Semantic Exploration from Language Abstractions and Pretrained Representations

Allison C. Tam
DeepMind
London, UK
actam@deepmind.com

Neil C. Rabinowitz
DeepMind
London, UK
ncr@deepmind.com

Andrew K. Lampinen
DeepMind
London, UK
lampinen@deepmind.com

Nicholas A. Roy
DeepMind
London, UK
nroy@deepmind.com

Stephanie C. Y. Chan
DeepMind
London, UK
scychan@deepmind.com

DJ Strouse
DeepMind
London, UK
strouse@deepmind.com

Jane X. Wang*
DeepMind
London, UK
wangjane@deepmind.com

Andrea Banino*
DeepMind
London, UK
abanino@deepmind.com

Felix Hill*
DeepMind
London, UK
felixhill@deepmind.com

Abstract

Continuous first-person 3D environments pose unique exploration challenges to reinforcement learning (RL) agents, because of their high-dimensional state and action spaces. These challenges can be ameliorated by using semantically meaningful state abstractions to define novelty for exploration. We propose that learned representations shaped by natural language provide exactly this form of abstraction. In particular, we show that vision-language representations, when pretrained on image captioning datasets sampled from the internet, can drive meaningful, task-relevant exploration and improve performance on 3D simulated environments. We also characterize why and how language provides useful abstractions for exploration by comparing the impacts of using representations from a pretrained model, a language oracle, and several ablations. We demonstrate the benefits of our approach in two very different task domains—one that stresses the identification and manipulation of everyday objects, and one that requires navigational exploration in an expansive world—as well as two popular deep RL algorithms: Impala and R2D2. Our results suggest that using language-shaped representations could improve exploration for various algorithms and agents in challenging environments.

1 Introduction

Exploration is one of the central challenges of reinforcement learning (RL). A popular way to increase an agent's tendency to explore is to augment trajectories with intrinsic rewards for reaching novel environment states. However, the success of this approach depends critically on which states are considered novel, which can in turn depend on how environment states are represented.

The literature on novelty-driven exploration describes several approaches to deriving state representations in RL agents [Burda et al., 2018b]. One of the most popular methods employs random features,

*Equal contribution

Preprint. Under review.

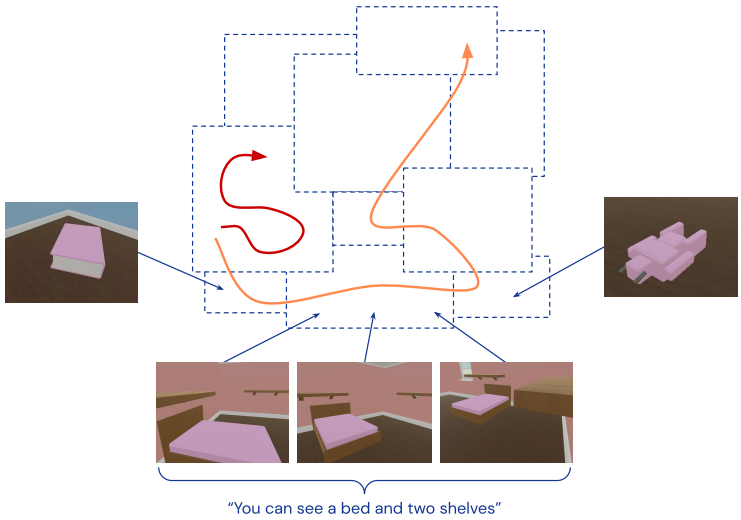


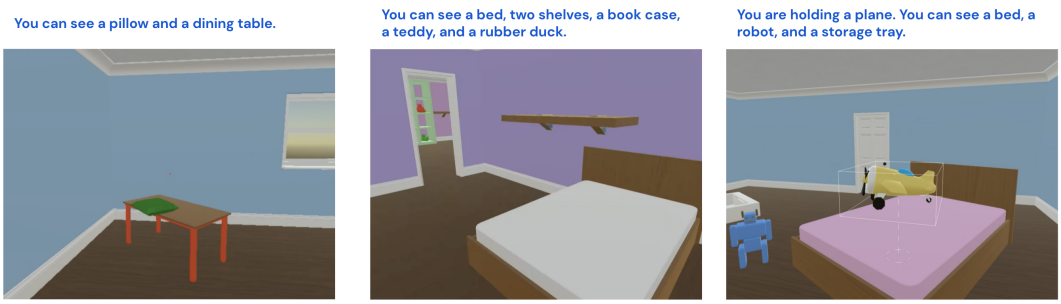
Figure 1: The dotted lines correspond to language abstractions, which group together semantically similar states. An agent exploring only based on visual novelty (red trajectory) might explore only a small part of the overall state space, while an agent exploring using language abstractions (orange trajectory) would cover much more of the state space, by seeking out semantically distinct states.

where the state is represented by embedding the visual observation with a fixed, randomly initialized target network [RND; Burda et al., 2018a]. Another popular method uses learned visual features, taken from an inverse dynamics model [NGU; Badia et al., 2020b]. While these approaches work well in classic 2D environments like Atari, it is less clear whether they are as effective in high-dimensional, partially-observable settings such as 3D environments. For instance, in these settings, different viewpoints of the same scene express similar semantic information but may map to distinct visual states/features. The difficulty of identifying a good mapping between visual state and feature space is further exacerbated by the fact that useful state abstractions are highly task dependent. For example, a task that involves tool use requires object-affordance abstractions, whereas a navigation task does not. Indeed, acquiring environment representations that support effective exploration is something of a chicken-and-egg problem—knowing whether two states should be considered similar or different requires the type of world understanding an agent can only acquire after effectively exploring its environment.

To overcome these challenges, we propose to endow agents with prior knowledge, in the form of abstractions derived from large vision-language models [e.g. Radford et al., 2021] that are pretrained on image captioning data. We hypothesize that representations acquired by vision-language pretraining drive effective, semantic exploration in 3D environments, because the representations are shaped by the unique abstract nature of natural language.

Several aspects of natural language suggest that it could provide a useful basis for directing novelty-based exploration. First, language is inherently abstract: language links superficially distinct, but causally-related situations by referring to them similarly, and contrasts between superficially similar, but causally-distinct states by describing them differently, thus outlining useful concepts [Lupyan et al., 2007, Lupyan, 2012]. Second, humans use language to communicate important information as efficiently as possible, without overspecifying [Grice, 1975, Hahn et al., 2020]. Thus, human language omits distracting irrelevant information and focuses on the aspects of the world that speakers consider important. For example, it is frequently observed that if an agent is rewarded for seeking novel experience, a TV with uncontrollable and unpredictable random noise outputs would be able to attract the agent's attention forever [Burda et al., 2018b]. However, a human caption would likely describe this situation as "a TV with no signal" regardless of the particular pattern; thus an agent exploring with language abstractions would quickly leave the TV behind. Figure 1 illustrates another conceptual example of how language abstractions can accelerate an agent's exploration.

We first demonstrate the value of this idea with a set of proof-of-concept experiments using a language oracle. We show that language is a useful abstraction for exploration not because it simply coarsens the state space, but rather because it coarsens the state space in a way that reflects the semantics



(a) Example instances of O_V and O_L from the Playroom environment.

(b) Example instances of O_V and O_L from City. Many different scenes can be associated with the same caption.

