Agent Frameworks

CS6960 MultiModal LLM Agents

Kenneth Marino

Slide credits: Yoav Artzi, Shuyan Zhou, Xinyun Chen, others

Announcements

- HW1 due in two weeks
  - See Syeda's note in announcements / read the new instructions
  - Basically just asking you to read the intro unit to the HF Agents tutorial and write some boilerplate code
- HW0
  - Hopefully you're done
  - Can give an extension if you recently joined the course
- Paper Presentations starting today
- Projects
  - Start thinking of ideas
  - We're working on a list of "starter ideas"
  - Read through papers in area you're interested in

Recommended Readings

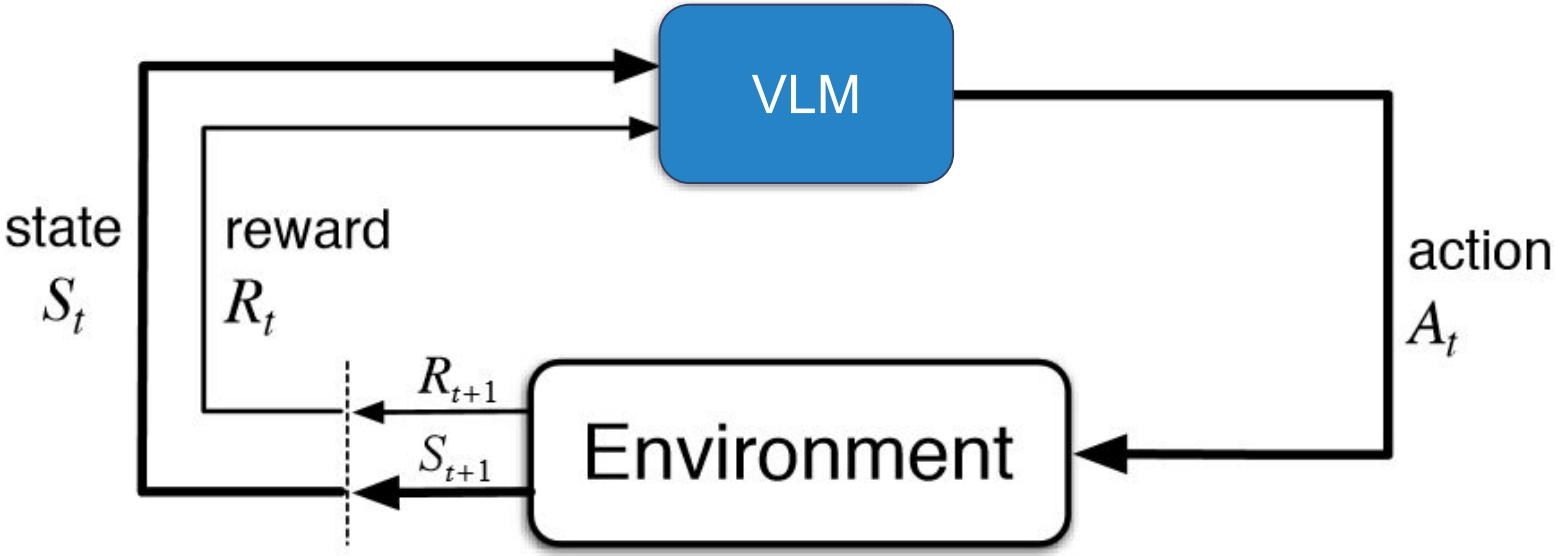
- Agent Courses/Tutorials
  - https://www.shuyanzhou.com/teaching/25fall-590/25fall-590.html
  - https://rdi.berkeley.edu/adv-llm-agents/sp25
  - https://agenticai-learning.org/sp25
- This week's papers
  - Chain-of-Thought Prompting Elicits Reasoning in Large Language Models
  - ReAct: Synergizing Reasoning and Acting in Language Models
  - Reflexion: Language Agents with Verbal Reinforcement Learning
  - Charts-of-Thought: Enhancing LLM Visualization Literacy through Structured Data Extraction
  - DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning

Lectures Going Forward

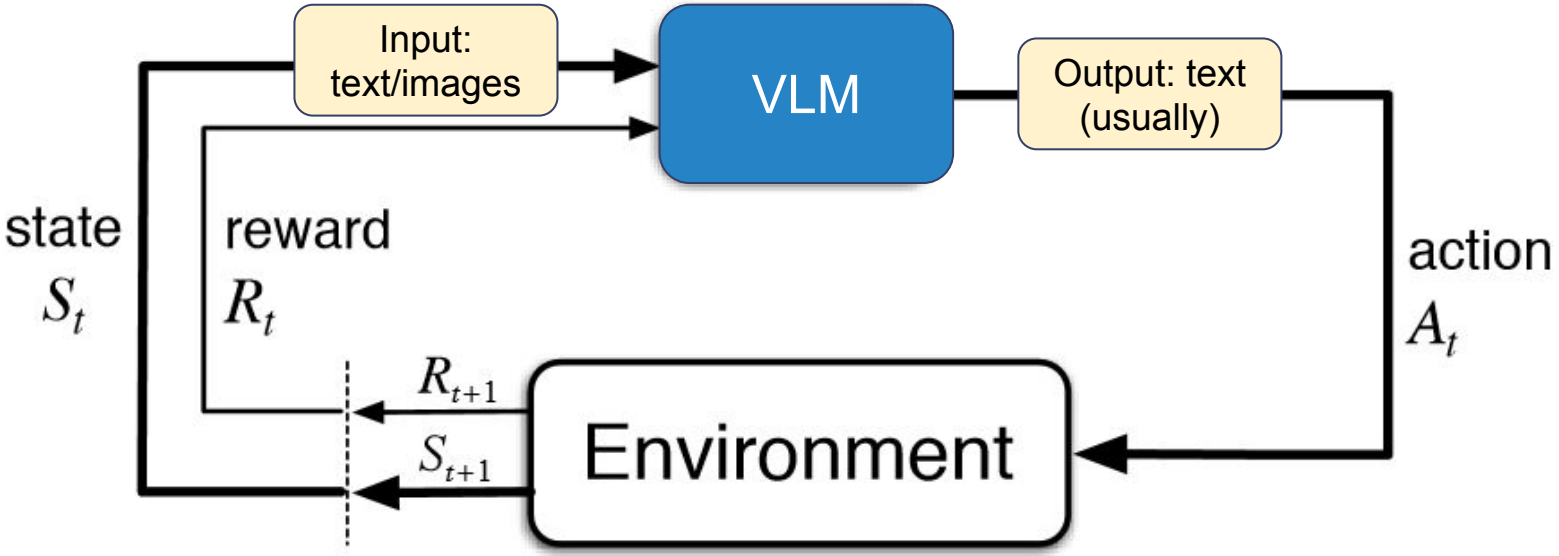
- Meant to be fairly short / high-level
  - More an introduction/framing of the topic than a deep dive
  - Maybe 20-40 mins depending on how much I have to say about the topic
- Imperative that you read papers and participate in class discussions to get full breadth of topic

