Pre-Trained Language Models for Interactive Decision-Making

Shuang Li [1*], Xavier Puig[1], Chris Paxton[2], Yilun Du[1], Clinton Wang[1], Linxi Fan[2], Tao Chen[1], De-An Huang[2], Ekin Akyürek[1], Anima Anandkumar[2,3,†], Jacob Andreas[1,†], Igor Mordatch[4,†], Antonio Torralba[1,†], Yuke Zhu[2,5,†]

[1]MIT, [2]Nvidia, [3]Caltech, [4]Google Brain, [5]UT Austin
Junior authors are ordered based on contributions and senior authors[†] are ordered alphabetically.

Overview

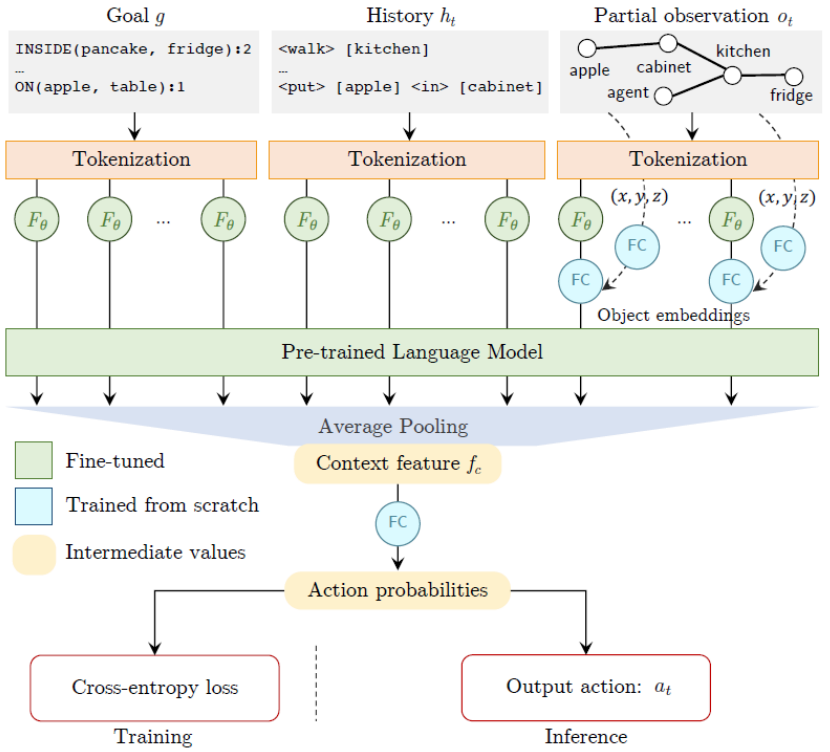
- Pre-trained LLM for decision-making and generalization.
- A new data gathering procedure.

To summarize, our work has four main contributions:

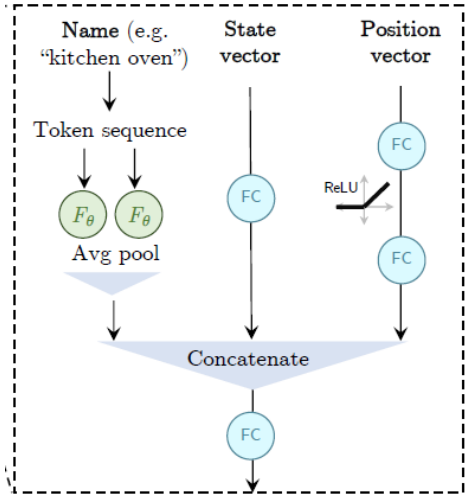
- First, we propose to use **pre-trained LMs as a general scaffold** for interactive decision-making across a variety of environments by converting all policy inputs into sequential data.
- Second, we demonstrate that **language modeling improves combinatorial generalization in policy learning**: initializing a policy with a pre-trained LM substantially improves out-of-distribution performance on novel tasks.
- Third, we integrate an **active data gathering** procedure into the proposed approach to further enable policy learning on environments without using pre-collected expert data.
- Finally, we perform several analyses to explain the generalization capabilities of pre-trained LMs, finding that natural strings are not needed to benefit from LM pre-training, but the sequential input encoding and weight pre-training are important.