Figure 2: Visual observations from the environment and example high-level captions generated by language oracle. Appendix Figure 9 contains more example captions.

of the environment. We then demonstrate that our results scale to real pretrained vision models, which are only implicitly supervised with language. We create language-augmented variants of two popular exploration methods from the literature, Random Network Distillation (RND; Burda et al. [2018b]) and Never Give Up (NGU; Badia et al. [2020b]). We evaluate the performance of the original and language-augmented algorithms on object manipulation, search, and navigation tasks in two challenging Unity-based 3D environments: Playroom (a simulated rendering of a playroom containing toys and furniture) and City (a simulation of a large city). Our results show that language-based exploration with pretrained vision-language representations increases sample efficiency on Playroom tasks, requiring as little as half as many updates to learn object interactions. It also doubles the visited areas in City, compared to baseline methods. We show that language-based exploration is effective for both IMPALA [Espeholt et al., 2018] and R2D2 [Kapturowski et al., 2018] agents.

2 Related Work

Exploration in RL Classical exploration strategies include ϵ-greedy action selection [Sutton and Barto, 2018], state-counting [Strehl and Littman, 2008, Bellemare et al., 2016, Martin et al., 2017, Machado et al., 2020, Badia et al., 2020b], curiosity driven exploration [Schmidhuber, 1991], and intrinsic motivation methods [Oudeyer et al., 2007]. Our work is part of this last class of methods, where the agent is given an intrinsic reward that encourages the visitation of diverse states over time [Oudeyer and Kaplan, 2009]. Intrinsic reward can be derived from a number of measures: novelty [Savinov et al., 2018, Zhang et al., 2021a, Burda et al., 2018a, Zhang et al., 2021b], prediction-error [Pathak et al., 2017, Badia et al., 2020b], ensemble disagreement [Pathak et al., 2019, Flennerhag et al., 2020], or information gain [Houthooft et al., 2016]. One family of methods gives intrinsic reward for following a curriculum of goals [Campero et al., 2020, Colas et al., 2019, Racaniere et al., 2019]. Others use novelty measures to identify interesting states from which they can perform additional learning [Ecoffet et al., 2021, Zha et al., 2021]. These methods encourage exploration in different ways, but they all rely on visual state representations that are learned jointly with the policy. While this paper focuses on novelty-based intrinsic reward, our methodology is relatively agnostic to the exploration method. We specifically demonstrate the benefits of language in RND and NGU; however, we suggest that many other exploration methods could be improved by using language abstractions and pretrained embeddings to represent the state space.

Pre-training representations for RL One common way pre-training has been used in RL is to improve the representations that are used in the policy network. Self-supervised representation learning techniques distill knowledge from external datasets to produce downstream features that are helpful in virtual environments [Du et al., 2021, Xiao et al., 2022]. Some recent work shows benefits from pre-training on more general, large-scale datasets. Pretrained CLIP features have been used in a number of recent robotics papers to speed up control and navigation tasks. The features can condition the policy network [Khandelwal et al., 2021], or they can be fused throughout the visual encoder to integrate semantic information about the environment [Parisi et al., 2022]. Pretrained language models have also been shown to provide useful initializations from which to train policies to imitate offline trajectories [Reid et al., 2022, Li et al., 2022]. The success of these methods demonstrate that large pretrained models contain prior knowledge that can be useful for RL. While the existing literature uses pretrained embeddings directly, we instead allow the policy network to learn its own representations flexibly from scratch and only use pretrained embeddings to guide exploration. We imagine that combining both approaches is possible in future work.

Language for exploration Some recent works have used language to structure agent learning, by either using language-defined subgoals to encourage exploration or providing task-specific reward shaping [Mirchandani et al., 2021, Colas et al., 2020, Goyal et al., 2019]. Schwartz et al. [2019] constructs a custom semantic parser for VizDoom and shows that representing states with language, rather than vision, leads to faster learning. Chaplot et al. [2020] tackles navigation in 3D by constructing a semantic map of the environment from pretrained SLAM modules and language-defined object categories. Work concurrent to ours by Mu et al. [2022] shows how language, in the form of hand-crafted annotations from the BabyAI platform, can be used to explore more effectively in 2D environments. These works demonstrate the value of language abstractions: the ability to ignore extraneous noise and highlight relevant, salient features in the environment. Additionally, each of these prior methods relies on environment-specific semantic parsers or environment annotations. In contrast, by exploiting powerful pre-trained vision-language models, our approach can be applied to *any* visually-naturalistic environment, including 3D environments, which have not been widely studied in prior exploration work. Our method could even potentially improve exploration for physical robots, but we leave that for future work.

3 Problem Formulation

We consider a goal-conditioned Markov decision process defined by tuple $(\mathcal{S}, \mathcal{A}, \mathcal{G}, P, R_e, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{G}$ is the goal space defined by natural language instructions, $P : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ specifies the environment dynamics, $R_e : \mathcal{S} \times \mathcal{G} \to R_e$ is the extrinsic reward, and γ is the discount factor. State $\mathbf{s_t}$ is presented to the agent as visual observation O_V. Although it is not necessary for our method, in certain experiments we also assume access to a language oracle $\mathcal{O} : \mathcal{S} \to \mathcal{L}$ that provides natural language descriptions of the state, O_L. Note that O_L is distinct from the language instruction $g \in \mathcal{G}$, which is sampled from a goal distribution at the start of an episode.

We use goal-conditioned reinforcement learning to produce a policy $\pi_g(\cdot \mid O_V)$ that maximizes the expected reward $\mathbb{E}[\sum_{t=0}^{H} \gamma^t (r_t^e + \beta r_t^i)]$, where H is the horizon, r_t^e is the extrinsic reward, r_t^i is the intrinsic reward, and β is a tuned hyperparameter. The intrinsic reward is goal-agnostic and is computed with access to either O_V or O_L.

4 Method

Our method builds on two popular exploration algorithms: Random Network Distillation (RND; Burda et al. [2018b]) and Never Give Up (NGU; Badia et al. [2020b]). These algorithms were chosen to demonstrate the value of language under two different exploration paradigms. While both methods reward visiting novel states, they differ on several dimensions: the novelty horizon (lifetime versus episodic), how the history of past visited states is retained (parametric versus non-parametric), and how states are represented (random features versus learned controllable states).

Name	Trainable Network	Target Function
Vis-RND	$f_V : O_V \to \mathbb{R}^k$	randomly initialized, fixed $\hat{f}$
Lang-RND	$f_L : O_L \to \mathbb{R}^k$	randomly initialized, fixed $\hat{f}$
LD	$f_C : O_V \to O_L$	O_L from language oracle

Table 1: Summary of family of RND-inspired methods. Intrinsic reward is derived from the prediction error between the outputs of the trainable network and some target quantity. The rightmost column refers to how the targets are generated.

Name	Embedding Type	Required Input
Vis-NGU	Controllable State	Vision
Lang-NGU	BERT	Language
	$\text{CLIP}_{\text{text}}$	Language
	ALM_{text}	Language
LSE-NGU	$\text{CLIP}_{\text{image}}$	Vision
	$\text{ALM}_{\text{image}}$	Vision

Table 2: Summary of NGU variants.

4.1 Random Network Distillation (RND)

Our RND-inspired family of methods rewards lifetime novelty. Generically, the intrinsic reward is derived from the prediction error between a trainable network and some target value (Table 1). The trainable network is learned jointly with the policy network, although they do not share any parameters. As the agent trains over the course of its lifetime, the prediction error for frequently-visited states decreases, and the associated intrinsic reward diminishes consequently. Intuitively, the weights of the trainable network implicitly stores the state visitation counts.

