



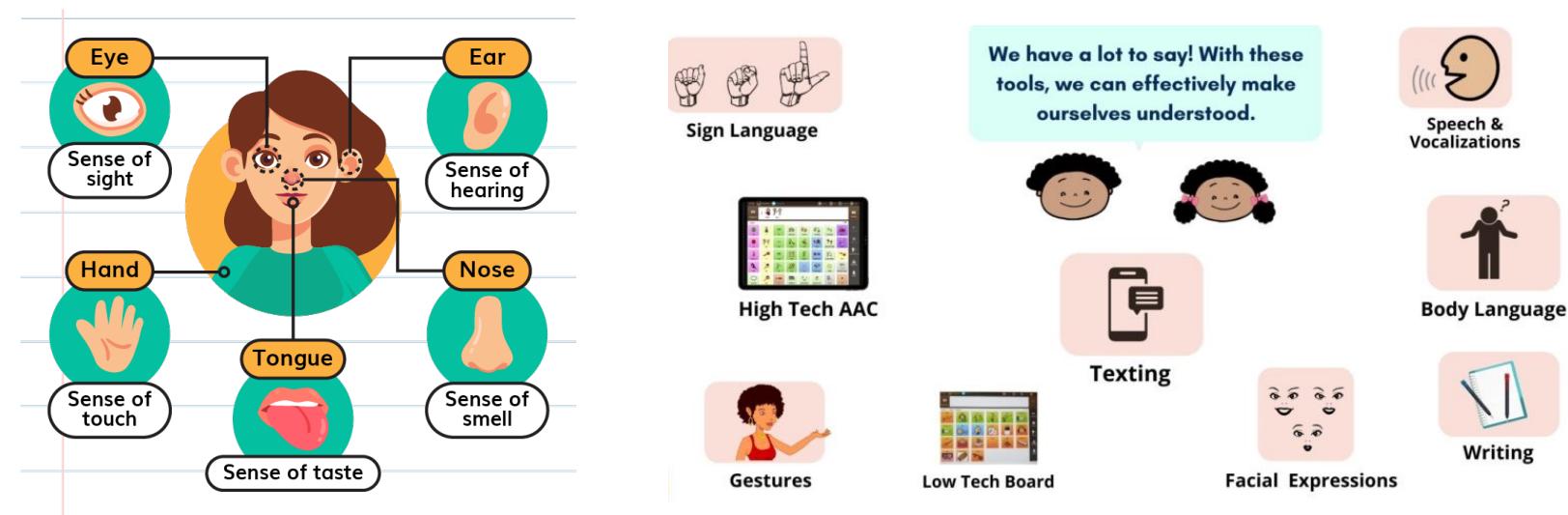
# Connecting Language to the World

CSCI 601-471/671 (NLP: Self-Supervised Models)

<https://self-supervised.cs.jhu.edu/sp2024/>

# Connecting Language to the World

- So far, we have focused on only “language” as our primary focus.
- But there are other modalities in which humans communicate with the world.



# Connecting LMs to the World: Chapter Plan

---

1. Connecting vision and language
2. Generative vision-language models
3. Other modalities [speech, audio, ...]
4. From language to code
5. From language to action

**Chapter goal:** Getting comfortable with thinking about extending LLMs to other modalities of the world and their limits.

# Connecting vision and language

# Computer Vis

- 1960s - First computer vis

MASSACHUSETTS INSTITUTE OF TECHNOLOGY  
PROJECT MAC

Artificial Intelligence Group  
Vision Memo. No. 100.

July 7, 1966

## THE SUMMER VISION PROJECT

Seymour Papert

The summer vision project is an attempt to use our summer workers effectively in the construction of a significant part of a visual system.

The particular task was chosen partly because it can be segmented into sub-problems which will allow individuals to work independently and yet participate in the construction of a system complex enough to be a real landmark in the development of "pattern recognition".

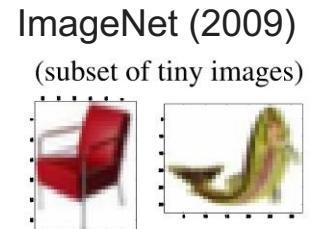
# Computer Vision Abridged History

- 1960s - First computer vision projects (MIT summer project)
- 2000s - Emergence of “tasks” and benchmarking in computer vision

PASCAL Visual Object Classes (2005-2012)

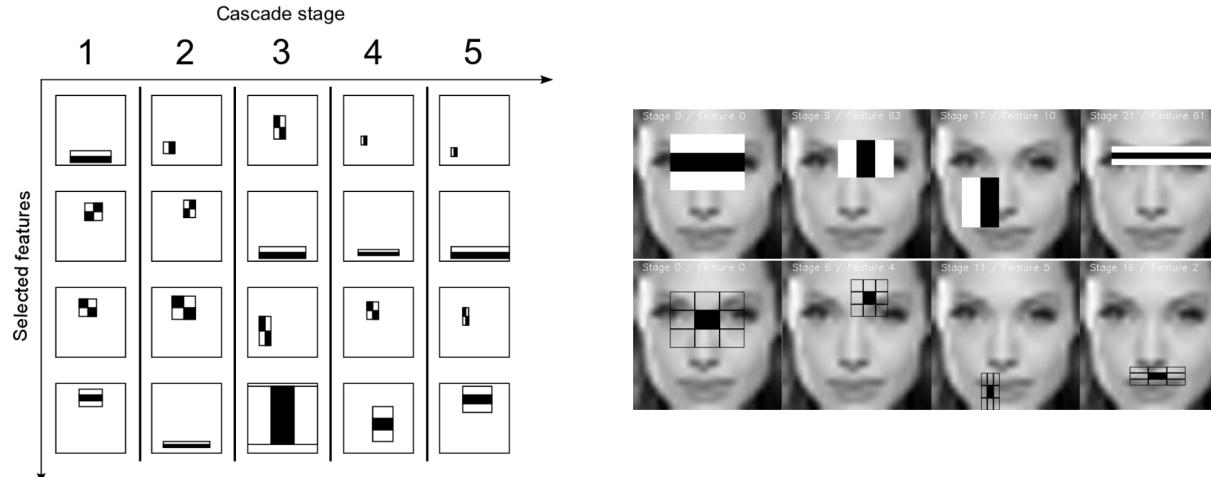


[https://en.wikipedia.org/wiki/List\\_of\\_datasets\\_in\\_computer\\_vision\\_and\\_image\\_processing](https://en.wikipedia.org/wiki/List_of_datasets_in_computer_vision_and_image_processing)



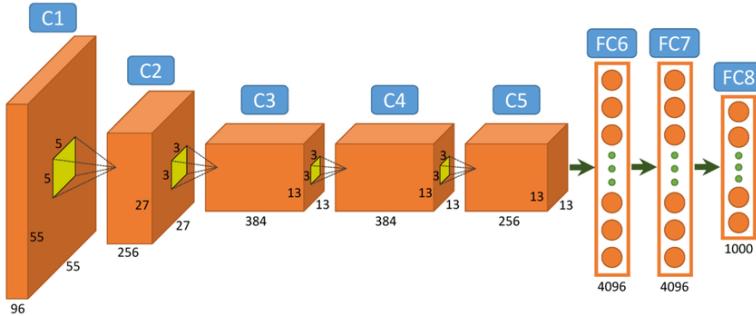
# Computer Vision Abridged History

- 1960s - First computer vision projects (MIT summer project)
- 2000s - Emergence of “tasks” and benchmarking in computer vision
- 2000s - Shallow classifiers and feature engineering (e.g., Viola & Jones algorithm)

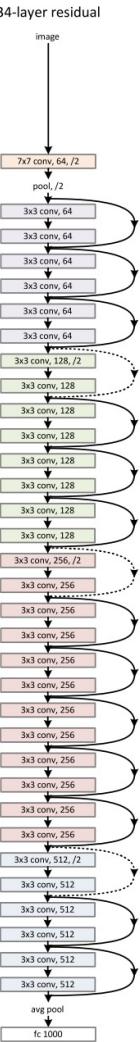


# Computer Vision Abridged History

- 1960s - First computer vision projects (MIT summer project)
- 2000s - Emergence of “tasks” and benchmarking in computer vision
- 2000s - Shallow classifiers and feature engineering
- 2012 - Deep Learning revolution:
  - Success of Convolutional neural nets in ImageNet



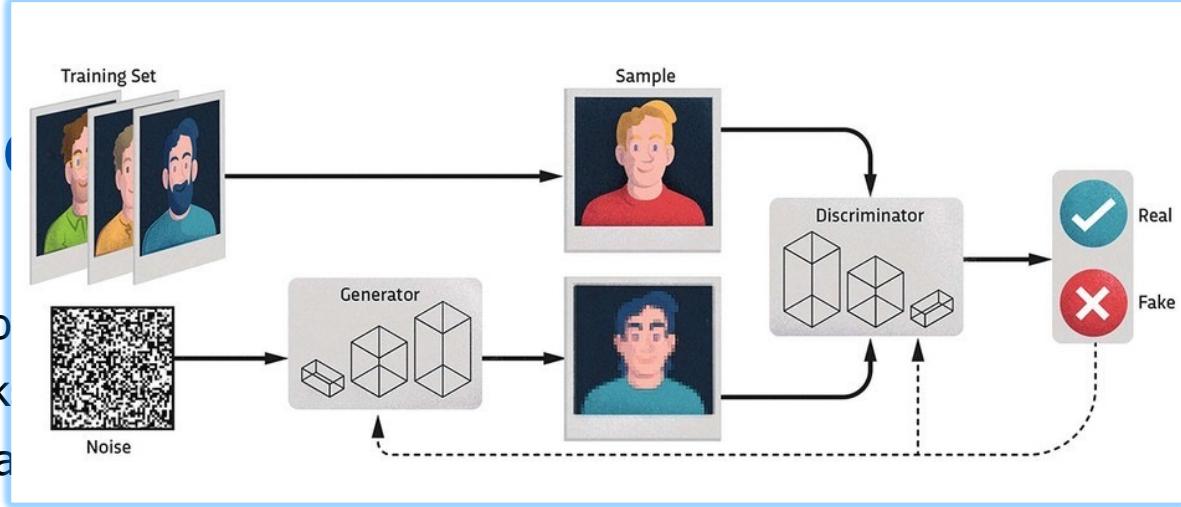
ImageNet Classification with Deep Convolutional Neural Networks, 2012



Deep Residual Learning for Image Recognition, 2015

# Computer Vision

- 1960s - First computer vision systems
- 2000s - Emergence of “task specific” systems
- 2000s - Shallow classifiers and feature extraction
- 2012 - Deep Learning revolution:
  - Success of Convolutional neural nets in ImageNet
  - Unification of architectures
  - Rise of image generation (VAEs, GANs, etc.)



# Computer Vision Abridged History

---

- 1960s - First computer vision projects (MIT summer project)
- 2000s - Emergence of “tasks” and benchmarking in computer vision
- 2000s - Shallow classifiers and feature engineering
- 2012 - Deep Learning revolution:
  - Success of Convolutional neural nets in ImageNet
  - Unification of architectures
  - Rise of image generation (VAEs, GANs, etc.)
- 2020s - Era of Vision Transformer
  - Stronger connection to language
  - Better generative models
  - Further unification of models and tasks

This is where we begin!

# Let's Consider Images – How to Encode?

---



# Vision Transformers

---

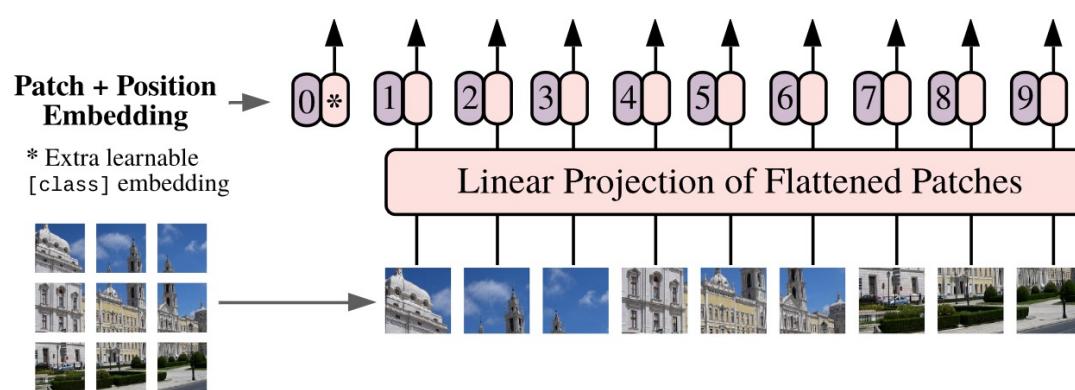
## Patch + Position Embedding

\* Extra learnable [class] embedding

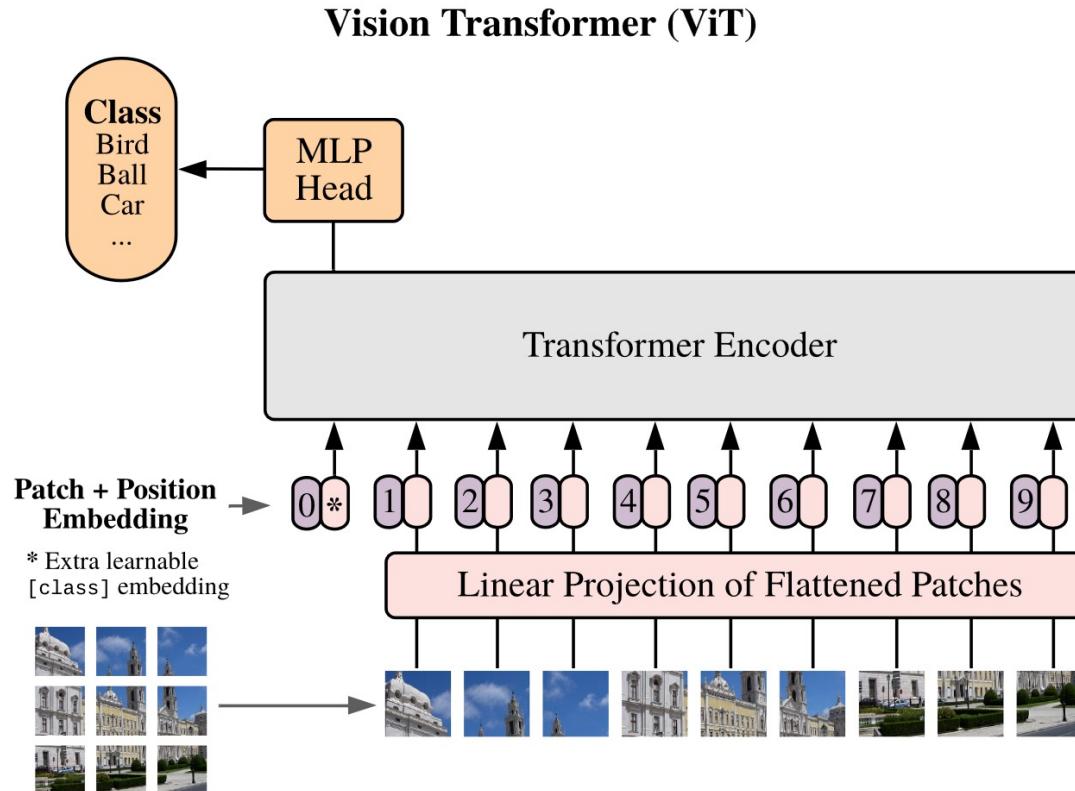


tokenize" the image by cutting it into patches of  $16px^2$ , and treating each patch as a token, e.g. embedding it into input space

# Vision Transformers



# Vision Transformers



# What about paired image-text – How to Encode?



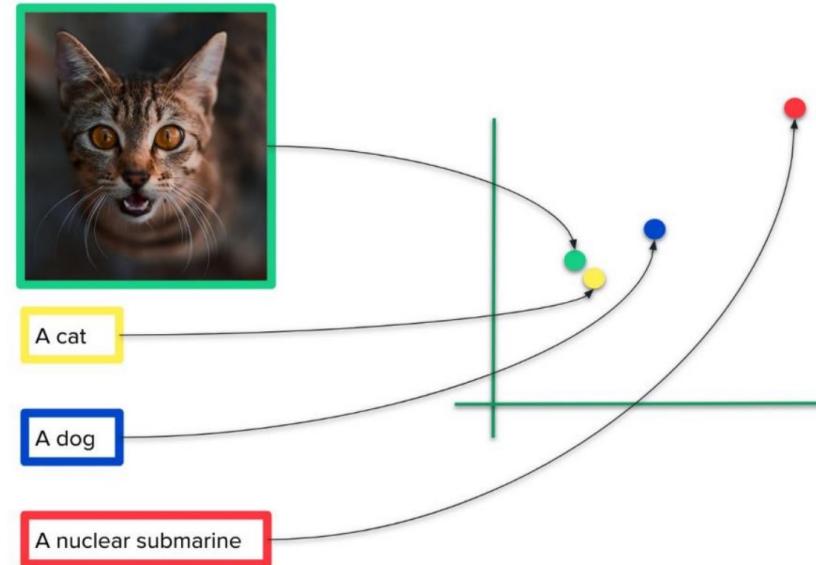
Basilica of St. John Lateran



House with Chimeras – Kiev

# What about paired image-text – How to Encode?

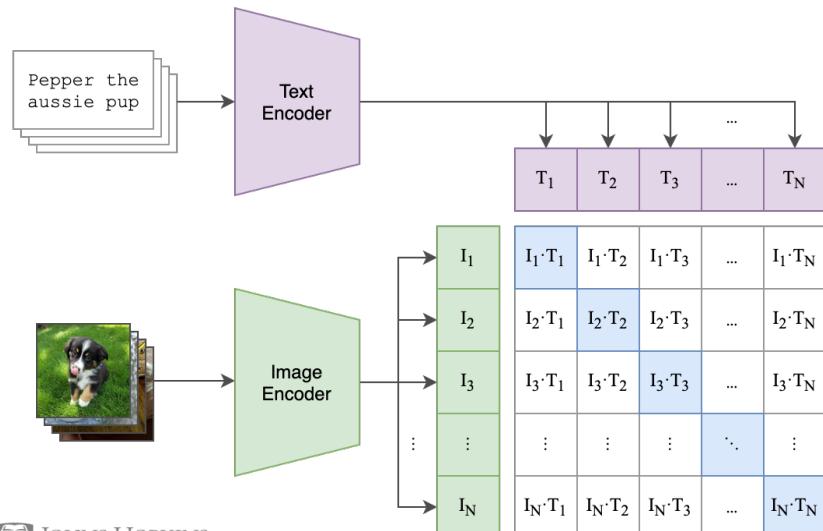
- The idea is to create a space to represent both semantics of language and image.
- Then, create a model that can align semantically-equivalent text and images nearby.



# Contrastive Language–Image Pre-training (CLIP)

- Training: simple **contrastive loss** between image-text pairs
  - Learning image representations from web-scale noisy text supervision

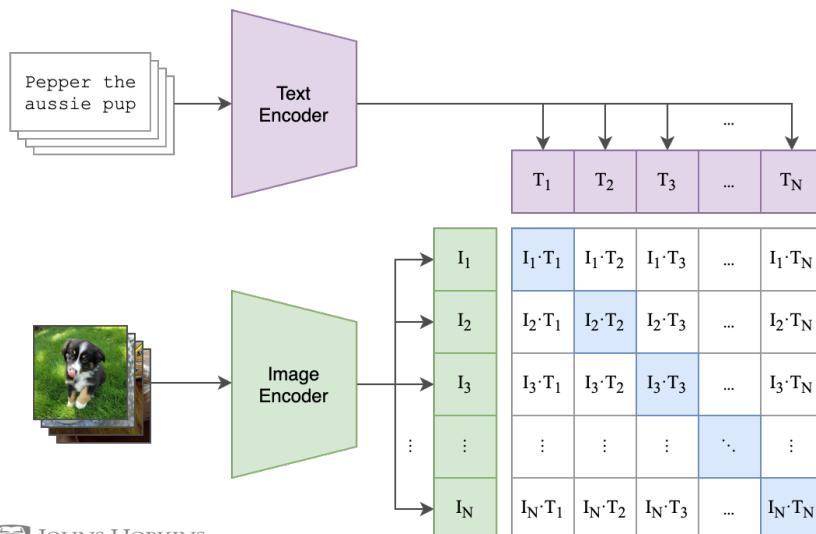
(1) Contrastive pre-training



# Contrastive Language–Image Pre-training (CLIP)

- Given a batch of  $N$  (image, text) pairs, predict which of the  $N \times N$  possible (image, text) pairings across a batch occurred.

## (1) Contrastive pre-training



```
# image_encoder - ResNet or Vision Transformer
# text_encoder - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l] - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T) #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

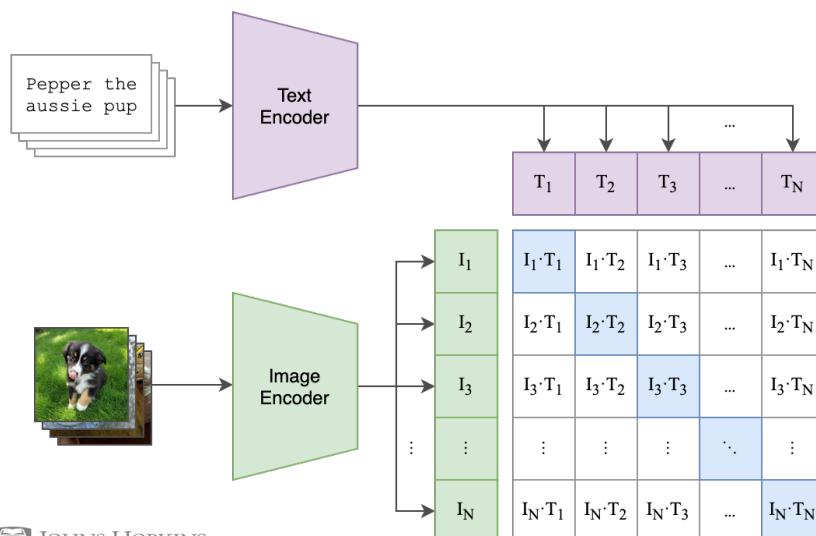
# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss = (loss_i + loss_t)/2
```

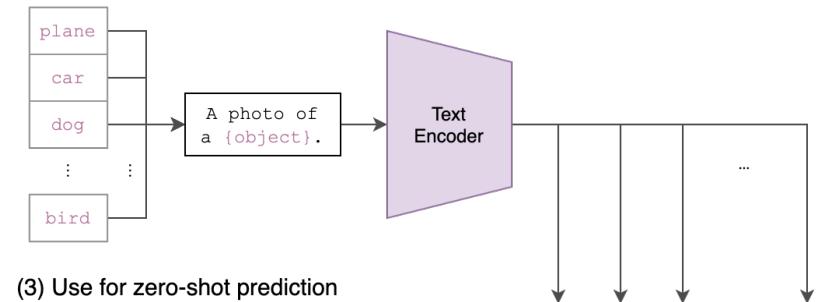
# What can CLIP do?

- Downstream: **zero-shot image** classification and image-text retrieval
  - Image classification can be reformatted as a retrieval task via considering the semantics behind label name.

