over just the presence of correct actions*. Indeed, in a pick and place task, the mere fact that the pick and place actions were performed may be sufficient to judge that the task involves picking a certain object and placing it in a specific place. However, an ideal reward function would assign higher reward to the correct temporal order, than the mere use of pick and place actions.

Perturbed Instructions. Alter the instructions specifying the task such that identifying attributes of the objects that the task is supposed to be performed on are converted to attributes of other objects present in the scene. We specifically focus on the following perturbations (see Fig. 5 for an overview).

1. $\texttt{PI}_{\texttt{Obj}}$: Alter the value of `object` and `container` identifiers from target objects/containers to another set of objects/containers present in the scene.
2. $\texttt{PI}_{\texttt{Col}}$: Alter the `chars` of target objects/containers by replacing their *color* with the color of another object present in the scene.
3. $\texttt{PI}_{\texttt{Tex}}$: Alter the `chars` of target objects/containers by replacing their *texture* with the texture of another object present in the scene.
4. $\texttt{PI}_{\texttt{Comb}}$: Combine all alterations listed above into a single altered instruction.

The motivations behind defining the above perturbed instructions is similar to that of the perturbed trajectories. For example, $\texttt{PI}_{\texttt{Obj}}$ alters the instruction such that the task to be performed remains the same, e.g., still a pick and place task, but the specific objects that are supposed to be acted upon are changed. This implies the objects the oracle policy acts upon, i.e., the objects specified in the original, unaltered instruction, will be different from the objects this perturbed instruction specifies. Assigning a lower reward to this perturbed instruction–trajectory pair, when compared to the correct one, indicates *our pipeline yields a reward function that accounts for the precise physical specification, i.e., the actions being performed and what objects they are performed on.*

	Maximum likelihood					Contrastive					Success detection				
Task	Correct	PT_{Rev}	PT_{Rep}	PT_{Len}	PT_{Inc}	Correct	PT_{Rev}	PT_{Rep}	PT_{Len}	PT_{Inc}	Correct	PT_{Rev}	PT_{Rep}	PT_{Len}	PT_{Inc}
T1	**-0.80**	-1.18	-1.14	-7.03	-37.49	**-1.52**	-77.28	-44.30	-25.49	-49.51	**-1.25**	-2.97	-2.28	-13.07	-39.51
T2	**-3.60**	-5.95	-6.59	-18.19	-63.92	**-6.42**	-93.89	-64.60	-38.71	-69.44	**-7.14**	-13.98	-10.92	-25.80	-77.56
T3	**-1.79**	-2.32	-2.59	-9.61	-33.85	**-3.11**	-45.90	-22.52	-15.99	-28.89	**-3.18**	-4.40	-3.80	-10.80	-38.85
T4	**-6.22**	-6.33	-7.53	-26.06	-87.25	**-9.71**	-9.83	-10.08	-43.20	-94.69	**-9.84**	-9.97	-10.04	-25.27	-101.98
T5	**-4.56**	-5.72	-7.16	-15.21	-37.85	**-6.55**	-32.30	-19.33	-19.21	-34.81	**-6.84**	-9.44	-8.58	-13.06	-35.60
T6	-4.10	**-4.08**	-4.12	-11.24	-34.83	-4.65	**-4.61**	-4.68	-10.26	-33.98	-4.92	**-4.86**	-4.88	-11.76	-32.94
T7	**-1.84**	-2.22	-2.72	-21.45	-99.06	**-4.38**	-19.04	-11.53	-45.36	-129.28	**-4.07**	-5.05	-4.89	-52.59	-147.13

Table 1: **Average log likelihood of `instruction` given correct versus perturbed trajectories (higher is better).** We compute the likelihood of the instruction specifying the task given several alterations of the correct trajectory, as specified in Sec. 4. The average is computed over all trajectories corresponding to a task (∼4300 per task). Broadly, the results show the use of FoMo as rewards is indeed inline with desired behavior when the `interface` is trained via the contrastive or success detection protocol: for the correct trajectories, the average likelihood of the instruction is generally much higher than the altered variants. For a subset of the tasks (e.g., T6 or Rotate), the pipeline struggles with assigning noticeably higher reward to the correct trajectory compared with the reverse or repeated one.

