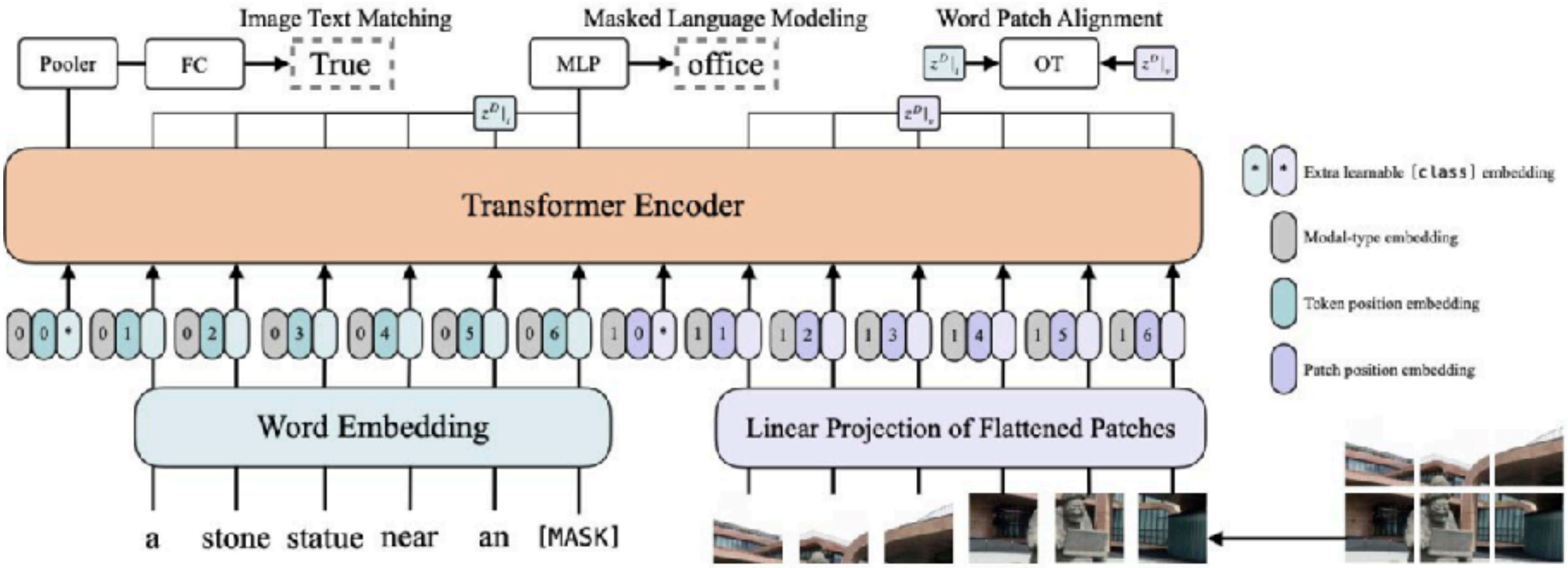
Vision and Language

Berkeley NLP

slides from: Daniel Fried, Yonatan Bisk, L-P Morency