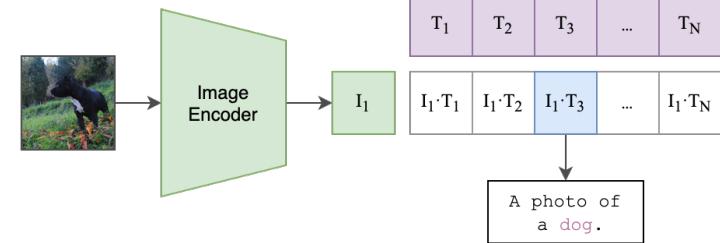
(1) Contrastive pre-training



(2) Create dataset classifier from label text



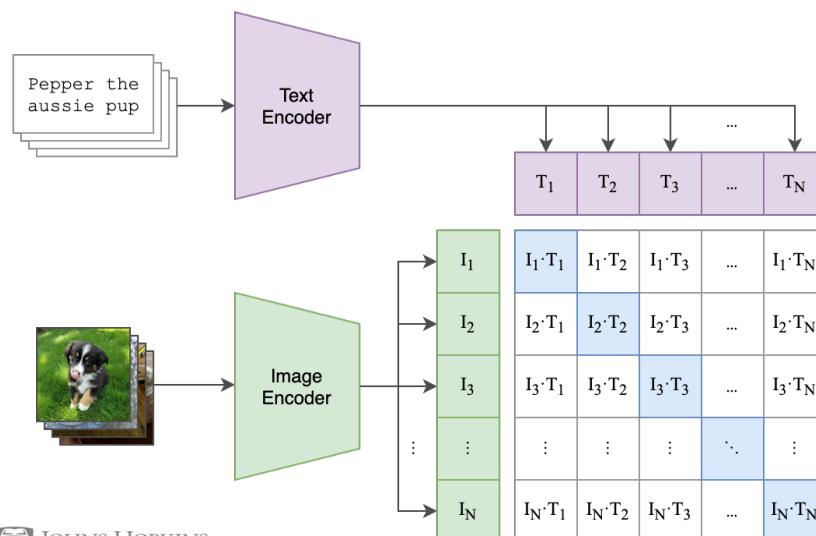
(3) Use for zero-shot prediction



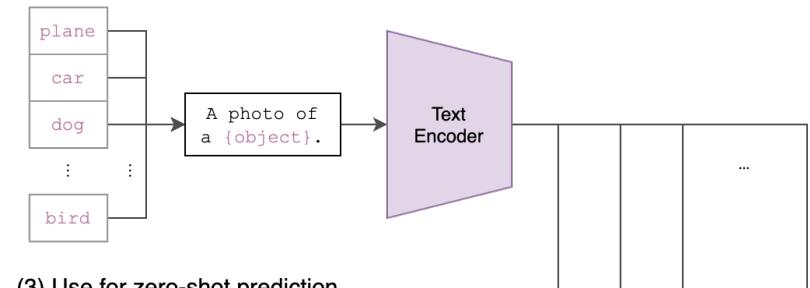
# What can CLIP do?

- Consider classifying photos of “dogs” vs “cats”—for each image, check if CLIP predicts text description “a photo of a dog” or “a photo of a cat” is more likely to be paired with it.

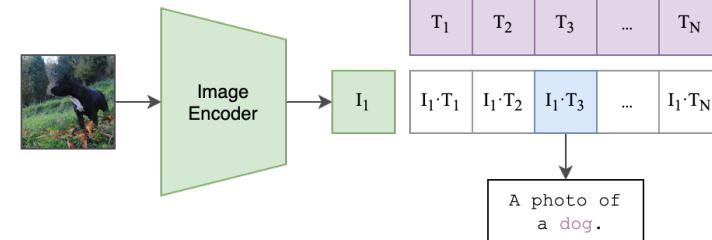
(1) Contrastive pre-training



(2) Create dataset classifier from label text



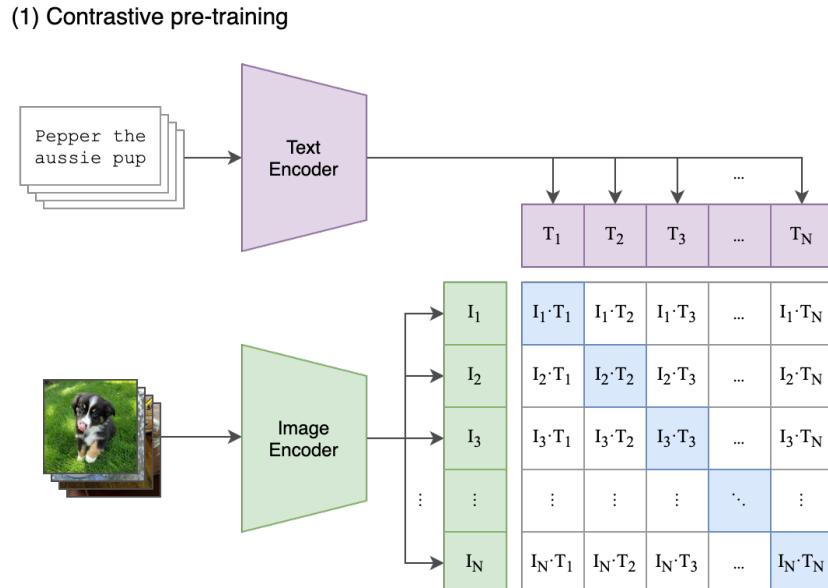
(3) Use for zero-shot prediction



# What can CLIP do?

CLIP evaluates associations between image-text pairs:

- Image Classification
- Image Searching
- ...



# What can CLIP do?

CLIP evaluates associations between image-text pairs:

- Image Classification
- Image Searching
- ...

<https://huggingface.co/openai/clip-vit-large-patch14>



Possible class names (comma-separated)

Johns Hopkins, Stanford, Berkeley, UPenn

Compute

Computation time on cpu: cached

Johns Hopkins	0.992
UPenn	0.008
Stanford	0.000
Berkeley	0.000

# What can CLIP do?

CLIP evaluates associations between image-text pairs:

- Image Classification
- Image Searching
- ...

<https://huggingface.co/openai/clip-vit-large-patch14>



Possible class names (comma-separated)

Johns Hopkins, Stanford, Berkeley, UPenn

Compute

Computation time on cpu: 0.623 s



# What can CLIP do?

CLIP evaluates associations between image-text pairs:

- Image Classification
- Image Searching
- ...

<https://huggingface.co/openai/clip-vit-large-patch14>



Possible class names (comma-separated)

Johns Hopkins, Stanford, Berkeley, UPenn

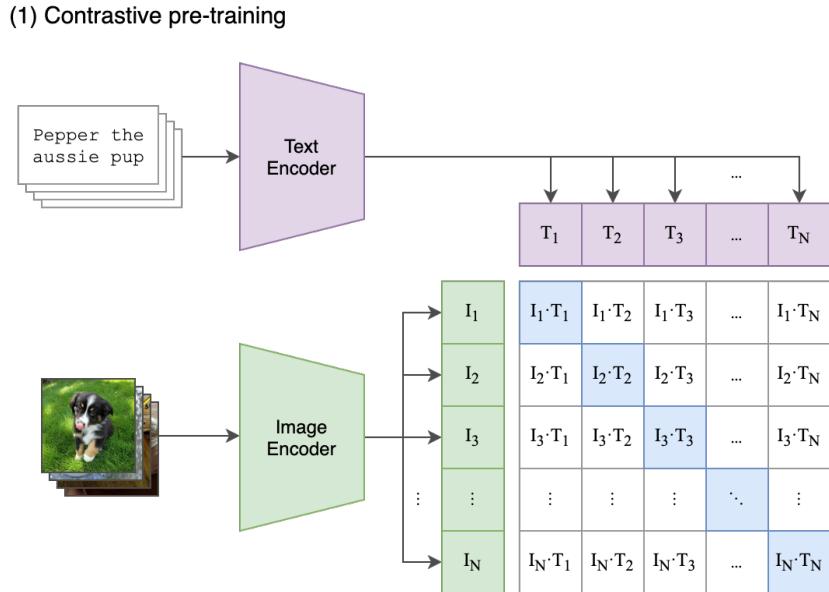
Compute

Computation time on cpu: 0.891 s

UPenn	0.928
Johns Hopkins	0.071
Stanford	0.000
Berkeley	0.000

# What can't CLIP do?

- No generation capabilities
- Prompting / In-Context Learning
  - Few-shot captioning



# What can't CLIP do?

<https://huggingface.co/openai/clip-vit-large-patch14>

- No generation capabilities
- Prompting / In-Context Learning
  - Few-shot captioning
- CLIP is not perfect.



Possible class names (comma-separated)

too many fingers, not enough fingers, just the right amount of fingers

Compute

not enough fingers

0.461

just the right amount of fingers

0.339

too many fingers

0.200

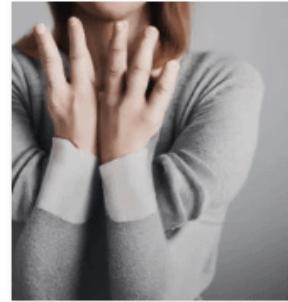
JSON Output

Maximize

# What can't CLIP do?

<https://huggingface.co/openai/clip-vit-large-patch14>

- No generation capabilities
- Prompting / In-Context Learning
  - Few-shot captioning
- CLIP is not perfect.



Possible class names (comma-separated)

too many fingers, not enough fingers, just the right amount of fingers

Compute

Computation time on cpu: 0.251 s

just the right amount of fingers 0.412

too many fingers 0.316

not enough fingers 0.272

# What can't CLIP do?

- No generation capabilities
- Prompting / In-Context Learning
  - Few-shot captioning
- CLIP is not perfect.

<https://huggingface.co/openai/clip-vit-large-patch14>



Possible class names (comma-separated)

black and white cat, black and white dog, brown cat, brown dog

Compute

Computation time on Intel Xeon 3rd Gen Scalable cpu: 0.625 s

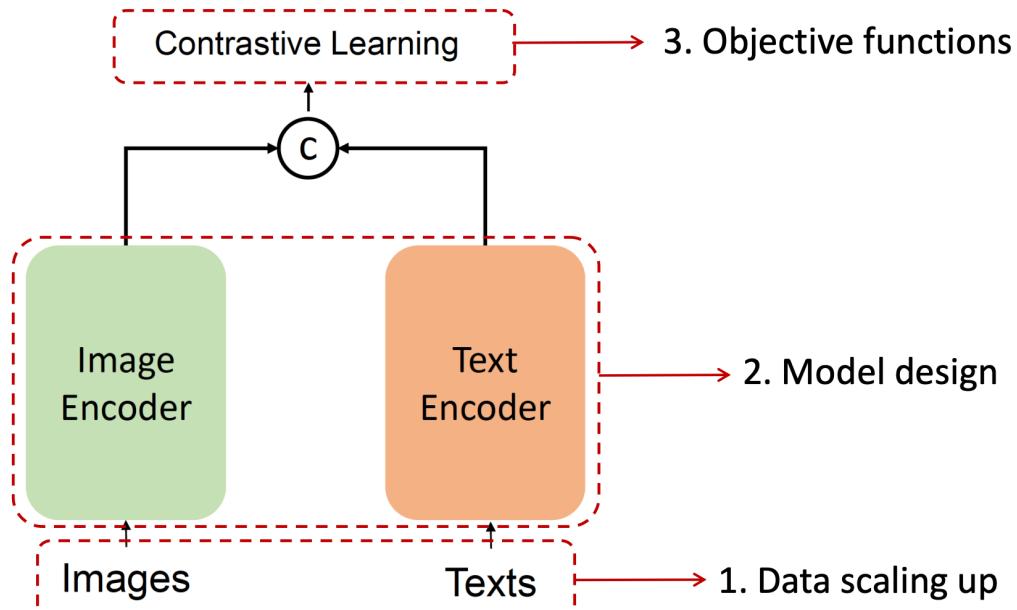
black and white dog	0.460
black and white cat	0.262
brown cat	0.181
brown dog	0.097

↔ JSON Output

Maximize

# What happened after CLIP?

- Ton of follow-up work on improve its design:



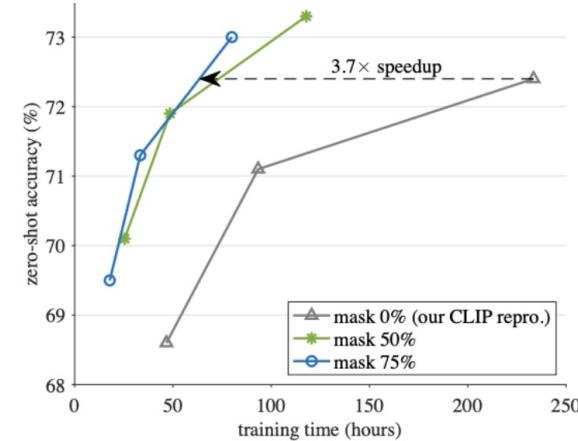
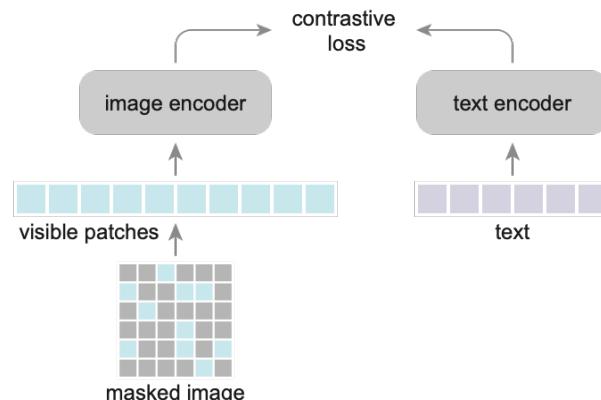
# What happened after CLIP?

- Open-source reproduction and scaling up
  - Open-source model: OpenCLIP
  - Pre-training on LAION-5B dataset

	Data	Arch.	ImageNet	VTAB+	COCO
CLIP [55]	WIT-400M	L/14	75.5	55.8	61.1
Ours	LAION-2B	L/14	75.2	54.6	71.1
Ours	LAION-2B	H/14	<u>78.0</u>	<u>56.4</u>	<u>73.4</u>

# What happened after CLIP?

- Open-source reproduction and scaling up
- Modifying the training process
  - A notable example is FLIP
    - Randomly masking out image patches
    - Does not hurt the performance, yet it improves the training efficiency



Scaling Language-Image Pre-training via Masking, 2022

# What happened after CLIP?

- Open-source reproduction and scaling up
- Modifying the training process
- Adding more modalities:
  - ImageBind: linking 7 modalities

<https://facebookresearch.github.io/ImageBind>  
ImageBind: One Embedding Space To Bind Them All, 2023

1) Cross-Modal Retrieval

Audio	Images & Videos			Depth	Text
					<p>A fire crackles while a pan of food is frying on the fire. "Fire is crackling then wind starts blowing." "Firewood crackles then music..."</p>
					<p>A baby is crying while a toddler is laughing. "A baby is laughing while an adult is laughing." "A baby laughs and something..."</p>

2) Embedding-Space Arithmetic

3) Audio to Image Generation

# What happened after CLIP?

---

- Open-source reproduction and scaling up
- Modifying the training process
- Adding more modalities:
- Generative models:
  - Text → Image
  - Image, Text → Image
  - Image, Image → Text
  - ...

Forthcoming

# Summary

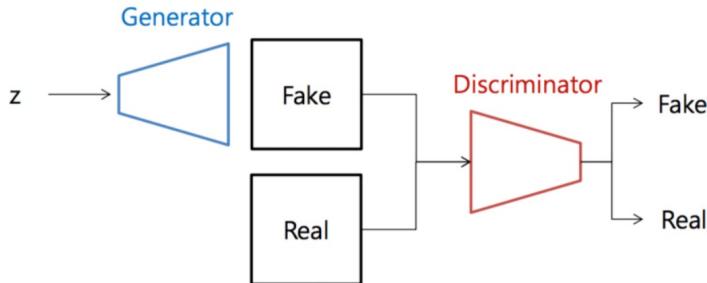
---

- The same computational architecture (e.g., Transformer) can represent different modalities.
- One can build models that embed different modalities in the same space.
- Next: generative vision-language models.

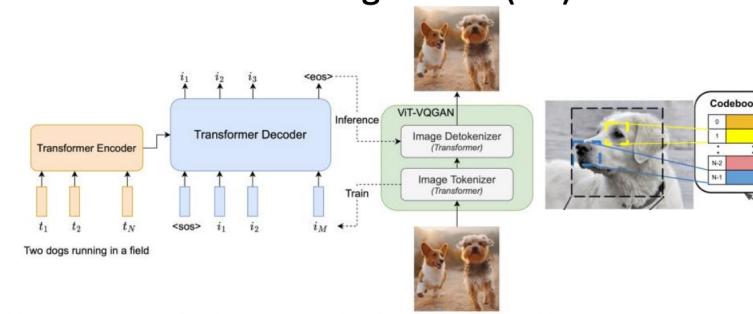
# Generative Vision-Language Models

# Image Generation Toolkit

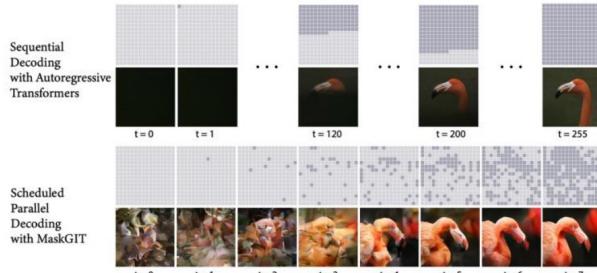
## Generative Adversarial Networks (GAN)



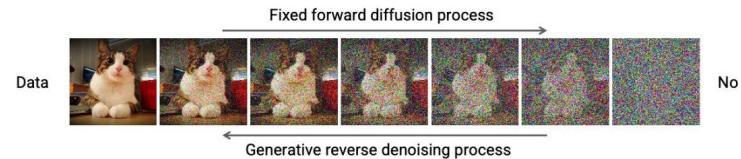
## Auto-regressive (AR)



## Non-AR Transformer



## Diffusion



# Image Generation Toolkit: Diffusions

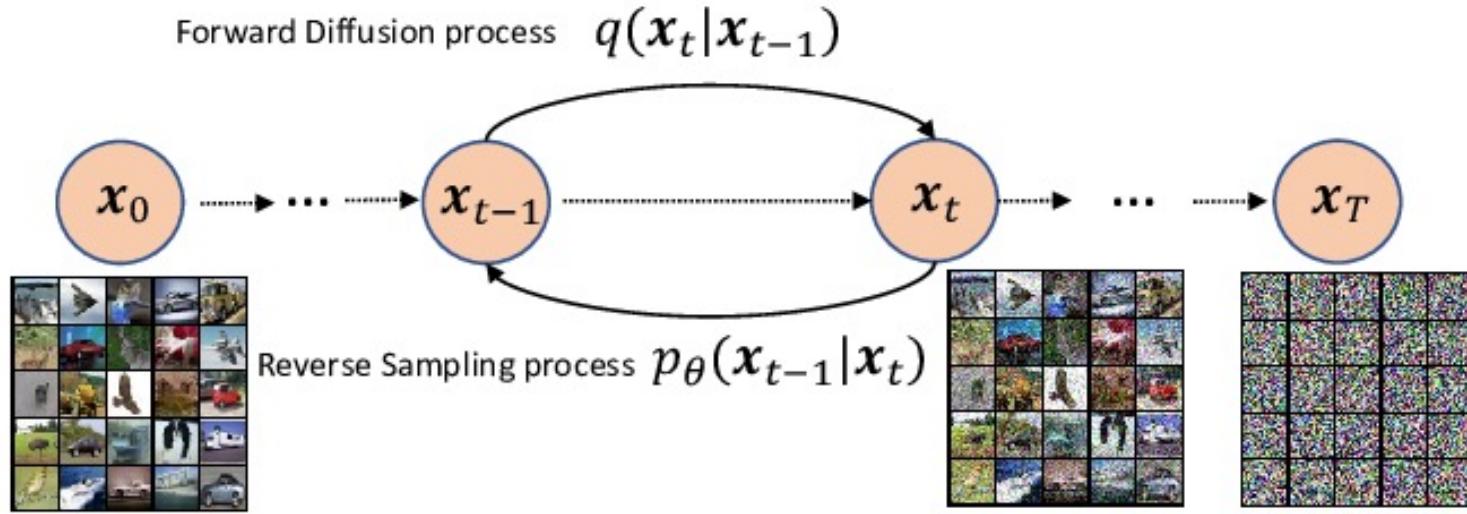
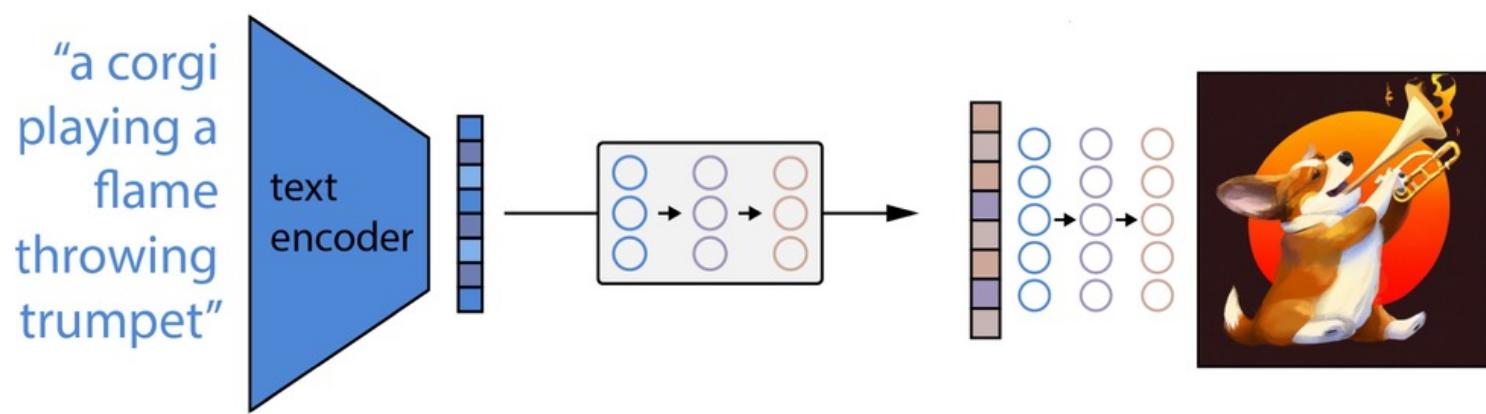


Image generation is out of scope for us. You can learn more by taking a computer vision class or watching the online tutorials. For example:

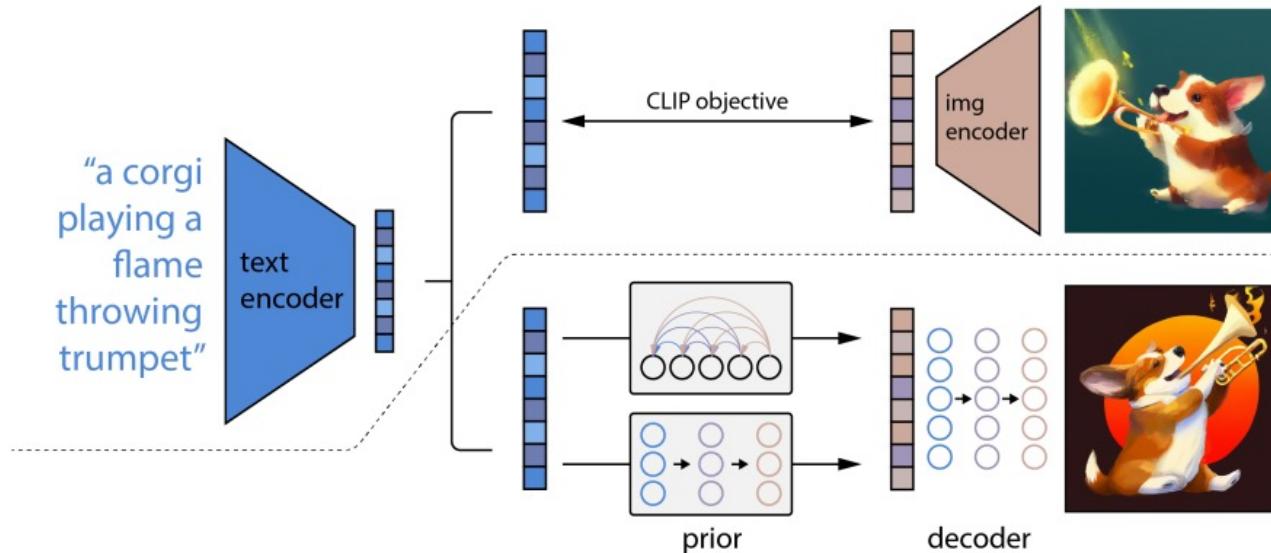
<https://cvpr2022-tutorial-diffusion-models.github.io/>