Any Questions

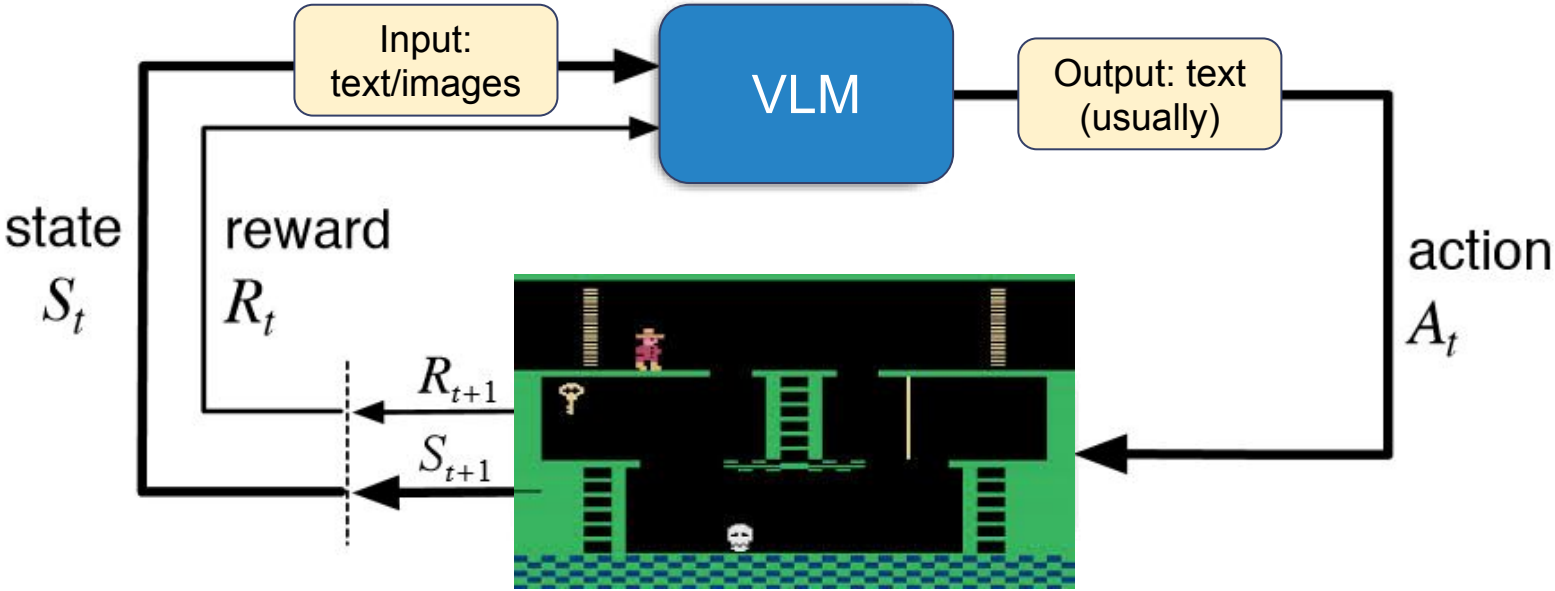
What does VLM agent actually look like?



What does VLM agent actually look like?



What does VLM agent actually look like?



Other “stuff”