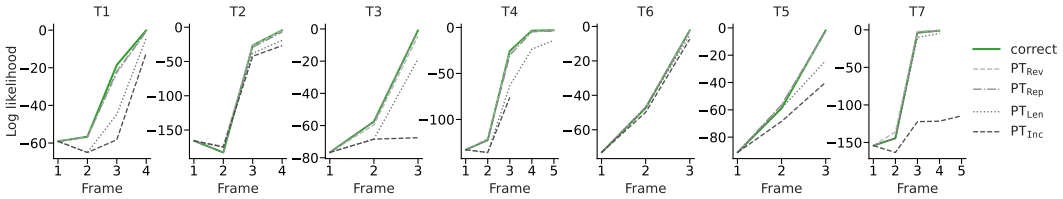
	Maximum likelihood					Contrastive					Success detection				
Task	GT	PI_{Obj}	PI_{Col}	PI_{Tex}	PI_{Comb}	GT	PI_{Obj}	PI_{Col}	PI_{Tex}	PI_{Comb}	GT	PI_{Obj}	PI_{Col}	PI_{Tex}	PI_{Comb}
T1	*-0.81*	-12.96	-4.17	-12.07	-30.06	*-1.49*	-12.34	-4.23	-11.24	-26.72	*-1.24*	-12.25	-4.11	-9.92	-27.00
T2	*-3.61*	-18.27	-8.60	-25.72	-47.75	*-6.43*	-21.87	-11.09	-23.86	-44.28	*-7.15*	-20.81	-11.07	-22.95	-38.51
T3	*-1.84*	*-1.84*	-7.36	-27.14	-31.58	*-3.10*	*-3.10*	-8.08	-24.15	-27.66	*-3.23*	*-3.23*	-8.08	-22.98	-26.49
T4	*-6.18*	-36.17	-14.20	-31.10	-68.44	*-9.71*	-40.91	-16.05	-28.34	-63.82	*-9.87*	-37.82	-16.11	-27.19	-59.78
T5	*-4.50*	-17.48	-8.91	-28.79	-48.45	*-6.53*	-18.54	-10.47	-25.18	-42.91	*-6.81*	-18.35	-10.62	-24.85	-42.25
T6	-4.10	*-3.99*	-6.38	-12.97	-23.88	*-4.66*	*-4.67*	-6.64	-12.43	-20.00	-4.92	*-4.85*	-7.02	-12.89	-20.05
T7	*-1.85*	-41.31	-10.76	-42.59	-86.14	*-4.38*	-39.83	-11.47	-35.48	-72.73	*-4.07*	-42.81	-11.25	-36.05	-78.31

Table 2: **Average log likelihood of perturbed instructions given the correct trajectory (higher is better).** We compute the likelihood of instructions altered in the manner specified in Sec. 4, while using the oracle trajectory corresponding to the unaltered `instruction` . The average is computed over all trajectories corresponding to a task (∼4300 per task). Broadly, the results show the use of FoMo as rewards is indeed inline with desired behavior: for all tasks, only for the correctly specified instructions, the average likelihood of the instruction is higher than the altered variants. We show the highest likelihoods in italics, but note that we are comparing likelihoods of texts of slightly varying lengths (generally the difference is 2–4 tokens).

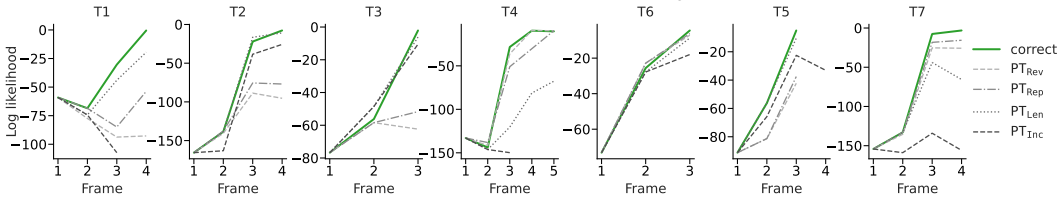
5 Results: FoMo is rewarding

With our experimental setup and evaluation protocols set up, we are now ready to answer our motivating question: "Can foundation models be cast as reward functions?" We instantiate our framework (see Sec. 3) by using an off-the-shelf, 300M parameter ViT from the PyTorch image models library hosted on HuggingFace [158]. The model is trained on the ImageNet-21K dataset using the protocol proposed by Steiner et al. [159]. We use the recently released Mosaic pretrained transformers (MPT) [160] as the LLM workhorse in our pipeline, focusing on the 1B parameter model that was pretrained using the RedPajama dataset [161] and further instruction fine-tuned using the Databricks Dolly dataset [162]. We use approximately 270K offline trajectories from an oracle policy from VIMA(Uni) to train our `interface` models and 30K trajectories to evaluate the overall reward function it yields (trajectories are approximately uniformly distributed across tasks). Training occurs at a batch-size of 8 for maximum likelihood and batch-size of 4 for contrastive and success detection protocols. Note the LVM and LLM remain frozen and their parameters are not changed at all. The oracle utilizes privileged state information to perform the specified task, though it can at times fail due to imperfect inverse kinematics by the underlying PyBullet engine.