# Text to Image Generation

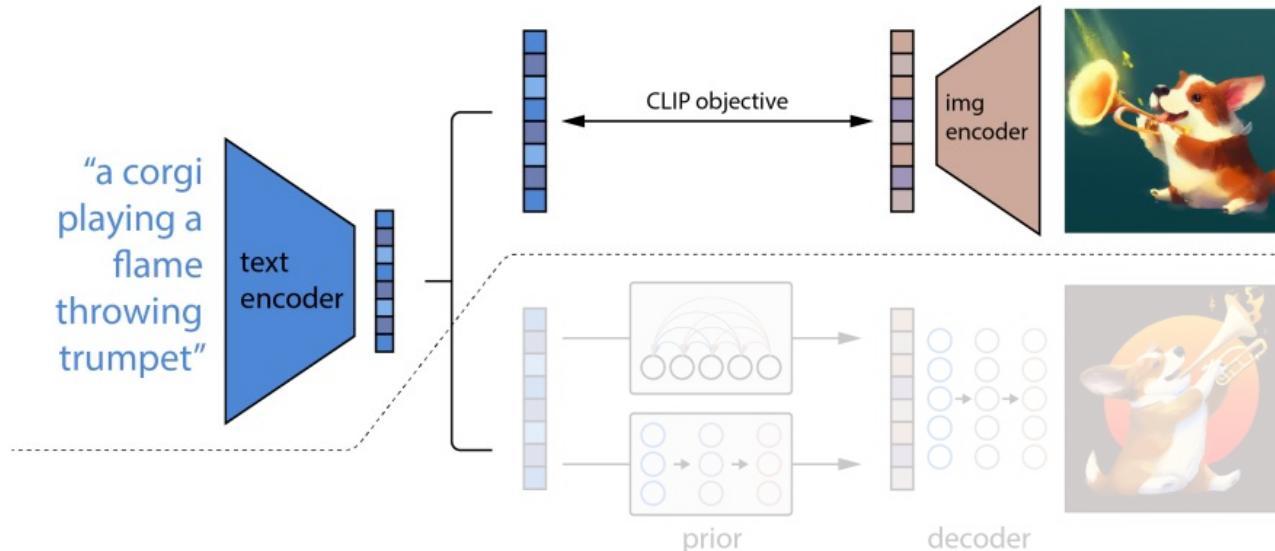


# DALL-E



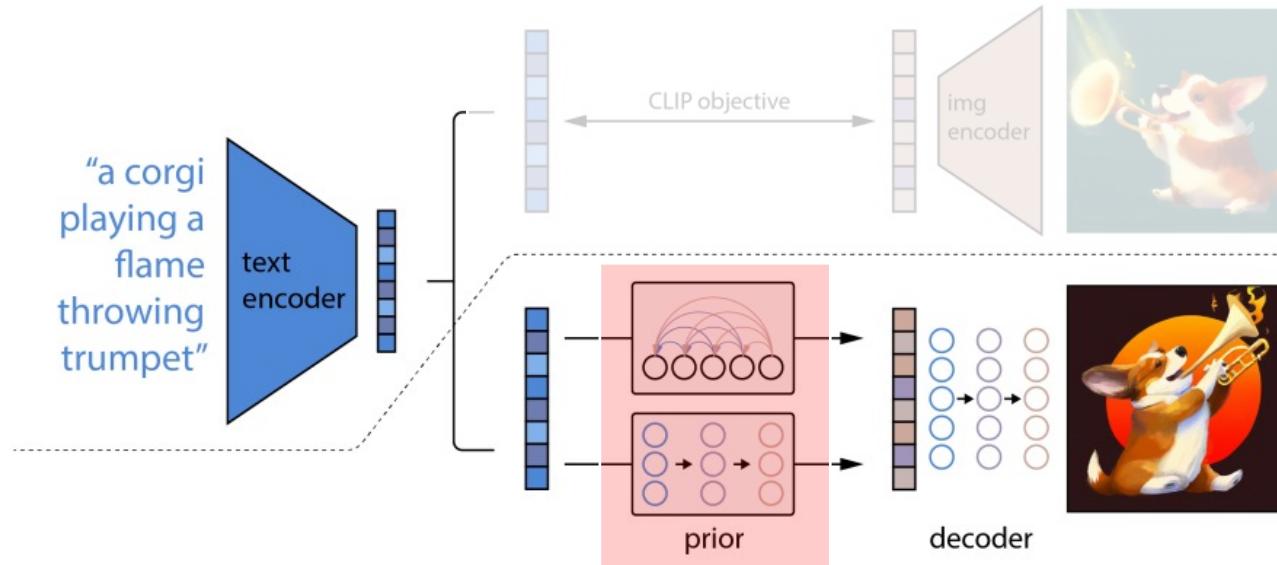
# DALL-E

- DALL-E is built on top of a pre-trained CLIP model.
  - This frozen model provides the representations of text and images.



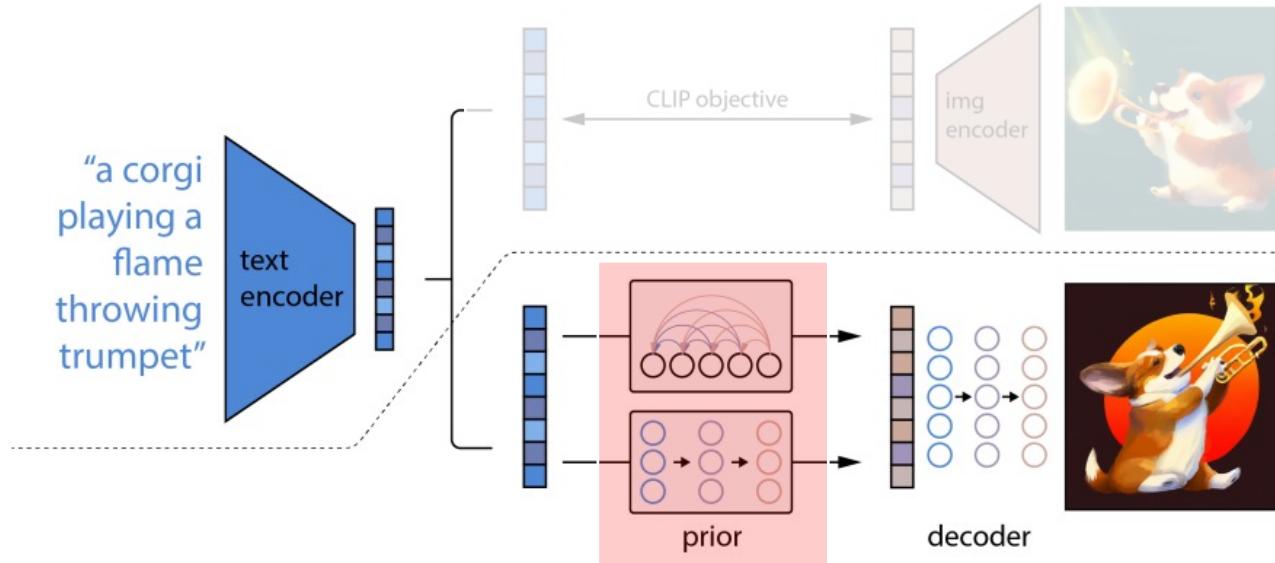
# DALL-E

- First, a text prompt is input into a **text encoder** that is trained to map the prompt to a representation space.



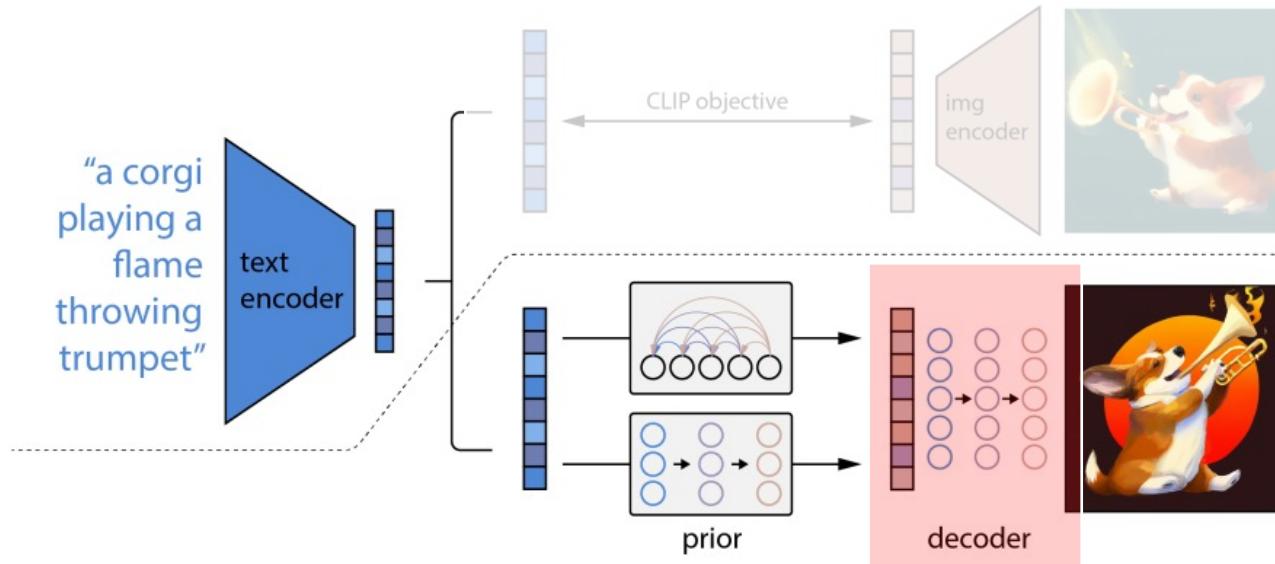
# DALL-E

- A **prior** maps the text encoding to a corresponding **image encoding** that captures the semantic information of the prompt contained in the text encoding.



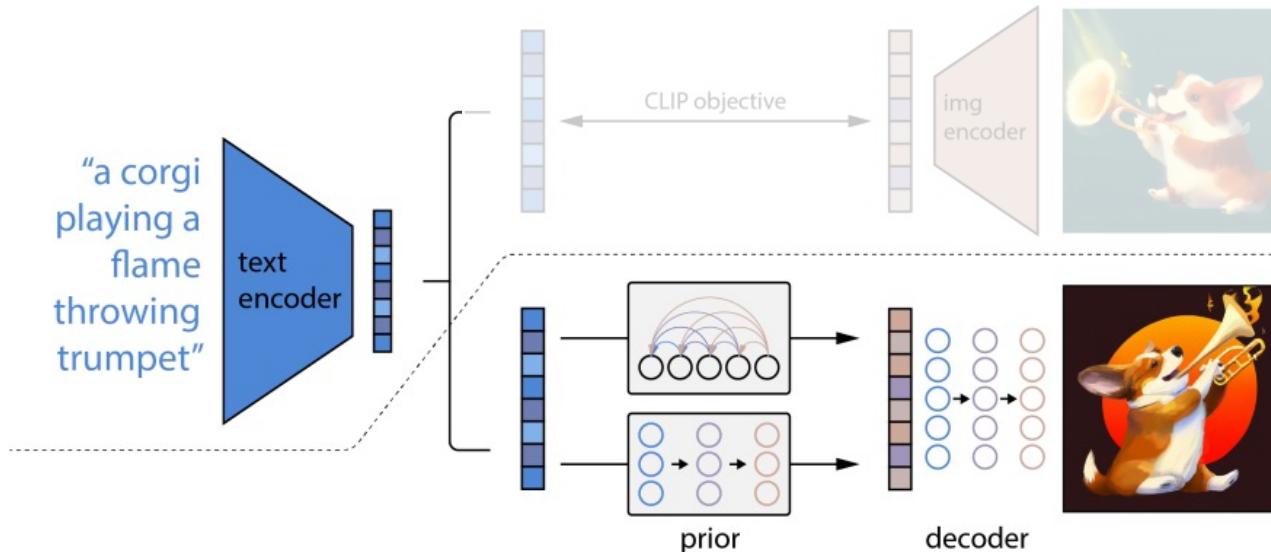
# DALL-E

- Finally, an **image decoder** stochastically generates an image which is a visual manifestation of this semantic information.



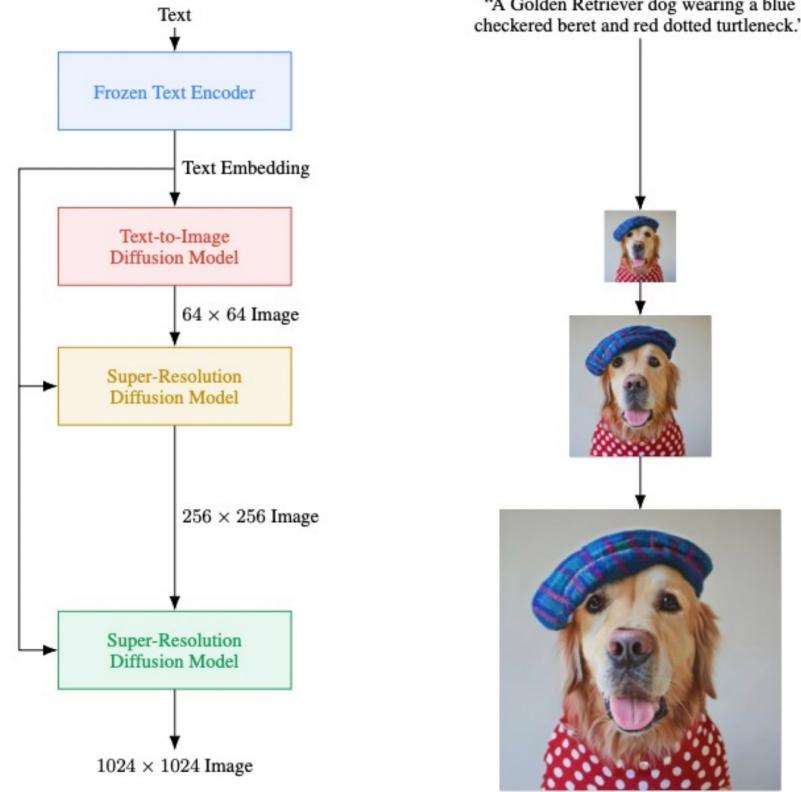
# DALL-E

- [Ignoring some details] Both modules are implemented using “diffusion models”.



# Imagen

- Simpler than DALL-E
- Key components:
  - Frozen language model providing text embeddings to all diffusion models.
    - Human raters prefer T5-XXL as the text encoder over CLIP encoder.
  - Cascaded diffusion models providing effective way to generate high-quality images.



# Imagen





Generate an image of a dancing elephant



Images

Videos

Free

Perspectives

App

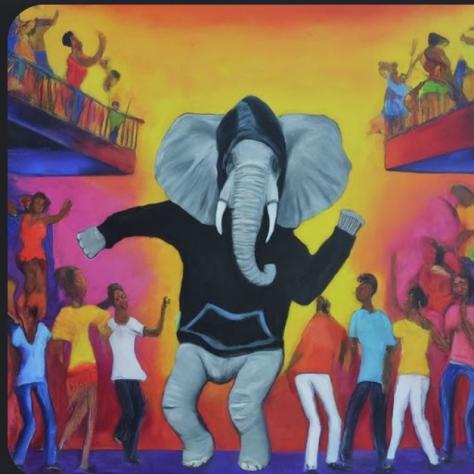
News

Shopping

Books

Maps

Generative AI is experimental. Learn more ::



*Generative imagery may display inaccurate, misleading or offensive images that do not represent Google's views*

Images

Perspectives

Videos

Shopping

Books

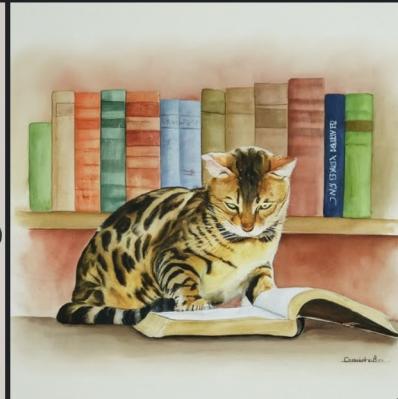
News

Maps

Flights

Finance

Generative AI is experimental. Learn more



Generative AI is experimental. Learn more



# A ton of work on $T \rightarrow V$

- Text to video



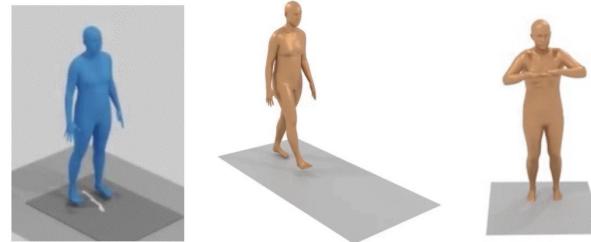
Text-to-Video generation: "a panda is playing guitar on times square"

- Text to 3D shapes

"A living room with a lit furnace, couch, and cozy curtains, bright lamps that make the room look well-lit."



- Text to motions/navigation



What about models that generate **text**?

# Image-to-Text Generative Models

- Model architectures:
  - Pre-trained image encoders
  - Pre-trained language models
  - Modules (to be trained) to connect the two modalities

A dog lying on the grass next to a frisbee



Language

Image

Language Model

Connection Module

Vision Encoder



# LLaVA Model

Sometimes I just look at pictures of the earth from space and I marvel at how beautiful it all is.



User  
LLaVA

Can you explain this meme in detail?

The meme in the image is a creative and humorous take on food, with a focus on chicken nuggets, as the center of the universe. The meme begins with the phrase "Sometimes I just look at pictures of the Earth from space and I marvel at how beautiful it all is..." The main image features a tray of cooked chicken nuggets, with the chicken pieces positioned to resemble the continents and islands on a world map. The meme playfully suggests that the chicken nuggets represent the Earth, and the various locations depicted in the photo are actually chicken nugget versions of different places. The punchline of the meme appears in smaller text at the bottom, reading "I mean, it's not the real Earth, but how beautiful it is all is." This lighthearted and entertaining meme combines food and humor, using the image of chicken nuggets to create a funny and imaginative representation of the world.

---

User  
GPT-4 [36]

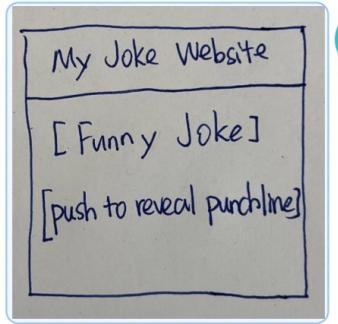
Can you explain this meme?

This meme is a joke that combines two unrelated things: pictures of the earth from space and chicken nuggets. The text of the meme suggests that the image below is a beautiful picture of the earth from space. However, the image is actually of chicken nuggets arranged to vaguely resemble a map of the world. The humor in this meme comes from the unexpected juxtaposition of the text and the image. The text sets up an expectation of a majestic image of the earth, but the image is actually something mundane and silly.

<https://arxiv.org/pdf/2304.08485.pdf>

Visual Instruction Tuning, 2023

# LLaVA Model



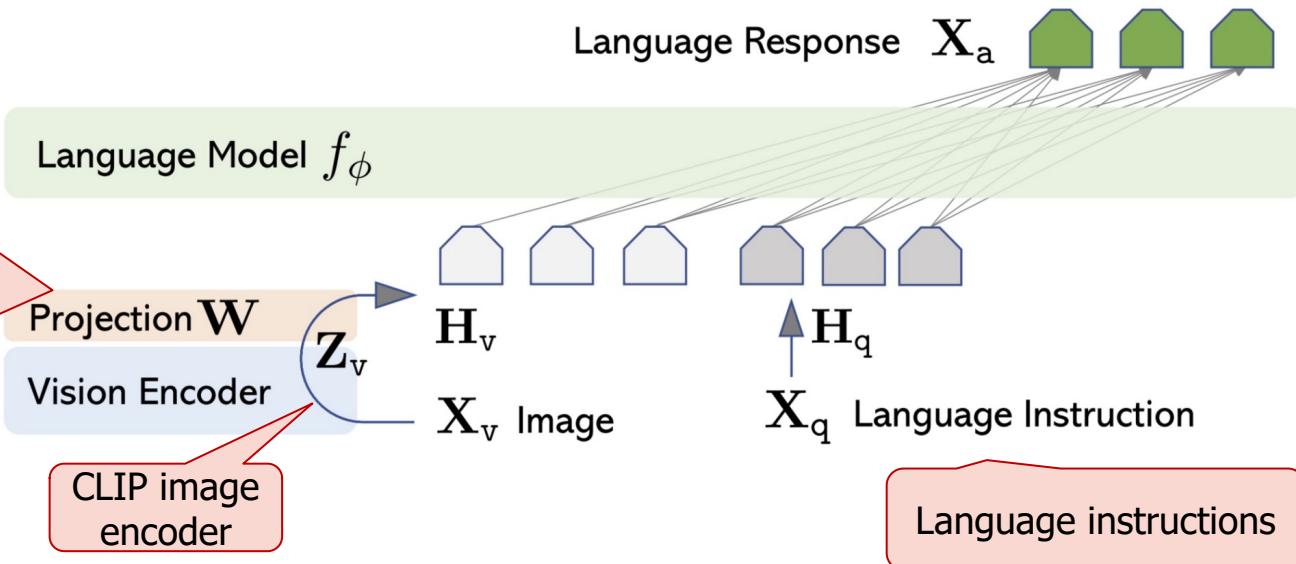
Turn this mock-up into an interactive website using html/js, replace the joke with a real joke.



```
<!DOCTYPE html>
<html>
<head>
<title>My Joke Website</title>
<style>
body { background-color: white; font-family: Arial, sans-serif; }
.container { display: flex; flex-direction: column; align-items: center; justify-content: center; height: 100vh; }
h1 { font-size: 36px; margin-bottom: 20px; }
p { font-size: 20px; }
button { background-color: blue; color: white; border: none; padding: 10px 20px; font-size: 18px; cursor: pointer; }
</style>
</head>
<body>
<div class="container">
<h1>My Joke Website</h1>
<p id="punchline">Welcome to my joke website. Push the button to reveal the punchline!</p>
<button onclick="revealPunchline()">Push me!</button>
</div>
<script>
function revealPunchline() {
  document.getElementById("punchline").innerHTML = "The punchline is: 'Why was the math book sad? Because it had too many problems.'";
}
</script>
</body>
</html>
```

# LLaVA Architecture

This projection matrix maps the embedded image to the same dimension as the LLM input (word embeddings).



# LLaVA: GPT4-assisted data generation

- **The raw data:** image captioning datasets
  - Consists of images, captions and bounding boxes of objects

## Context type 1: Captions

A group of people standing outside of a black vehicle with various luggage.

Luggage surrounds a vehicle in an underground parking area

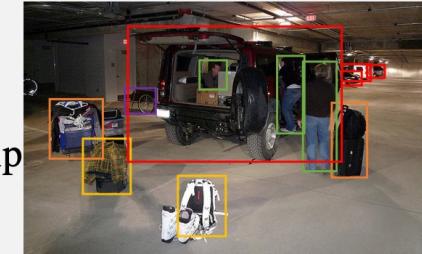
People try to fit all of their luggage in an SUV.

The sport utility vehicle is parked in the public garage, being packed for a trip

Some people with luggage near a van that is transporting it.

## Context type 2: Boxes

person: [0.681, 0.242, 0.774, 0.694], backpack: [0.384, 0.696, 0.485, 0.914], suitcase: ...<omitted>



- **Augmentation:** it uses GPT4 to augment these with detailed captions.

## Context type 1: Captions

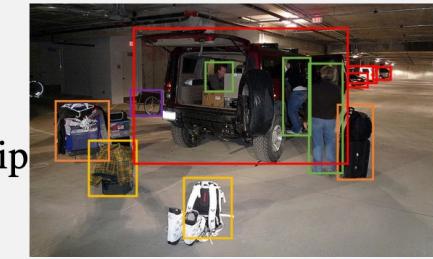
A group of people standing outside of a black vehicle with various luggage.