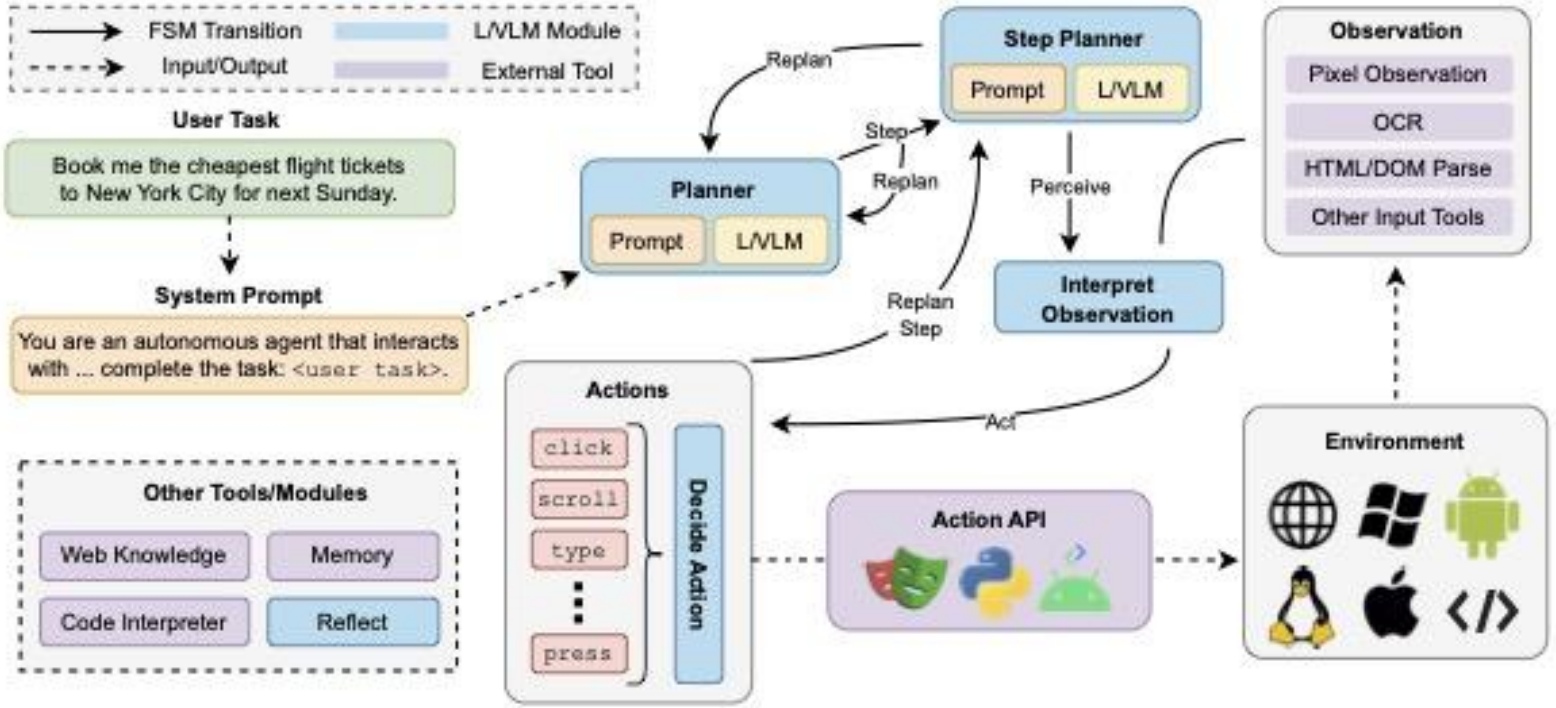


Image credit: Farhan Ishmam

Other “stuff”

Frameworks for LLMs to be better at actions (reflect, plan, replan, etc)