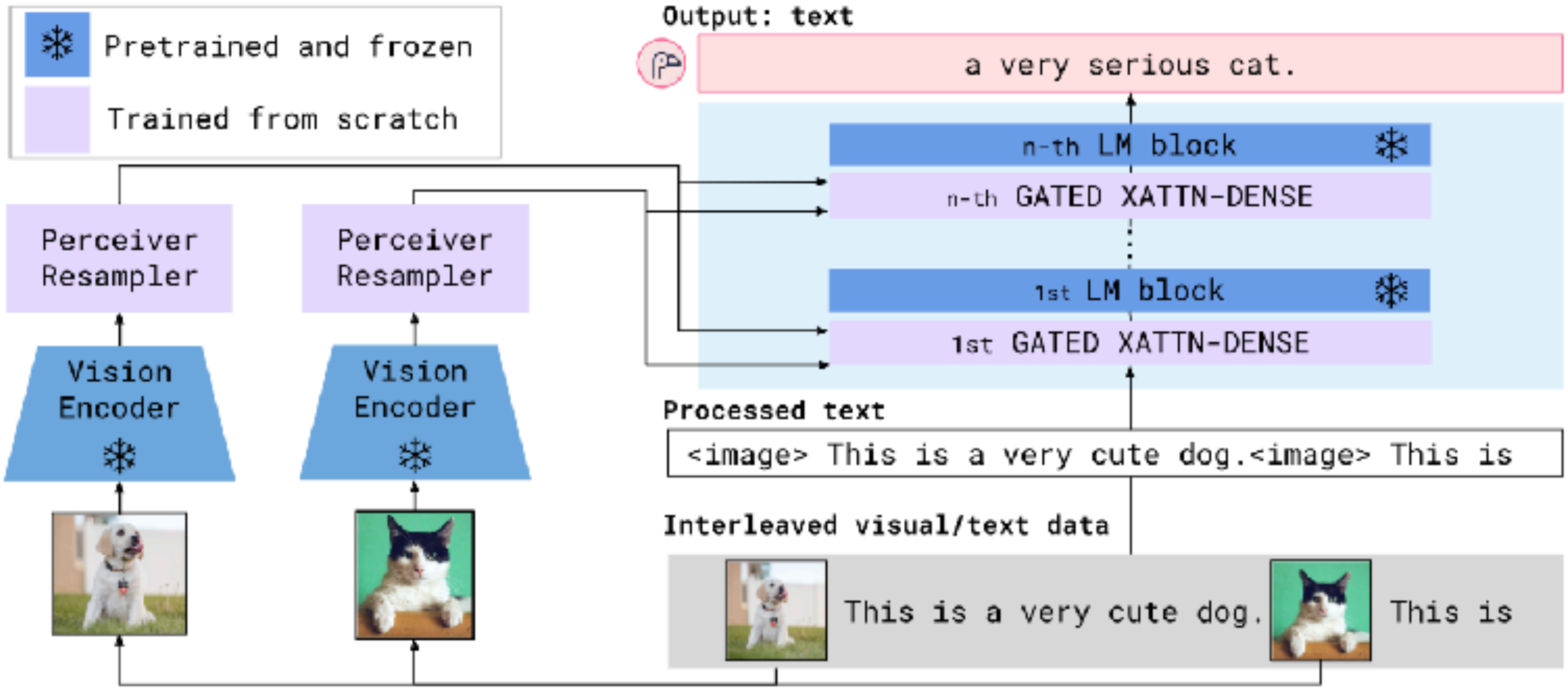
Joint Encoding: Multimodal Transformers

ViLT (Kim et al. 2021), encoder-only model (like BERT)