For clarity, we refer to the baseline published by Burda et al. [2018b] as **Vis-RND**. The trainable network in Vis-RND $f_V : O_V \to \mathbb{R}^k$ maps the visual state to random features. The random features are produced by a fixed, randomly initialized network $\hat{f}_V$. Both f_V and $\hat{f}_V$ share the same architecture: a ResNet followed by a MLP. The intrinsic reward is the mean squared error $\|f_V(O_V) - \hat{f}_V(O_V)\|^2$.

Lang-RND is a variant in which the trainable network $f_L : O_L \to \mathbb{R}^k$ maps the oracle caption to random features. The f_L and $\hat{f}_L$ architectures are the same, consisting of a LSTM and MLP. The intrinsic reward is the mean squared error between the outputs of f_L and $\hat{f}_L$.

LD is loosely inspired by RND in that the novelty signal comes from the prediction error. However, instead of learning to produce random features, the trainable network learns to caption the visual state, i.e. $f_C : O_V \to O_L$. The network architecture consists of a CNN encoder and LSTM decoder. The intrinsic reward is the negative log-likelihood between the output of f_C and the oracle caption. In LD, the exploration dynamics not only depends on how frequently states are visited but also the alignment between language and the visual world. We leverage LD in Section 6.1 to test whether this caption-denoted alignment is necessary for directing semantic exploration.

4.2 Never Give Up (NGU)

To more clearly isolate our impact, we use a simplified version of the full NGU agent, which encourages only episodic novelty. State representations along the trajectory are written to a non-parametric episodic memory buffer. The intrinsic reward reflects how novel the current state is relative to the states visited so far in the episode. Novelty is a function of the L2 distances between the current state and the representations stored in the memory buffer. Intrinsic reward is higher for larger distances.

More details can be found in the original paper; however, we differ in two key ways. While Badia et al. [2020a] proposes learning a family of policy networks that are capable of different levels of exploration, we train one policy network that maximizes reward $r = r_e + \beta r_i$ for a fixed hyperparameter β. We also fix the long-term novelty modulator α to be 1, essentially removing it.

Since the intrinsic reward relies on comparing state representations from the buffer, changing the embedding function greatly influences exploration (Table 2). The baseline method, **Vis-NGU**, uses a k-dimensional controllable state taken from an inverse dynamics model. The inverse dynamics model is trained jointly with the policy, but the two networks do not share any parameters.

Lang-NGU relies on the language oracle. It uses a pretrained frozen language encoder to embed the oracle caption O_L. We compare language embeddings from BERT [Devlin et al., 2018], CLIP [Radford et al., 2021], Small-ALM, and Med-ALM. The ALM models are trained on the ALIGN dataset [Jia et al., 2021] but use different architectures. Small-ALM uses a ResNet-50 image encoder [He et al., 2016]; Med-ALM uses a 377M parameter NFNet [Brock et al., 2021]. The language backbones are based on BERT and are all in the range of 70-90M parameters. We do not finetune on

environment-specific data; this preserves the real world knowledge acquired during pretraining and demonstrates its benefit without requiring any environment-specific language captions.

LSE-NGU does not use the language oracle. It uses a pretrained frozen image encoder to embed the visual observation O_V. We use the image encoder from CLIP or ALM, which are trained on captioning datasets to produce outputs that are close to the corresponding language embeddings. The human-generated captions structure the visual embedding space to reflect the features most pertinent to humans, so the output can be thought of as Language Supervised Embedding (LSE).

The primary benefit of LSE-NGU is that it can be applied to environments without a language oracle or language annotations. CLIP and ALM are trained on real world data, so they would work best on realistic 3D environments. However, we imagine that the dataset or pretraining process can be tailored to maximize transfer to a desired target environment.

5 Experimental Setup

5.1 Environments

Previous exploration work benchmarked algorithms on video games, such as 2D grid-world MiniHack and Montezuma's Revenge, or 3D first-person shooter Vizdoom. In this paper, we focus on first-person 3D environments that are simulated with the Unity physics engine. Our 3D environments are meant to mimic familiar scenes from the real world (Figure 2).

Playroom Our first domain, Playroom, is a randomly-generated house containing everyday household items (e.g. bed, bathtub, tables, chairs, toys). The agent's action set consists of 46 discrete actions that involve locomotion primitives and object manipulation, such as holding and rotating.

We study two settings in Playroom. In the first setting, the agent is confined to a single room with 3-5 objects and is given a `lift` or `put` instruction. At the start of an episode, the set of objects are sampled from a larger set of everyday objects (i.e. a candle, cup, hairdryer). Object colors and sizes are also randomized, adding superficial variance to different semantic categories. The instructions take the form: `"Lift a <object>"` or `"Put a <object> on a {bed, tray}"`. With a `lift` goal, the episode ends with reward 1 or 0 whenever any object is lifted. With a `put` goal, the episode ends with reward 1 when the condition is fulfilled. This setting tests spatial rearrangement skills.

In the second setting, the agent is placed in a house with 3-5 different rooms, and is given a `find` instruction of the form `"Find a {teddy bear, rubber duck}"`. Every episode, the house is randomly generated with the teddy and duck hidden amongst many objects, furniture, and decorations. The target objects can appear in any room— either on the floor, on top of tables, or inside bookshelves. The agent is randomly initialized and can travel throughout the house and freely rearrange objects. The episode ends early with reward 1 when the agent pauses in front of the desired object. The `find` task requires navigation/search skills and tests the ability to ignore the numerous distractor objects.

City Our second domain, City, is an expansive, large-scale urban environment. Each episode, a new map is generated; shops, parks, and buildings are randomly arranged in city blocks. Daylight is simulated, such that the episode starts during the morning and ends at nighttime. The agent is randomly initialized and is instructed to "explore the city." It is trained to maximize its intrinsic reward and can take the following actions: `move_{forward,backward,left,right}`, `look_{left,right}`, and `move_forward_and_look_{left,right}`. We divide up the map into a 32×32 grid and track how many unique bins are visited in an episode.

Additionally, City does not provide explicit visual or verbal signage to disambiguate locations. As such, systematic exploration is needed to maximize coverage. In contrast to Playroom, City tests long horizon credit assignment. A Playroom episode lasts only 600 timesteps, whereas a City episode lasts 11,250 and requires hundreds of timesteps to fully traverse the map even once. The area of the City covers about 1.5km, which models a 2-by-2 grid of real world blocks.

5.2 Captioning Engine

We equip the environment with a language oracle that generates language descriptions of the scene, O_L, based on the Unity state, s (Figure 2). In Playroom, the caption describes if and how the

agent interacts with objects and lists what is currently within the agent's visual field. In City, O_L generally describes the object that the agent is directly looking at, although the captions themselves do not disambiguate the agent's locations. Since these captions are generated from a Unity state, these descriptions may not be as varied or rich as a human's, but they can nonetheless be generated accurately and reliably, and at scale.

5.3 Training Details

At test time, the agent receives image observation O_V and language-specified goal G. The policy network never requires caption O_L to act. During training, the exploration method calculates the intrinsic reward from O_L or O_V.

We show that language-based exploration is compatible with both policy gradient and Q-learning algorithms. We use Impala [Espeholt et al., 2018] on Playroom and R2D2 on City [Kapturowski et al., 2018]. Q-learning is more suitable for the City, because the action space is more restricted compared to the one needed for Playroom tasks.