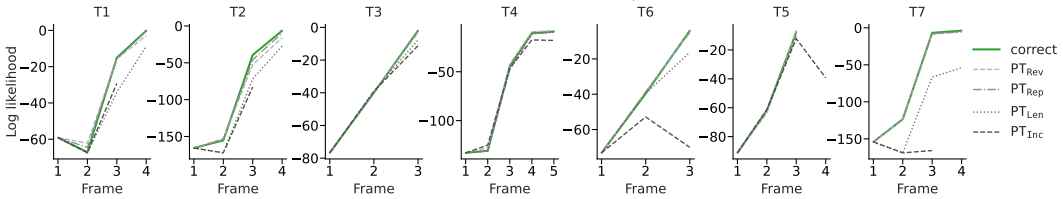
We report the following sets of results: (i) likelihood of `instruction` given the oracle versus perturbed trajectories, averaged over all instruction–trajectory pairs of a task (Tab. 1); (ii) likelihood of correct versus perturbed `instruction` given the oracle trajectory for the correct instruction, averaged over all instruction–trajectory pairs of a task (Tab. 2); and (iii) progress of the likelihood of the correct instruction given oracle versus perturbed trajectories is plotted as a function ofobservations



(a) Maximum likelihood training.



(b) Contrastive training.



(c) Success detection.

Figure 7: **FoMo rewards when training with different training protocols.** We plot `instruction` log-likelihoods as a function of actions taken in the environment in 7 trajectories, showing results for the different training protocols proposed in Sec. 3.1. On most tasks and at most steps, the correct policy achieves higher rewards than the perturbed behaviors when the interface is trained via the contrastive or success detection training protocols. However, mere maximum likelihood training, the simplest of our protocols, clearly suffers from shortcut solutions and yields similar results for both correct and perturbed trajectories.

seen after the execution of an action (Fig. 7), an evaluation inspired by the "intentscoring" experiment by Karamcheti et al. [125].

We find that across almost all tasks, the correct behaviours achieve higher likelihoods than all perturbed trajectories when the contrastive and success detection training protocols are used. The maximum likelihood objective, our simplest protocol, however clearly suffers from shortcut solutions: indeed, in Tab. 1 and Fig. 7, we see this protocol yields similar likelihoods for both correct trajectories and the perturbed ones (e.g., repeat frames yield very similar results). These results provide evidence that while FoMo rewards are viable and that agents trained on them may learn the intended behaviours, the training protocol must be devised with caution depending on the task and setup.

6 Future work

In this work, we focused on demonstrating the viability of using foundation models as reward functions by comparing the rewards achieved from various policies. Motivated by the promising results, the logical next step is to train agents on these reward functions with off-the-shelf RL algorithms. The VIMA environment used in this study provided a practical benchmark, but is not very representative of the real-world environments. Future work should study richer environments and training on large egocentric datasets. Finally, the use of foundation models to define desirable behaviors opens up the possibility of exploiting in-context learning as a means of adopting behaviours out of domain.

Acknowledgements

We would like to thank Pietro Mazzaglia and Risto Vuorio for fruitful discussions.

References

[1] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[2] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[3] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[4] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[5] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[6] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017.

[7] Thomas Anthony, Zheng Tian, and David Barber. Thinking fast and slow with deep learning and tree search. *Advances in neural information processing systems*, 30, 2017.

[8] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

[9] Brendan O'Donoghue, Remi Munos, Koray Kavukcuoglu, and Volodymyr Mnih. Combining policy gradient and q-learning. *arXiv preprint arXiv:1611.01626*, 2016.

[10] Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. Bridging the gap between value and policy based reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

[11] John Schulman, Xi Chen, and Pieter Abbeel. Equivalence between policy gradients and soft q-learning. *arXiv preprint arXiv:1704.06440*, 2017.

[12] Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[13] Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. Benchmarking model-based reinforcement learning. *arXiv preprint arXiv:1907.02057*, 2019.

[14] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

[15] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

[16] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019.

[17] Jessy Lin, Yuqing Du, Olivia Watkins, Danijar Hafner, Pieter Abbeel, Dan Klein, and Anca Dragan. Learning to model the world with language. *arXiv preprint arXiv:2308.01399*, 2023.

[18] Arthur Guez, Mehdi Mirza, Karol Gregor, Rishabh Kabra, Sébastien Racanière, Théophane Weber, David Raposo, Adam Santoro, Laurent Orseau, Tom Eccles, et al. An investigation of model-free planning. In *International Conference on Machine Learning*, pages 2464–2473. PMLR, 2019.

[19] Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. Improving sample efficiency in model-free reinforcement learning from images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10674–10681, 2021.

[20] Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning*, pages 5639–5650. PMLR, 2020.

[21] Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. *arXiv preprint arXiv:2007.05929*, 2020.

