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

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 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*

Interpreted by:

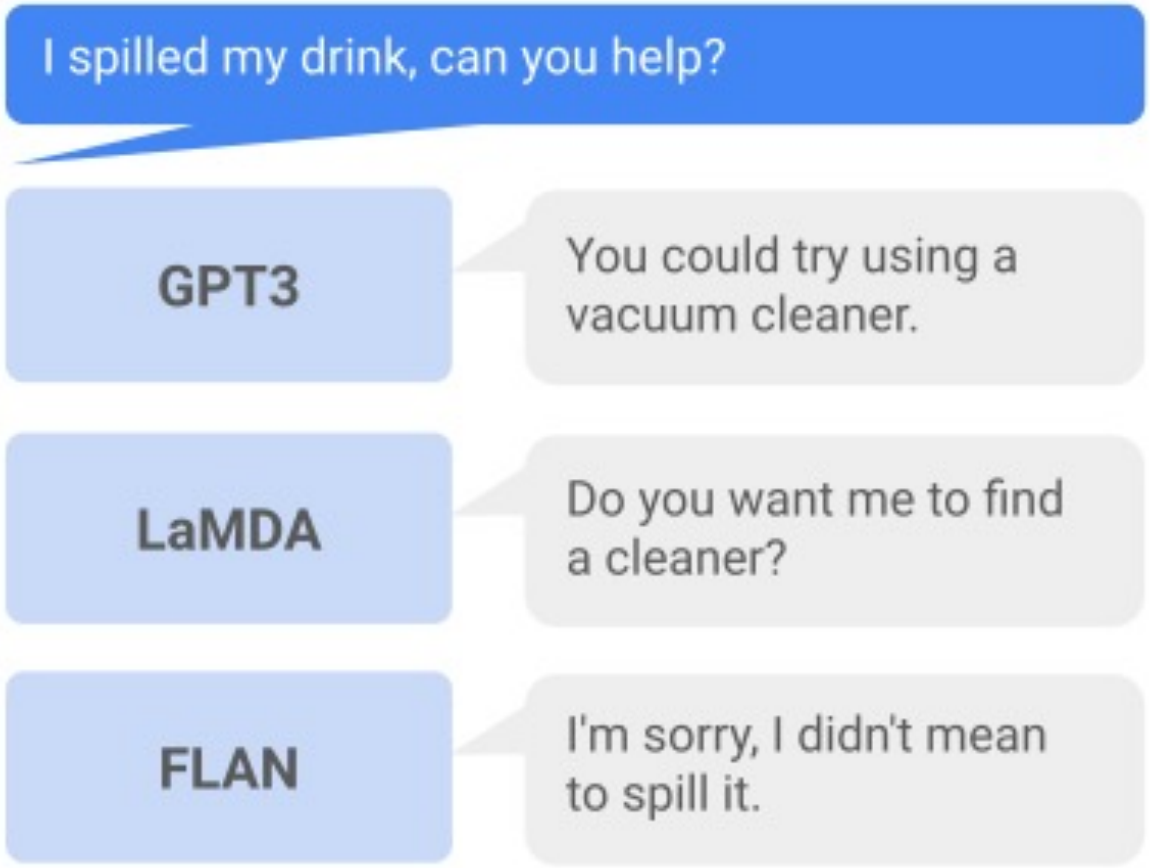
John Tan Chong Min

Main Question

Can we use a LLM to directly predict the next action to attain the goal in an environment?