Luggage surrounds a vehicle in an underground parking area

People try to fit all of their luggage in an SUV.

The sport utility vehicle is parked in the public garage, being packed for a trip

Some people with luggage near a van that is transporting it.



## Context type 2: Boxes

person: [0.681, 0.242, 0.774, 0.694], backpack: [0.384, 0.696, 0.485, 0.914], suitcase: ...<omitted>

## Response type 1: conversation

Question: What type of vehicle is featured in the image?

Answer: The image features a black sport utility vehicle (SUV) ...<omitted>

## Response type 2: detailed description

The image is an underground parking area with a black sport utility vehicle (SUV) parked. There are three people in the scene, with one person standing closer to the left side of the vehicle, another person in the middle, and the third person on the right side. They are all working together to pack their luggage into the SUV for a trip. ...<omitted>

## Response type 3: complex reasoning

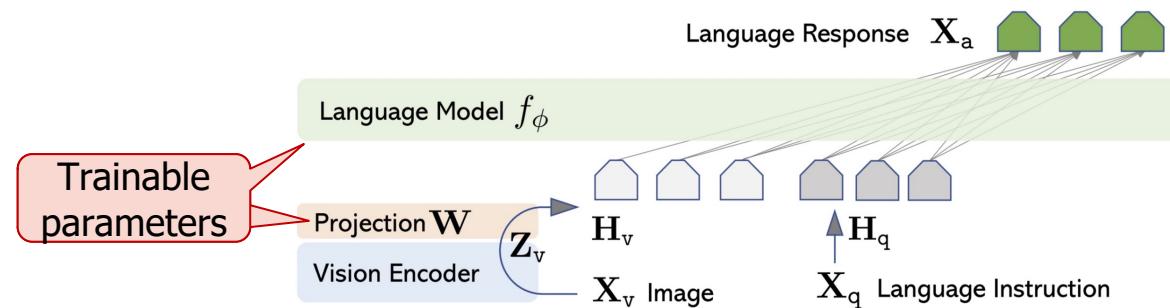
Question: What challenges do these people face?

Answer: In the image, a group of people is standing outside a black SUV in a parking area, surrounded by various pieces of luggage, including suitcases and backpacks. They are facing the challenge of fitting all their luggage into the black SUV. There are multiple suitcases and backpacks to be packed, which suggests that the group has a significant amount of belongings ...<omitted>



# LLaVA Architecture: Training

- **Step 1:** Feature alignment — aligning the representation of Vision Encoder and LLM
  - Both Vision Encoder and LLM are kept frozen.
  - The only training parameter is  $W$  (the projection matrix)
- **Step 2:** End-to-end fine-tuning
  - Vision Encoder is kept frozen. The training params are  $W$  and LLM.



# Many open-source models ...

---



- BLIP/BLIP2 [Li et al. 2023]
- Instruct-BLIP [Dai et al. 2023]
- MiniGPT4 [Zhu et al. 2023]
- CoCa [Yu et al. 2022]
- Shikra [Chen et al. 2023]
- LLaVA 1.5 [Liu et al. 2023]
- ...

# Multi-modal GPT4

- Model: unknown
- Strong zero-shot visual understanding & reasoning capability
- How do we build opensource models that are as good as GPT4?

## GPT-4 visual input example, Extreme Ironing:

User      What is unusual about this image?

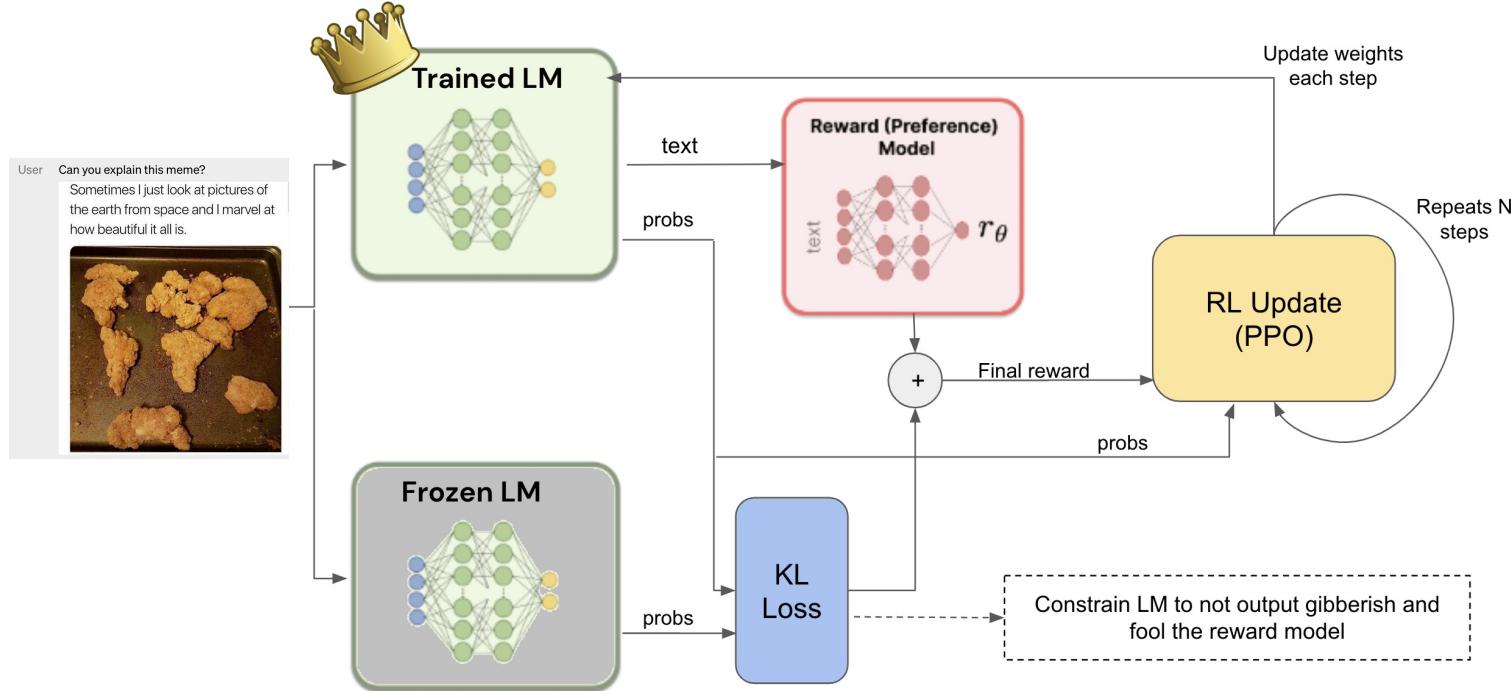


Source: <https://www.barnorama.com/wp-content/uploads/2016/12/03-Confusing-Pictures.jpg>

GPT-4

The unusual thing about this image is that a man is ironing clothes on an ironing board attached to the roof of a moving taxi.

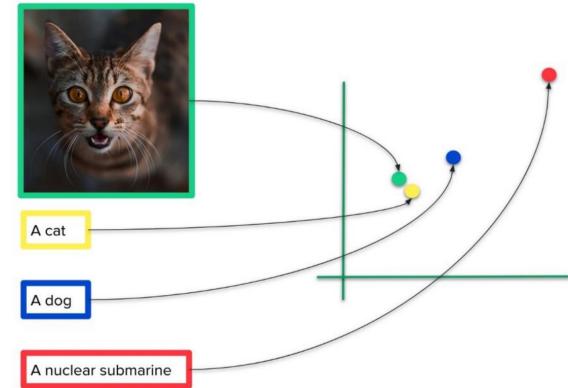
# Multimodal - RLHF



# Summary

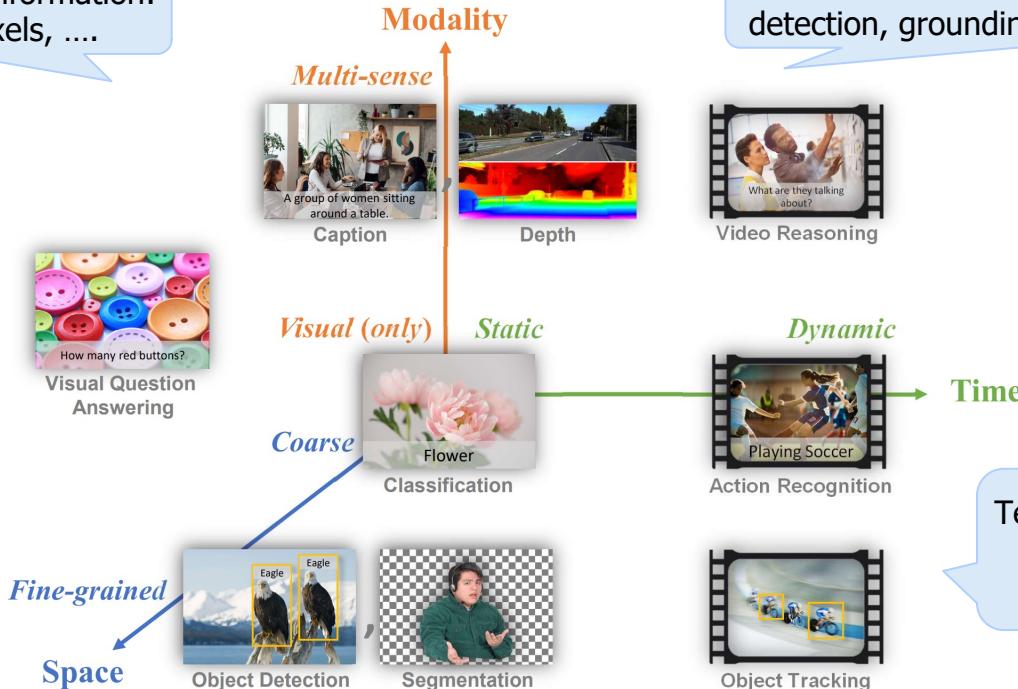
---

- There are massive amounts of unimodal data in the world.
  - We can build strong unimodal self-supervised models.
- Multi-modal models: the key challenge is **aligning** the representations.
- Many recent successes but a lot of challenges remain.
  - Making the models efficient.
  - Further scaling up — data scaling and model scaling
  - Better alignment of the representations — more natural/richer signals for alignment.



# Challenges for Future Multi-modal Models

Different modalities of information:  
Images, regions, pixels, ....



Different types  
of inputs

Different tasks: Classification, captioning,  
detection, grounding, segmentation, depth, ...

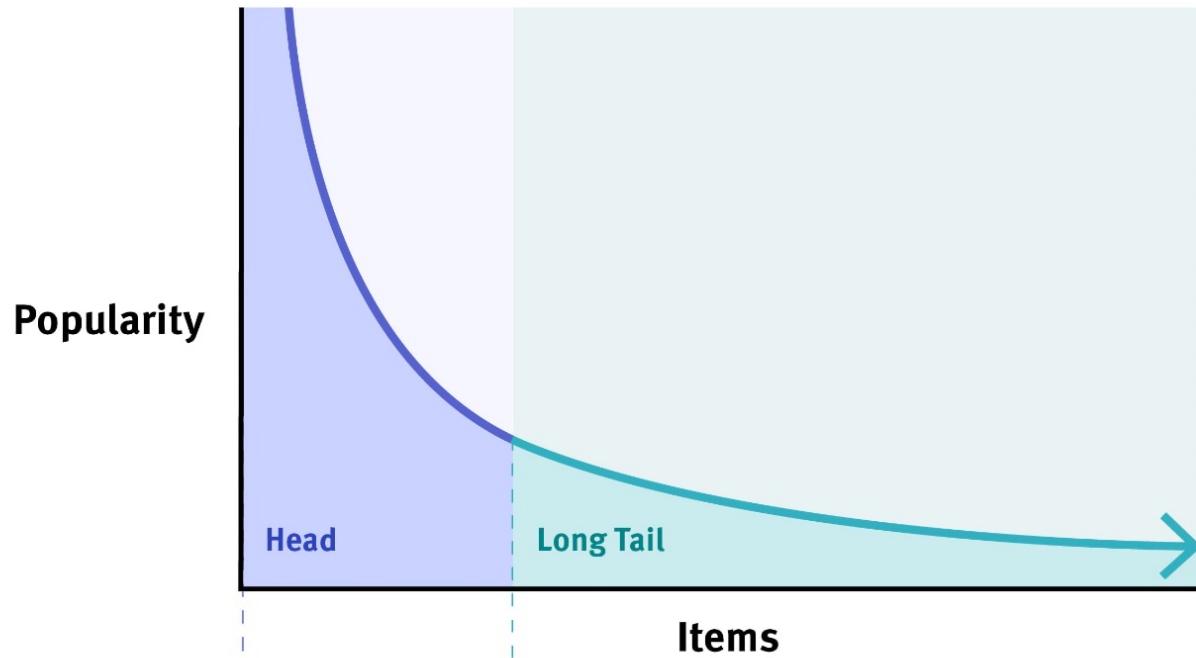
Temporality: static images,  
short videos, very long  
videos, ...

# Challenges for Future Multi-modal Models

---



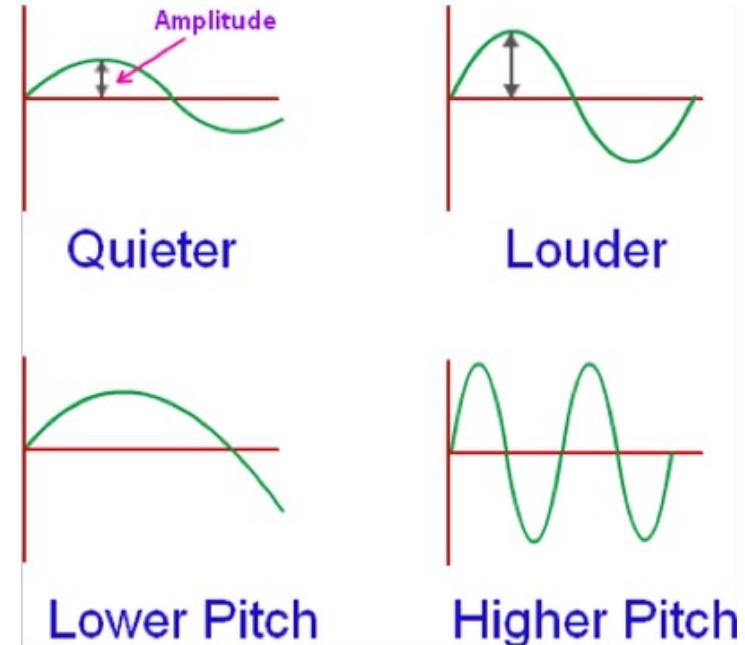
# The long tail: *Most things are infrequent*



# Transformers for Speech and Audio

# Dealing with Audio Data

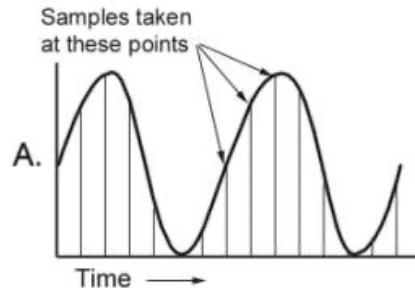
- Audio is originally continuous wave.



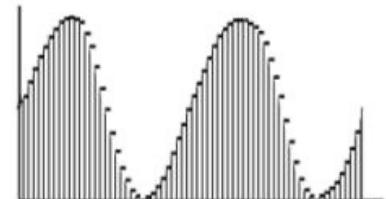
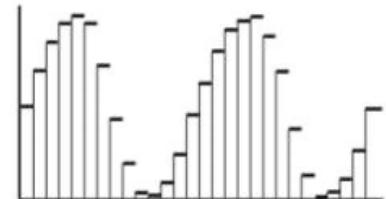
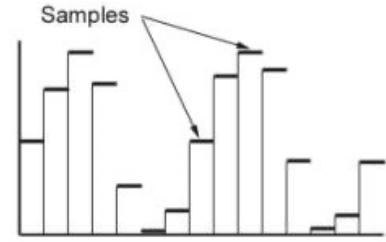
# Dealing with Audio Data

- Audio is originally continuous wave.
- When recording it, we sample from it.
- The choice of sampling rate determines the fidelity of the recording.
- If the sampling rate is too low, the digital sound will be muffled.
- Nyquist limit: the minimum rate.

Analog Wave

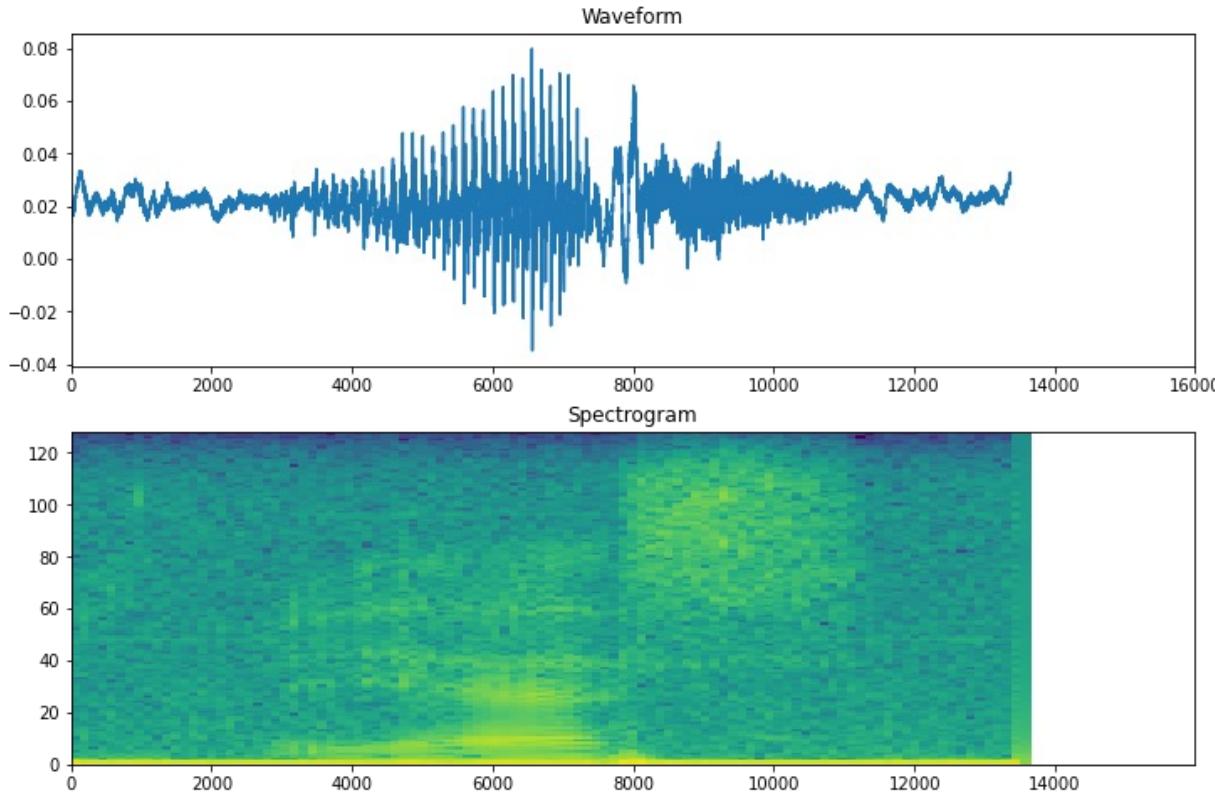


Digital Result

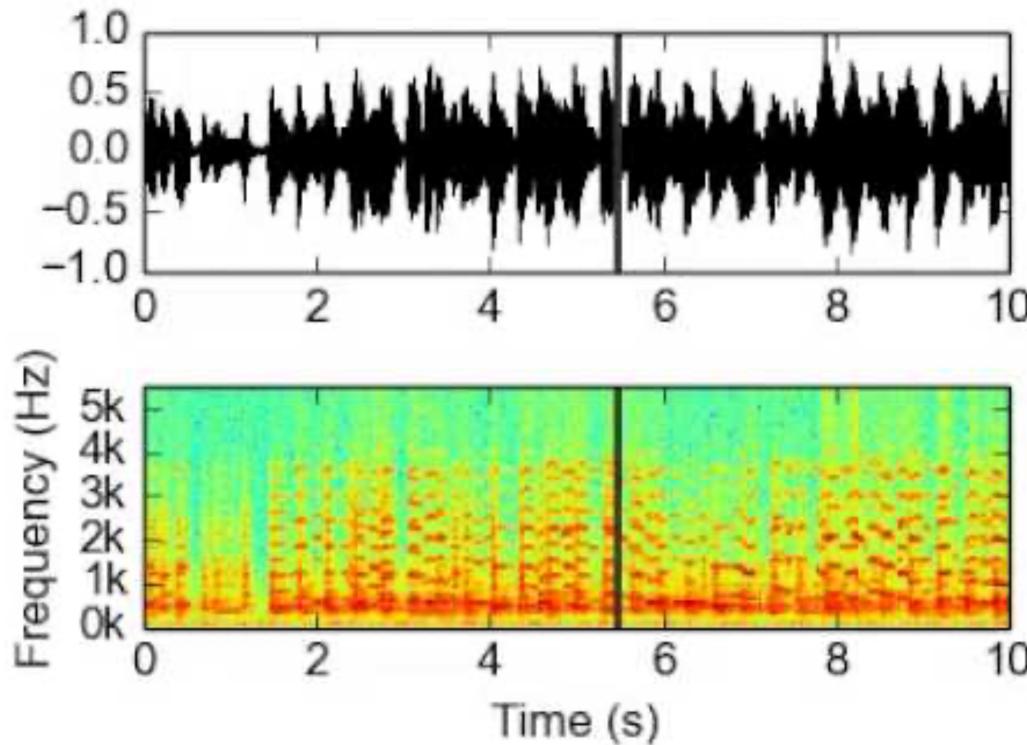


# Spectrogram

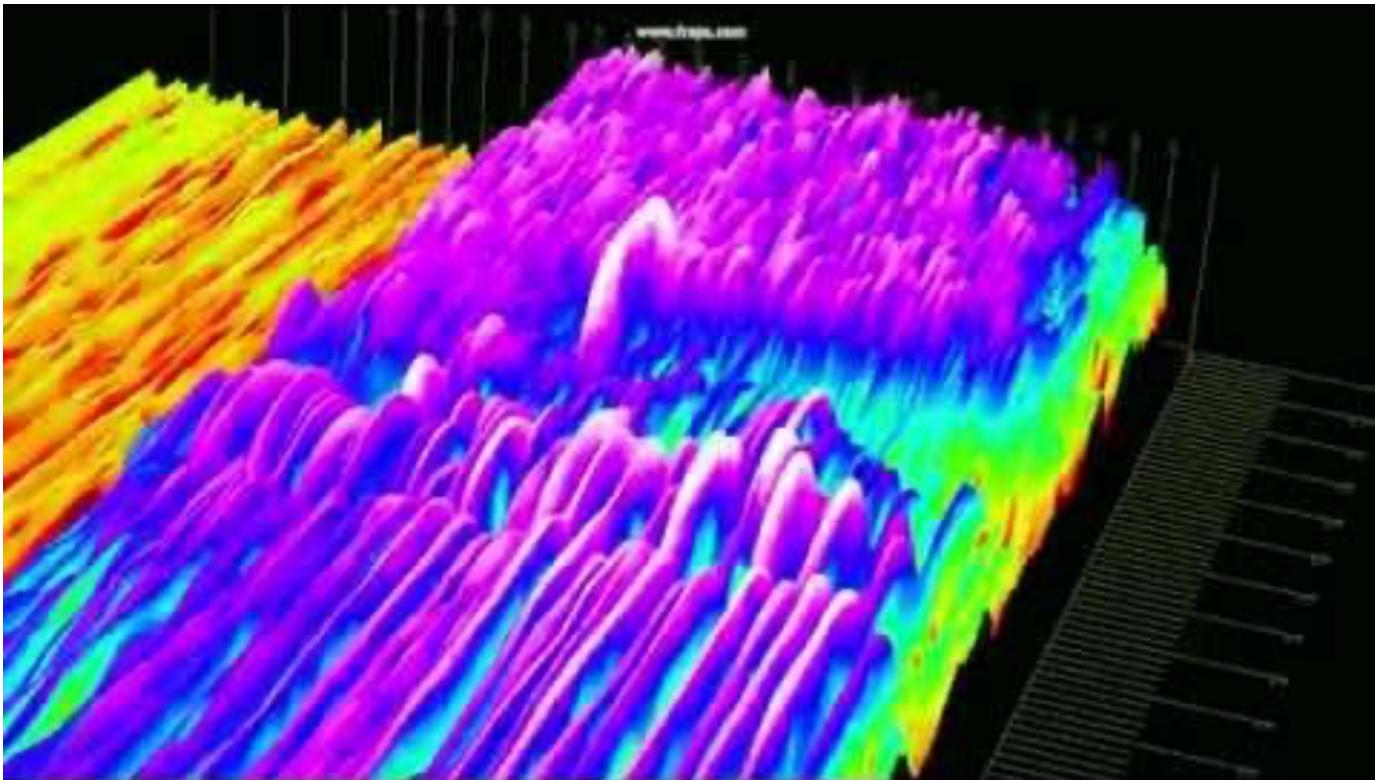
- A spectrogram shows the **frequency content (y-axis)** of an audio signal as it changes over time.
- In spectrogram, **magnitude** of the sound is shown by **color-coding**.