For both environments, the agent architecture consists of an image ResNet encoder and a language LSTM encoder that feed into a memory LSTM module. The policy and value heads are MLPs that receive the memory state as input. If the exploration method requires additional networks, such as the trainable network in RND or inverse dynamics model in NGU, they do not share any parameters with the policy or value networks. Hyperparameters and additional details are found in Appendix A.

6 Results

6.1 Motivation: Language is a Meaningful Abstraction

We share the intuition with other work [e.g. Mu et al., 2022] that language can be helpful for exploration. We design a set of experiments to show how and why this may be the case. Our analysis follows the desiderata outlined by Burda et al. [2018a]—prediction-error exploration ought to use a feature space that filters irrelevant information (compact) and contains necessary information (sufficient). Burda et al. [2018a] specifically studies RND and notes that the random feature space, the outputs of the random network, may not fully satisfy either condition. As such, we use the language variants of RND to frame this discussion.

We hypothesize that language abstractions are useful, because they (1) create a coarser state space and (2) divide the state space in a way that meaningfully aligns with the visual world (i.e. using semantics). First, if language provides a coarser state space, then the random feature space becomes more compact, and we can expect to see better exploration. We observe that Lang-RND significantly outperforms Vis-RND (Figure 4a). Lang-RND learns the `lift` task 33% faster and solves the `put` task before Vis-RND is able to get off the ground.

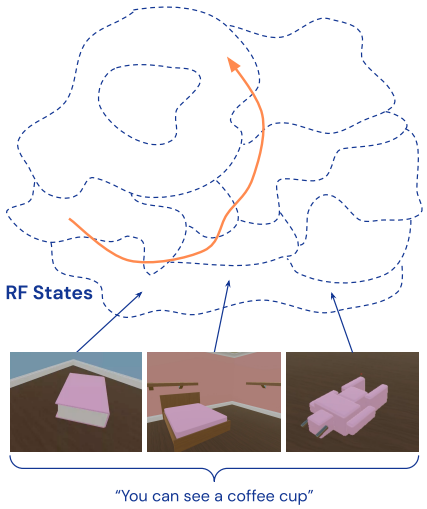
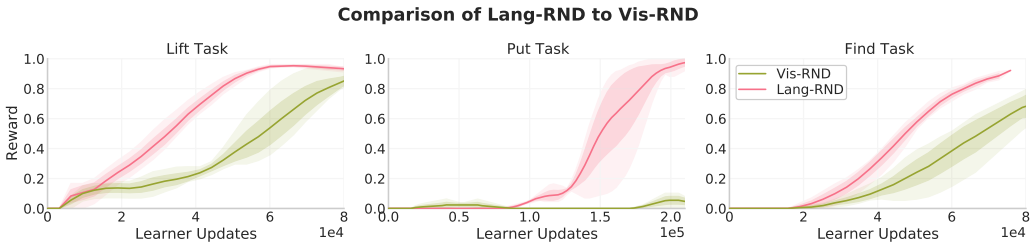
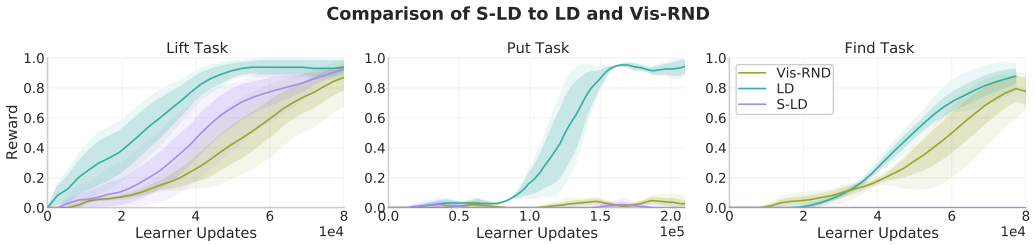


Figure 3: The dotted lines correspond to state abstractions given by the shuffled $\hat{f}_S$ used in S-LD. The states are grouped together based on similarities in the visual random feature space and assigned a label. Exploring in this shuffled space is less effective than exploring with semantically-meaningful abstractions shown in Figure 1.

Role of Semantics in Language-Based Exploration Second, we also ask whether semantics – that is *how* language divides up the state space – is critical for effective exploration. We use LD to test this hypothesis, precisely because the exploration in LD is motivated by modeling the semantic relationship between language and vision. We compare LD to a shuffled variation (S-LD), where we replace the particular semantic state abstraction that language offers with a statistically-matched



(a) Lang-RND outperforms Vis-RND by creating a coarser, more compact state space.



(b) LD outperforms S-LD. It is important how language abstractions carve up the state space.

Figure 4: Our comparisons demonstrate that language is useful for exploration, because it outlines a more abstract, semantically-meaningful state space. Results are shown with a 95% confidence band.

randomized abstraction (Figure 3). S-LD is similar to LD; the intrinsic reward is the prediction error of the captioning network. However, instead of targeting the language oracle output, in S-LD the network is trained to produce a different target caption $\widetilde{O_L}$ that may not correctly label the image. $\widetilde{O_L}$ is produced by a fixed, random mapping $\hat{f}_S : O_V \to \widetilde{O_L}$. $\hat{f}_S$ is constrained such that the marginal distributions $P(O_L) \approx P(\widetilde{O_L})$ are matched under trajectories produced by π_{LD}. Details on how S-LD is constructed can be found in Appendix A.4.

Thus, whereas the LD captions parcel up state space in a way that reflects the abstractions that language offers, the randomized mapping $\hat{f}_S$ parcels up the state space in a way that abstracts over random features of the visual space (Figure 3). We also control for the degree of compactness and coarseness of the resulting representation by maintaining the same marginal distribution of captions.

If semantics is crucial for exploration, then we expect to see LD outperform S-LD. This indeed holds experimentally (Figure 4b). We can also view these results under the Burda et al. [2018a] framework—$\hat{f}_S$ produces a compact feature space that may not be sufficient. The S-LD abstractions can group together visually similar, but semantically distinct states. One randomly sampled caption cannot sufficiently capture all the information in this group in a way and in a manner that is consistent with other shuffled captions. This may explain why S-LD learns faster than Vis-RND on the simpler `lift` task but fails on the more complex `put` and `find` tasks. The S-LD experiments imply that language abstractions are helpful for exploration because they expose not only a more compact, but also a more semantically meaningful state space.

6.2 Pretrained Vision-Language Representations Improve Exploration

Having shown in what ways language can be helpful for exploration, we now demonstrate how to practically incorporate pretrained visual-language representations within an RL agent. Such representations reap the same benefits as explicit language abstractions but have the further advantage of not relying on a language oracle provided by the environment, as we can directly make use of embeddings pretrained on captioning datasets. Because the intrinsic reward signal of NGU is derived directly from its learned embeddings, it serves as a natural baseline against which to compare, as we can simply switch the learned embeddings from NGU with those from the pretrained model. We evaluate three variants of NGU with different embeddings: (1) learned features from an inverse dynamics model (Vis-NGU, baseline), (2) pretrained text embeddings that require a language oracle (Lang-NGU), and (3) pretrained vision embeddings that do not require an oracle (LSE-NGU), but

nevertheless capture language semantics. See section 4.2 for details on the specific pretrained embeddings used. All results are shown with a 95% confidence interval.