Preliminary Video

Problem: LLMs may not output valid actions

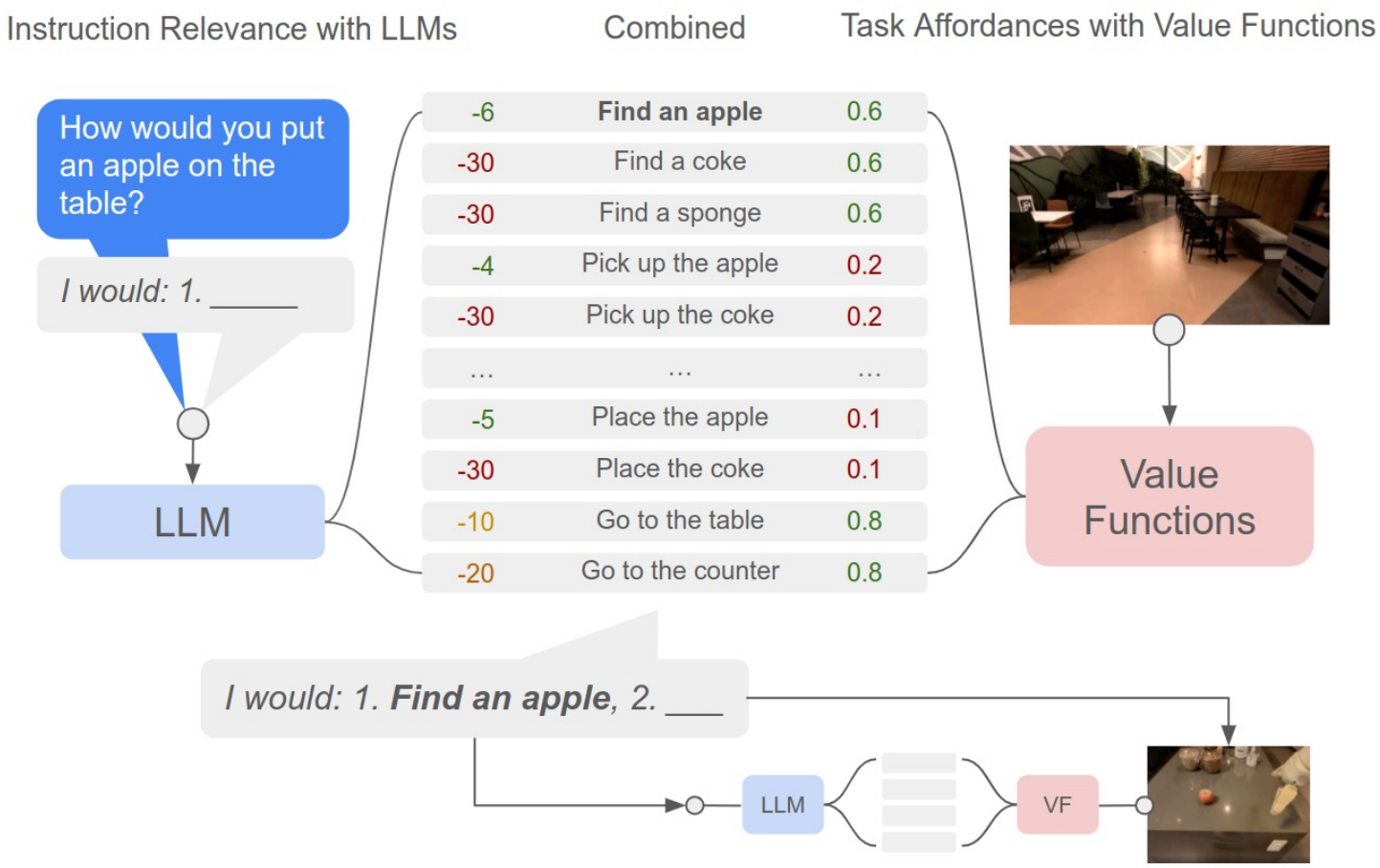


Potential Solutions

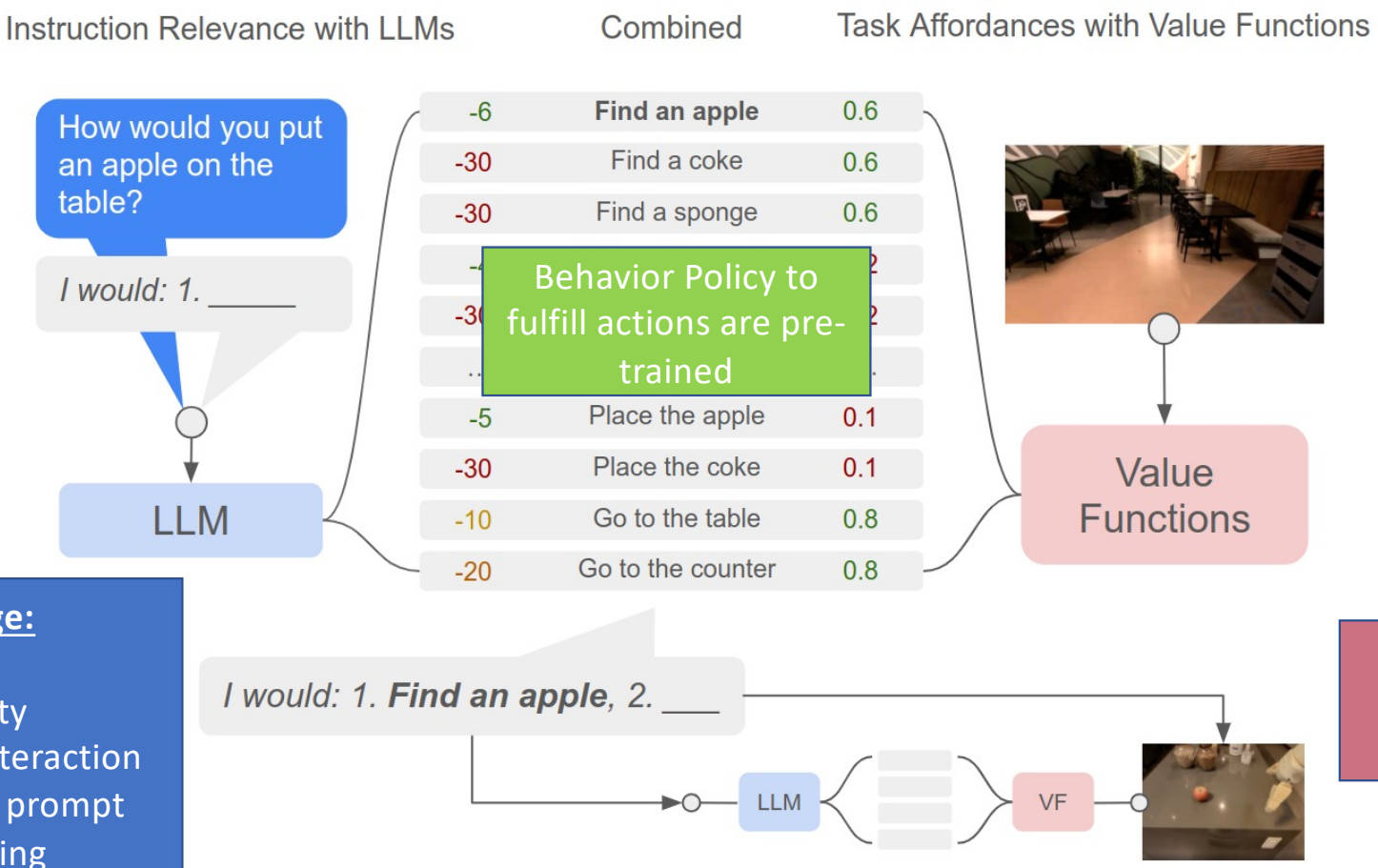
- Limit the output of the LLMs to only possible actions by the agent
- Have a way to do lookahead of various actions to goal state
  - AlphaGo/AlphaZero: Tree search
- **Have a way to heuristically determine value of immediate action**

Language X Affordance

LLMs with Value Functions



LLMs with Value Functions



Language:

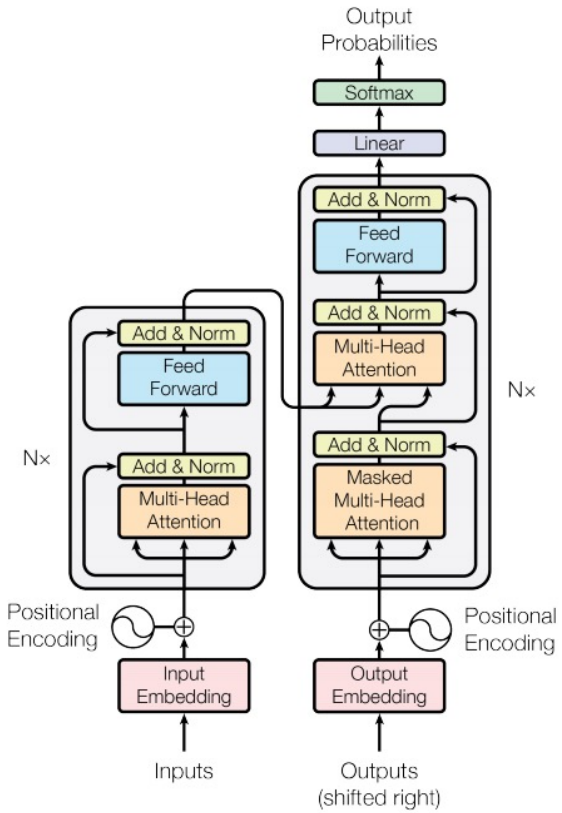
Flexibility
Ease of User Interaction
Goal-directed prompt engineering

Value Functions:

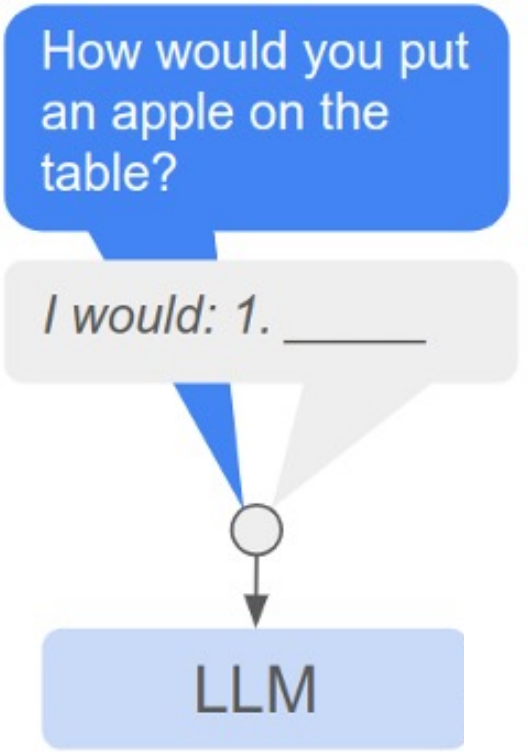
Feasibility of action

Multi-step prediction: Transformer at meta-level

I am a _____



Token-level



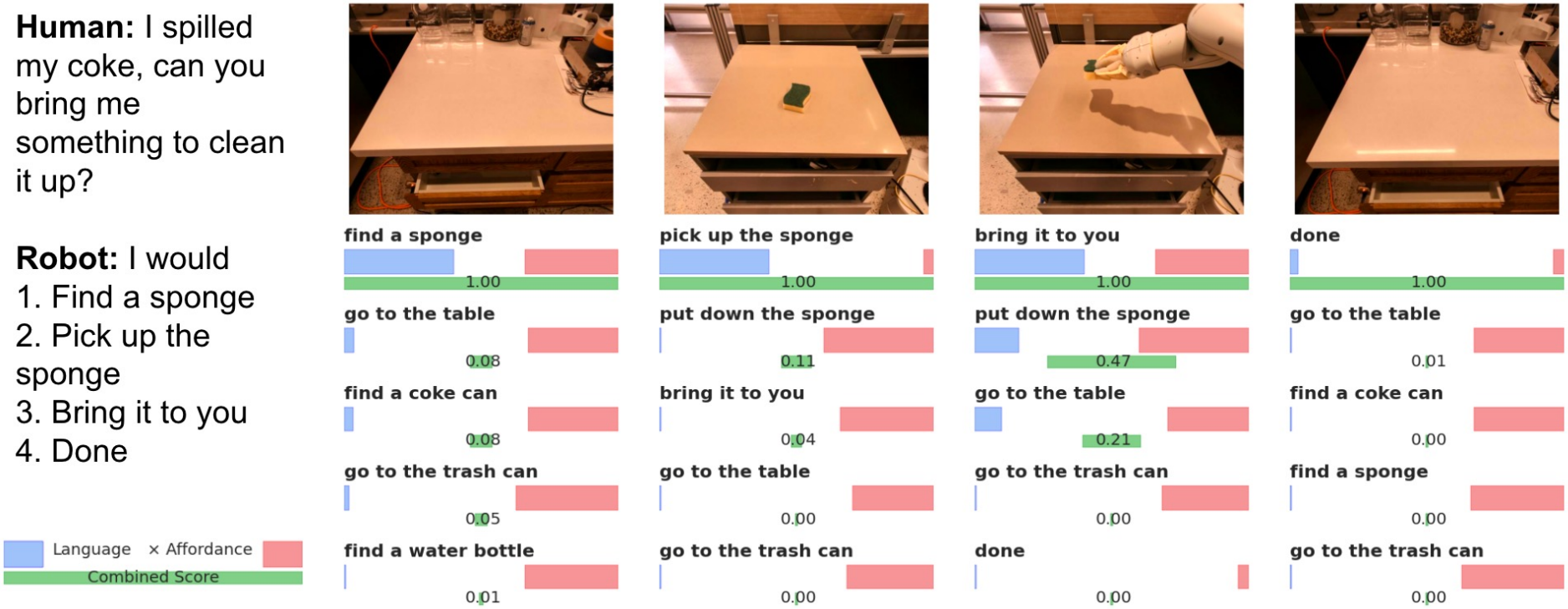
Action-level

Example Tasks

Example Task 1

Human: I spilled my coke, can you bring me something to clean it up?