# Spectrogram: Example

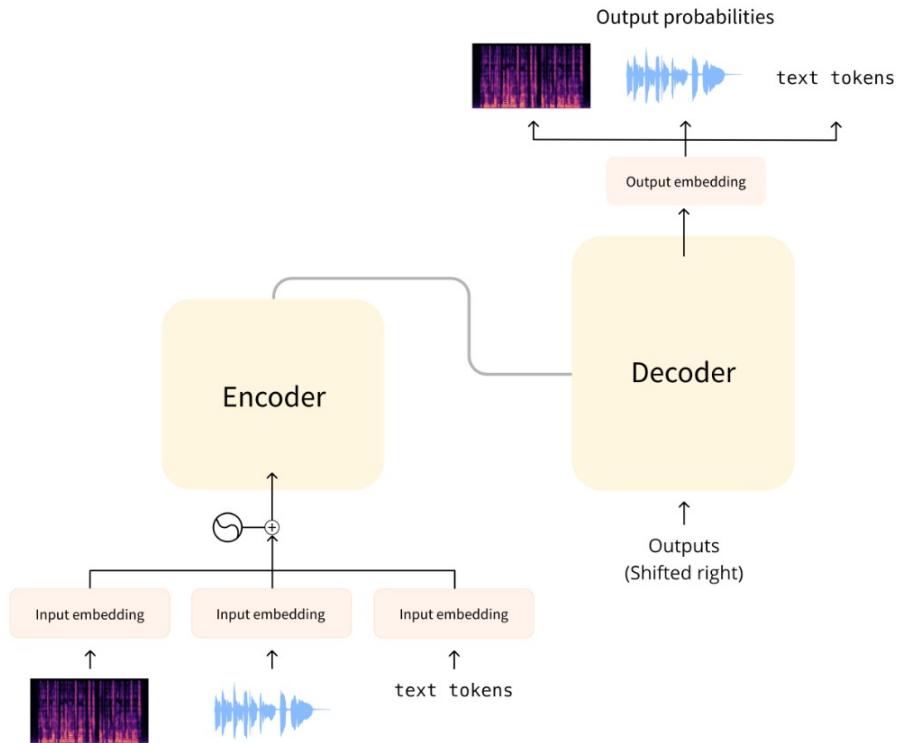


# Spectrogram: Example



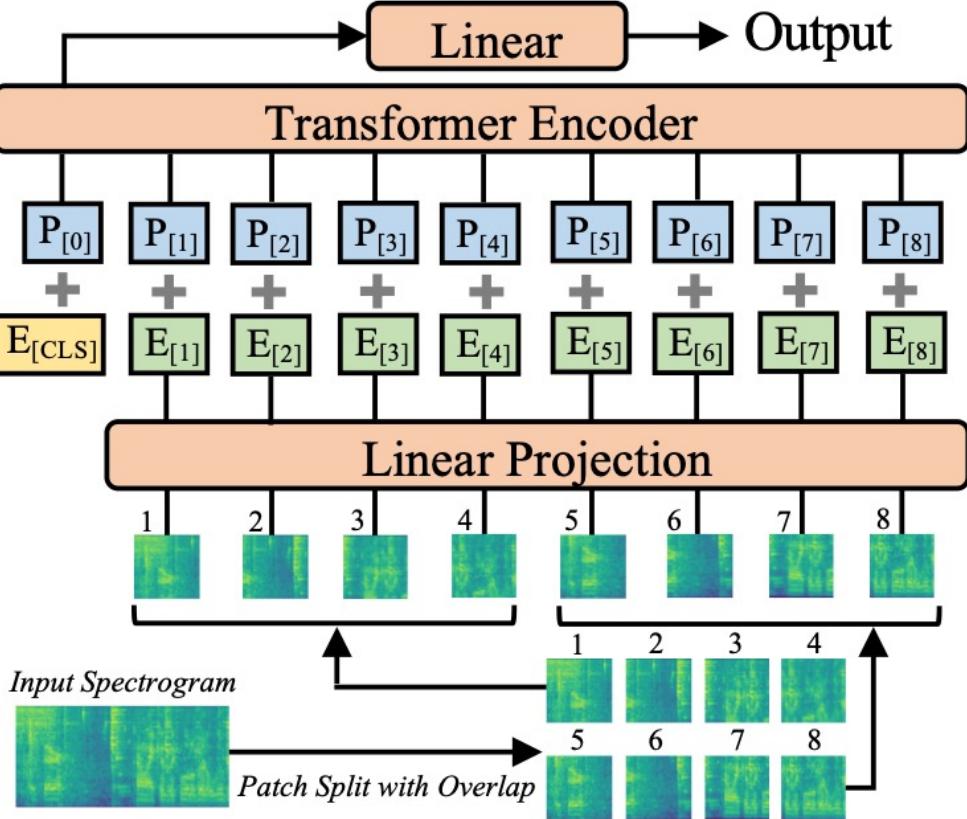
# Feeding Speech to Transformer

- Turn your data into a format that can be processed by Transformer.



# Audio Spectrogram Transformer (AST)

- Like ViT, AST splits the spectrogram into a sequence of partially overlapping  $16 \times 16$  images.
- This sequence of patches is then projected into a sequence of embeddings, and these are given to the transformer encoder.
- On top of this is a simple classification layer with sigmoid activation to map the hidden-states to classification probabilities.



# Birds of View of Speech Tasks

---

- The progress in speech/audio is not as mature as areas.
- There are various modality connections:
  - Speech/audio to text, Speech/audio to image, Text to speech/audio, Image to speech/audio, ...
- Recording audio is quite tricky:
  - Each microphone is different
  - Each room is different
  - Multiple audio sources (e.g., group of people)
  - ...
- There are tons of sounds in the world that are not easily captured.

# Summary

---

- Audio/speech via Transformers?
- Use the same old recipe: pre-process and feed it directly.
- How far are we from solving it? Quite far!!

# LMs and Code Generation

# How many people have used Github CoPilot?

---



GitHub  
Copilot

# Some examples

```
def solution(lst):
    """Given a non-empty list of integers, return the sum of all of the odd elements
    that are in even positions.

    Examples
    solution([5, 8, 7, 1]) ==>12
    solution([3, 3, 3, 3, 3]) ==>9
    solution([30, 13, 24, 321]) ==>0
    """
    return sum(lst[i] for i in range(0,len(lst)) if i % 2 == 0 and lst[i] % 2 == 1)
```

```
def encode_cyclic(s: str):
    """
    returns encoded string by cycling groups of three characters.
    """
    # split string to groups. Each of length 3.
    groups = [s[(3 * i):min((3 * i + 3), len(s))] for i in range((len(s) + 2) // 3)]
    # cycle elements in each group. Unless group has fewer elements than 3.
    groups = [(group[1:] + group[0]) if len(group) == 3 else group for group in groups]
    return "".join(groups)

def decode_cyclic(s: str):
    """
    takes as input string encoded with encode_cyclic function. Returns decoded string.
    """
    # split string to groups. Each of length 3.
    groups = [s[(3 * i):min((3 * i + 3), len(s))] for i in range((len(s) + 2) // 3)]
    # cycle elements in each group.
    groups = [(group[-1] + group[:-1]) if len(group) == 3 else group for group in groups]
    return "".join(groups)
```

# General Idea

---

The story is the same as what we have seen before! 😊

- 1. Pre-train on web-scale text/code data (what we have seen before)
- 2. More pre-training on a lot of [potentially, noisy] code data (**new**)
- 3. Fine-tune on smaller but cleaner data (**new**)

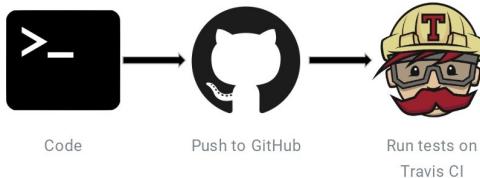
# Setup: Pre-train on Code Data

---

- Start with GPT-3 that is already pre-trained to on text and code.
- **Codex:** Starting from GPT-3, do continued training on code data.
  - Training data: 160GB of Python code (unlabeled!!)
- The goal is to evaluate model for:
  - Generating code from doc-string (implementing an idea)
  - Generating doc-string from code (explaining code)
- Why should this work?
  - Human written code often comes with comments (i.e., free supervision)!! 😍

# Fine-tuning on Clean[er] Data

- Competitive Programming (10,000 problems)
    - Problem descriptions as docstrings
    - These have unit test coverage
  - Continuous Integration (40,000 functions)
    - “Developers regularly merge code changes into a central repository, after which automated builds and tests are run.”
    - These come with free test functions.
- } They also do additional filtering.  
Filter out:
  - Low-quality docstring
  - Stateful functions that may be missing variables outside them.
- Approach:
  - Use Codex-12B to generate 100 samples per problem, discard the problem if no generation passes.



# Evaluation

---

- 164 hand-written problems
  - Hand-written to avoid overlap with the training data ("training data leakage")
    - Although in practice it's hard to control for this.
  - Evaluates language comprehension, reasoning, algorithms and simple math
- "Check if two words have the same characters."
- "Return median of elements in the list l."
- "sum\_to\_n is a function that sums numbers from 1 to n."
- "Given a non-empty list of integers lst. add the even elements that are at odd indices."
- "Return true if a given number is prime, and false otherwise."
- "Return n-th Fibonacci number."

# Evaluation

- Format:
  - function signature
  - docstring with examples
  - unit-tests

```
def vowels_count(s):
    """Write a function vowels_count which takes a
       string representing
       a word as input and returns the number of vowels in
       the string.
       Vowels in this case are 'a', 'e', 'i', 'o', 'u'.
       Here, 'y' is also a
       vowel, but only when it is at the end of the given
       word.
```

Example:

```
>>> vowels_count("abcde")
2
>>> vowels_count("ACEDY")
3
"""
```

```
def check(candidate):
    # Check some simple cases
    assert candidate("abcde") == 2, "Test 1"
    assert candidate("Alone") == 3, "Test 2"
    assert candidate("key") == 2, "Test 3"
    assert candidate("bye") == 1, "Test 4"
    assert candidate("keY") == 2, "Test 5"
    assert candidate("bYe") == 1, "Test 6"
    assert candidate("ACEDY") == 3, "Test 7"
    # Check some edge cases that are easy to work out by
    # hand.
    assert True, "This prints if this assert fails
2 (also good for debugging!)"
```

# Evaluation

- The traditional NLP metrics are not necessarily good for code generation.

Reference code

```
def f(a, b):
    c = a - b
    return c
```

Equivalent code

```
def f(a, b):
    summ = 0
    summ += a
    summ -= b
    return summ
```

BLEU = 66

Non-equivalent code

```
def f(a, b):
    c = a + b
    return c
```

BLEU = 81

- Instead, they measure functional correctness.
  - Whether the generated code (ignoring its details) passes all unit tests
  - This is the way humans evaluate correctness of the code rather than its content.

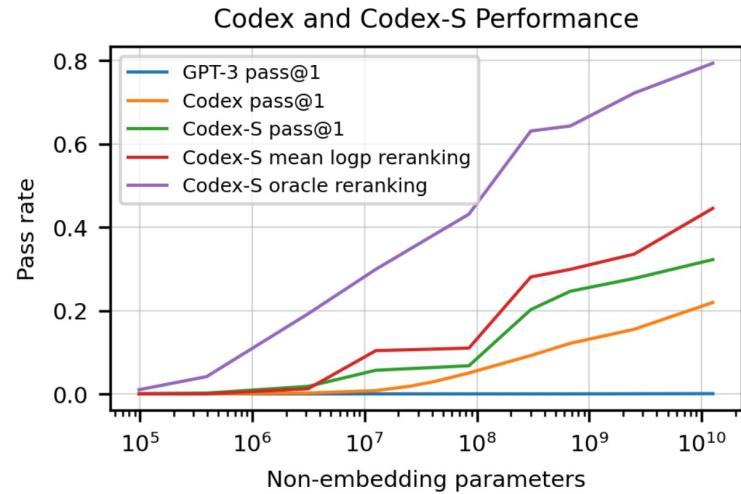
# Evaluation

---

- Given a prompt, generate k samples
  - For each, sample tokens until a stop sequence is encountered
- **pass@k:** the probability that **at least one** of the top k-generated code samples for a problem passes the unit tests.
  - How do you compute this?
    - Think about it! ;-)

# Results

- Scaling improves the results
- Just pre-training code (Codex) gives a major boost.
- Reranking heuristics:  
Generate 100 samples per problem and selecting the sample with the highest mean log-probability or by selecting the sample that passes the unit tests (oracle).



# Results: w/ Open-source Models

- Existing open-source models (GPT-J and GPT-Neo) know a surprising amount about code generation!!
- Reason: The Pile (used for GPT-J) contains a lot of code: 8% GitHub code, along with natural language data

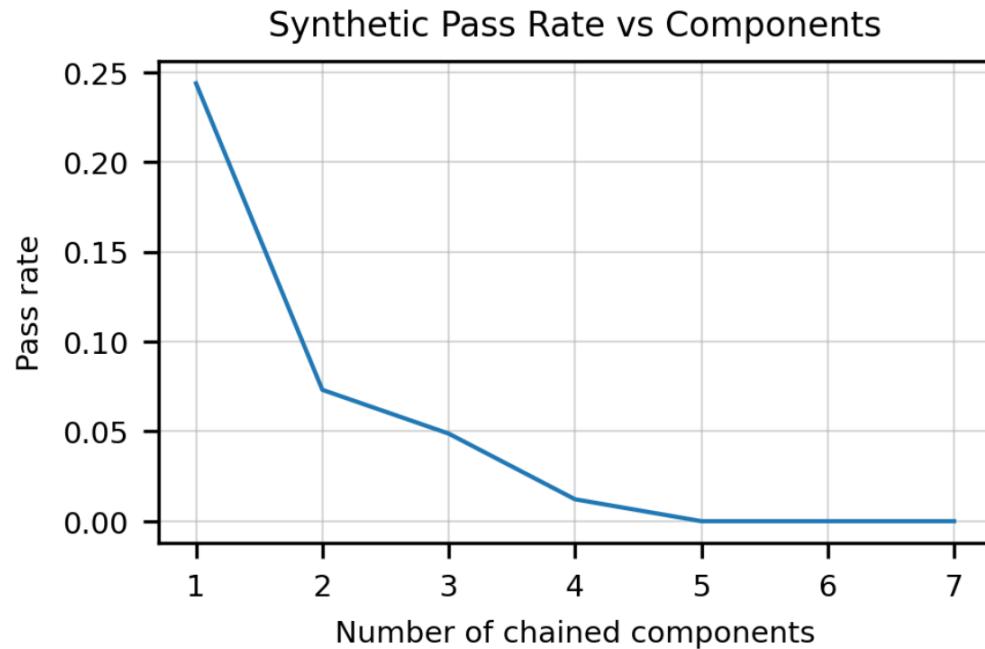
*Table 1.* Codex, GPT-Neo, & TabNine evaluations for HumanEval. We find that GPT-J pass@1 is between Codex-85M and Codex-300M performance.

	PASS @ $k$		
	$k = 1$	$k = 10$	$k = 100$
GPT-NEO 125M	0.75%	1.88%	2.97%
GPT-NEO 1.3B	4.79%	7.47%	16.30%
GPT-NEO 2.7B	6.41%	11.27%	21.37%
GPT-J 6B	11.62%	15.74%	27.74%
TABNINE	2.58%	4.35%	7.59%
CODEX-12M	2.00%	3.62%	8.58%
CODEX-25M	3.21%	7.1%	12.89%
CODEX-42M	5.06%	8.8%	15.55%
CODEX-85M	8.22%	12.81%	22.4%
CODEX-300M	13.17%	20.37%	36.27%
CODEX-679M	16.22%	25.7%	40.95%
CODEX-2.5B	21.36%	35.42%	59.5%
CODEX-12B	28.81%	46.81%	72.31%

# Result: Degradation with Length

---

- The longer the inputs instructions are, the lower the model performance is.



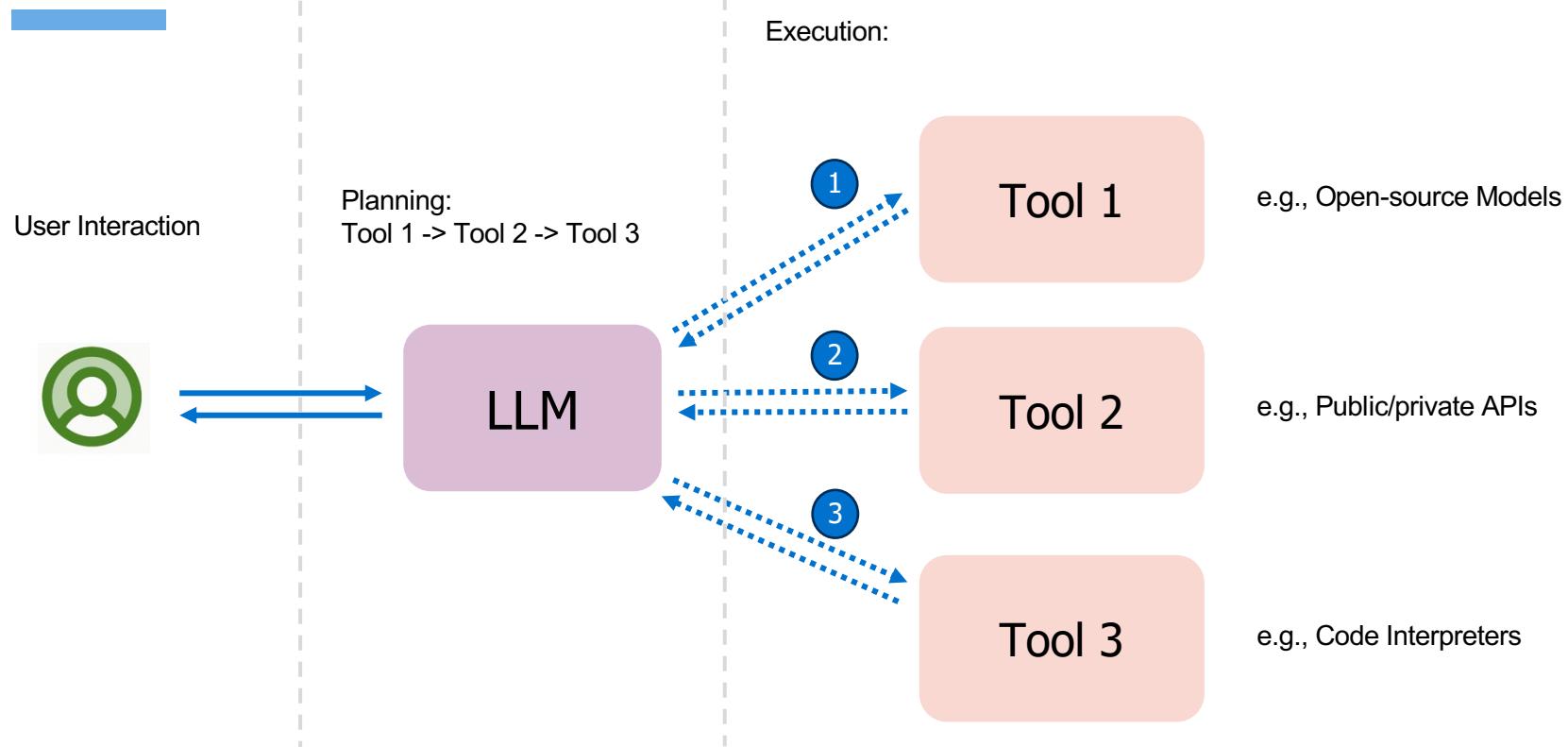
# Summary

---

- Our focus: building a bridge between natural language and code.
- Turns out the ideas in prior chapters go a long way!
- The importance of this is more than just increasing the productivity of programmers!
- In the next part, we will use this bridge to enable LMs speak to do various things!

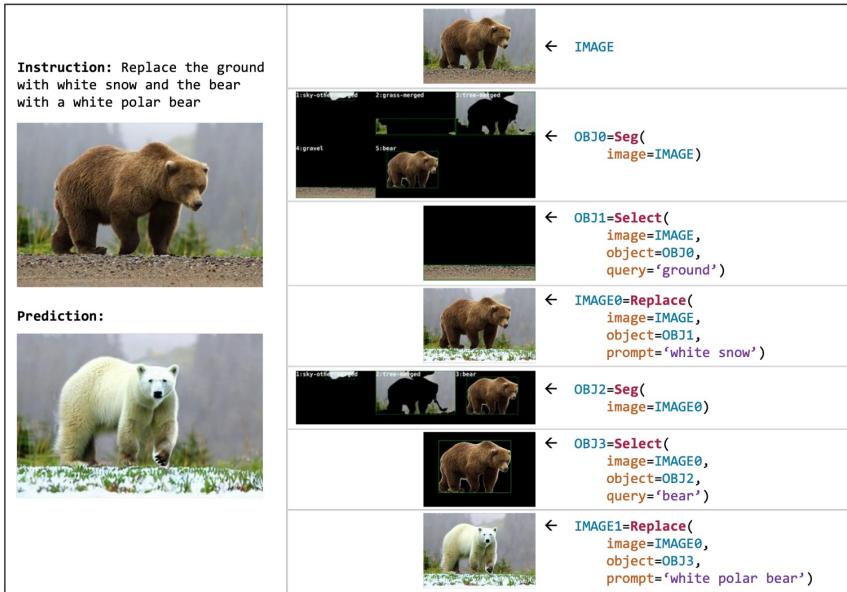
# Connecting Language to Actions/Tools

# Tool-Augmented LLMs

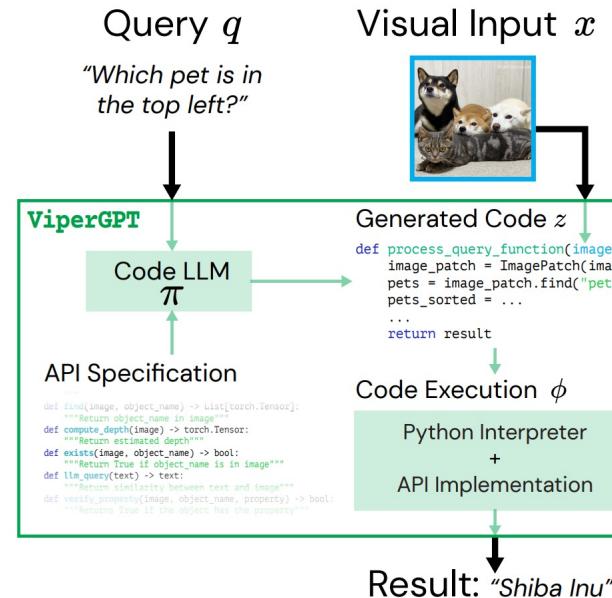


# Planning: Composing Tools via Code Generation

Visual Programming: Compositional visual reasoning without training



ViperGPT: Visual Inference via Python Execution for Reasoning



# MM-ReAct: Capabilities

## Visual Math and Text Reasoning

What is "a" in the first equation?  
What is y in the last equation?