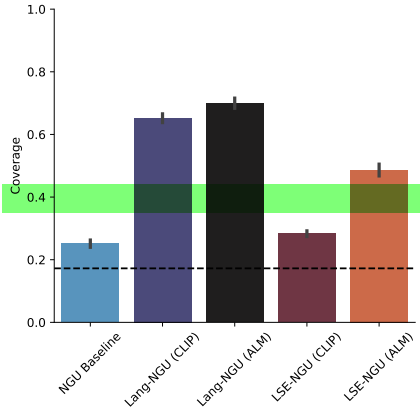


Figure 5: Coverage of City (number of bins reached on map) by variants of NGU agents using different state representations for exploration, normalized by coverage of an agent with access to ground-truth information (the NGU state representation is the global coordinate of the agent location). The dashed line indicates coverage of a uniform random policy. Error bars indicate standard error of the mean, calculated over 5 replicas. See Table 6 for absolute coverage numbers.

City We first compare how representations affect performance in a pure exploration setting. With no extrinsic reward, the agent is motivated solely by the NGU intrinsic reward to explore the City environment. We report how many unique areas the agent visits in an episode in Figure 5. All pretrained representations lead to significantly better exploration than the controllable states. Lang-NGU, which uses text embeddings of O_L, visits an area up to 3 times larger. Even without querying a language oracle, LSE-NGU achieves up to 2 times the coverage.

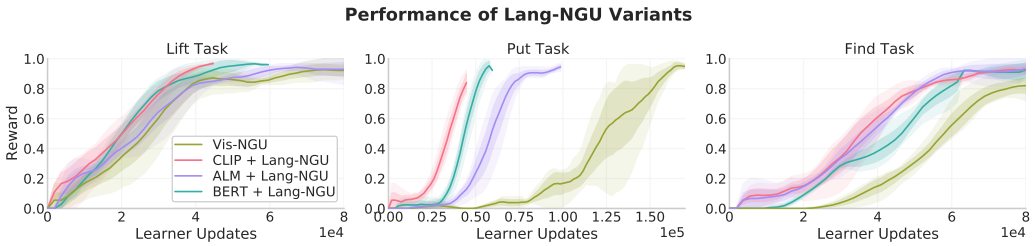
We note that optimizing for coverage only requires knowledge of an agent's global location, rather than generic scene understanding. Although vision-language representations likely contain knowledge more helpful for the latter, they still lead to improvements in City, simply because meaningful exploration is inherently semantic.

Playroom We show that pretrained vision-language representations significantly speed up learning across all Playroom tasks (Figure 6). Depending on the specific pretrained model, the various Lang-NGU agents improve sample efficiency by 50-70% on the `lift` and `put` tasks and 18-38% on the `find` task. Without accessing a language oracle, LSE-NGU variants also outperform baseline Vis-NGU. While pretrained embeddings significantly improve sample efficiency across all settings, the benefits are most pronounced for the `lift` and `put` tasks. This may be due to the fact that language is particularly well-suited for describing object relationships and interactions and the tasks center around object interactions.

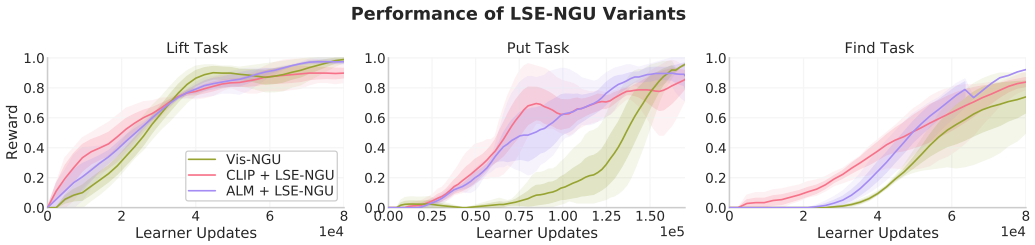
To that point, we measure agent-object interactions over the course of training. The best LSE-NGU agent and all Lang-NGU agents learn to foveate on and hold objects within 40k learning updates, whereas Vis-NGU agent takes 2 times longer to do so with the same frequency (Figure 7 and Appendix Figure 10). The LSE-NGU agent using CLIP embeddings learns object interactions at the same rate as Lang-NGU agents, showing that both the pretrained vision and language representations are similarly effective for encouraging object interactions. We note that there is some variability between the CLIP and ALM LSE-NGU agents. One consideration is that CLIP-style models are limited in their ability to understand complex multi-object scenes, such as those frequently encountered in the `find` task. However, as the performance of pretrained visual-language models improve, we can expect to see those same benefits transfer to this method and drive even better exploration.

7 Discussion

We have shown that language abstractions and pretrained vision-language representations improve the sample efficiency of existing exploration methods. This benefit is seen across on-policy and off-policy algorithms (Impala and R2D2), different 3D domains (Playroom and City), and various task specifications (intrinsically motivated navigation, lifting/putting, and searching). Furthermore, we carefully designed control experiments to understand how language contributes to better exploration. Our results are consistent with perspectives from cognitive science on the utility of human language—language is powerful because it groups together semantically similar states and refers to them with the same caption. Finally, we note that using pretrained vision-language representations to embed visual observations enables more effective exploration without requiring language during agent training. This is vital for scaling to to environments that do not have a language oracle or annotations.



(a) Representations come from text embeddings of O_L.



(b) Representations come from image embeddings, without using oracle language.

Figure 6: NGU agents that use pretrained representations (Lang-NGU and LSE-NGU) learn faster than those that use the controllable state (Vis-NGU).

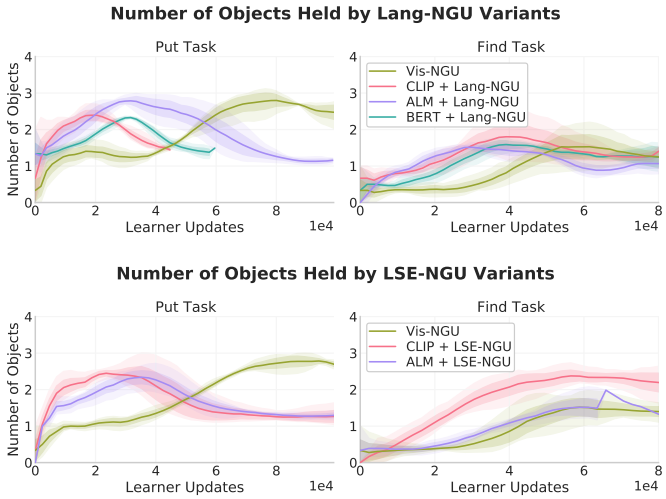


Figure 7: Lang-NGU and LSE-NGU agents learn to hold objects earlier in training compared to Vis-NGU agent. The benefit is larger for the `put` task, where the extrinsic reward also reinforces object interaction.

Limitations & future directions We highlight several avenues that can extend from our work. First, additional ablations may provide a more comprehensive understanding of how language abstractions affect representations. This could include comparing different types of captions offering varying levels of detail, or task-dependent descriptions. A comparison between pretrained vision-language representations and pretrained ImageNet representations would also be of interest. Because ImageNet representations are optimized for classification rather than encoding objects and relationship, we expect that ImageNet representations would perform more poorly, as shown by Du et al. [2021] and Parisi et al. [2022].

Second, it would be useful to investigate how to improve pretrained vision-language representations for exploration by finetuning on relevant datasets. The semantics of the dataset could even be tailored to task-specific abstractions to increase the quality of the learnt representations. Such approaches

would potentially enable our method to be applied to environments that are farther from the pretraining distribution, such as Atari.

Finally, we note that the benefits of LSE-NGU and similar methods are limited. Current pretrained vision-language models are susceptible to domain shift on virtual environments and have been shown to be less effective on multi-object scenes. Thus, larger or future iterations of pretrained models would presumably improve quality of these representations and lead to even more effective exploration.