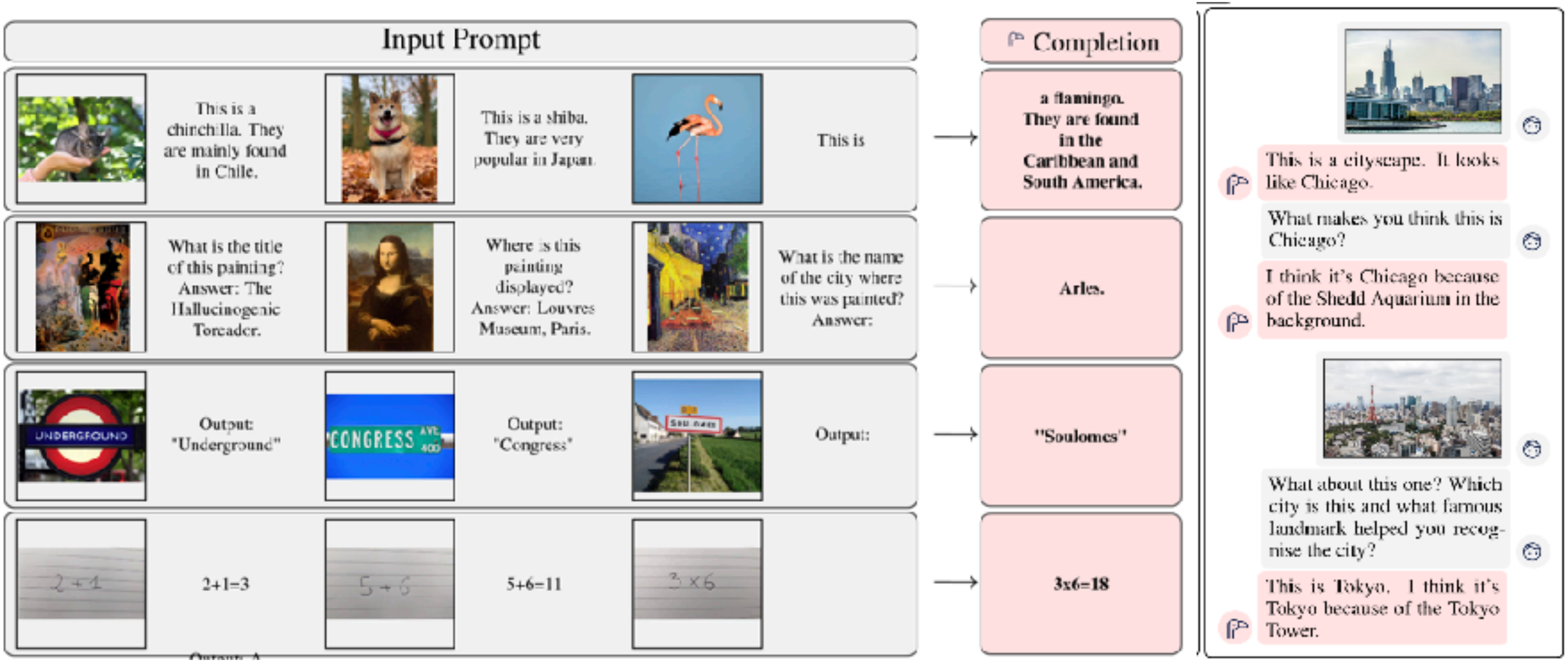
Joint Encoding: Multimodal Transformers

Flamingo, Alayrac et al. 2022

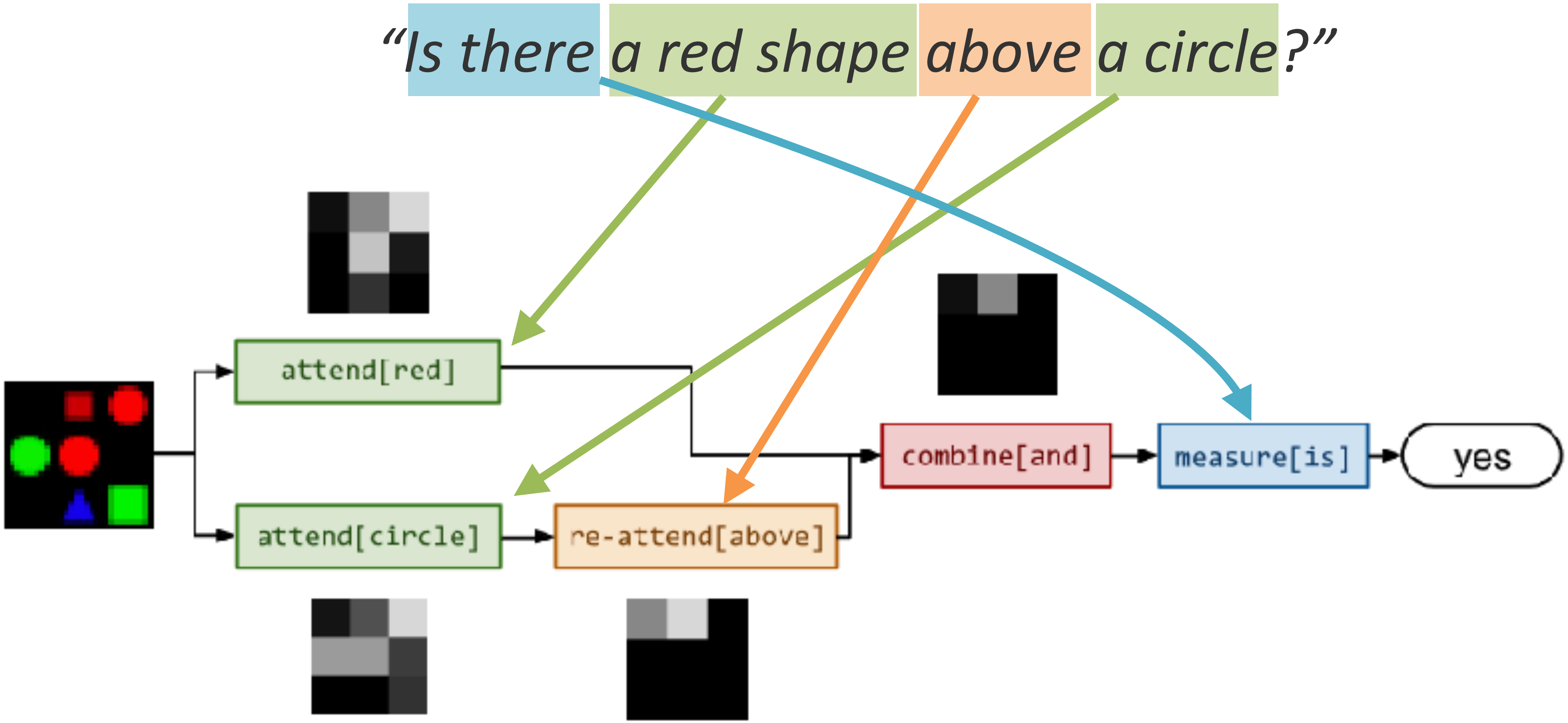


Joint Encoding: Multimodal Transformers

Flamingo, Alayrac et al. 2022