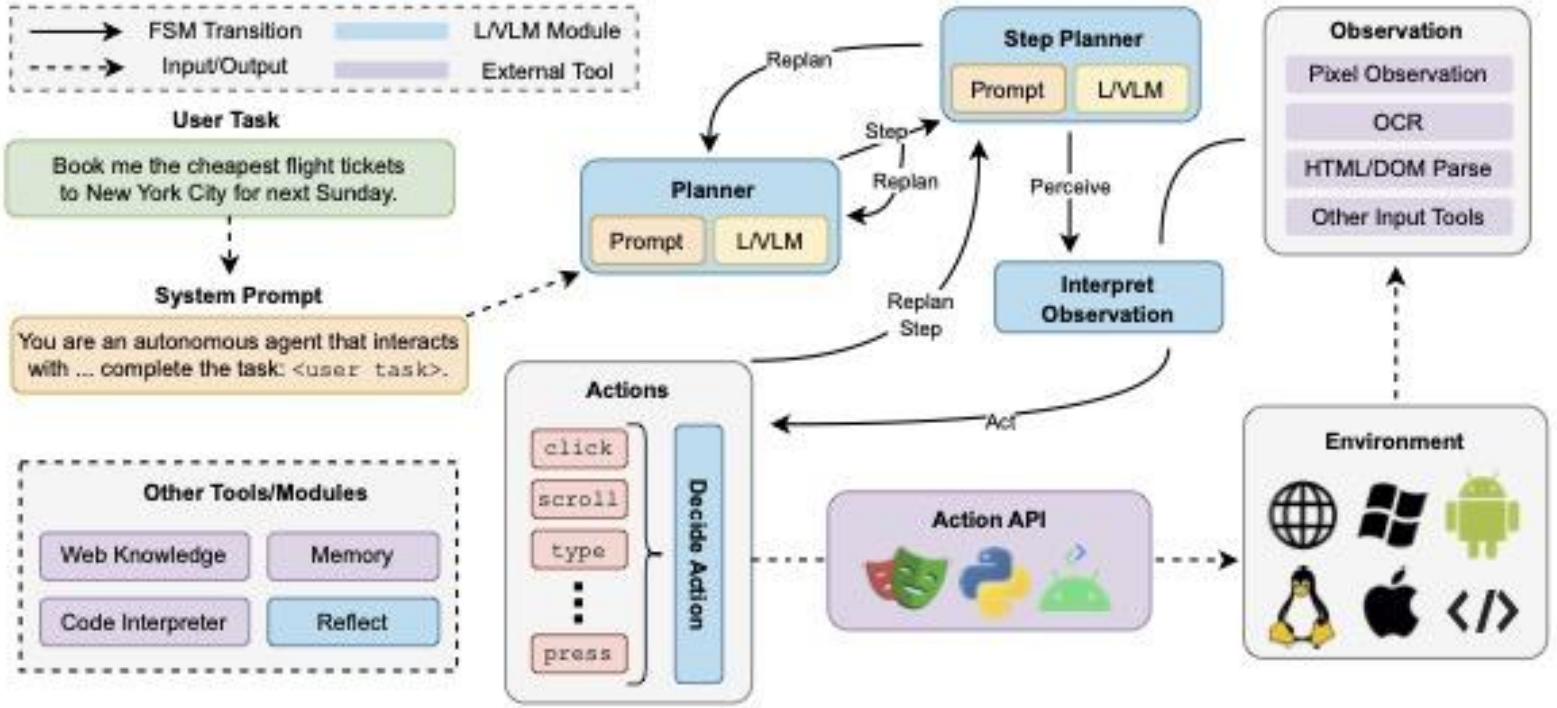


Image credit: Farhan Ishmam

Why do this?

What this means for VLM Agents

- Basics of environment, maximizing reward, policies, all apply
  - State/observation given by text image tokens
  - Output is text (then interpreted as environment actions)
  - Want to maximize some reward (answer a query, fill out a web form, do some action in the environment)
- Most useful algorithms to know
  - Imitation learning probably most common
    - When we talk about LLMs, this is exactly the same as Supervised Fine Tuning (SFT)
  - When we do RL with agents, PPO is most commonly used
- VLM agents also often work without any training!
  - Through prompting frameworks and other methods
  - Just relies on general ability of the base models!
  - **There will always be a limit on how good this will be / no way to make it better**

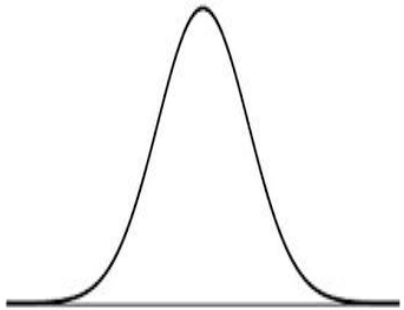
What this means for VLM Agents

- Basics of environment, maximizing reward, policies, all apply
  - State/observation given by text image tokens
  - Output is text (then interpreted as environment actions)
  - Want to maximize some reward (answer a query, fill out a web form, do some action in the environment)
- Most useful algorithms to know
  - Imitation learning probably most c
    - When we talk about LLMs, this is exa Fine Tuning (SFT)
  - When we do RL with agents, PPO
- VLM agents also often work with
  - Through prompting frameworks and other methods
  - Just relies on general ability of the base models!
  - **There will always be a limit on how good this will be / no way to make it better**

Is there a way to make VLM better without training?

Shift VLM’s distribution

- LLM/VLM
  - Pre-trained on Internet Data



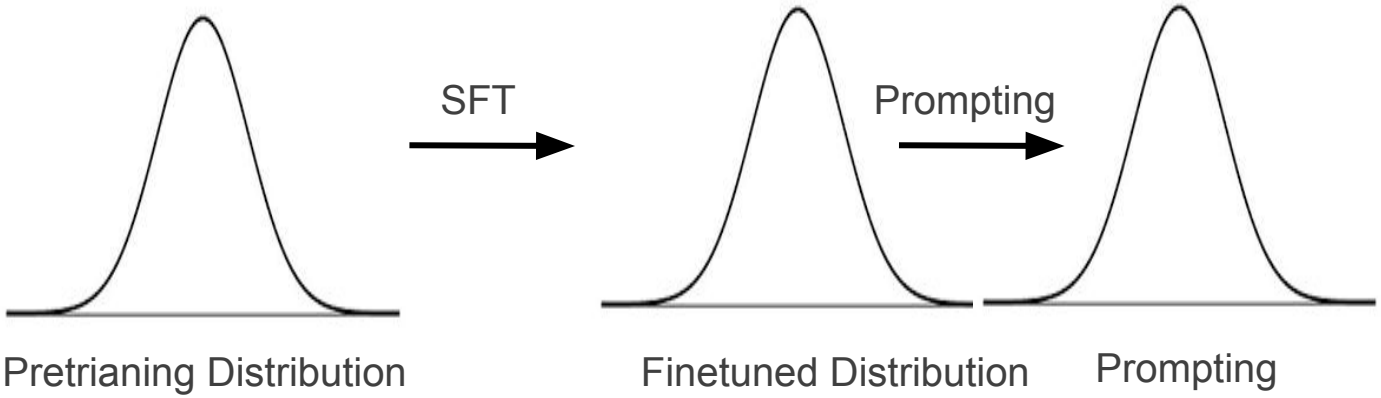
Pretrianing Distribution

Shift VLM’s distribution

- LLM/VLM
  - Pre-trained on Internet Data
  - Finetuned on some task specific data

Shift VLM's distribution

- LLM/VLM
  - Pre-trained on Internet Data
  - Finetuned on some task specific data
  - Use prompting to shift the input to get a different distribution