Acknowledgments and Disclosure of Funding

We would like to thank Iain Barr for ALM models and Nathaniel Wong and Arthur Brussee for the Playroom environment. For the City environment, we would like to thank Nick Young, Tom Hudson, Alex Platonov, Bethanie Brownfield, Sarah Chakera, Dario de Cesare, Marjorie Limont, Benigno Uria, Borja Ibarz and Charles Blundell. Moreover, for the City, we would like to extend our special thanks to Jayd Matthias, Jason Sanmiya, Marcus Wainwright, Max Cant and the rest of the Worlds Team.

References

A. P. Badia, B. Piot, S. Kapturowski, P. Sprechmann, A. Vitvitskyi, Z. D. Guo, and C. Blundell. Agent57: Outperforming the atari human benchmark. In *International Conference on Machine Learning*, pages 507–517. PMLR, 2020a.

A. P. Badia, P. Sprechmann, A. Vitvitskyi, D. Guo, B. Piot, S. Kapturowski, O. Tieleman, M. Arjovsky, A. Pritzel, A. Bolt, et al. Never give up: Learning directed exploration strategies. *arXiv preprint arXiv:2002.06038*, 2020b.

M. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016.

A. Brock, S. De, S. L. Smith, and K. Simonyan. High-performance large-scale image recognition without normalization. In *International Conference on Machine Learning*, pages 1059–1071. PMLR, 2021.

Y. Burda, H. Edwards, D. Pathak, A. Storkey, T. Darrell, and A. A. Efros. Large-scale study of curiosity-driven learning. *arXiv preprint arXiv:1808.04355*, 2018a.

Y. Burda, H. Edwards, A. Storkey, and O. Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018b.

A. Campero, R. Raileanu, H. Küttler, J. B. Tenenbaum, T. Rocktäschel, and E. Grefenstette. Learning with amigo: Adversarially motivated intrinsic goals. *arXiv preprint arXiv:2006.12122*, 2020.

D. S. Chaplot, D. P. Gandhi, A. Gupta, and R. R. Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. *Advances in Neural Information Processing Systems*, 33: 4247–4258, 2020.

C. Colas, P. Fournier, M. Chetouani, O. Sigaud, and P.-Y. Oudeyer. Curious: intrinsically motivated modular multi-goal reinforcement learning. In *International conference on machine learning*, pages 1331–1340. PMLR, 2019.

C. Colas, T. Karch, N. Lair, J.-M. Dussoux, C. Moulin-Frier, P. Dominey, and P.-Y. Oudeyer. Language as a cognitive tool to imagine goals in curiosity driven exploration. *Advances in Neural Information Processing Systems*, 33:3761–3774, 2020.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Y. Du, C. Gan, and P. Isola. Curious representation learning for embodied intelligence. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10408–10417, 2021.

A. Ecoffet, J. Huizinga, J. Lehman, K. O. Stanley, and J. Clune. First return, then explore. *Nature*, 590(7847):580–586, 2021.

L. Espeholt, H. Soyer, R. Munos, K. Simonyan, V. Mnih, T. Ward, Y. Doron, V. Firoiu, T. Harley, I. Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, pages 1407–1416. PMLR, 2018.

S. Flennerhag, J. X. Wang, P. Sprechmann, F. Visin, A. Galashov, S. Kapturowski, D. L. Borsa, N. Heess, A. Barreto, and R. Pascanu. Temporal difference uncertainties as a signal for exploration. *arXiv preprint arXiv:2010.02255*, 2020.

P. Goyal, S. Niekum, and R. J. Mooney. Using natural language for reward shaping in reinforcement learning. *arXiv preprint arXiv:1903.02020*, 2019.

H. P. Grice. Logic and conversation. In *Speech acts*, pages 41–58. Brill, 1975.

M. Hahn, D. Jurafsky, and R. Futrell. Universals of word order reflect optimization of grammars for efficient communication. *Proceedings of the National Academy of Sciences*, 117(5):2347–2353, 2020.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

R. Houthooft, X. Chen, Y. Duan, J. Schulman, F. De Turck, and P. Abbeel. Vime: Variational information maximizing exploration. *Advances in neural information processing systems*, 29, 2016.

C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. Le, Y.-H. Sung, Z. Li, and T. Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021.

S. Kapturowski, G. Ostrovski, J. Quan, R. Munos, and W. Dabney. Recurrent experience replay in distributed reinforcement learning. In *International conference on learning representations*, 2018.

A. Khandelwal, L. Weihs, R. Mottaghi, and A. Kembhavi. Simple but effective: Clip embeddings for embodied ai. *arXiv preprint arXiv:2111.09888*, 2021.

S. Li, X. Puig, Y. Du, C. Wang, E. Akyurek, A. Torralba, J. Andreas, and I. Mordatch. Pre-trained language models for interactive decision-making. *arXiv preprint arXiv:2202.01771*, 2022.

G. Lupyan. What do words do? toward a theory of language-augmented thought. In *Psychology of learning and motivation*, volume 57, pages 255–297. Elsevier, 2012.

G. Lupyan, D. H. Rakison, and J. L. McClelland. Language is not just for talking: Redundant labels facilitate learning of novel categories. *Psychological science*, 18(12):1077–1083, 2007.

M. C. Machado, M. G. Bellemare, and M. Bowling. Count-based exploration with the successor representation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):5125–5133, 2020.

J. Martin, S. N. Sasikumar, T. Everitt, and M. Hutter. Count-based exploration in feature space for reinforcement learning. *arXiv preprint arXiv:1706.08090*, 2017.

S. Mirchandani, S. Karamcheti, and D. Sadigh. Ella: Exploration through learned language abstraction. *Advances in Neural Information Processing Systems*, 34, 2021.

J. Mu, V. Zhong, R. Raileanu, M. Jiang, N. Goodman, T. Rocktäschel, and E. Grefenstette. Improving intrinsic exploration with language abstractions. *arXiv preprint arXiv:2202.08938*, 2022.

P.-Y. Oudeyer and F. Kaplan. What is intrinsic motivation? a typology of computational approaches. *Frontiers in neurorobotics*, 1:6, 2009.

P.-Y. Oudeyer, F. Kaplan, and V. V. Hafner. Intrinsic motivation systems for autonomous mental development. *IEEE transactions on evolutionary computation*, 11(2):265–286, 2007.

S. Parisi, A. Rajeswaran, S. Purushwalkam, and A. Gupta. The unsurprising effectiveness of pre-trained vision models for control. *arXiv preprint arXiv:2203.03580*, 2022.

D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR, 2017.

D. Pathak, D. Gandhi, and A. Gupta. Self-supervised exploration via disagreement. In *International conference on machine learning*, pages 5062–5071. PMLR, 2019.

S. Racaniere, A. Lampinen, A. Santoro, D. Reichert, V. Firoiu, and T. Lillicrap. Automated curriculum generation through setter-solver interactions. In *International conference on learning representations*, 2019.

A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

M. Reid, Y. Yamada, and S. S. Gu. Can wikipedia help offline reinforcement learning? *arXiv preprint arXiv:2201.12122*, 2022.

N. Savinov, A. Raichuk, R. Marinier, D. Vincent, M. Pollefeys, T. Lillicrap, and S. Gelly. Episodic curiosity through reachability. *arXiv preprint arXiv:1810.02274*, 2018.

J. Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. In *Proc. of the international conference on simulation of adaptive behavior: From animals to animats*, pages 222–227, 1991.