[22] Max Schwarzer, Nitarshan Rajkumar, Michael Noukhovitch, Ankesh Anand, Laurent Charlin, R Devon Hjelm, Philip Bachman, and Aaron C Courville. Pretraining representations for data-efficient reinforcement learning. *Advances in Neural Information Processing Systems*, 34: 12686–12699, 2021.

[23] Michael Laskin, Denis Yarats, Hao Liu, Kimin Lee, Albert Zhan, Kevin Lu, Catherine Cang, Lerrel Pinto, and Pieter Abbeel. Urlb: Unsupervised reinforcement learning benchmark. *arXiv preprint arXiv:2110.15191*, 2021.

[24] Victoria Krakovna. Specification gaming examples in ai, April 2018. URL https://docs.google.com/spreadsheets/d/e/2PACX-1vRPiprOaC3HsCf5Tuum8bRfzYUiKLRqJmbOoC-32JorNdfyTiRRsR7Ea5eWtvsWzuxo8bjOxCG84dAg/pubhtml.

[25] Joar Skalse, Nikolaus Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward gaming. *Advances in Neural Information Processing Systems*, 35: 9460–9471, 2022.

[26] Alexander Matt Turner, Dylan Hadfield-Menell, and Prasad Tadepalli. Conservative agency via attainable utility preservation. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 385–391, 2020.

[27] Alexander Pan, Kush Bhatia, and Jacob Steinhardt. The effects of reward misspecification: Mapping and mitigating misaligned models. *arXiv preprint arXiv:2201.03544*, 2022.

[28] Saurabh Arora and Prashant Doshi. A survey of inverse reinforcement learning: Challenges, methods and progress. *Artificial Intelligence*, 297:103500, 2021.

[29] Andrew Y Ng, Stuart Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[30] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.

[31] Haoyang Cao, Samuel Cohen, and Lukasz Szpruch. Identifiability in inverse reinforcement learning. *Advances in Neural Information Processing Systems*, 34:12362–12373, 2021.

[32] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287. Citeseer, 1999.

[33] Abhishek Gupta, Aldo Pacchiano, Yuexiang Zhai, Sham Kakade, and Sergey Levine. Unpacking reward shaping: Understanding the benefits of reward engineering on sample complexity. *Advances in Neural Information Processing Systems*, 35:15281–15295, 2022.

[34] Yujing Hu, Weixun Wang, Hangtian Jia, Yixiang Wang, Yingfeng Chen, Jianye Hao, Feng Wu, and Changjie Fan. Learning to utilize shaping rewards: A new approach of reward shaping. *Advances in Neural Information Processing Systems*, 33:15931–15941, 2020.

[35] Andrew Szot, Amy Zhang, Dhruv Batra, Zsolt Kira, and Franziska Meier. Bc-irl: Learning generalizable reward functions from demonstrations. *arXiv preprint arXiv:2303.16194*, 2023.

[36] Zhengyao Jiang, Tianjun Zhang, Michael Janner, Yueying Li, Tim Rocktäschel, Edward Grefenstette, and Yuandong Tian. Efficient planning in a compact latent action space. *arXiv preprint arXiv:2208.10291*, 2022.

[37] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

[38] Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34:1273–1286, 2021.

[39] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

[40] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

[41] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

[42] Tim Pearce, Tabish Rashid, Anssi Kanervisto, Dave Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, et al. Imitating human behaviour with diffusion models. *arXiv preprint arXiv:2301.10677*, 2023.

[43] Jifeng Hu, Yanchao Sun, Sili Huang, SiYuan Guo, Hechang Chen, Li Shen, Lichao Sun, Yi Chang, and Dacheng Tao. Instructed diffuser with temporal condition guidance for offline reinforcement learning. *arXiv preprint arXiv:2306.04875*, 2023.

[44] Pim De Haan, Dinesh Jayaraman, and Sergey Levine. Causal confusion in imitation learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[45] Joar Skalse and Alessandro Abate. Misspecification in inverse reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 15136–15143, 2023.

[46] Joar Max Viktor Skalse, Matthew Farrugia-Roberts, Stuart Russell, Alessandro Abate, and Adam Gleave. Invariance in policy optimisation and partial identifiability in reward learning. In *International Conference on Machine Learning*, pages 32033–32058. PMLR, 2023.

[47] Lev McKinney, Yawen Duan, David Krueger, and Adam Gleave. On the fragility of learned reward functions. *arXiv preprint arXiv:2301.03652*, 2023.

[48] Eric J Michaud, Adam Gleave, and Stuart Russell. Understanding learned reward functions. *arXiv preprint arXiv:2012.05862*, 2020.