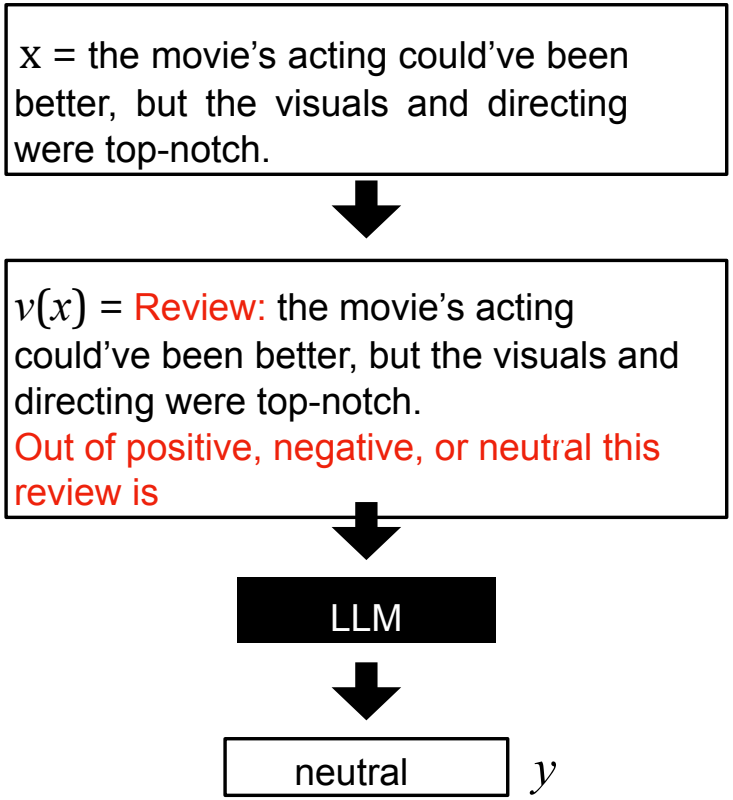
Prompting!

- Simple way to turn LLMs into task-specific models
  - You benefit from the rich representations of the LLM
  - Finetuned on some task specific data
- LLMs offer a completely new mode of operation that does not require any changes to its parameters: prompting
  - With or without annotated examples: zero-shot or in-context learning (few-shot)
  - With or without intermediate reasoning steps: **chain-of-thought prompting**
  - Prompt to Act and plan - ReAct
  - Prompt to self-correct mistakes - Reflexion

Material from Yoav Artzi

Zero-shot Prompting

- Input: single unlabeled example x
- Output: the label *y*
- The task (and output) can be any text-to-text task: classification, summarization, translation
- Pre-processing: wrap x in a verbalizer template
- The template controls the output

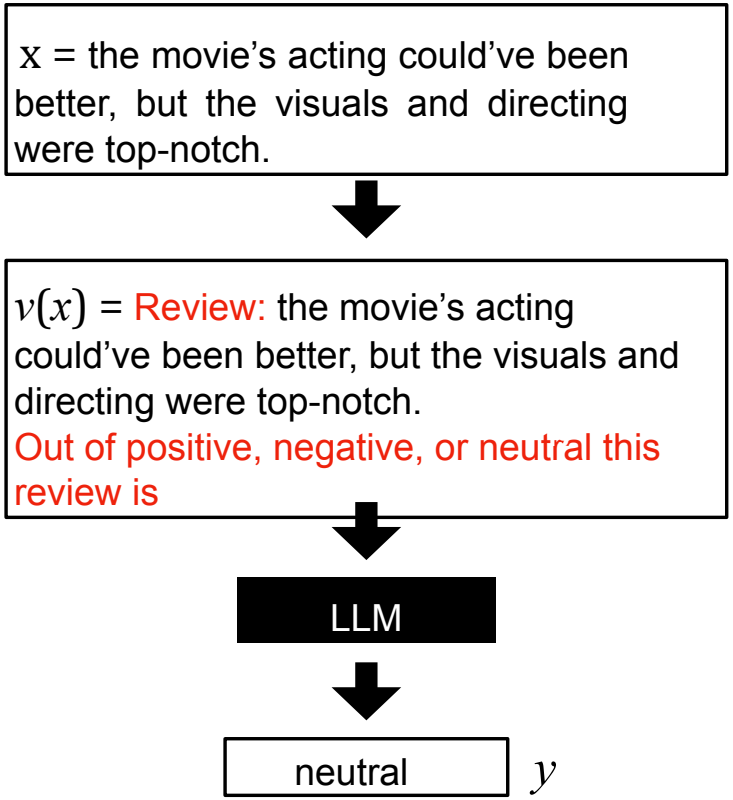


Material from Yoav Artzi

Zero-shot Prompting

- I
- C
- T
  t
  s
- P er template
- The template controls the output

- We've made a task more accurate by prompting the LLM with information about how we want it to perform the task
- In this case - what is the output space we want for the task
- Can instantly improve performance with no additional data (zero-shot, zero extra data)



Material from Yoav Artzi

Sensitivity and Variability

- Prompts can even be sensitive to minor cosmetic changes
- Can influence performance in unexpected ways
- Can think of them as (very complex) hyper-parameters