$5 + a = 7$	$40 + y = 45$
$10 + y = 15$	$39 + a = 41$
$20 + a = 22$	$24 + y = 29$
$48 + a = 50$	$15 + a = 17$
$10 + a = 12$	$34 + y = 39$

The value of "a" in the first equation " $5 + a = 7$ " is 2.

The value of "y" in the last equation " $34 + y = 39$ " is 5.



What is funny about it?



The humor in the comic may come from the situation, the characters, or the dialogue. The person is feeling unwell and is asked if they are pregnant or have the coronavirus, which is a current and relevant topic. The humor may come from the absurdity of the situation or the unexpectedness of the question.

## Spatial/Coordinate Understanding

Where is the frisbee?  
What is the relative position of the frisbee and the man?



The frisbee is located at coordinates (63,119,133,177).



The frisbee is located above the man in the image. The man is falling on the ground while the frisbee is in the air. The frisbee is also closer to the top of the image than the man..

## Visual Planning and Prediction

What should I do after kneading the dough, before dividing it into pieces? For how long?



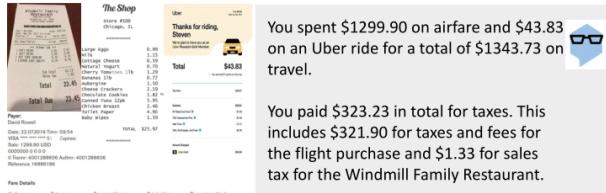
After kneading the dough, you should put it in a warm place before dividing it into pieces, according to the recipe in the image.

You should put the dough in a warm place for 1 hour.



## Multi-Image Reasoning

How much in total did I spend on travel?  
How much in total did I pay for taxes?

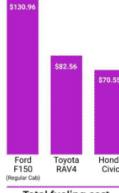


You spent \$1299.90 on airfare and \$43.83 on an Uber ride for a total of \$1343.73 on travel.

You paid \$323.23 in total for taxes. This includes \$321.90 for taxes and fees for the flight purchase and \$1.33 for sales tax for the Windmill Family Restaurant.

## Multi-Hop Document Understanding

Excluding the regular cab, what is the average total fueling cost?



The average total fueling cost excluding the Ford F150 (Regular Cab) is \$76.55. This is calculated by taking the sum of the total fueling cost of Toyota RAV4 and Honda Civic and dividing it by 2. ( $\$82.56 + \$70.55$ ) / 2 =  $\$76.55$ .



## Open-World Concept Understanding

What is the brand of these drink? Which one is unsweetened?



The brand of these drinks is "BETTER THAN MILK". The unsweetened one is "BETTER THAN MILK organic OAT DRINK".



Who are the celebrities in this image?



Keira Knightley and Natalie Portman are the celebrities in this image.



Is it safe to eat this mushroom?



Morel mushrooms are generally considered safe to eat when properly cooked.

## Video Summarization/Event Localization

Can you breakdown this video tutorial to step-by-step instructions, with start and end time?



Here's a step-by-step breakdown of the tutorial with start and end time.

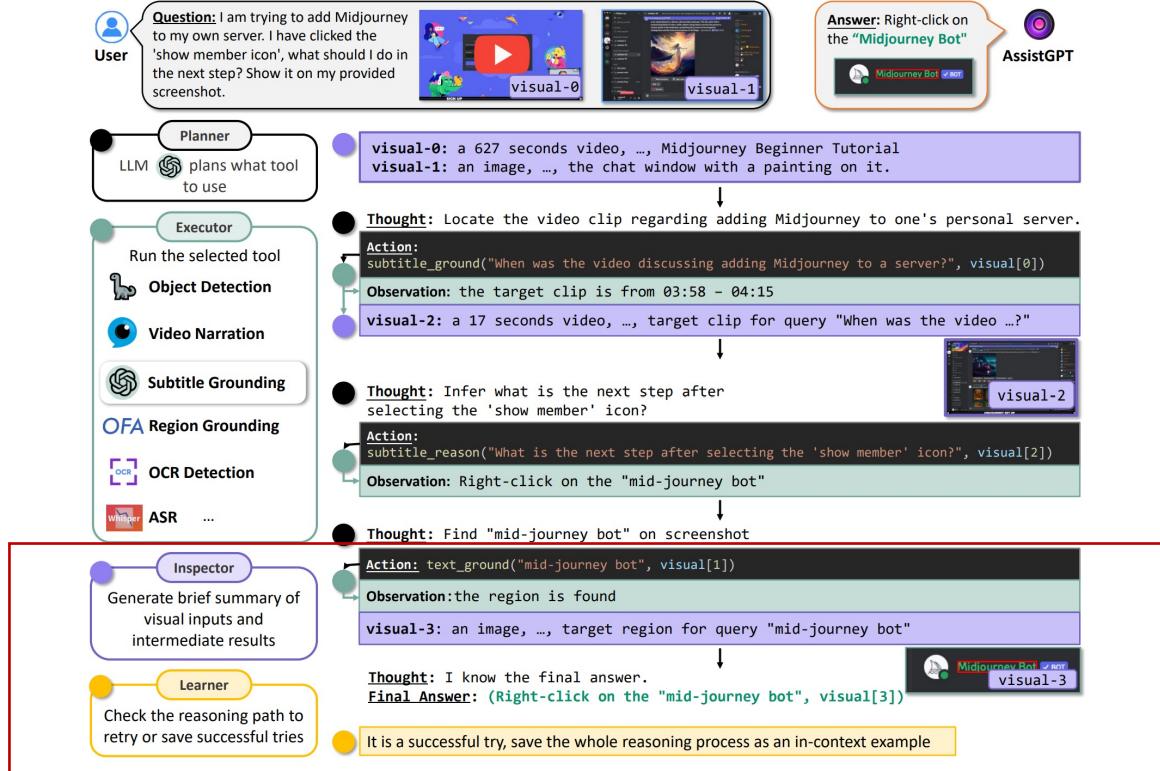


... Add Lettuce    Add Tomato    Add Bacon ...

- TBD: expand on ReACT?

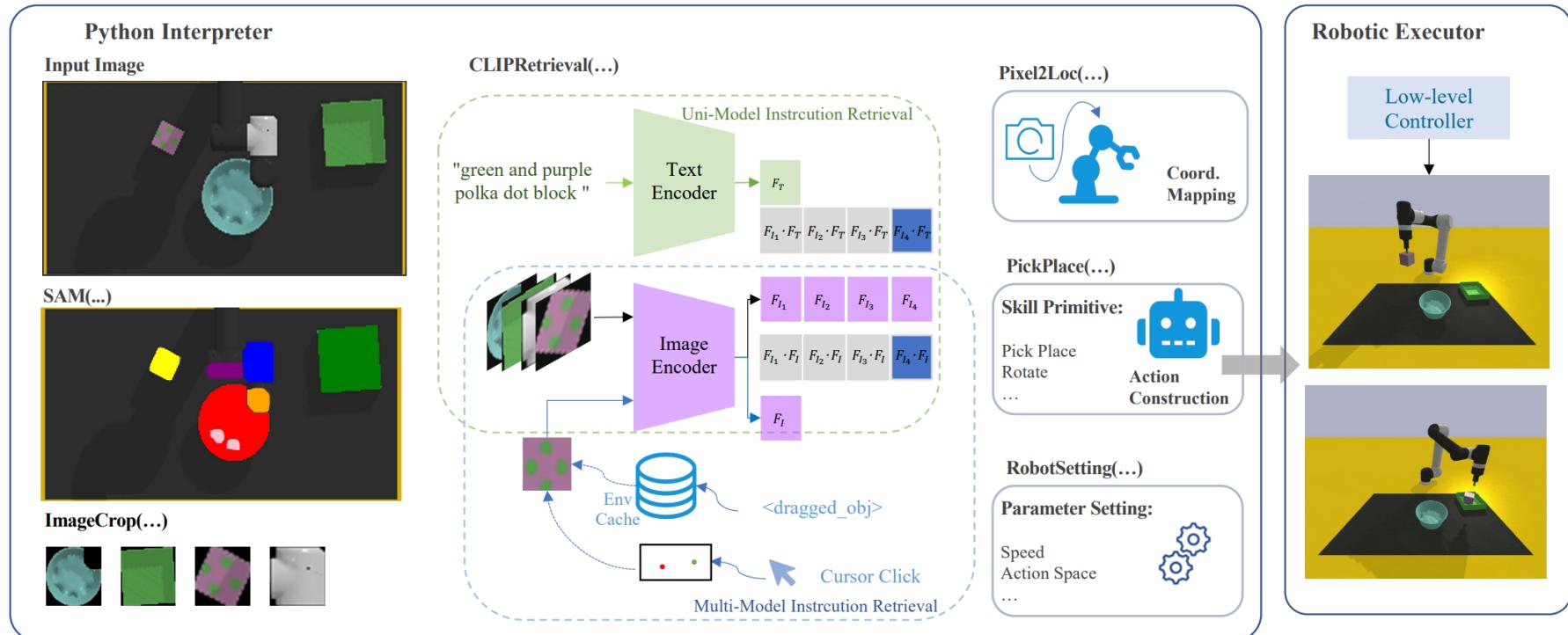
# Tool Using with Self-Assessment

AssistGPT: A General Multi-modal Assistant that can Plan, Execute, Inspect, and Learn



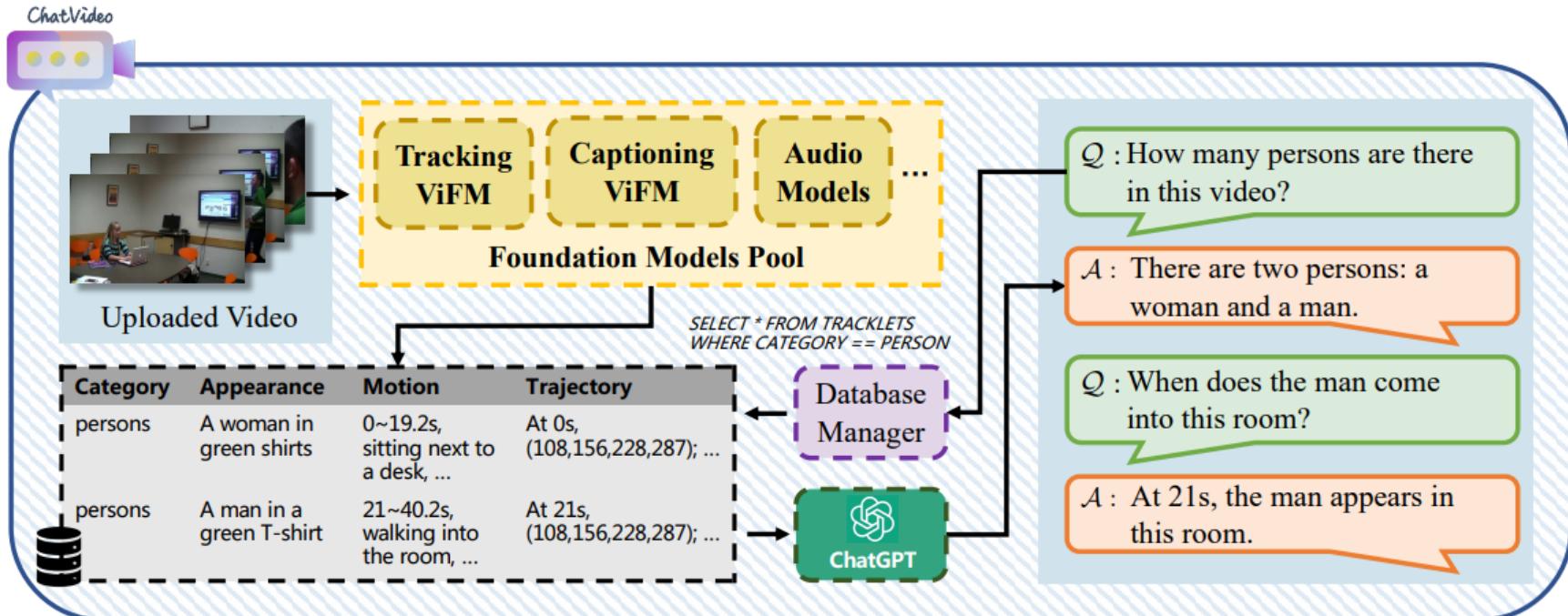
# Applications : Robotics

Instruct2Act: Mapping Multi-modality Instructions to Robotic Actions with Large Language Model



# Applications: Multi-Channel Video Understanding

ChatVideo: A Tracklet-centric Multimodal and Versatile Video Understanding System



# SayCan: Grounding Language in Affordances

We need to ground the language model in tasks that are feasible within a specific real-world context.

I spilled my drink, can you help?

GPT3

You could try using a vacuum cleaner.

LaMDA

Do you want me to find a cleaner?

FLAN

I'm sorry, I didn't mean to spill it.

I spilled my drink, can you help?

LLM

"find a cleaner"  
"find a sponge"  
"go to the trash can"  
"pick up the sponge"  
"try using the vacuum"

Value Functions

"find a cleaner"  
"find a sponge"  
"go to the trash can"  
"pick up the sponge"  
"try using the vacuum"

SayCan

"find a sponge"  
"go to the trash can"  
"pick up the sponge"  
"try using the vacuum"



I would:

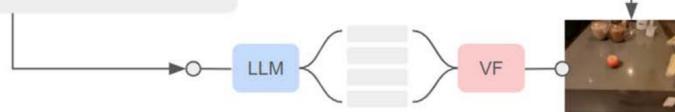
1. find a sponge
2. pick up the sponge
3. come to you
4. put down the sponge
5. done

# SayCan: Grounding Language in Affordances



Note that the LM scores options instead of doing generation

*I would: 1. Find an apple, 2. \_\_\_\_*



# SayCan + Chain-of-Thought

---

## **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.

# PaLM-E: Embodied Multimodal LM

Encoding embodied observations as language tokens.

## Mobile Manipulation



Human: Bring me the rice chips from the drawer. Robot: 1. Go to the drawers, 2. Open top drawer. I see . 3. Pick the green rice chip bag from the drawer and place it on the counter.

## Visual Q&A, Captioning ...



Given . Q: What's in the image? Answer in emojis.  
A: 🍏🍌🍇🍐🍊🍒.

## PaLM-E: An Embodied Multimodal Language Model

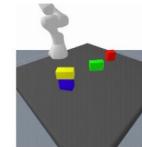
Given `<emb>` ... `<img>` Q: How to grasp blue block? A: First, grasp yellow block

Large Language Model (PaLM)

Control

A: First, grasp yellow block and ...

## Task and Motion Planning



Given `<emb>` Q: How to grasp blue block?  
A: First grasp yellow block and place it on the table, then grasp the blue block.

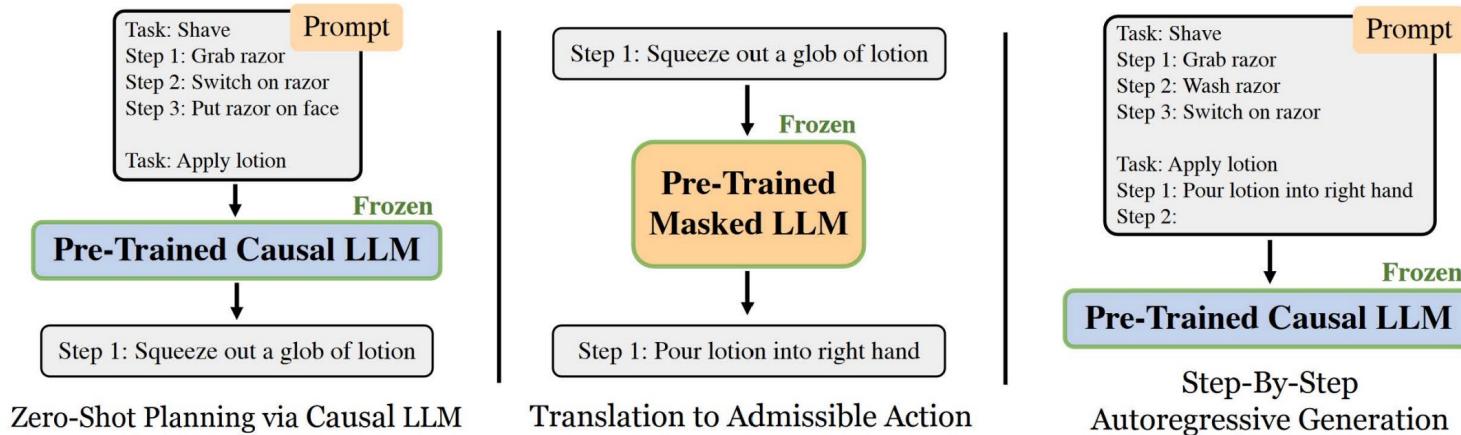
## Tabletop Manipulation



Given `<img>` Task: Sort colors into corners.  
Step 1. Push the green star to the bottom left.  
Step 2. Push the green circle to the green star.

# LMs as Planners

- Key Idea: Large language models learn rich procedural knowledge and such knowledge could be extracted.



# LMs as Planners

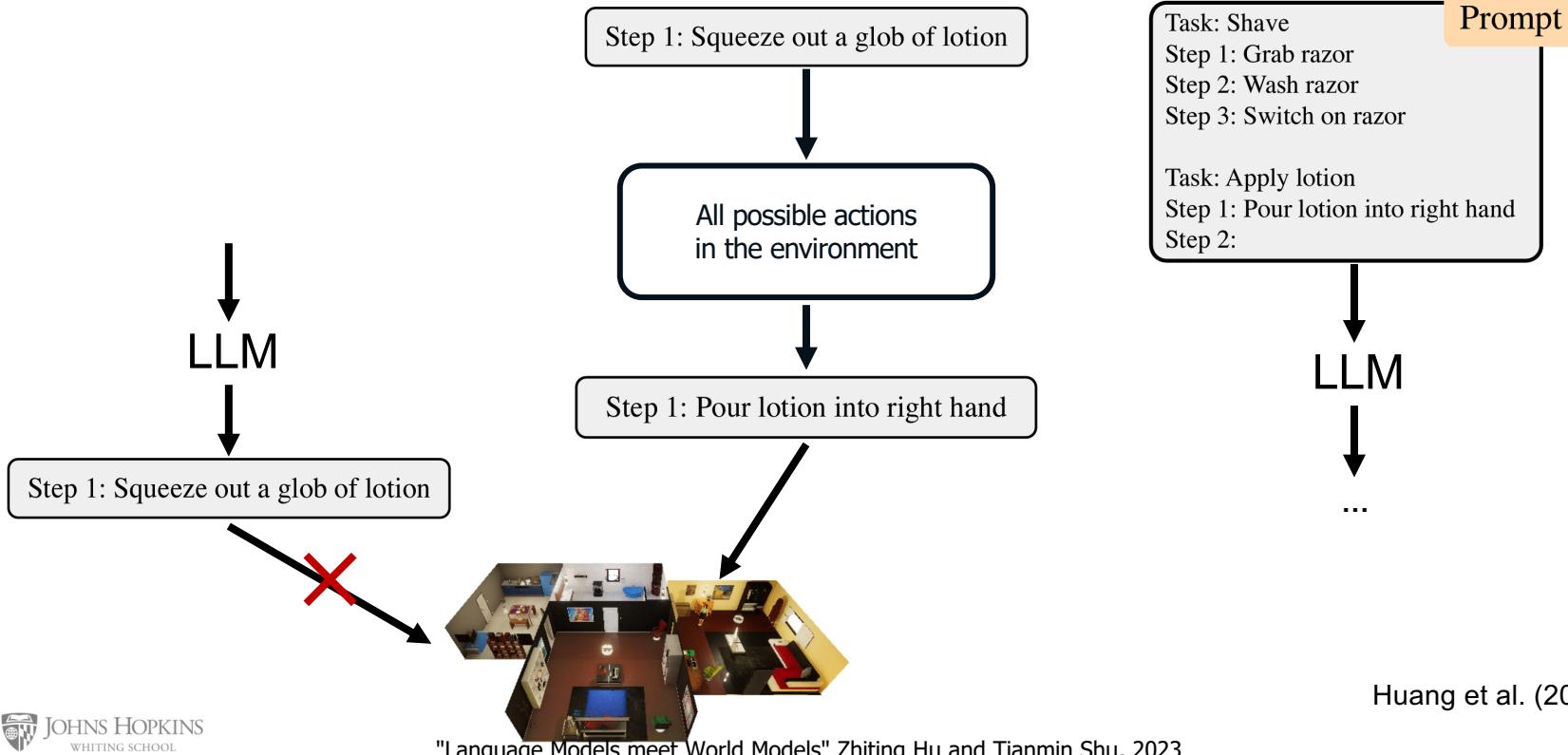
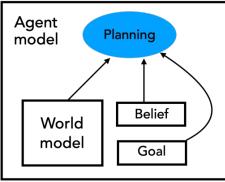
- Impressive results.
- Challenge: verification; groundability to real-world videos.