[49] Adam Gleave, Michael Dennis, Shane Legg, Stuart Russell, and Jan Leike. Quantifying differences in reward functions. *arXiv preprint arXiv:2006.13900*, 2020.

[50] Jeremy Tien, Jerry Zhi-Yang He, Zackory Erickson, Anca Dragan, and Daniel S Brown. Causal confusion and reward misidentification in preference-based reward learning. In *The Eleventh International Conference on Learning Representations*, 2023.

[51] Tom Everitt, Marcus Hutter, Ramana Kumar, and Victoria Krakovna. Reward tampering problems and solutions in reinforcement learning: A causal influence diagram perspective. *Synthese*, 198(Suppl 27):6435–6467, 2021.

[52] Pedro A Ortega, Markus Kunesch, Grégoire Delétang, Tim Genewein, Jordi Grau-Moya, Joel Veness, Jonas Buchli, Jonas Degrave, Bilal Piot, Julien Perolat, et al. Shaking the foundations: delusions in sequence models for interaction and control. *arXiv preprint arXiv:2110.10819*, 2021.

[53] Risto Vuorio, Johann Brehmer, Hanno Ackermann, Daniel Dijkman, Taco Cohen, and Pim de Haan. Deconfounded imitation learning. *NeurIPS workshop on deep reinforcement learning*, 2022.

[54] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[55] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[56] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[57] Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

[58] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[59] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[60] Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

[61] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[62] Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207*, 2021.

[63] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

[64] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

[65] Nitarshan Rajkumar, Raymond Li, and Dzmitry Bahdanau. Evaluating the text-to-sql capabilities of large language models. *arXiv preprint arXiv:2204.00498*, 2022.

[66] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint. arXiv:2108.07258*, 2021.

[67] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

[68] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

[69] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[70] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

[71] Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. Multitask prompted training enables zero-shot task generalization. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=9Vrb9D0WI4.

[72] Yuming Jiang, Shuai Yang, Tong Liang Koh, Wayne Wu, Chen Change Loy, and Ziwei Liu. Text2performer: Text-driven human video generation. *arXiv preprint arXiv:2304.08483*, 2023.

[73] Qinkai Zheng, Xiao Xia, Xu Zou, Yuxiao Dong, Shan Wang, Yufei Xue, Zihan Wang, Lei Shen, Andi Wang, Yang Li, et al. Codegeex: A pre-trained model for code generation with multilingual evaluations on humaneval-x. *arXiv preprint arXiv:2303.17568*, 2023.

[74] Erik Nijkamp, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. Codegen: An open large language model for code with multi-turn program synthesis. *arXiv preprint arXiv:2203.13474*, 2022.

[75] Machel Reid, Yutaro Yamada, and Shixiang Shane Gu. Can wikipedia help offline reinforcement learning? *arXiv preprint arXiv:2201.12122*, 2022.

[76] Yaru Hao, Haoyu Song, Li Dong, Shaohan Huang, Zewen Chi, Wenhui Wang, Shuming Ma, and Furu Wei. Language models are general-purpose interfaces. *arXiv preprint arXiv:2206.06336*, 2022.

[77] Jack Merullo, Louis Castricato, Carsten Eickhoff, and Ellie Pavlick. Linearly mapping from image to text space. *arXiv preprint arXiv:2209.15162*, 2022.

[78] Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. Multimodal few-shot learning with frozen language models. *Advances in Neural Information Processing Systems*, 34:200–212, 2021.

[79] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.

[80] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

[81] Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. Imagic: Text-based real image editing with diffusion models. *arXiv preprint arXiv:2210.09276*, 2022.

[82] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022.

[83] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

[84] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. *arXiv preprint arXiv:2211.10440*, 2022.

[85] Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

[86] Yan Ding, Xiaohan Zhang, Chris Paxton, and Shiqi Zhang. Leveraging commonsense knowledge from large language models for task and motion planning. In *RSS 2023 Workshop on Learning for Task and Motion Planning*, 2023.

[87] Badr AlKhamissi, Millicent Li, Asli Celikyilmaz, Mona Diab, and Marjan Ghazvininejad. A review on language models as knowledge bases. *arXiv preprint arXiv:2204.06031*, 2022.

[88] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

[89] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*, 2022.

[90] Aman Madaan, Shuyan Zhou, Uri Alon, Yiming Yang, and Graham Neubig. Language models of code are few-shot commonsense learners. *arXiv preprint arXiv:2210.07128*, 2022.

[91] Jiaxin Huang, Shixiang Shane Gu, Le Hou, Yuexin Wu, Xuezhi Wang, Hongkun Yu, and Jiawei Han. Large language models can self-improve. *arXiv preprint arXiv:2210.11610*, 2022.