E. Schwartz, G. Tennenholtz, C. Tessler, and S. Mannor. Language is power: Representing states using natural language in reinforcement learning. *arXiv preprint arXiv:1910.02789*, 2019.

A. L. Strehl and M. L. Littman. An analysis of model-based interval estimation for markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.

R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

T. Xiao, I. Radosavovic, T. Darrell, and J. Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022.

D. Zha, W. Ma, L. Yuan, X. Hu, and J. Liu. Rank the episodes: A simple approach for exploration in procedurally-generated environments. *arXiv preprint arXiv:2101.08152*, 2021.

T. Zhang, P. Rashidinejad, J. Jiao, Y. Tian, J. E. Gonzalez, and S. Russell. Made: Exploration via maximizing deviation from explored regions. *Advances in Neural Information Processing Systems*, 34, 2021a.

T. Zhang, H. Xu, X. Wang, Y. Wu, K. Keutzer, J. E. Gonzalez, and Y. Tian. Noveld: A simple yet effective exploration criterion. *Advances in Neural Information Processing Systems*, 34, 2021b.

A Training Details

We use a distributed RL training setup with 256 parallel actors. For Impala agents, the learner samples from a replay buffer that acts like a queue. For R2D2 agents, the learner samples from a replay buffer using prioritized replay. Training took 8-24 hours per experiment on a 2×2 TPUv2.

All agents share the same policy network architecture and hyperparameters (Table 3). We use an Adam optimizer for all exerpiments. The hyperparameters used to train Impala [Espeholt et al., 2018] and R2D2 [Kapturowski et al., 2018] are mostly taken from the original implementations. For Impala, we set the unroll length to 128, the policy network cost to 0.85, and state-value function cost to 1.0. We also use two heads to estimate the state-value functions for extrinsic and intrinsic reward separately. For R2D2, we set the unroll length to 100, burn-in period to 20, priority exponent to 0.9, Q-network target update period to 400, and replay buffer size to 10,000.

Table 3: Common architecture and hyperparameters for all agents.

Setting	Value
Image resolution:	96x72x3
Number of action repeats:	4
Batch size:	32
Agent discount γ:	0.99
Learning rate:	0.0003
ResNet num channels (policy):	16, 32, 32
LSTM hidden units (memory):	256

A.1 Hyperparameters for RND-Inspired Agents

For the RND-inspired exploration agents, the hyperparameters for training the trainable network are taken from the original implementation [Burda et al., 2018b]. We perform a random hyperparameter search over a range of values for the intrinsic reward scale, V-Trace entropy cost, and the learning rate for the trainable network. The values can be found in Table 4. We also normalize all intrinsic reward with the rolling mean and standard deviation.

Table 4: Hyperparameters for the family of RND-inspired agents.

Environment	Setting	Vis-RND	Lang-RND	LD	S-LD
Playroom `lift`, `put`	Intrinsic reward scale β	1.4e-4	3.2e-6	1.1e-5	1.4e-6
	Entropy cost (V-Trace)	1.2e-4	8.1e-5	2.7e-5	7.6e-5
	Learning rate (trainable network)	1.2e-3	5.3e-4	1.7e-3	1.8e-4
Playroom `find`	Intrinsic reward scale β	9.9e-5	7.2e-4	4.1e-5	4.1e-5
	Entropy cost (V-Trace)	4.4e-5	8e-5	4.3e-5	4.3e-5
	Learning rate (trainable network)	5.4e-4	1e-3	2e-3	2e-3

A.2 Hyperparameters for NGU Variants

We use a simplified version of the full NGU agent as to focus on the episodic novelty component. One major difference is that we only learn one value function, associated with a single intrinsic reward scale β and discount factor γ. The discount factor γ is 0.99 and we sweep for β (Table 5). Another major difference is that our lifetime novelty factor α is always set to 1.

The NGU memory buffer is set to 12,000, so that it can always has the capacity to store the the entire episode. The buffer is reset at the start of the episode. The intrinsic reward is calculated from a kernel operation over state representations stored in the memory buffer. We use the same kernel function and associated hyperparameters (e.g. number of neighbors, cluster distance, maximum similarity) found in Badia et al. [2020b].

For Vis-NGU, the 32-dimension controllable states come from a learned inverse dynamics model. The inverse dynamics model is trained with an Adam optimizer (learning rate 5e-4, $\beta_1 = 0.0$, $\beta_2 = 0.95$, ϵ is 6e-6). For the variants, we use frozen pretrained representations from BERT, ALM, or CLIP.

These pretrained embeddings are size 768. We use representations from Med-ALM on Playroom tasks. However, we use Small-ALM on City, where the episode is magnitudes longer, in order to offset the increased inference time of the ALM model.

We notice that it is crucial for the pretrained representations to only be added to the buffer every 8 timesteps. Meanwhile, Vis-NGU adds controllable states every timestep, which is as or more effective than every 8. This may be due to some interactions between the kernel function and the smoothness of the learned controllable states. We also find that normalizing the intrinsic reward, like in RND, is helpful in some settings. We use normalization for Lang-NGU and LSE-NGU on the Playroom tasks.

Table 5: Hyperparameters for the family of NGU agents on the Playroom tasks. All agents except Lang-NGU use a scaling factor of 0.01 in City. Lang-NGU uses 0.1.

Environment	Method	Embedding Type	Intrinsic Reward Scale β	Entropy cost
Playroom `lift`, `put`	Vis-NGU	Controllable State	3.1e-7	6.2e-5
	Lang-NGU	BERT	0.029	2.4e-4
		$\text{CLIP}_{\text{text}}$	0.02	2.3e-4
		ALM_{text}	0.0035	9.1e-5
	LSE-NGU	$\text{CLIP}_{\text{image}}$	0.078	2.9e-5
		$\text{ALM}_{\text{image}}$	0.059	1.7e-4
Playroom `find`	Vis-NGU	Controllable State	3.1e-6	2.6e-5
	Lang-NGU	BERT	0.029	2.4e-4
		$\text{CLIP}_{\text{text}}$	0.0047	4.1e-5
		ALM_{text}	0.013	1.9e-4
	LSE-NGU	$\text{CLIP}_{\text{image}}$	0.0083	6.7e-5
		$\text{ALM}_{\text{image}}$	0.0051	1.2e-4

A.3 Engineering Considerations

Integrating large pretrained models into RL frameworks is nontrivial. High quality pretrained models are orders of magnitudes larger than policy networks, introducing challenges with inference speed. Slow inference not only increases iteration and training times but also may push learning further off-policy in distributed RL setups [e.g. Espeholt et al., 2018, Kapturowski et al., 2018]. We mitigate this issue in Lang-NGU and LSE-NGU by only performing inference computations when it is necessary. We compute the pretrained representations only when they are required for adding to the NGU buffer (i.e. every 8 timesteps).

A.4 S-LD Construction

As explained in Section 6.1, we compare the efficacy of exploration induced by the language distillation (LD) method with that induced by a shuffled variant, S-LD. LD employs an exploration bonus based on the prediction error of a captioning network $f_C : O_V \to O_L$, where the target value is the oracle caption. Our goal with S-LD is to determine whether the efficacy of the exploration is due to the specific abstractions induced by f_C, or whether it is due to some low level statistical property (e.g. the discrete nature of O_L, or the particular marginal output distribution).

We sample a fixed, random mapping $\hat{f_S} : O_V \to O_L$ while trying to match the low-level statistics of f_C. To do this, we construct a (fixed, random) smooth partitioning of O_V into discrete regions, which in turn are each assigned to a unique caption from the output space of f_C. The regions are sized to maintain the same marginal distribution of captions, $P_{\pi_{LD}}(L) \approx P_{\pi_{S-LD}}(L)$.