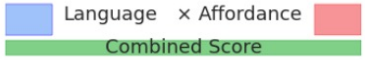
Robot: I would
1. Find a sponge
2. Pick up the sponge
3. Bring it to you
4. Done

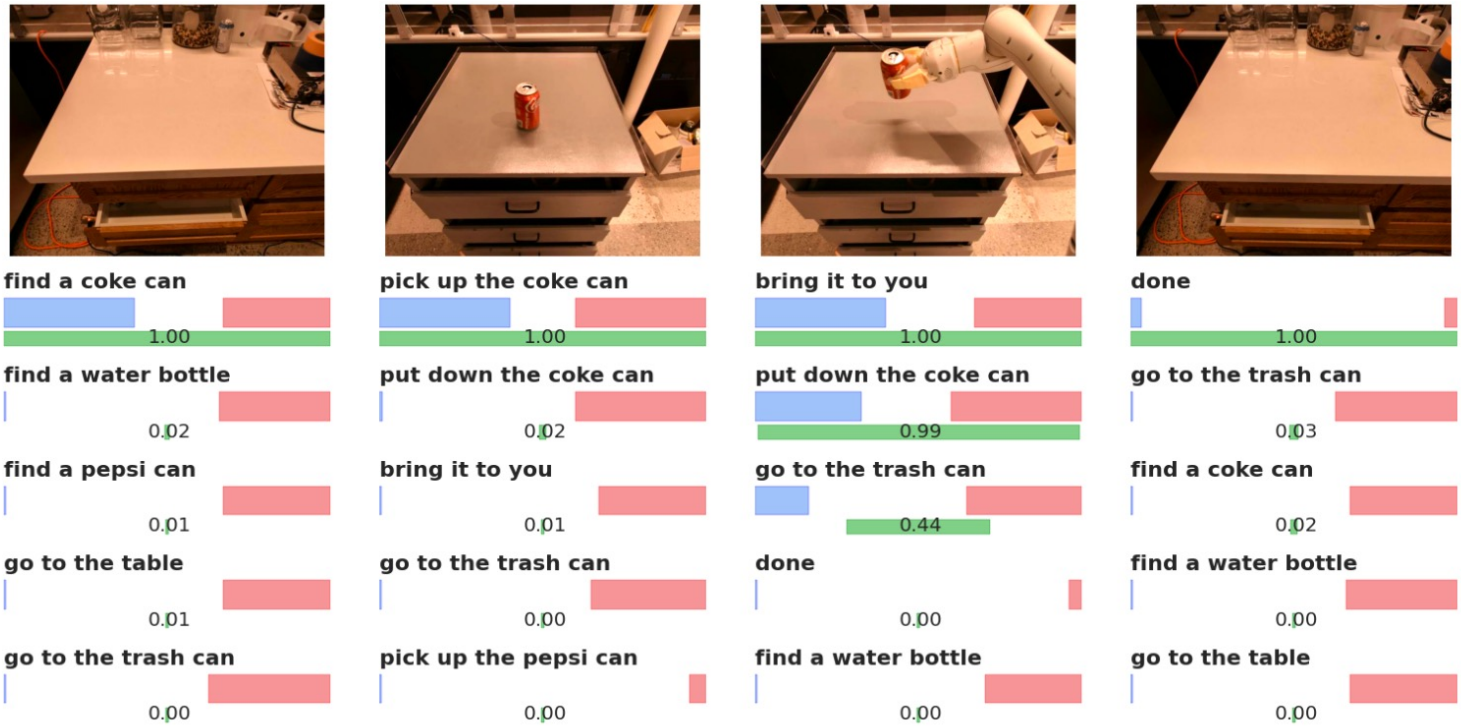


Example Task 2

Human: I spilled my coke, can you bring me a replacement?

Robot: I would
1. Find a coke can
2. Pick up the coke can
3. Bring it to you
4. Done