[92] Shuofei Qiao, Yixin Ou, Ningyu Zhang, Xiang Chen, Yunzhi Yao, Shumin Deng, Chuanqi Tan, Fei Huang, and Huajun Chen. Reasoning with language model prompting: A survey. *arXiv preprint arXiv:2212.09597*, 2022.

[93] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. A survey for in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.

[94] Karthik Valmeekam, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. Large language models still can't plan (a benchmark for llms on planning and reasoning about change). *arXiv preprint arXiv:2206.10498*, 2022.

[95] Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Yumao Lu, Zicheng Liu, and Lijuan Wang. An empirical study of gpt-3 for few-shot knowledge-based vqa. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3081–3089, 2022.

[96] Andy Zeng, Maria Attarian, Brian Ichter, Krzysztof Choromanski, Adrian Wong, Stefan Welker, Federico Tombari, Aveek Purohit, Michael Ryoo, Vikas Sindhwani, et al. Socratic models: Composing zero-shot multimodal reasoning with language. *arXiv preprint arXiv:2204.00598*, 2022.

[97] Chenyang Lyu, Minghao Wu, Longyue Wang, Xinting Huang, Bingshuai Liu, Zefeng Du, Shuming Shi, and Zhaopeng Tu. Macaw-llm: Multi-modal language modeling with image, audio, video, and text integration. *arXiv preprint arXiv:2306.09093*, 2023.

[98] Mostafa Abdou, Artur Kulmizev, Daniel Hershcovich, Stella Frank, Ellie Pavlick, and Anders Søgaard. Can language models encode perceptual structure without grounding? a case study in color. *arXiv preprint arXiv:2109.06129*, 2021.

[99] Apoorv Khandelwal, Luca Weihs, Roozbeh Mottaghi, and Aniruddha Kembhavi. Simple but effective: Clip embeddings for embodied ai. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14829–14838, 2022.

[100] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with multimodal prompts. *arXiv preprint arXiv:2210.03094*, 2022.

[101] Wenlong Huang, Fei Xia, Dhruv Shah, Danny Driess, Andy Zeng, Yao Lu, Pete Florence, Igor Mordatch, Sergey Levine, Karol Hausman, et al. Grounded decoding: Guiding text generation with grounded models for robot control. *arXiv preprint arXiv:2303.00855*, 2023.

[102] Fangchen Liu, Hao Liu, Aditya Grover, and Pieter Abbeel. Masked autoencoding for scalable and generalizable decision making. *arXiv preprint arXiv:2211.12740*, 2022.

[103] Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Brianna Zitkovich, Fei Xia, Chelsea Finn, et al. Open-world object manipulation using pre-trained vision-language models. *arXiv preprint arXiv:2303.00905*, 2023.

[104] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[105] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[106] Danny Driess, Ingmar Schubert, Pete Florence, Yunzhu Li, and Marc Toussaint. Reinforcement learning with neural radiance fields. *arXiv preprint arXiv:2206.01634*, 2022.

[107] Yilun Dai, Mengjiao Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *arXiv preprint arXiv:2302.00111*, 2023.

[108] Edwin Zhang, Yujie Lu, William Wang, and Amy Zhang. Lad: Language augmented diffusion for reinforcement learning. *arXiv preprint arXiv:2210.15629*, 2022.

[109] Suraj Nair, Eric Mitchell, Kevin Chen, Silvio Savarese, Chelsea Finn, et al. Learning language-conditioned robot behavior from offline data and crowd-sourced annotation. In *Conference on Robot Learning*, pages 1303–1315. PMLR, 2022.

[110] Sai Vemprala, Rogerio Bonatti, Arthur Bucker, and Ashish Kapoor. Chatgpt for robotics: Design principles and model abilities. *Microsoft Auton. Syst. Robot. Res*, 2:20, 2023.

[111] Hao Liu, Lisa Lee, Kimin Lee, and Pieter Abbeel. Instruction-following agents with jointly pre-trained vision-language models. *arXiv preprint arXiv:2210.13431*, 2022.

[112] Pierre-Louis Guhur, Shizhe Chen, Ricardo Garcia Pinel, Makarand Tapaswi, Ivan Laptev, and Cordelia Schmid. Instruction-driven history-aware policies for robotic manipulations. In *Conference on Robot Learning*, pages 175–187. PMLR, 2023.

[113] Austin W Hanjie, Victor Y Zhong, and Karthik Narasimhan. Grounding language to entities and dynamics for generalization in reinforcement learning. In *International Conference on Machine Learning*, pages 4051–4062. PMLR, 2021.

[114] Corey Lynch and Pierre Sermanet. Language conditioned imitation learning over unstructured data. *arXiv preprint arXiv:2005.07648*, 2020.

[115] Prasoon Goyal, Raymond J Mooney, and Scott Niekum. Language-guided task adaptation for imitation learning. *arXiv preprint arXiv:2301.09770*, 2023.