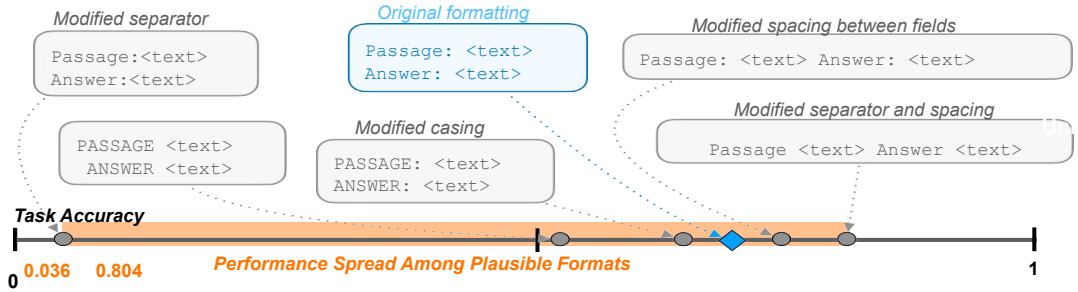


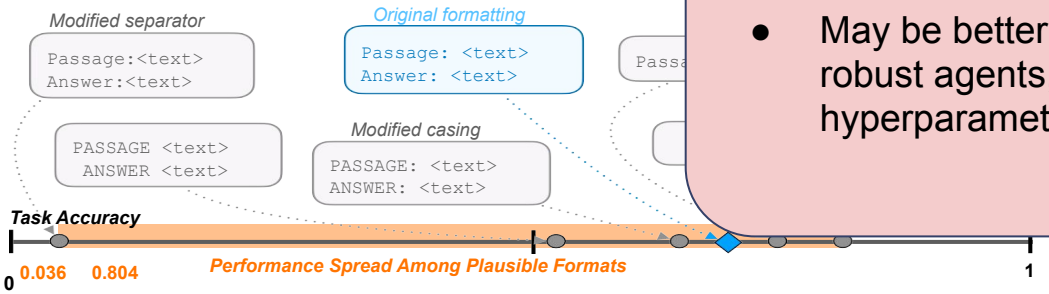
Figure 1: Slight modifications in prompt format templating may lead to significantly different model performance for a given task. Each `<text>` represents a different variable-length placeholder to be replaced with actual data samples. Example shown corresponds to 1-shot LLaMA-2-7B perfor- mances for task280 from SuperNaturalInstructions (Wang et al., 2022). This StereoSet-inspired task (Nadeem et al., 2021) requires the model to, given a short passage, classify it into one of four types of stereotype or anti-stereotype (gender, profession, race, and religion).

[Sclar et al. 2023]

Material from Yoav Artzi

Sensitivity and Variability

- Prompts can even be sensitive to minor cosmetic changes
- Can influence performance in ur
- Can think of them as (very comp



- This is one reason prompting can be a less idea strategy
- May be better to SFT/RL train for more robust agents, avoid prompt hyperparameter search

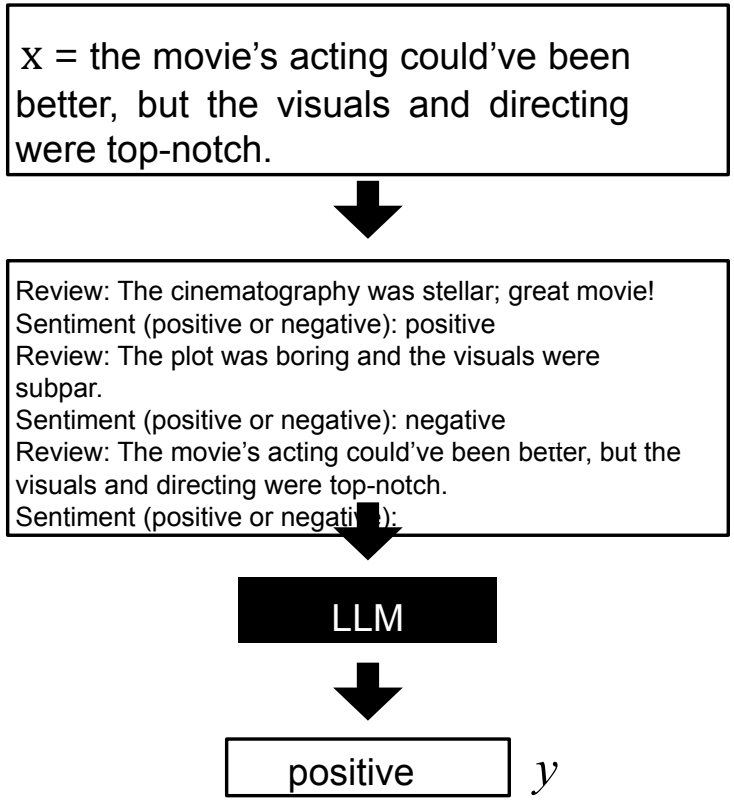
Figure 1: Slight modifications in prompt format templating may lead to significantly different model performance for a given task. Each `<text>` represents a different variable-length placeholder to be replaced with actual data samples. Example shown corresponds to 1-shot LLaMA-2-7B perfor- mances for task280 from SuperNaturalInstructions (Wang et al., 2022). This StereoSet-inspired task (Nadeem et al., 2021) requires the model to, given a short passage, classify it into one of four types of stereotype or anti-stereotype (gender, profession, race, and religion).

[Sclar et al. 2023]

Material from Yoav Artzi

Few-shot / In-context Learning (ICL)

- LLMs have the ability to "learn" to complete tasks through training in the prompt
- The recipe is simple:
  - Take a small number of annotated training example $\{(\bar{x}^{(i)}, \bar{y}^{(i)}\}_{i=1}^{N}$
  - Convert them using verbalizer v templates
  - Concatenate them and follow with the target input $\bar{x}$
  - The completion will be the label of the input