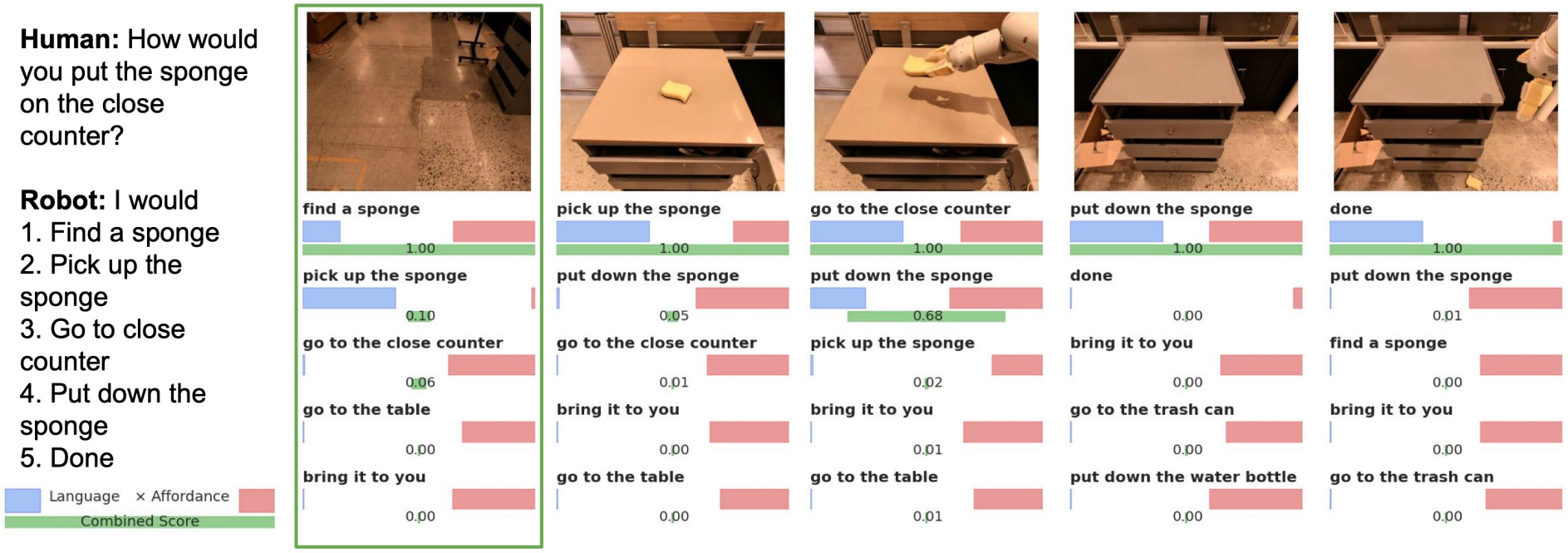




Example Task 3 – Affordance cueing actions

Human: How would you put the sponge on the close counter?

Robot: I would
1. Find a sponge
2. Pick up the sponge
3. Go to close counter
4. Put down the sponge
5. Done



Overall Algorithm

SayCan Overall Algorithm

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

LLM Scoring

- For each possible action, calculate the sum of log probabilities of the LLM predicting the tokens describing the action

- Action which is more natural will have a lower sum of log probabilities
  - Action is more likely to be predicted by the LLM

Affordance Function Scoring

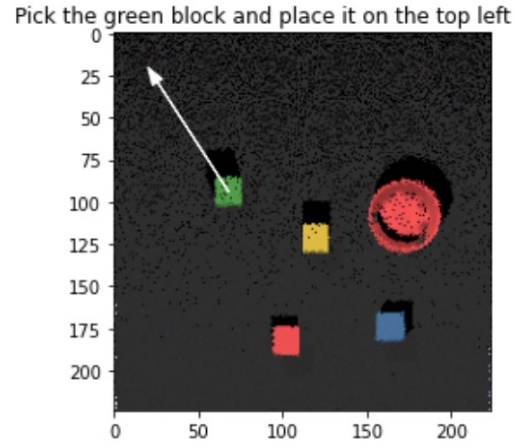
- Denotes the feasibility of performing the action in the environment
- Can be heuristically-encoded
- Can also be the value function of typical RL algorithms (e.g. Temporal Difference Learning)

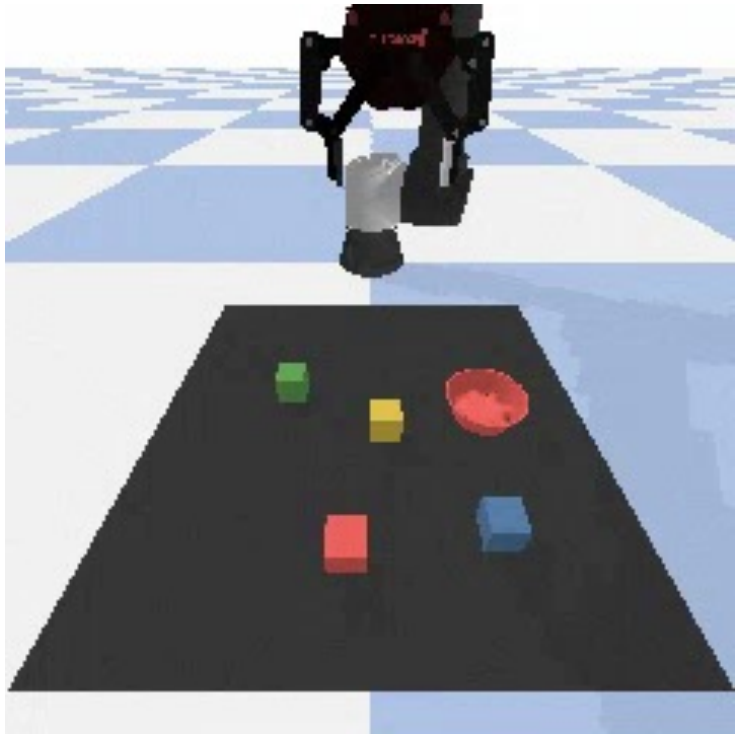
Code Walkthrough

Pick and Place

Pick and Place (Solution – 1st step)

```
**** Solution ****
objects = [yellow block, green block, red bowl, red block, blue block]
# put all the blocks in different corners.
Step 0: robot.pick_and_place(green block, top left corner)
Step 1: robot.pick_and_place(red block, top right corner)
Step 2: robot.pick_and_place(yellow block, bottom left corner)
Step 3: robot.pick_and_place(blue block, bottom right corner)
Initial state:
GPT-3 says next step: Pick the green block and place it on the top left corner.
```





Experimental Evaluation

Office Kitchen, Mock Kitchen

Experiment Setup

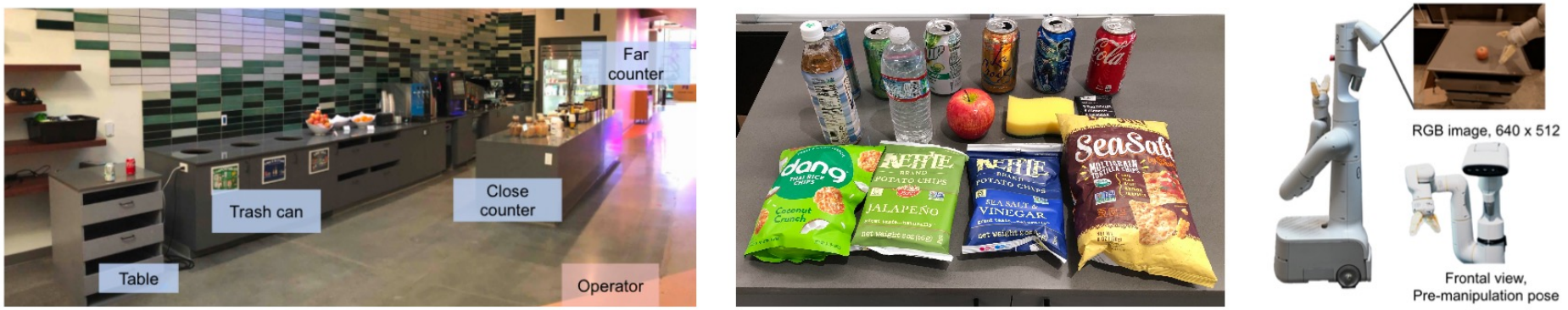


Figure 4: The experiments were performed in an office kitchen and a mock kitchen mirroring this setup, with 5 locations and 15 objects. The robot is a mobile manipulator with policies trained from an RGB observation.