Human	GPT-3 175B	Translated GPT-3 175B	Task: Complete Amazon Turk Surveys	Task: Get Glass of Milk
<b>Task: Throw away paper</b> Step 1: Walk to home office Step 2: Walk to desk Step 3: Find desk Step 4: Turn to desk Step 5: Find chair Step 6: Sit on chair Step 7: Find check Step 8: Grab check Step 9: Squeeze check Step 10: Stand up Step 11: Walk to trashcan Step 12: Put check on trashcan	<b>Task: Throw away paper</b> Step 1: Walk to home office Step 2: Walk to wastebasket Step 3: Find wastebasket Step 4: Grab wastebasket Step 5: Walk to desk Step 6: Drop paper in wastebasket	<b>Task: Throw away paper</b> Step 1: Walk to home office Step 2: Walk to table Step 3: Find table Step 4: Turn to table Step 5: Find paper Step 6: Grab paper Step 7: Walk to trashcan Step 8: Open trashcan Step 9: Put paper on trashcan Step 10: Close trashcan	<p>Walk to Home Office      Sit on Chair Switch on Computer      Look at Computer</p>	<p>Walk to Kitchen      Open Fridge Grab Milk      Close Fridge</p>
<b>Task: Brush teeth</b> Step 1: Go to bathroom	<b>Task: Brush teeth</b> Step 1: Walk to bathroom Step 2: Walk to sink Step 3: Find toothbrush Step 4: Pick up toothbrush Step 5: Put toothbrush in mouth Step 6: Move brush around mouth for two minutes Step 7: Spit out toothpaste and brush into sink Step 8: Turn on water in sink and rinse brush for one minute Step 9: Turn off water in sink and return brush to cupboard	<b>Task: Brush teeth</b> Step 1: Walk to bathroom Step 2: Open door Step 3: Walk to sink Step 4: Put pot on sink Step 5: Put brush on toothbrush Step 6: Turn to toothpaste Step 7: Put toothpaste on toothbrush Step 8: Put teeth on toothbrush		



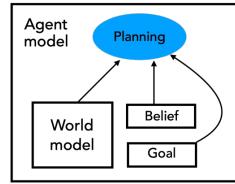
# Planning with Environment Feedback

# Language models as (autoregressive) planners



Huang et al. (2022)

# Language models as (autoregressive) planners



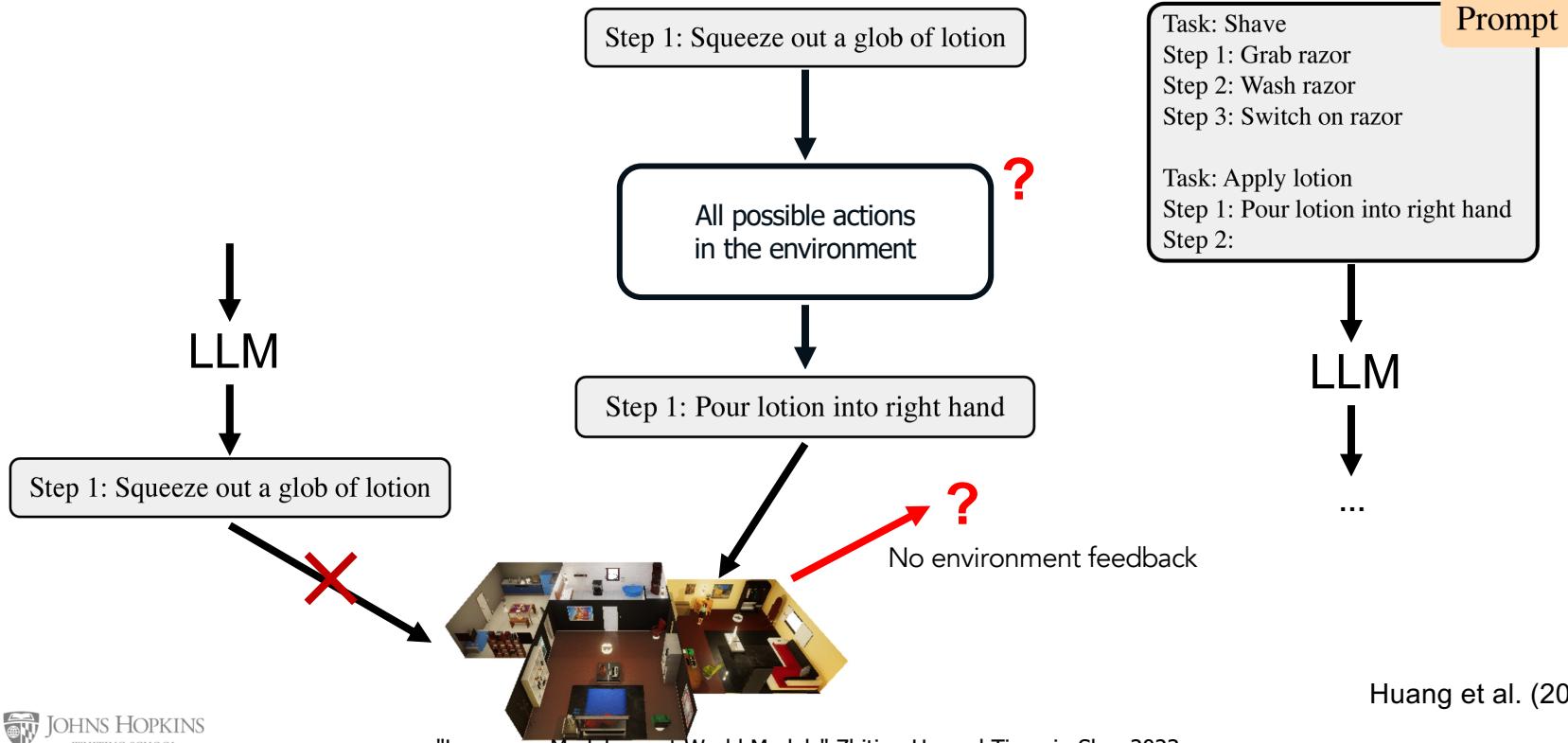
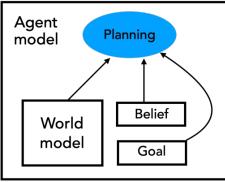
Empty Dishwasher



Organize Closet

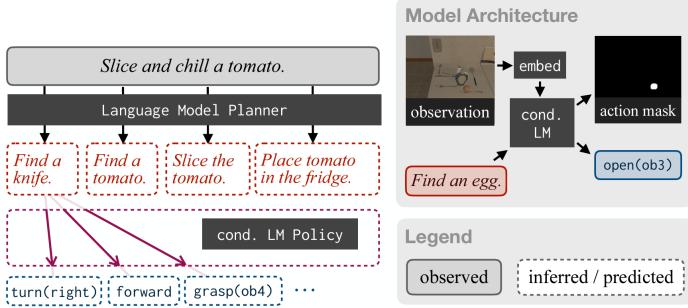
Huang et al. (2022)

# Language models as (autoregressive) planners

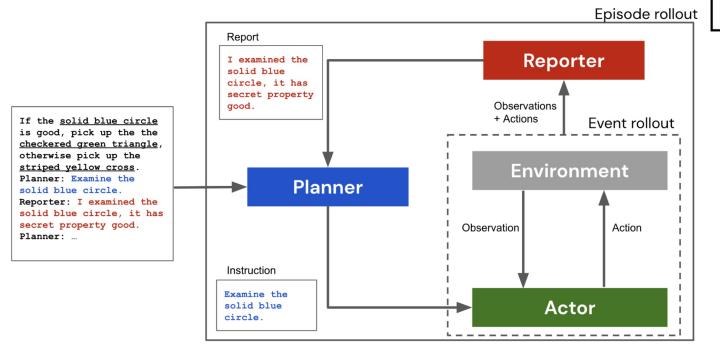


Huang et al. (2022)

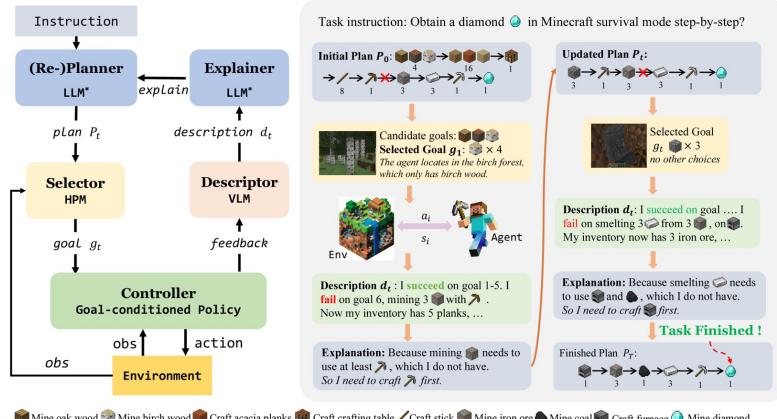
# Interactive planner



Sharma et al. (2021)

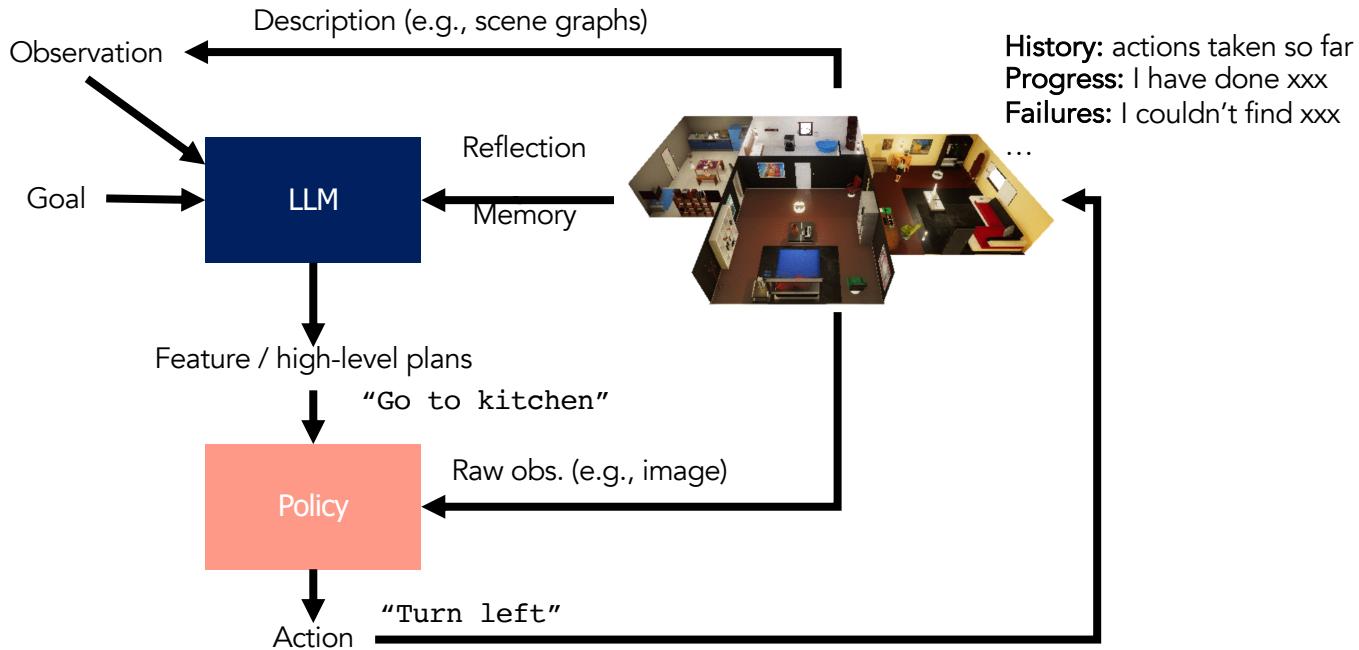
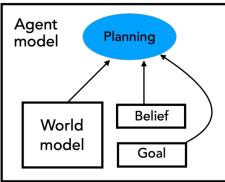


Dasgupta et al. (2022)



Wang et al. (2023)

# Interactive planner



# Sustained Embodied Reasoning in Rich Environments

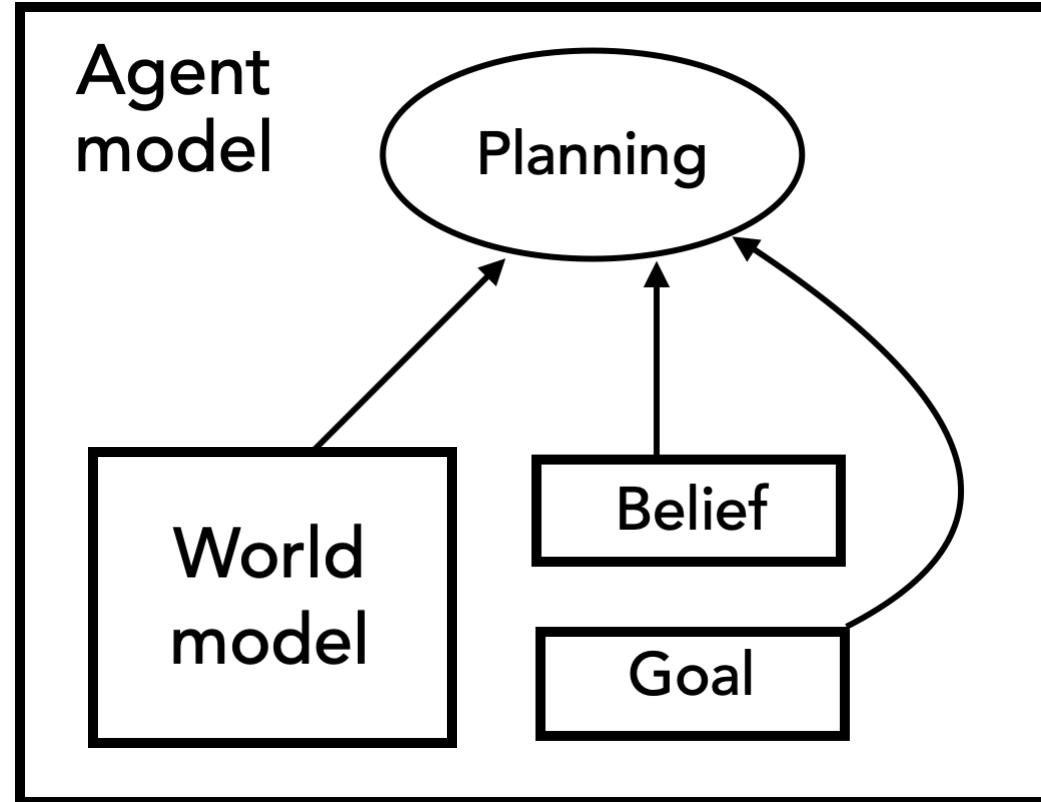
# Limits in LLMs

---



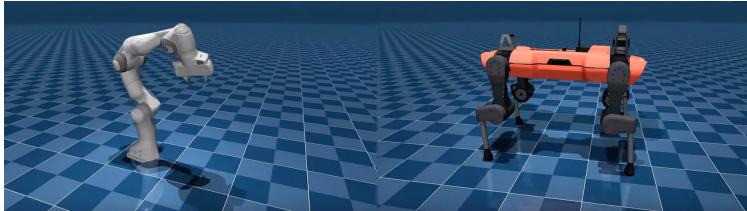
Does this person need help?

# World models and agent models



# Simulators as world models

- Physics engines / embodied simulators  
MuJoCo



Todorov et al. (2012)

AI2-THOR



Kolve et al. (2017)

ThreeDWorld



iGibson 2.0



Habitat 2.0



Gan et al. (2020)

Li et al. (2021)

Szot et al. (2021)

# Simulators as world models

- Embodied simulators + synthetic humans

VirtualHome 2.0



Puig et al. (2021)

Habitat 3.0



Puig et al. (2023)

# Video prediction for robot planning

Simulating long sequence of robot executions.

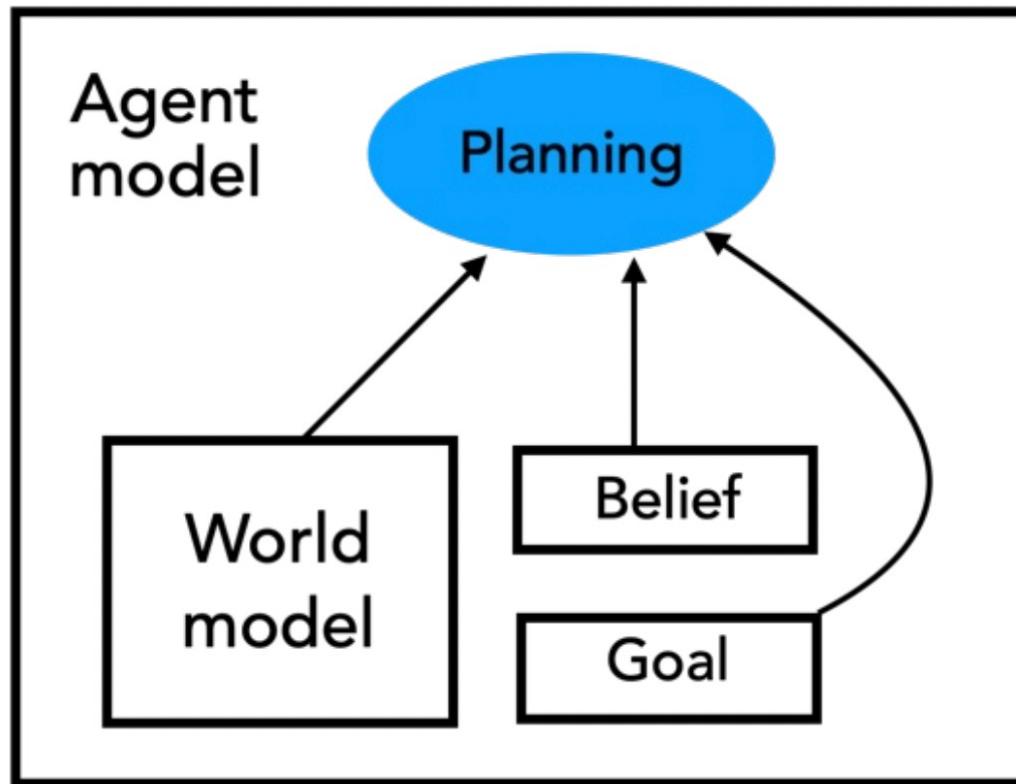
Step 1:



Yang et al. (2023)

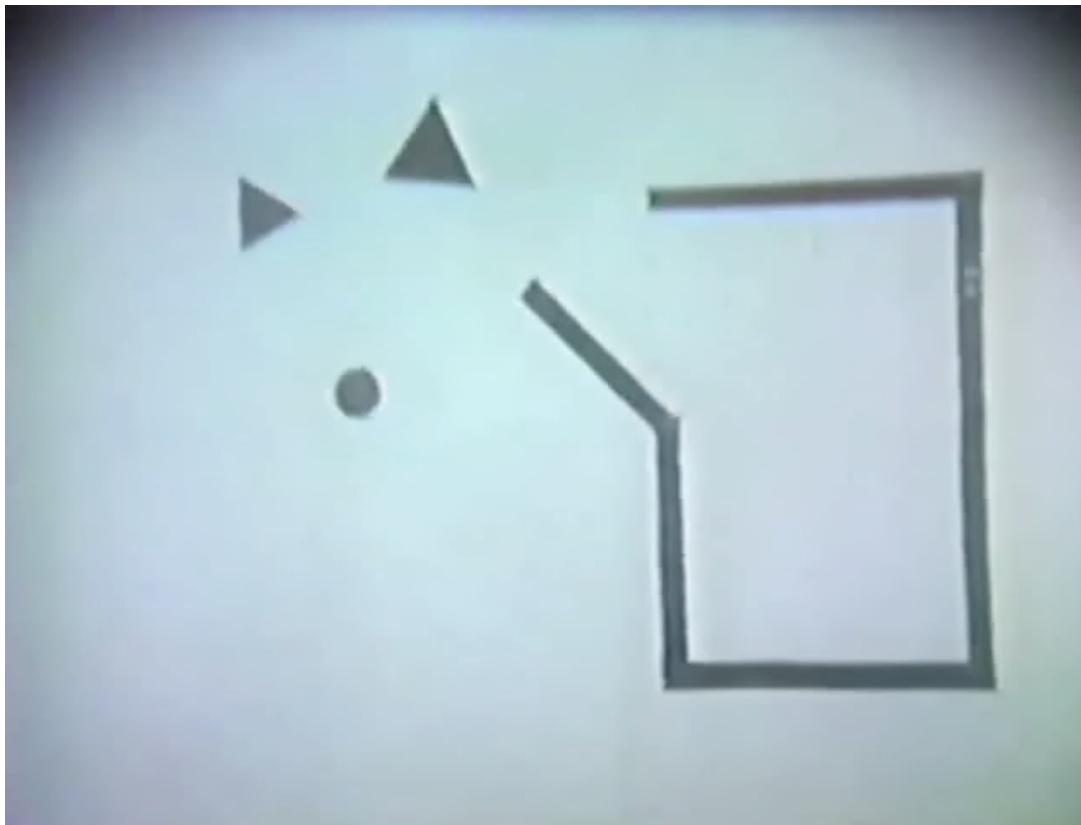
"Language Models meet World Models" Zhiting Hu and Tianmin Shu, 2023

# Language models as (autoregressive) planners



# Humans represent agents differently from objects

---



Heider & Simmel (1944)

# Humans represent agents differently from objects

## Strengths

strong, weak

## Goals

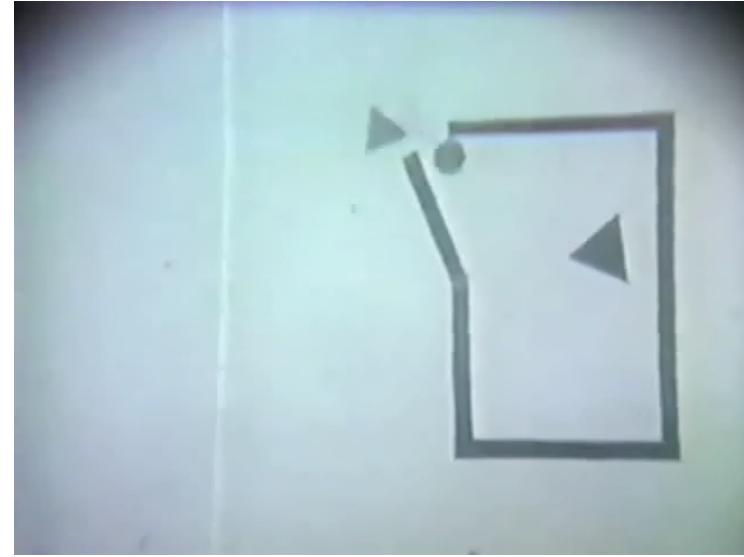
helping, hurting, escaping

## Relationships

friends, enemies

## Moral judgment

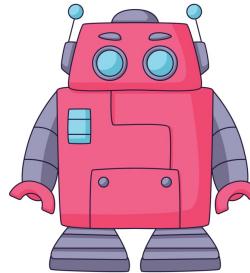
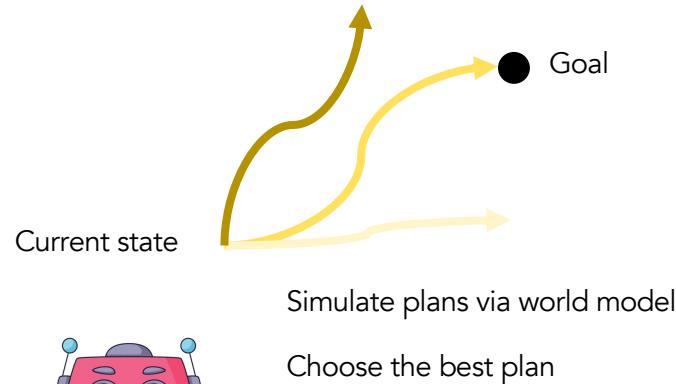
good guy, bully



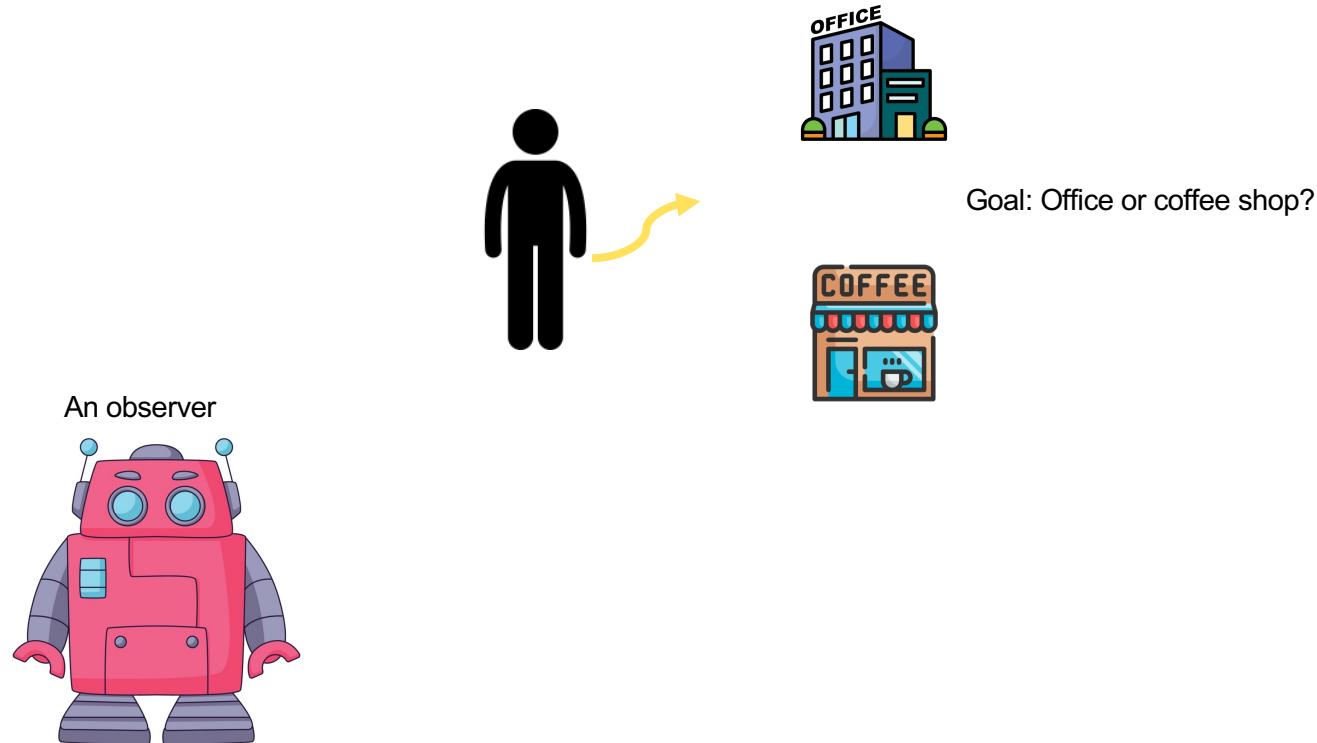
(size / velocity / angle...)