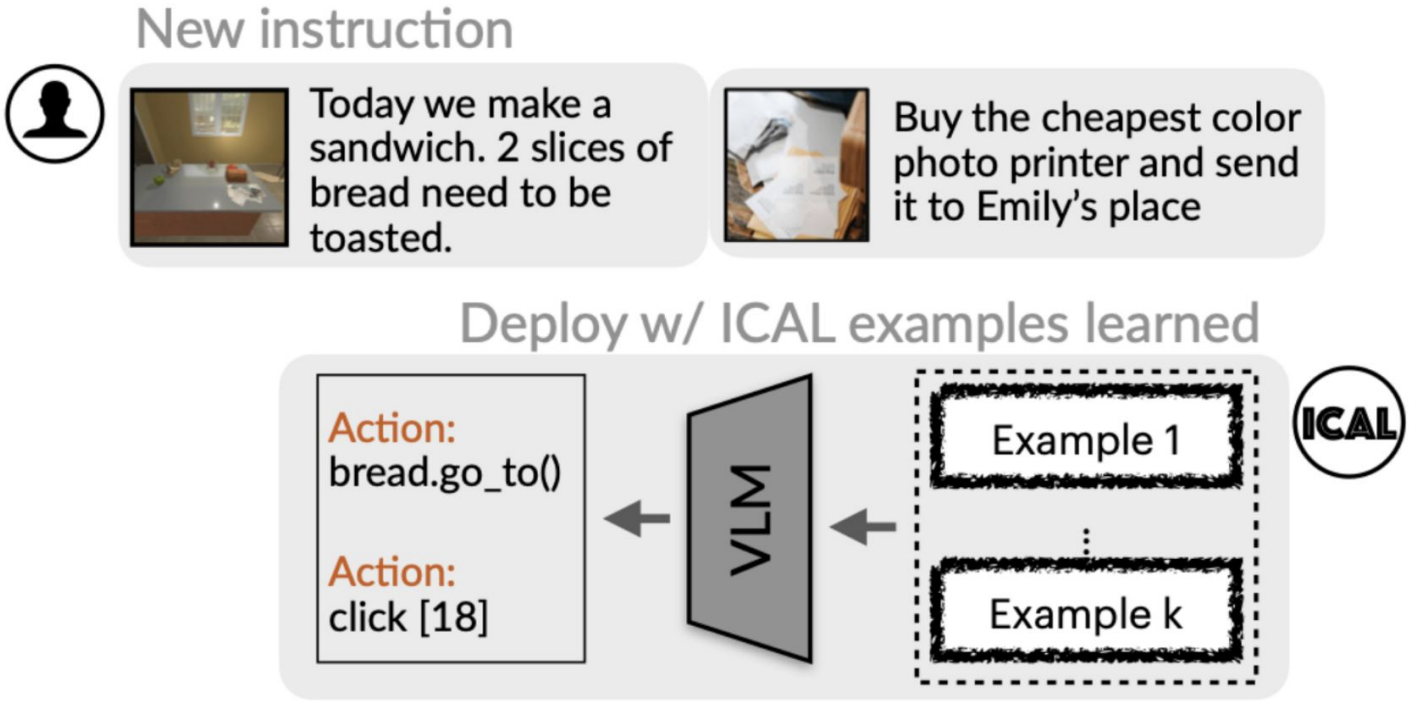
Material from Yoav Artzi

In-context Learning (ICL)

- Providing ICL examples almost always leads to significant improvements
- Benefits tend to diminish with more examples

Material from Yoav Artzi

ICL in Agents



Gabriel Sarch, Lawrence Jang, Michael Tarr, William W Cohen, Kenneth Marino, Katerina Fragkiadaki. "VLM agents generate their own memories: Distilling experience into embodied programs of thought." *NeurIPS* 2024.

Chain-of-thought (COT) Prompting

- Can we get agents to “think” through decisions?
- Some tasks require multiple reasoning steps
- Directly generating the answer requires the model internally do the reasoning steps (or shortcut somehow)
- It is empirically useful to:
  - Show the model examples of the reasoning steps through ICL
  - And then have it explicitly generate the reasoning steps

Material from Yoav Artzi

Chain-of-thought (COT) Prompting

Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ✗

Chain-of-Thought Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The answer is 9. ✓

[Wei et al. 2022]

Chain-of-thought (COT) Prompting

Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?
A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ✗

Chain-of-Thought Prompting

M

Q:
ten
ten
A:
eac

Q:
ma
do

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The answer is 9. ✓

- This is covered in one of the presentations today, so not going too deep
- Bottom line: prompting alone can encourage LLM to reason better!

[Wei et al. 2022]

Thinking without Prompting?

- Can we train LLMs to think through their reasoning?
- Will hear more about this later
- Use RL to train models to do reasoning

Many classes of inference-time LLMs

- Are there inference-time / prompting techniques for agents?

ReAct (Yao et al 2023)

- **Loop:** Goal → Observation → (Reasoning → Action) → Observation → ….
- Treat **reasoning itself** as an action that changes the **internal** state of the agent, rather than the environment.

Question: Aside from the Apple remote, what other devices can control the program Apple remote was originally designed for?

Thought 1: I need to search Apple Remote and find the program it was originally designed to interact with.

Act 1: `Search[Apple Remote]`
Obs 1: The Apple Remote is a remote control introduced in October 2005 by Apple […] originally designed to control the Front Row media center program […]

Thought 2: Apple Remote was originally designed to control the Front Row media center program. I need to search Front Row next and find what other device can control it.

Act 2: `Search[Front Row]`
[…]

Slides from Shuyan Zhou

ReAct (Yao et al 2023)

```
def react_agent(goal, max_steps=10):
  observation = get_initial_observation()

  for step in range(max_steps):
    # Generate thought about current situation
    thought = llm.generate(f"Goal: {goal}\nObservation: {observation}\nThought:")

    # Decide on action based on thought and observation
    action = llm.generate(f"Goal: {goal}\nObservation: {observation}\nThought: {thought}\r

    # Execute action in environment
    env.execute(action)
```

ReAct effectively defines a straightforward for loop

Slides from Shuyan Zhou

ReAct (Yao et al 2023)

```
def react_agent(goal, max_steps=10):
  observation = get_initial_observation()

  for step in range(max_steps):
    # Generate thought about current situation
    thought = llm.generate(f"Goal: {goal}\nObservation: {observation}\nThought:")

    # Decide on action based on thought and observation
    action = llm.generate(f"Goal: {goal}\nObservation: {observation}\nThought: {thought}\r

    # Execute action in environment
    env.execute(action)
```