Training Value Function with RL

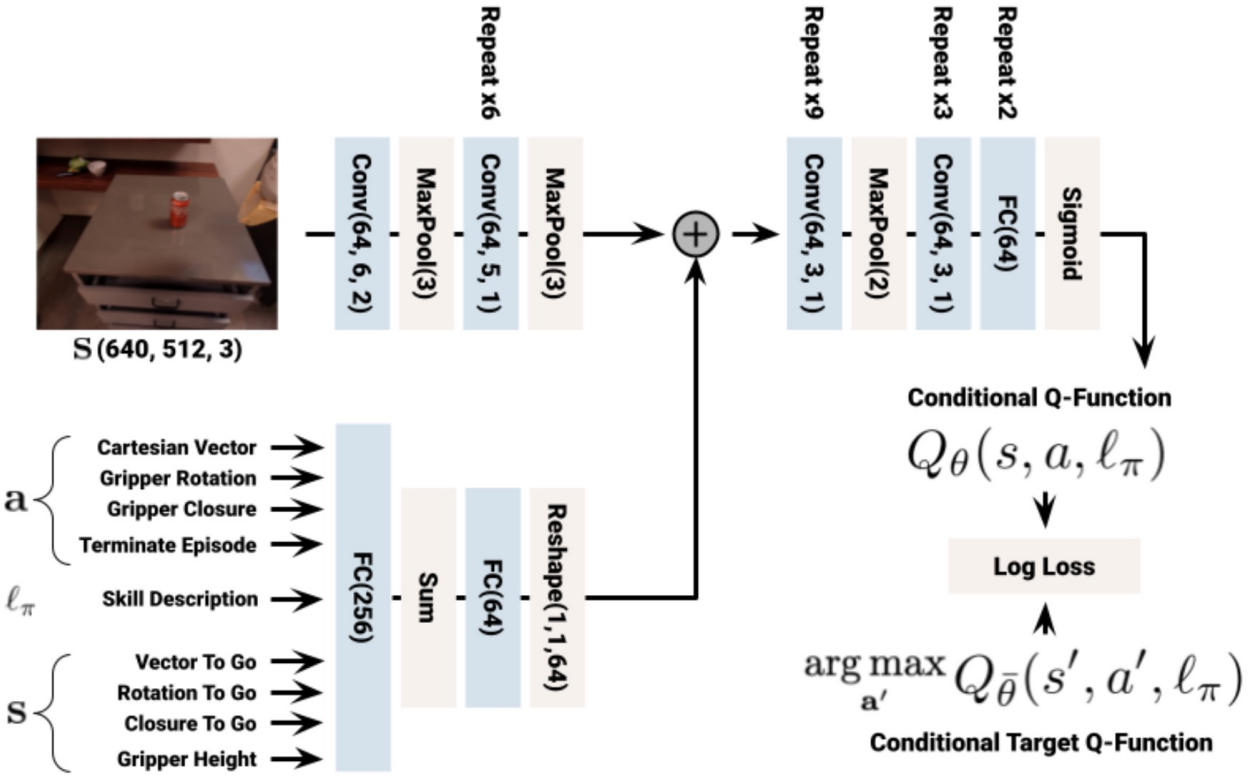


Figure 9: Network architecture in RL policy

- RL model trained with **16 TPUv3 chips for about 100 hours**, with another **3000 CPU workers** to collect episodes and another **3000 CPU workers** to compute gradient targets
- Episodes are sparse rewards of 0 or 1

Types of instructions (goals)

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

Table 1: **List of instruction family definitions:** We evaluate the algorithm on 101 instructions. We group the instructions into different families, with each family focusing on testing one aspect of the proposed method.

Metrics

- **Plan Success Rate:** Whether the skills selected by the model are correct for the instruction, regardless of whether or not they actually successfully executed

- **Evaluation Success Rate:** Whether the full system performs the desired instruction successfully
  - Rated by 3 humans, at least 2 out of 3 for success

Results: LLMs and Affordance are important

		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%

Table 2: Success rates of instructions by family. PaLM-SayCan achieves a planning success rate of 84% and execution success rate of 74% in the training environment and 81% planning and 60% execution in a real kitchen. *No VF* uses the maximum score skill from the LLM, *Generative (Gen.)* uses a generative LLM and then projects to the nearest skill via USE embeddings, *BC NL* uses the policy with the natural language instruction, and *BC USE* uses the policy with the natural language instruction projected to the nearest skill via USE embeddings.

65% failures from LLM, 35% from affordances

More Complex LLM leads to better results

		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%

Table 3: Success rates of instructions by family. SayCan achieves a planning success rate of 84% and execution success rate of 74% with PaLM and FLAN achieves 70% planning and 61% success. SayCan scales and improves with improved LLMs.

Better Planning: Chain-of-thought reasoning

CHAIN-OF-THOUGHT PLANNING ROLLOUT WITH PALM-SAYCAN.

Human: Can you bring a fruit-flavored drink without caffeine?

Explanation: The user has asked for a drink that is fruit-flavored and does not have caffeine, I will bring the lime soda.