Policy Network

- Encoding: translate goal/history into a templated English sentence.
- For observation, add FCs to encode state vector and position vector.
- Language Model: GPT-2 with fine-tuning.



We first examine whether pre-trained LMs provide effective initializers when states and action histories are represented as natural language strings. We encode the inputs to the policy—including observations, goals, and action histories—as sequences of words. These word sequences are passed to the LM (using its pre-trained word embedding layer F_θ) and used to obtain contextualized token representations. Token representations are averaged and used to predict actions. We design a policy network following the general policy framework proposed in Figure 1.



Policy Learning with Active Data Gathering

- Collecting expert data is sometimes challenging.
- Similar to Hindsight Experience Replay (HER)
- Three stages: exploration**, relabeling,** policy update.
- Relabeling: relabel useful sub-trajectory of failure samples for training.

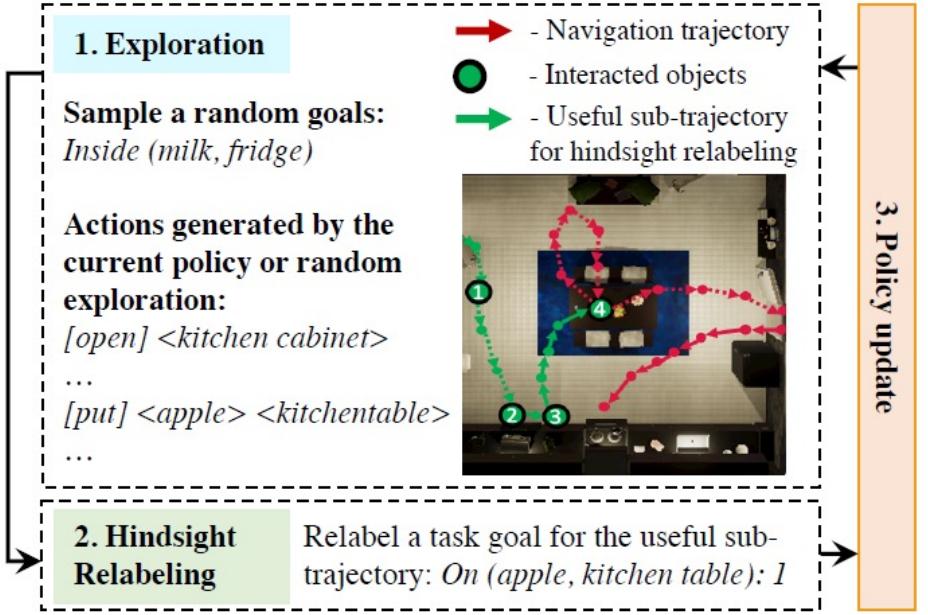


Figure 2: **LID with the active data gathering procedure.** By iteratively repeating the exploration, hindsight relabeling, and policy update, LID with active data gathering can learn an effective policy without using pre-collected expert data.

Experiment

- VitualHome:
- Three aspects: In-Distribution, Novel Scenes, Novel Tasks
- Baselines: Recurrent Network(LSTM), MLP, MLP-1
- BabyAI:
- Baselines: BabyAI-Ori, the method used in original paper.

We use the standard training and test data provided by [16]. In BabyAI, performing well on unseen test tasks with new environment layouts and goals requires combinatorial reasoning.