ReAct effectively defines a straightforward for loop

Slides from Shuyan Zhou

Planning Beyond ReAct

Besides reason about the immediate observation, the agent can also

- Break high-level task into subtasks.
- Examine the history, perform backtracking or re-planning if necessary

Hence, the control loop can be more complicated.

```
def planning_agent(goal, max_steps=10):
  # Initial planning phase
  plan = llm.generate(f"Create plan for: {goal}")
  subtasks = parse_plan_into_subtasks(plan)
```

```
  for subtask in subtasks:
    # Execute subtask using ReAct loop
    while not is_subtask_completed(subtask):
      thought = llm.generate(f"Current subtask: {subtask}")
      action = llm.generate(f"Based on thought, what action?")
      observation = env.execute(action)
```

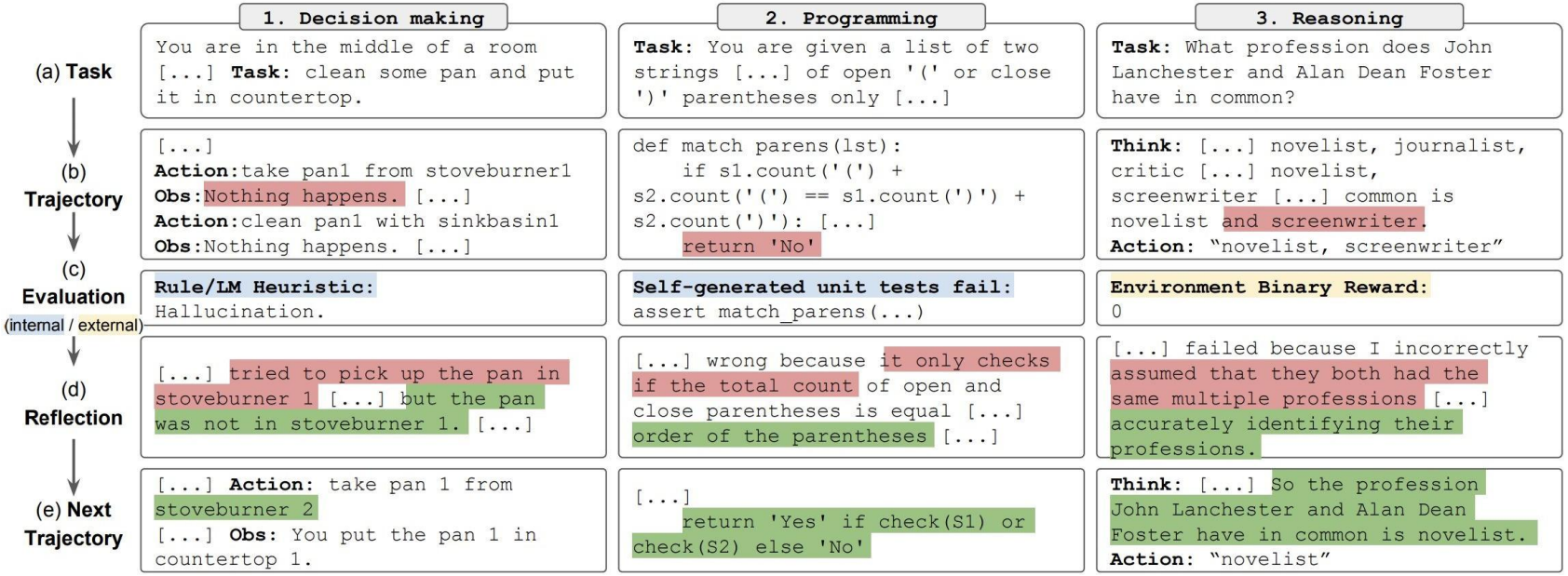
```
      # Replan if stuck
      if should_replan(observation):
        subtasks = replan(goal, current_progress)
        break
```

Slides from Shuyan Zhou

Correcting mistakes

- Can VLM agents recognize/fix their own mistakes
- Can they **reflect** and figure out where they went wrong

Reflexion and Self-Refine



- LLM generates feedback on its output. Use external evaluation when available.
- LLM self-refines its output given both internal feedback and external evaluation.

Shinn et al., Reflexion: Language Agents with Verbal Reinforcement Learning, NeurIPS 2023.
Madaan et al., Self-Refine: Iterative Refinement with Self-Feedback, NeurIPS 2023.

Slides from Xinyun Chen

Beyond Singe Agent

- **Multi-agent systems**: Multiple agents collaborate or compete
  - e.g., code review (writer/reviewer), debate systems, hierarchical teams

```
def multi_agent_collaboration(goal, agents):
  # Distribute work among agents
  subtasks = decompose_task(goal, num_agents=len(agents))
```

```
  for round in range(max_rounds):
    results = []
    for i, agent in enumerate(agents):
      # Each agent works on their subtask
      result = agent.execute(subtasks[i])
      results.append(result)
```

```
    # Agents communicate and share results
    shared_knowledge = aggregate_results(results)

    # Update each agent's understanding
    for agent in agents:
      agent.update_knowledge(shared_knowledge)

    if is_goal_achieved(shared_knowledge, goal):
      break
```

Slides from Shuyan Zhou

Takeaways

- We want some way to get VLMs/LLMs more in distribution for agent tasks
  - Ideally: SFT/RL can do this
  - In a pinch, prompting techniques very effective
- Get agents in distribution
  - Zero-shot prompting
  - In context Learning
- Get agents to think
  - With prompting in context (CoT)
  - With RL training (DeepSeek)
- Prompt agents to behave more like agents
  - Think/Act/Observe ReAct
  - Reflect on mistakes - Reflexion

Any Questions

Questions

Now for the presentations!