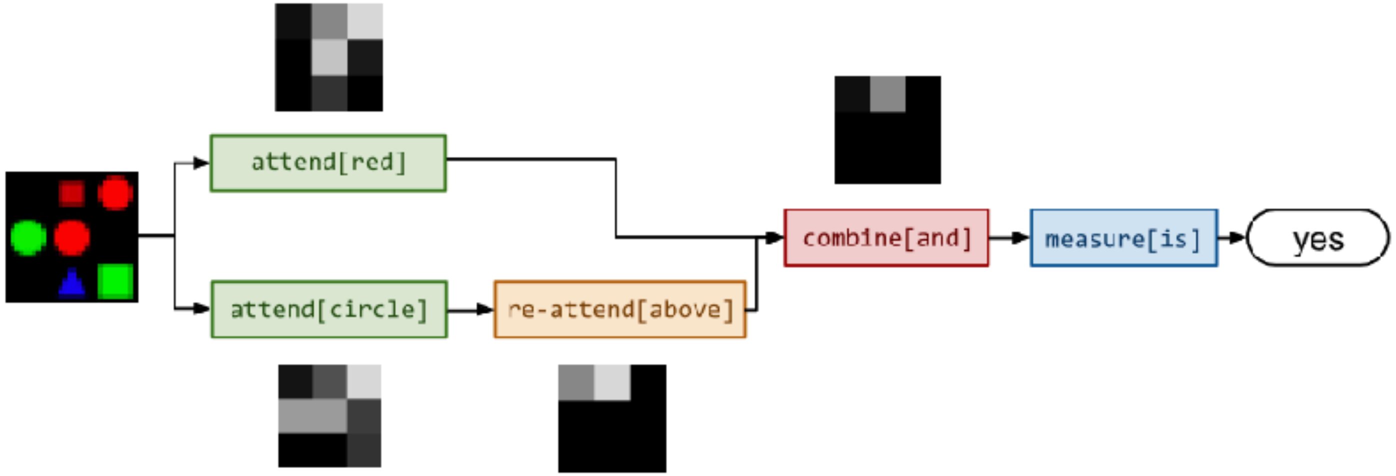
Neuromodular Approaches



Neural Module Networks, Andreas et al. 2017

Neuromodular Approaches

"Is there a red shape above a circle?"