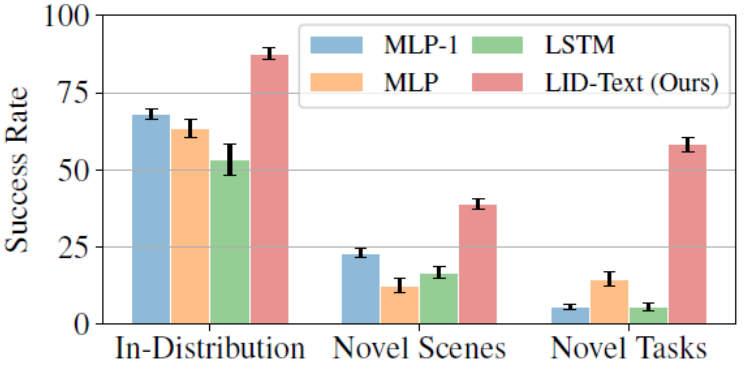


Figure 3: **Comparisons of the proposed method and baselines on VirtualHome.** All the methods are trained on expert data using imitation learning. *MLP-1*, *MLP*, and *LSTM* are baselines without using the pre-trained LM. The proposed method, LID-*Text (Ours)*, outperforms all baselines.

Tasks	Methods	Number of Demos				
		100	500	1K	5K	10K
GoToRedBall	BabyAI-Ori [16]	81.0	96.0	99.0	99.5	99.9
	LID-Text (Ours)	**93.9**	**99.4**	**99.7**	**100.0**	**100.0**
GoToLocal	BabyAI-Ori [16]	55.9	84.3	98.6	**99.9**	99.8
	LID-Text (Ours)	**64.6**	**97.9**	**99.0**	99.5	99.5
PickupLoc	BabyAI-Ori [16]	28.0	58.0	93.3	97.9	**99.8**
	LID-Text (Ours)	**28.7**	**73.4**	**99.0**	**99.6**	**99.8**
PutNextLocal	BabyAI-Ori [16]	**14.3**	16.8	43.4	81.2	97.7
	LID-Text (Ours)	11.1	**93.0**	**93.2**	**98.9**	**99.9**

Table 1: **Success rates on BabyAI tasks.** All the methods are trained on offline expert data using imitation learning. LID-*Text (Ours)* outperforms BabyAI-Ori, the method used in the original paper [16].

Experiment

- Pre-trained Language Model with Active Data Gathering (**LID-ADG**)

	In-Distribution	Novel Scenes	Novel Tasks
Random	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
Goal-Object	0.8 ± 0.5	0.0 ± 0.0	0.4 ± 0.4
PPO	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
DQN+HER	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
LID-ADG (Ours)	$\mathbf{46.7 \pm 2.7}$	$\mathbf{32.2 \pm 3.3}$	$\mathbf{25.5 \pm 4.1}$

Table 2: **Comparisons of methods without using expert data on VirtualHome.** LID-*ADG (Ours)* is the only successful approach.

	In-Distribution	Novel Scenes	Novel Tasks
LID-ADG (Ours)	46.7 ± 2.7	$\mathbf{32.2 \pm 3.3}$	25.5 ± 4.1
PPO (LID-ADG Init)	$\mathbf{53.7 \pm 3.5}$	30.2 ± 3.4	$\mathbf{27.8 \pm 2.7}$
DT (LID-ADG Data)	42.4 ± 1.5	21.6 ± 2.48	16.8 ± 1.0

Table 3: The proposed method with active data gathering, LID-ADG (Ours), can be used as an policy initializer for online RL or a data provider for offline RL.

Analysis

The pre-trained LM policy, fine-tuned on either expert data or actively gathered data, exhibits effective combinatorial generalization. Is this simply because LMs are effective models of relations between natural language descriptions of states and actions [1], or because they provide a more general framework for combinatorial generalization in decision-making? We hypothesize and investigate three possible factors to understand the sources of such combinatorial generalization. We use policies trained on the expert data as an example to explain the experiments.

- **Input Encoding Scheme**: text/index/unnatural(randomly map each token to new token)

Methods	Number of Demos					
	100	500	1K	5K	10K	20K
LID-Text (Ours)	**8.8 ± 1.4**	**22.2 ± 1.7**	26.8 ± 1.0	46.0 ± 1.0	**58.2 ± 1.2**	58.2 ± 1.6
LID-Index (Ours)	6.4 ± 0.6	18.0 ± 3.8	18.8 ± 1.0	45.5 ± 2.1	54.6 ± 0.8	57.8 ± 0.9
LID-Unnatural (Ours)	6.8 ± 1.3	18.6 ± 2.1	**27.0 ± 1.1**	**47.2 ± 1.7**	55.8 ± 0.8	**58.8 ± 0.9**

This result indicates that the effectiveness of pre-trained LMs in compositional generalization is not unique to natural language strings, but can be leveraged from arbitrary encodings, although adapting the model to arbitrary encodings may require more training data.