A big triangle moves back and forth, while a small triangle and a small circle rotate 360°...

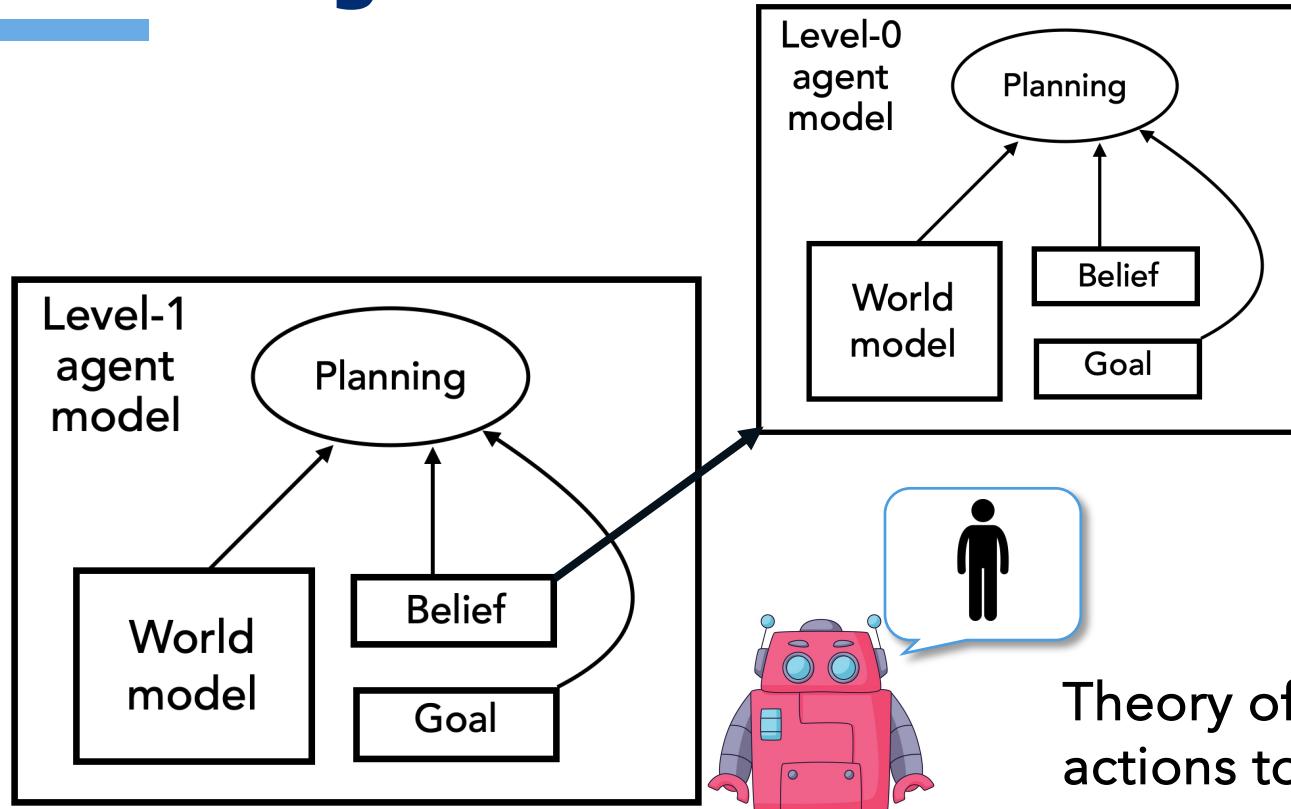
# Level-0 agent models for embodied tasks



# Level-1 agent models for social reasoning

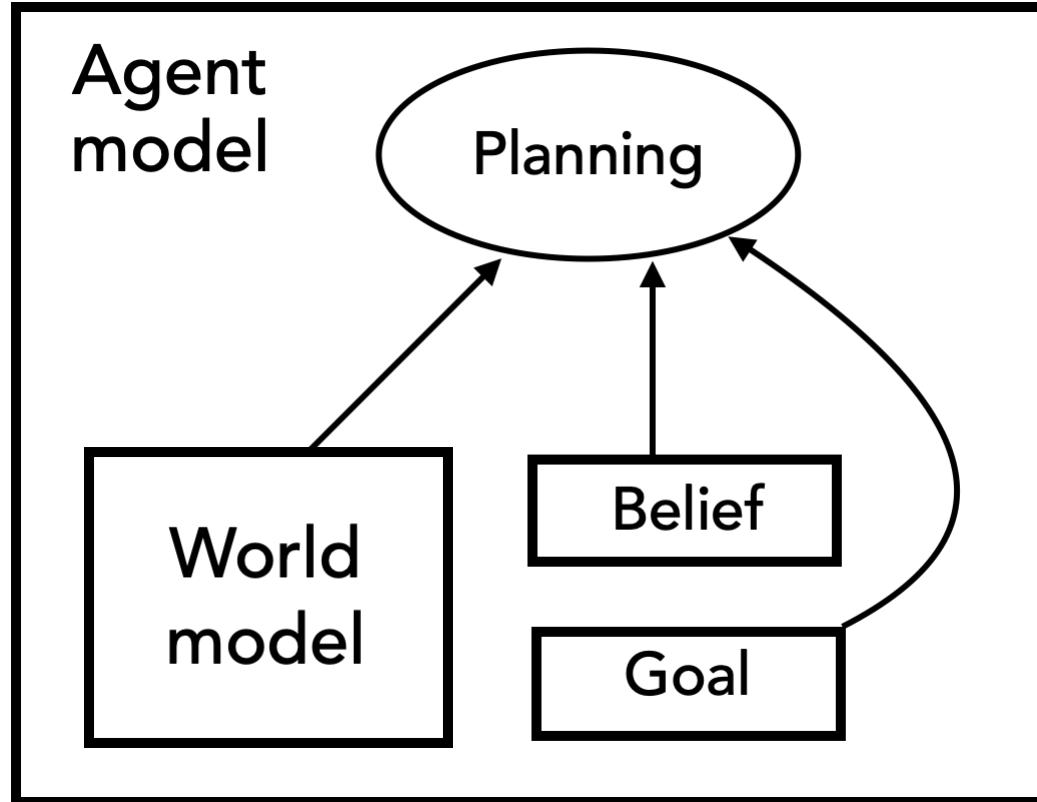


# Level-1 agent models for social reasoning

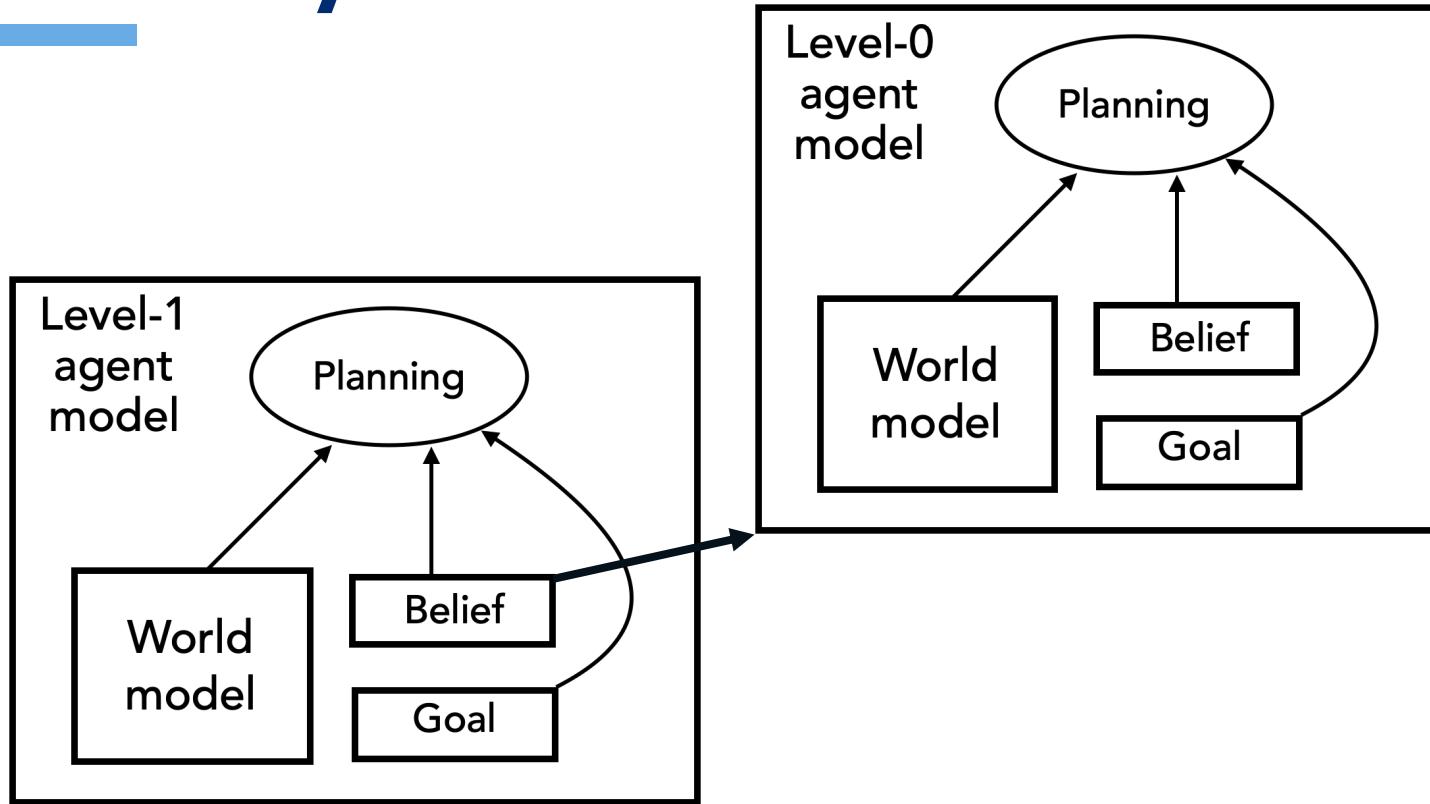


Theory of Mind: mapping actions to mental state

# Summary



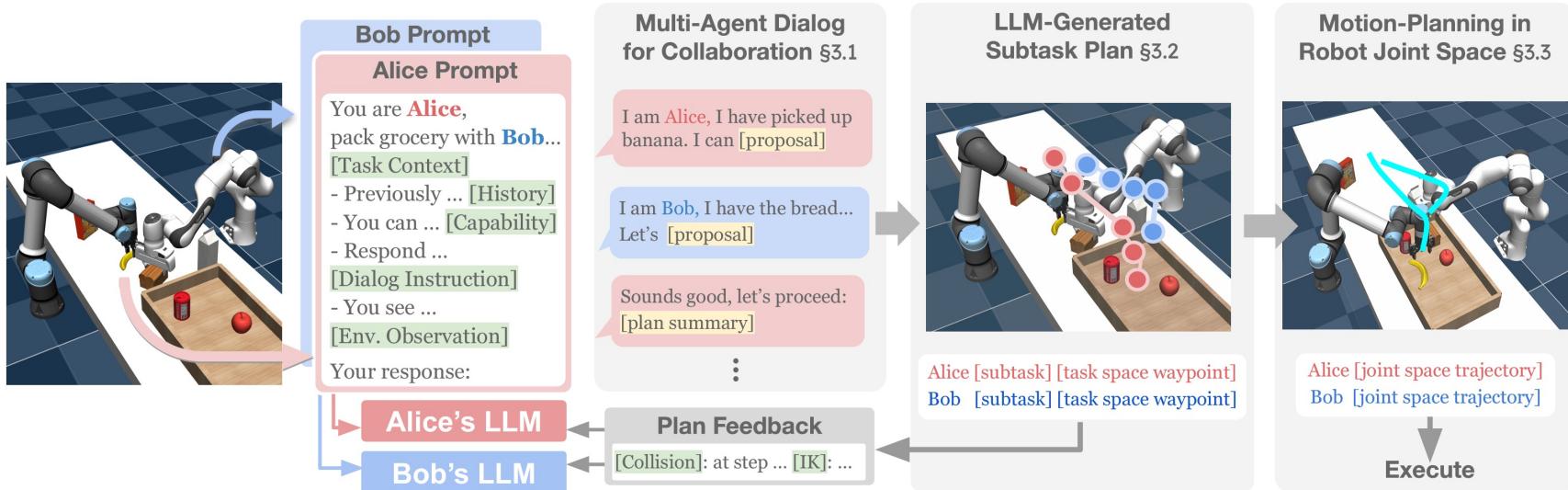
# Summary



Another issue is interaction  
with more than one agents

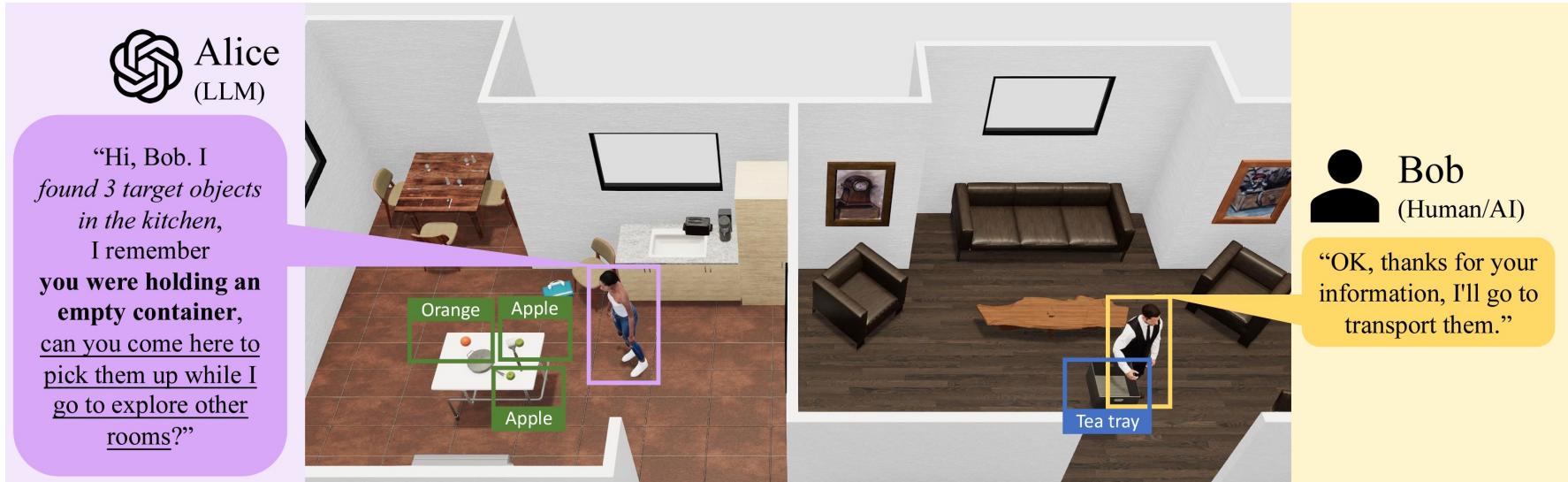
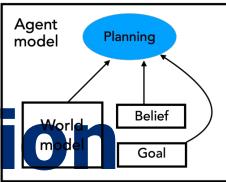
# Multi-agent planning and communication

- Multi-agent collaboration



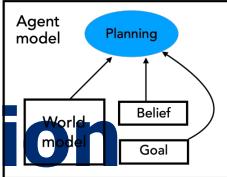
Mandi et al. (2023)

# Multi-agent planning and communication



Zhang et al. (2023)

# Multi-agent planning and communication

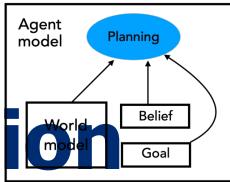


- Social



Park et al. (2023)

# Multi-agent planning and communication



- Social interaction between *multiple* simulated agents

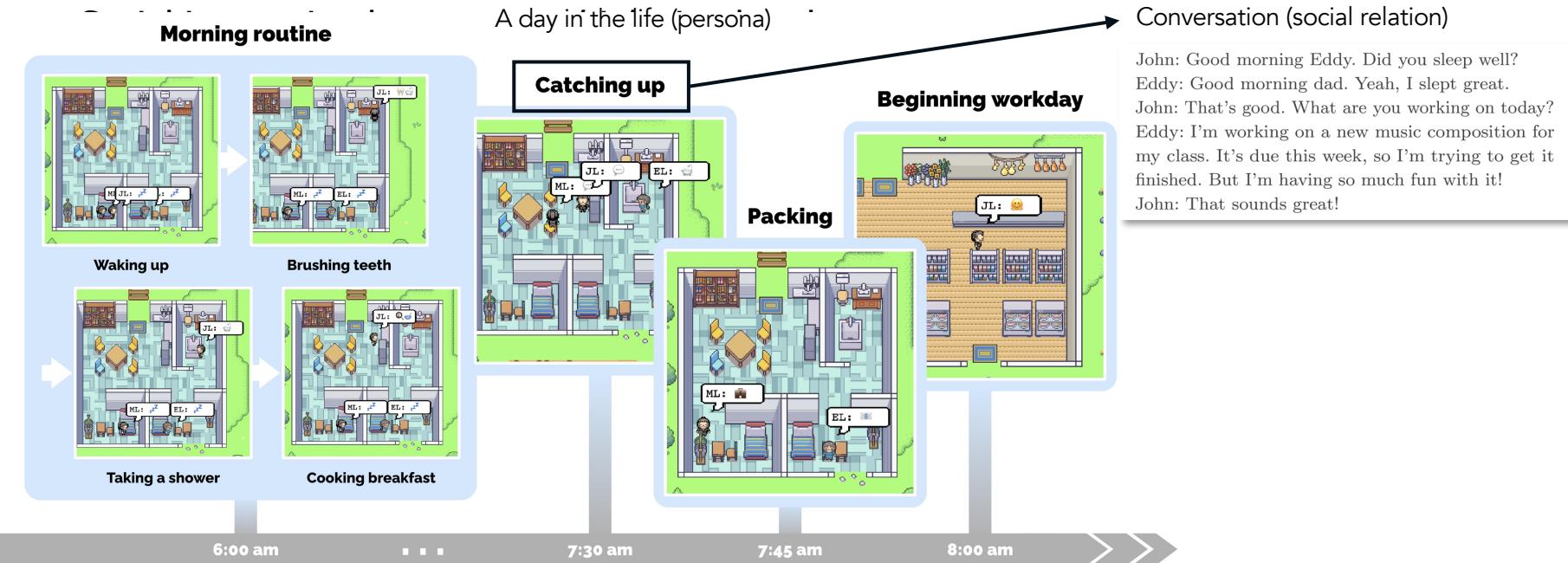
- Persona

John Lin is a pharmacy shopkeeper at the Willow Market and Pharmacy who loves to help people.

- Social relationships with other agents

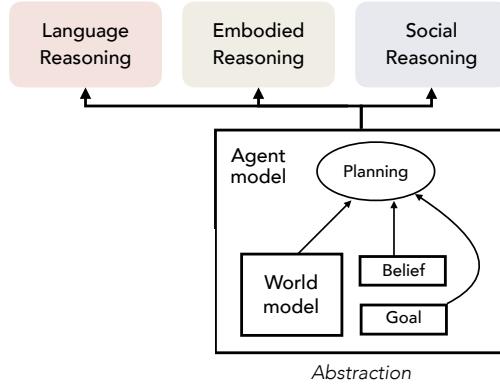
...  
John Lin is living with his wife, Mei Lin, who is a college professor, and on, Eddy Lin, who is a student studying music theory

# Multi-agent planning and communication



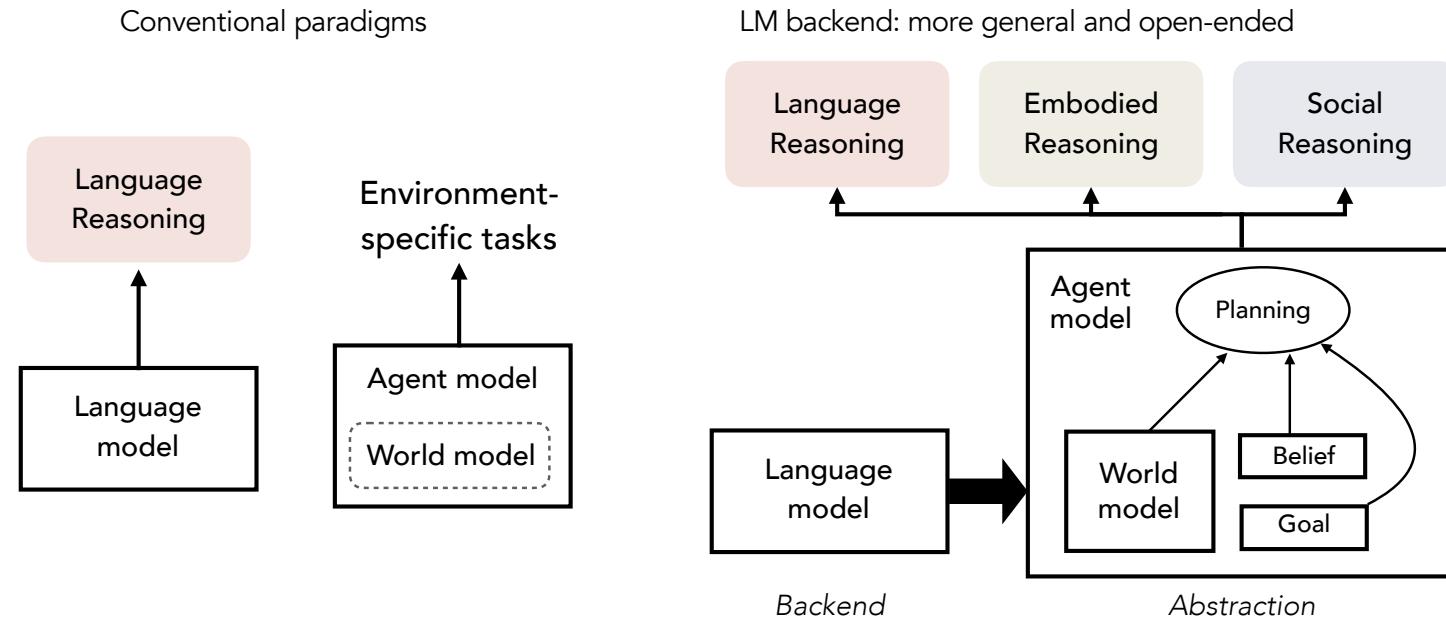
# Summary

- Model-based reasoning
  - Data-efficient
  - Generalizes to new scenarios well
- Traditionally, world models and agent models are all domain specific
- Can we leverage the open-endedness of LMs to construct world and agent models?
- Language models as backend



# Summary

- Language models as the backend for world models and agent models, supporting model-based reasoning



# Summary

---

- Language models as the backend for world models and agent models, supporting model-based reasoning
- While language alone is not sufficient for many tasks, language models can be connected with models operating on other modalities to achieve multimodal capacity
- There is still a need for manually crafted components (such as belief / memory modules) or conventional methods (such as classical planning)
- Enhancing the language model backend
  - Multimodality?
  - Single-model or modular design?
  - ...
- Questions?

# Envisioning the future

---

- More modalities — combinations of video (2D, 3D), text, code, etc.
- Large models and more efficient scaling
- More breath — more data and more types of data
- Interaction with physical world — models with hands and actuators
- Better personalization — these agents should serve your 🤝 needs
- Better human-machine teaming
  - CoPilot for coding
  - CoPilot for writing
  - ....
  - CoPilot for life!!!

- <https://sites.google.com/princeton.edu/cos597f>