[116] Arthur Bucker, Luis Figueredo, Sami Haddadin, Ashish Kapoor, Shuang Ma, and Rogerio Bonatti. Latte: Language trajectory transformer. *arXiv preprint arXiv:2208.02918*, 2022.

[117] Karl Schmeckpeper, Oleh Rybkin, Kostas Daniilidis, Sergey Levine, and Chelsea Finn. Reinforcement learning with videos: Combining offline observations with interaction. *arXiv preprint arXiv:2011.06507*, 2020.

[118] Pierre Sermanet, Kelvin Xu, and Sergey Levine. Unsupervised perceptual rewards for imitation learning. *arXiv preprint arXiv:1612.06699*, 2016.

[119] Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Gupta. The unsurprising effectiveness of pre-trained vision models for control. In *International Conference on Machine Learning*, pages 17359–17371. PMLR, 2022.

[120] Shuang Li, Xavier Puig, Chris Paxton, Yilun Du, Clinton Wang, Linxi Fan, Tao Chen, De-An Huang, Ekin Akyürek, Anima Anandkumar, et al. Pre-trained language models for interactive decision-making. *Advances in Neural Information Processing Systems*, 35:31199–31212, 2022.

[121] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[122] Samir Yitzhak Gadre, Mitchell Wortsman, Gabriel Ilharco, Ludwig Schmidt, and Shuran Song. Cows on pasture: Baselines and benchmarks for language-driven zero-shot object navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23171–23181, 2023.

[123] David Brandfonbrener, Ofir Nachum, and Joan Bruna. Inverse dynamics pretraining learns good representations for multitask imitation. *arXiv preprint arXiv:2305.16985*, 2023.

[124] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022.

[125] Siddharth Karamcheti, Suraj Nair, Annie S Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-driven representation learning for robotics. *arXiv preprint arXiv:2302.12766*, 2023.

[126] Nicklas Hansen, Hao Su, and Xiaolong Wang. Stabilizing deep q-learning with convnets and vision transformers under data augmentation. *Advances in neural information processing systems*, 34:3680–3693, 2021.

[127] Yanjie Ze, Nicklas Hansen, Yinbo Chen, Mohit Jain, and Xiaolong Wang. Visual reinforcement learning with self-supervised 3d representations. *IEEE Robotics and Automation Letters*, 8(5): 2890–2897, 2023.

[128] Nicklas Hansen, Zhecheng Yuan, Yanjie Ze, Tongzhou Mu, Aravind Rajeswaran, Hao Su, Huazhe Xu, and Xiaolong Wang. On pre-training for visuo-motor control: Revisiting a learning-from-scratch baseline. *arXiv preprint arXiv:2212.05749*, 2022.

[129] Hao Liu and Pieter Abbeel. Aps: Active pretraining with successor features. In *International Conference on Machine Learning*, pages 6736–6747. PMLR, 2021.

[130] Zhecheng Yuan, Zhengrong Xue, Bo Yuan, Xueqian Wang, Yi Wu, Yang Gao, and Huazhe Xu. Pre-trained image encoder for generalizable visual reinforcement learning. *arXiv preprint arXiv:2212.08860*, 2022.

[131] Shikhar Bahl, Russell Mendonca, Lili Chen, Unnat Jain, and Deepak Pathak. Affordances from human videos as a versatile representation for robotics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13778–13790, 2023.

[132] Hao Liu and Pieter Abbeel. Behavior from the void: Unsupervised active pre-training. *Advances in Neural Information Processing Systems*, 34:18459–18473, 2021.

[133] Bowen Baker, Ilge Akkaya, Peter Zhokov, Joost Huizinga, Jie Tang, Adrien Ecoffet, Brandon Houghton, Raul Sampedro, and Jeff Clune. Video pretraining (vpt): Learning to act by watching unlabeled online videos. *Advances in Neural Information Processing Systems*, 35: 24639–24654, 2022.

[134] Ilija Radosavovic, Tete Xiao, Bike Zhang, Trevor Darrell, Jitendra Malik, and Koushil Sreenath. Learning humanoid locomotion with transformers. *arXiv preprint arXiv:2303.03381*, 2023.

[135] Younggyo Seo, Danijar Hafner, Hao Liu, Fangchen Liu, Stephen James, Kimin Lee, and Pieter Abbeel. Masked world models for visual control. In *Conference on Robot Learning*, pages 1332–1344. PMLR, 2023.

[136] Younggyo Seo, Junsu Kim, Stephen James, Kimin Lee, Jinwoo Shin, and Pieter Abbeel. Multi-view masked world models for visual robotic manipulation. *arXiv preprint arXiv:2302.02408*, 2023.