- **Sequential Input Representation/Favorable Weight Initialization**:

No-Seq: input encoding is not sequential
No-Pretrain: train from scratch
No-FT: no fine-tuning

	In-Distribution	Novel Tasks
LID-Text (Ours)	87.6 ± 1.9	**58.2 ± 2.3**
No-Seq	74.0 ± 2.3	2.0 ± 0.6
No-Pretrain	**90.8 ± 2.0**	47.0 ± 2.8
No-FT	51.2 ± 4.5	17.0 ± 2.9

Do As I Can, Not As I Say: Grounding Language in Robotic Affordances

[1] Michael Ahn*, Anthony Brohan*, Noah Brown*, Yevgen Chebotar*, Omar Cortes*, Byron David*, Chelsea Finn*, Chuyuan Fu†, Keerthana Gopalakrishnan*, Karol Hausman*, Alex Herzog†, Daniel Ho†, Jasmine Hsu*, Julian Ibarz*, Brian Ichter*, Alex Irpan*, Eric Jang*, Rosario Jauregui Ruano*, Kyle Jeffrey*, Sally Jesmonth*, Nikhil J Joshi*, Ryan Julian*, Dmitry Kalashnikov*, Yuheng Kuang*, Kuang-Huei Lee*, Sergey Levine*, Yao Lu*, Linda Luu*, Carolina Parada*, Peter Pastor†, Jornell Quiambao*, Kanishka Rao*, Jarek Rettinghouse*, Diego Reyes*, Pierre Sermanet*, Nicolas Sievers*, Clayton Tan*, Alexander Toshev*, Vincent Vanhoucke*, Fei Xia*, Ted Xiao*, Peng Xu*, Sichun Xu*, Mengyuan Yan†, Andy Zeng*