Robot: 1. find a lime soda, 2. pick up the lime soda, 3. bring it to you, 4. put down the lime soda, 5. done

Human: Bring me the spicy chips.

Explanation: The user has asked for chips that are spicy, I will bring the Jalapeno chips.

Robot: 1. find the jalapeno chips, 2. pick up the jalapeno chips, 3. bring it to you, 4. put down the jalapeno chips, 5.done

Human: The snack is too small, bring me something more filling.

Explanation: The user has asked for a snack that is more filling, I will bring the multigrain chips.

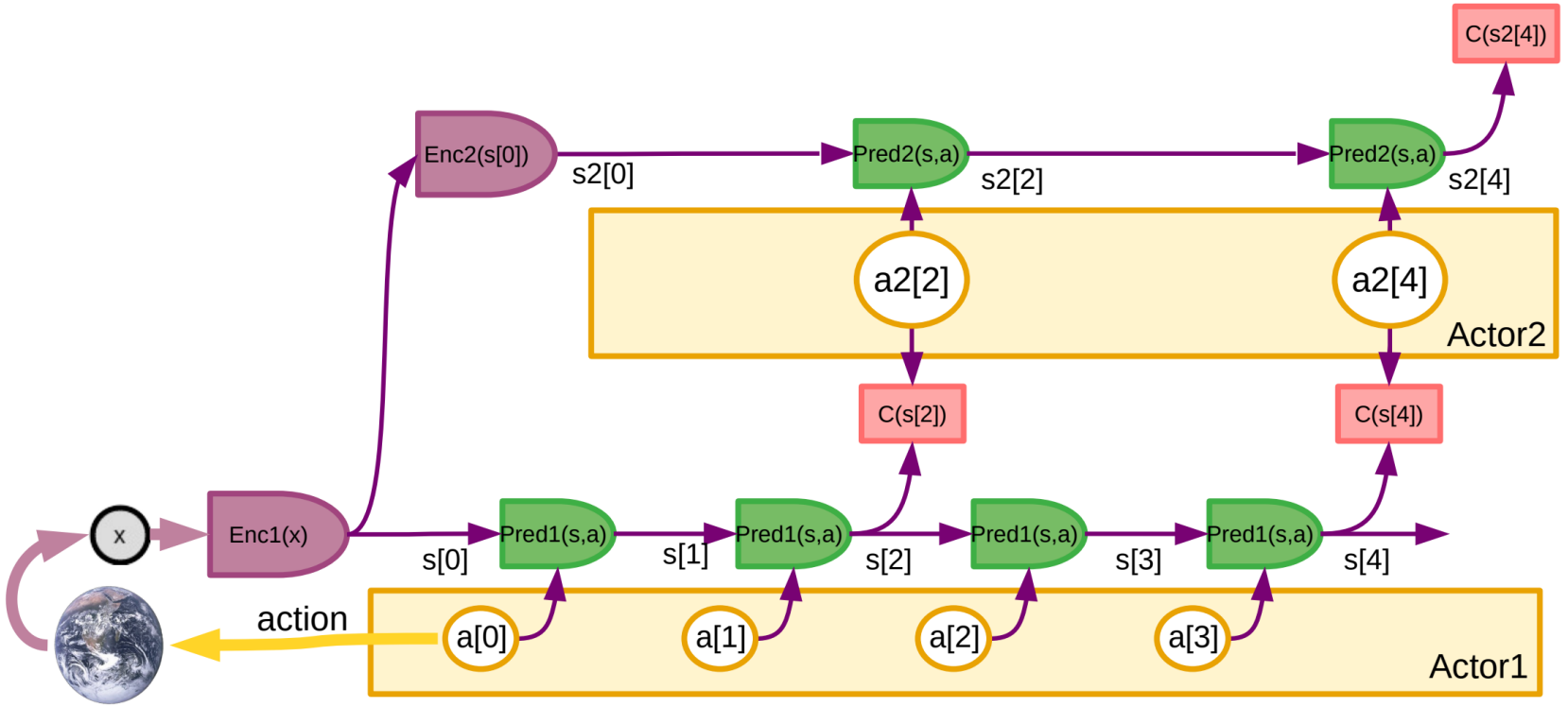
Robot: 1. find the multigrain chips, 2. pick up the multigrain chips, 3. bring it to you, 4. put down the multigrain chips, 5. done

Table 4: Chain-of-thought planning rollout with PaLM-SayCan. The highlighted part is the chain of thought generated by PaLM-SayCan.