[137] Alejandro Escontrela, Ademi Adeniji, Wilson Yan, Ajay Jain, Xue Bin Peng, Ken Goldberg, Youngwoon Lee, Danijar Hafner, and Pieter Abbeel. Video prediction models as rewards for reinforcement learning. *arXiv preprint arXiv:2305.14343*, 2023.

[138] Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montse Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, et al. Language to rewards for robotic skill synthesis. *arXiv preprint arXiv:2306.08647*, 2023.

[139] Jessy Lin, Daniel Fried, Dan Klein, and Anca Dragan. Inferring rewards from language in context. *arXiv preprint arXiv:2204.02515*, 2022.

[140] Minae Kwon, Sang Michael Xie, Kalesha Bullard, and Dorsa Sadigh. Reward design with language models. *arXiv preprint arXiv:2303.00001*, 2023.

[141] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[142] Yuchen Cui, Scott Niekum, Abhinav Gupta, Vikash Kumar, and Aravind Rajeswaran. Can foundation models perform zero-shot task specification for robot manipulation? In *Learning for Dynamics and Control Conference*, pages 893–905. PMLR, 2022.

[143] Yecheng Jason Ma, William Liang, Vaidehi Som, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. Liv: Language-image representations and rewards for robotic control. *arXiv preprint arXiv:2306.00958*, 2023.

[144] Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

[145] Nicholas Waytowich, Sean L Barton, Vernon Lawhern, and Garrett Warnell. A narration-based reward shaping approach using grounded natural language commands. *arXiv preprint arXiv:1911.00497*, 2019.

[146] Prasoon Goyal, Scott Niekum, and Raymond J Mooney. Using natural language for reward shaping in reinforcement learning. *arXiv preprint arXiv:1903.02020*, 2019.

[147] Prasoon Goyal, Raymond J Mooney, and Scott Niekum. Zero-shot task adaptation using natural language. *arXiv preprint arXiv:2106.02972*, 2021.

[148] Parsa Mahmoudieh, Deepak Pathak, and Trevor Darrell. Zero-shot reward specification via grounded natural language. In *International Conference on Machine Learning*, pages 14743–14752. PMLR, 2022.

[149] Linxi Fan, Guanzhi Wang, Yunfan Jiang, Ajay Mandlekar, Yuncong Yang, Haoyi Zhu, Andrew Tang, De-An Huang, Yuke Zhu, and Anima Anandkumar. Minedojo: Building open-ended embodied agents with internet-scale knowledge. *arXiv preprint arXiv:2206.08853*, 2022.

[150] Martha Lewis, Qinan Yu, Jack Merullo, and Ellie Pavlick. Does clip bind concepts? probing compositionality in large image models. *arXiv preprint arXiv:2212.10537*, 2022.

[151] Tian Yun, Usha Bhalla, Ellie Pavlick, and Chen Sun. Do vision-language pretrained models learn composable primitive concepts? *arXiv preprint arXiv:2203.17271*, 2022.

[152] Mert Yuksekgonul, Federico Bianchi, Pratyusha Kalluri, Dan Jurafsky, and James Zou. When and why vision-language models behave like bags-of-words, and what to do about it? In *The Eleventh International Conference on Learning Representations*, 2022.

[153] Tristan Thrush, Ryan Jiang, Max Bartolo, Amanpreet Singh, Adina Williams, Douwe Kiela, and Candace Ross. Winoground: Probing vision and language models for visio-linguistic compositionality. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5238–5248, 2022.

[154] Sukai Huang, Nir Lipovetzky, and Trevor Cohn. A reminder of its brittleness: Language reward shaping may hinder learning for instruction following agents. *arXiv preprint arXiv:2305.16621*, 2023.

[155] Yuqing Du, Ksenia Konyushkova, Misha Denil, Akhil Raju, Jessica Landon, Felix Hill, Nando de Freitas, and Serkan Cabi. Vision-language models as success detectors. *arXiv preprint arXiv:2303.07280*, 2023.

[156] Or Honovich, Uri Shaham, Samuel R Bowman, and Omer Levy. Instruction induction: From few examples to natural language task descriptions. *arXiv preprint arXiv:2205.10782*, 2022.

[157] Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=gEZrGCozdqR.

[158] Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019.

[159] Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *arXiv preprint arXiv:2106.10270*, 2021.

[160] MosaicML. Mosaic pretrained transformers (mpt). https://huggingface.co/mosaicml/mpt-1b-redpajama-200b-dolly, 2023.

[161] Together Computer. Redpajama: An open source recipe to reproduce llama training dataset, April 2023. URL https://github.com/togethercomputer/RedPajama-Data.

[162] Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. Free dolly: Introducing the world's first truly open instruction-tuned llm, 2023. URL https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-llm.