*Robotics at Google, †Everyday Robots

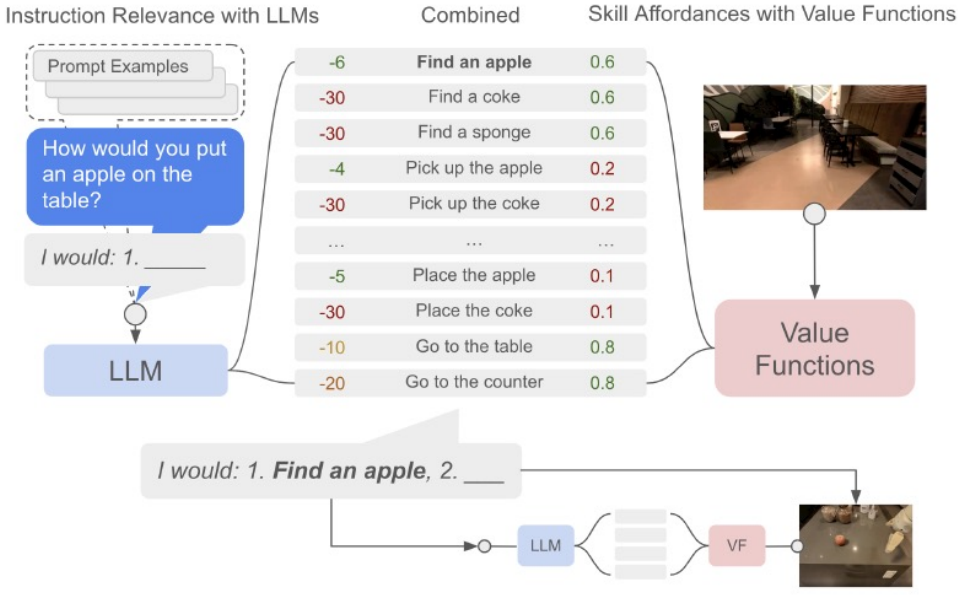
SayCan

- **Problem Statement**:
- Given a natural language instruction i, a set of skills Π, where each skill performs a short task, with a language description ℓ_π, and an affordance function $p(c_\pi|s, \ell_\pi)$.
- **Language-Conditioned Robotic Control Policies**
- Train **a set of low-level skills**, with policy(BC), value function and language description.
- Using LLM and prompts to get $p(\ell_\pi|i, \ell_{\pi_{n-1}}, ..., \ell_{\pi_0})$

Algorithm 1 SayCan

Given: A high level instruction i, state s_0, and a set of skills Π and their language descriptions ℓ_Π

$n = 0, \pi = \emptyset$
while $\ell_{\pi_{n-1}} \neq$ "done" **do**
  $\mathcal{C} = \emptyset$
  for $\pi \in \Pi$ and $\ell_\pi \in \ell_\Pi$ **do**
    $p_\pi^{\text{LLM}} = p(\ell_\pi|i, \ell_{\pi_{n-1}}, ..., \ell_{\pi_0})$ ▷ Evaluate scoring of LLM
    $p_\pi^{\text{affordance}} = p(c_\pi|s_n, \ell_\pi)$ ▷ Evaluate affordance function
    $p_\pi^{\text{combined}} = p_\pi^{\text{affordance}} p_\pi^{\text{LLM}}$
    $\mathcal{C} = \mathcal{C} \cup p_\pi^{\text{combined}}$
  end for
  $\pi_n = \arg\max_{\pi \in \Pi} \mathcal{C}$
  Execute $\pi_n(s_n)$ in the environment, updating state s_{n+1}
  $n = n + 1$
end while



SayCan

- Language-Conditioned Robotic Control Policies
- Train **a set of low-level skills**, with policy(BC), value function(RL) and language description.
- Using LLM and prompt to get $p(\ell_\pi | i, \ell_{\pi_{n-1}}, \ldots, \ell_{\pi_0})$