Connection to Fast & Slow

Hierarchical Planning using LLMs

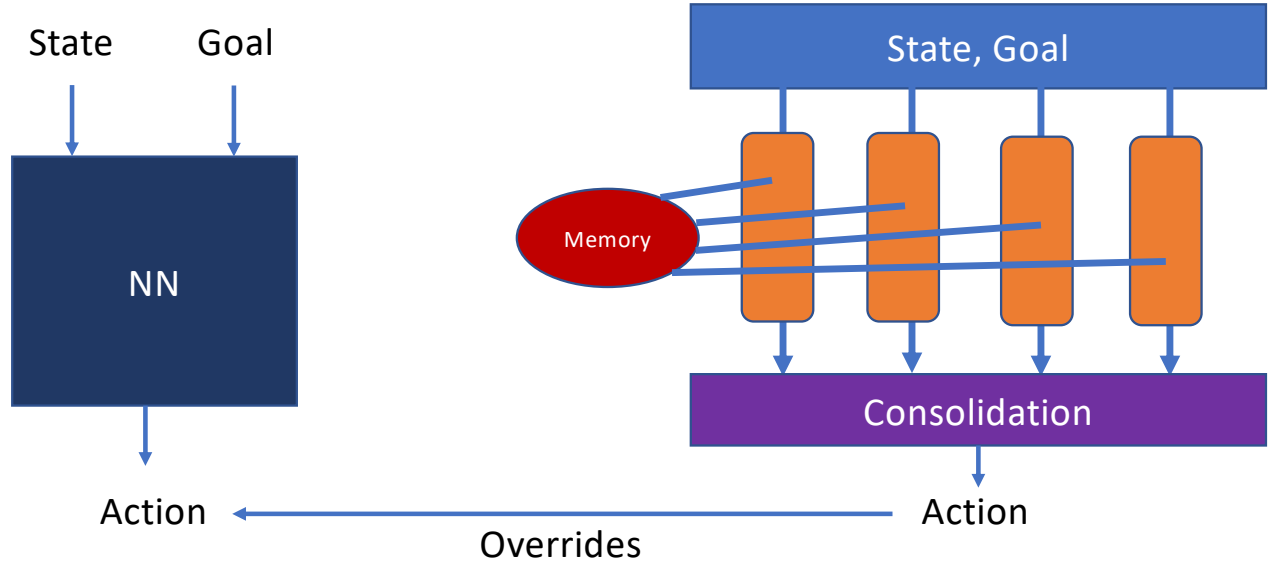
- Meta-level planning using SayCan



A Path Towards Autonomous Machine Intelligence. Yann LeCun. 2022.

Linkage to Fast and Slow

- Goals here are sub-goals which can be determined by a meta-model (e.g. SayCan)
- Perhaps fast and slow component can be a variant of SayCan?



SayCan's LLMs as goal-based action selection?

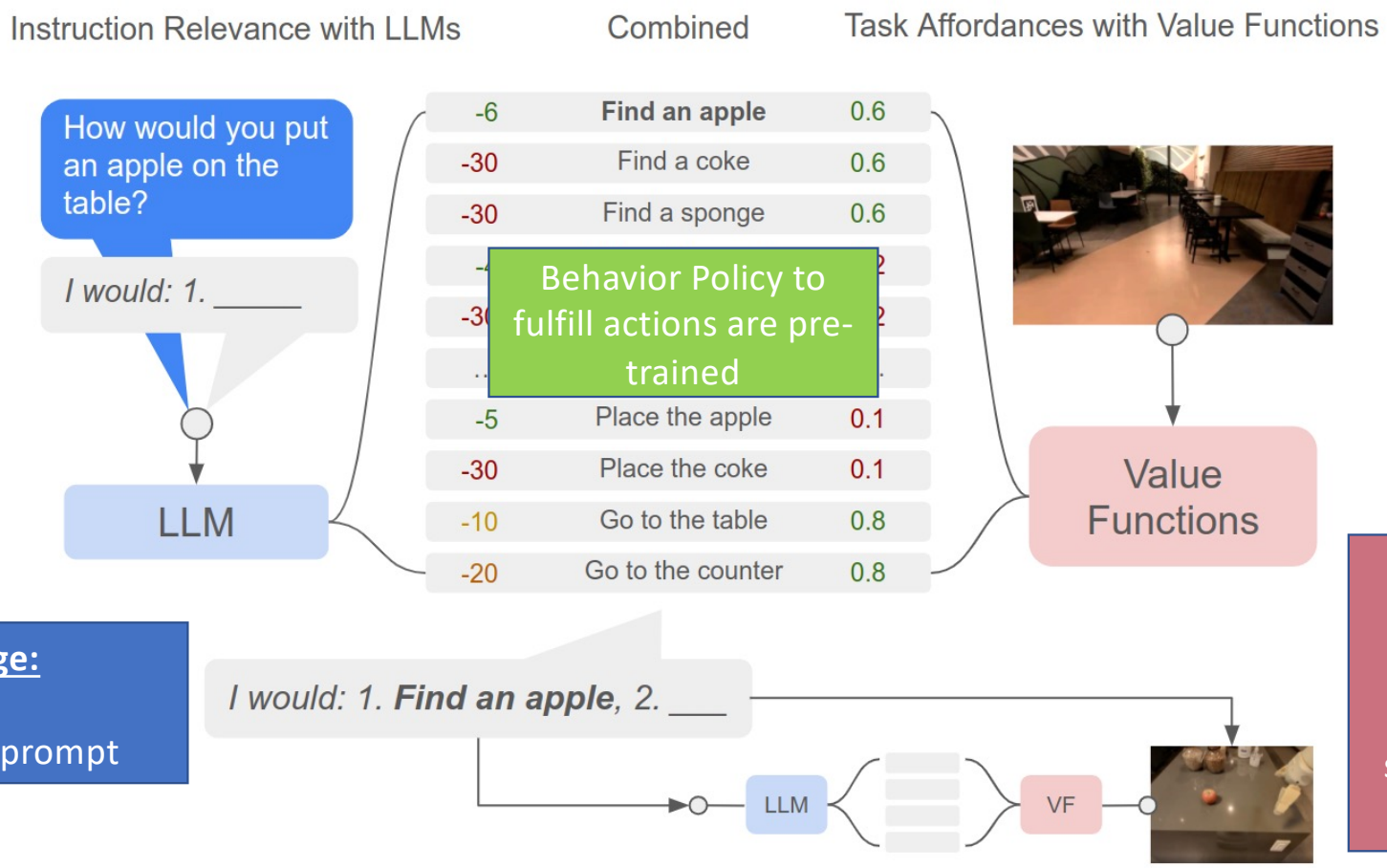
Memory lookahead as feasibility of attaining goal?

Neural Networks: Fast retrieval, slow learning

Memory: Slow retrieval, fast learning

Connection to Reliable GPT

Reliable GPT



Language:

Answering a prompt

Value Function:

Ensure output is referenceable in some memory bank etc.

Questions to Ponder

- Actions considered are finite. How do we add an unlimited amount of axiomatic actions?
- Is LLM too complicated an architecture for picking the next action?
- Currently the model is only doing a one-step lookahead. How can we better solve long-horizon tasks?
- Can we use something other than a value function to score affordances?
- Can we make this architecture solve generic unseen (dynamic) tasks?