More precisely, our procedure for constructing $\hat{f_S}$ is as follows. We build $\hat{f_S}$ as the composition of three functions, $\hat{f_S} = g_3 \circ g_2 \circ g_1$:

- $g_1 : O_V \to \mathbb{R}$ is a fixed, random function, implemented using a neural network as in Vis-RND.
- $g_2 : \mathbb{R} \to Cat(K)$ is a one-hot operation, segmenting $g_1(O_V)$ into K non-overlapping, contiguous intervals.

- $g_3 : Cat(K) \rightarrow O_L$ is a fixed mapping from the discrete categories to set of captions produced by the language oracle, f_C.

To ensure that the marginal distribution of captions seen in f_C was preserved in f_S, we first measure the empirical distribution of oracle captions encountered by π_{LD}, a policy trained to maximize the LD intrinsic reward. We denote the observed marginal probability of observing caption l_i as $P_{\pi_{LD}}(O_L = l_i) = q_i$ (where the ordering of captions $l_i \in O_L$ is fixed and random). We then measure the empirical distribution of g_1 under the same action of the same policy, $P_{\pi_{LD}}(g_1(O_V))$, and define the boundaries of the intervals of g_2 by the quantiles of this empirical distribution so as to match the CDF of $\mathbf{q}$, i.e. so that $P_{\pi_{LD}}(g_2 \circ g_1(O_V) = i) = q_i$. Finally, we define g_3 to map from the i^{th} category to the corresponding caption l_i, so that $P_{\pi_{LD}}(g_3 \circ g_2 \circ g_1(O_V) = l_i) = q_i$.

B Additional ablation: Pretrained controllable states

The state representations used by Vis-NGU are trained at the same time as the policy, whereas the pretrained representations used by Lang-NGU and LSE-NGU are frozen during training. To isolate the effect of the knowledge encoded by the representations, we perform an additional experiment where we pretrain the controllable states and freeze them during training. We use the weights of the inverse dynamics model from a previously trained Vis-NGU agent. Figure 8 shows that pretrained Vis-NGU learns at the same rate as Vis-NGU (if not slower). Thus, the increased performance in Lang-NGU and LSE-NGU agents is due to the way the vision-language embeddings are pretrained on captioning datasets. This furthermore suggests that the converged controllable states do not fully capture the knowledge needed for efficient exploration and in fact may even hurt exploration at the start of training by focusing on the wrong parts of the visual observation.

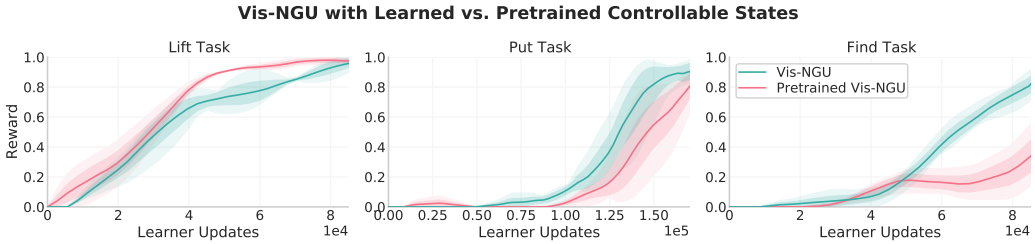
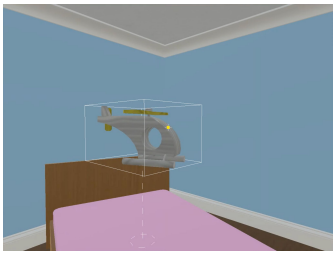


Figure 8: Vis-NGU with a pretrained inverse dynamics model learns slower than the baseline Vis-NGU agent that uses online learned controllable states.

C Additional Figures

You can see two arm chairs, an ottoman, and a shelf.



You are holding a helicopter. You can see a bed.

You are looking at a post office in a light grey stucco shops in a apartment building.

You can see a bed, a shelf, a bookcase, a teddy, and a rubber duck.

You can see a storage tray, a train, a robot, and a car.

You are looking at a blue grey brick offices in a office building.

You can see an armchair, an ottoman, a shelf, and a teddy.

You are holding a potted plant. You can see a storage tray and a bed.

You are looking at a green door in a dark grey painted brick ground floor apartments in a apartment building.

You can see a bathtub and a rubber duck.

You can see a bed, a storage tray, a rocket, a bus, and a mug.

You are looking at a fire hydrant in a park.

Figure 9: Example scenes and associated captions from multi-room Playroom used for the `find` task (left column), single-room Playroom used for the `lift` and `put` tasks (middle column), and City (right column).

Table 6: Mean and standard error of coverage (number of bins reached on map) by variants of NGU agents using different state representations. The City consists of 1024 total bins, although not all are reachable. With the ground truth (continuous) embedding type, the NGU state representation is the global coordinate of the agent location. With the ground truth (discrete) embedding type, the representation is a one-hot encoding of the bins. A non-adaptive, uniform random policy is also included as baseline ('N/A - Random Actions').

Embedding Type	**Coverage (number of bins)**
Ground Truth (continuous)	346 ± 3.2
Ground Truth (discrete)	539 ± 3.5
N/A – Random Actions	60 ± 0.53
NGU with Controllable State	83 ± 6.9
Lang-NGU with CLIP	225 ± 8.9
Lang-NGU with Small-ALM	241 ± 9.0
LSE-NGU with CLIP	100 ± 2.2
LSE-NGU with Small-ALM	162 ± 5.6

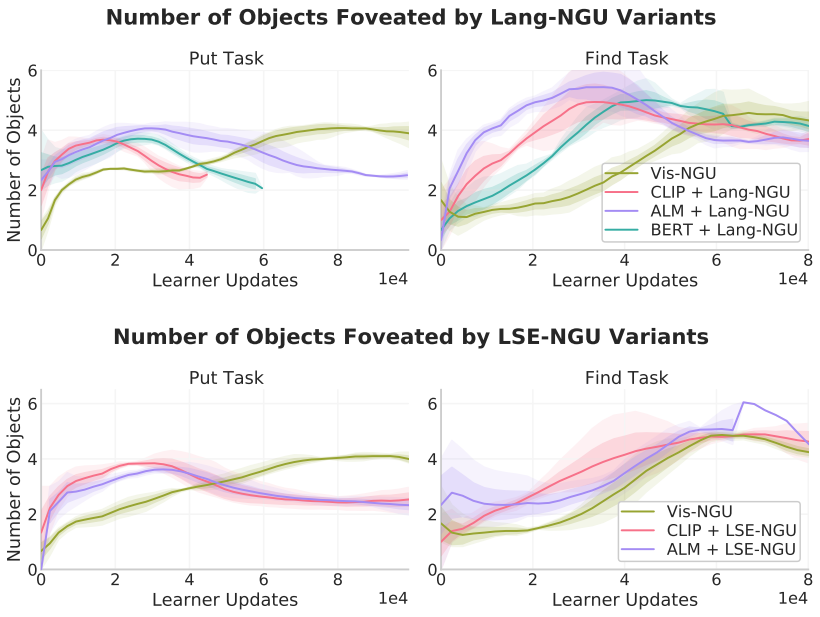


Figure 10: Lang-NGU and LSE-NGU agents learn to foveate on objects earlier in training compared to the Vis-NGU agent.