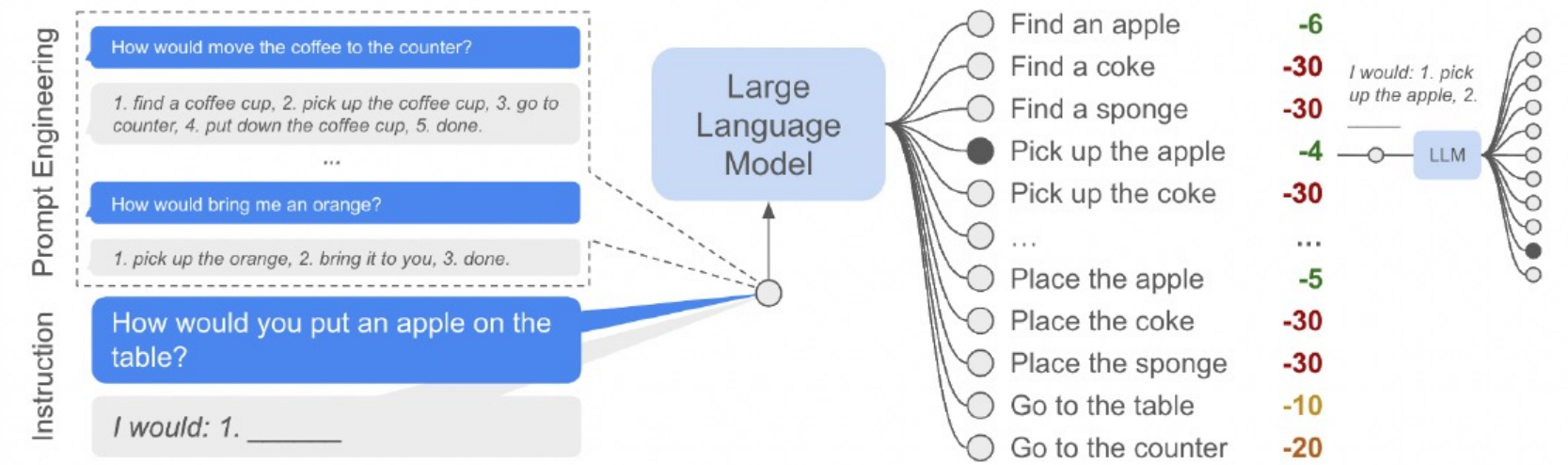


Figure 12: A scoring language model is queried with a prompt-engineered context of examples and the high-level instruction to execute and outputs the probability of each skill being selected. To iteratively plan the next steps, the selected skill is added to the natural language query and the language model is queried again.

Experiment

- Metrics.
- **Plan success rate** (can achieve?)
- **Execution success rate** (achieved?)
- Ask human to justify.
- Ablations
- **LLM**: directly feed instruction into policy net.
- **Value Functions**: "No VF" removes VF; "Gen." uses LLM to scoring.
- **Different LLM**: 540B vs 137B

Instruction Family	Num	Explanation	Example Instruction
NL Single Primitive	15	NL queries for a single primitive	Let go of the coke can
NL Nouns	15	NL queries focused on abstract nouns	Bring me a fruit
NL Verbs	15	NL queries focused on abstract verbs	Restock the rice chips on the far counter
Structured Language	15	Structured language queries, mirror NL Verbs	Move the rice chips to the far counter.
Embodiment	11	Queries to test SayCan's understanding of the current state of the environment and robot	Put the coke on the counter. (starting from different completion stages)
Crowd-Sourced	15	Queries in unstructured formats	My favorite drink is redbull, bring one
Long-Horizon	15	Long-horizon queries that require many steps of reasoning	I spilled my coke on the table, throw it away and bring me something to clean

		Mock Kitchen		Kitchen		No Affordance		No LLM	
		PaLM-SayCan	**PaLM-SayCan**	**PaLM-SayCan**	**PaLM-SayCan**	**No VF**	**Gen.**	**BC NL**	**BC USE**
Family	**Num**	Plan	Execute	Plan	Execute	Plan	Plan	Execute	Execute
NL Single	15	100%	100%	93%	87%	73%	87%	0%	60%
NL Nouns	15	67%	47%	60%	40%	53%	53%	0%	0%
NL Verbs	15	100%	93%	93%	73%	87%	93%	0%	0%
Structured	15	93%	87%	93%	47%	93%	100%	0%	0%
Embodiment	11	64%	55%	64%	55%	18%	36%	0%	0%
Crowd Sourced	15	87%	87%	73%	60%	67%	80%	0%	0%
Long-Horizon	15	73%	47%	73%	47%	67%	60%	0%	0%
Total	101	84%	74%	81%	60%	67%	74%	0%	9%

		PaLM-SayCan		FLAN-SayCan	
Family	**Num**	Plan	Execute	Plan	Execute
NL Single	15	100%	100%	67%	67%
NL Nouns	15	67%	47%	60%	53%
NL Verbs	15	100%	93%	80%	67%
Structured	15	93%	87%	100%	87%
Embodiment	11	64%	55%	64%	55%
Crowd Sourced	15	87%	87%	73%	67%
Long-Horizon	15	73%	47%	47%	33%
Total	101	84%	74%	70%	61%

(2) *Generative*, which uses the generative output of the LLM and then projects each planned skill to its maximal cosine similarity skill via USE embeddings.

Experiment

- Add New Capabilities.
- **Adding Skills**
- SayCan is capable of integrating new skills by simply adding the new skills as options for the LLM and providing accompanying value functions and add an example in the prompt with that skill.
- **Chain of Thought Reasoning**

  A few successful rollouts of the model at evaluation time is shown in Table 4. As we can see, with chain of thought prompting, the model can handle negations and tasks that require reasoning.

- **Multilingual Queries**
- There is almost no performance drop in planning success rate when changing the queries from English to Chinese, French and Spanish.