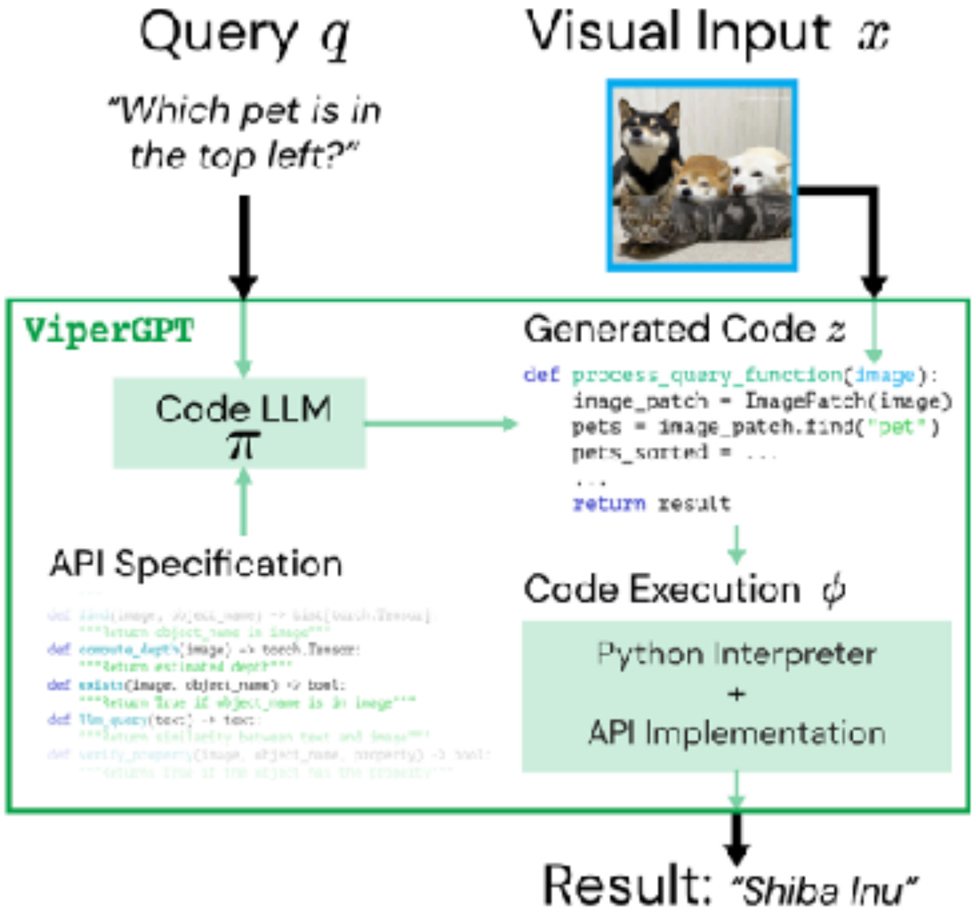


- Map *x* to some structured representation $\phi_l(x)$
- Manipulate image $\phi_w(i)$ according to components of this structured representation

Neural Module Networks, Andreas et al. 2017

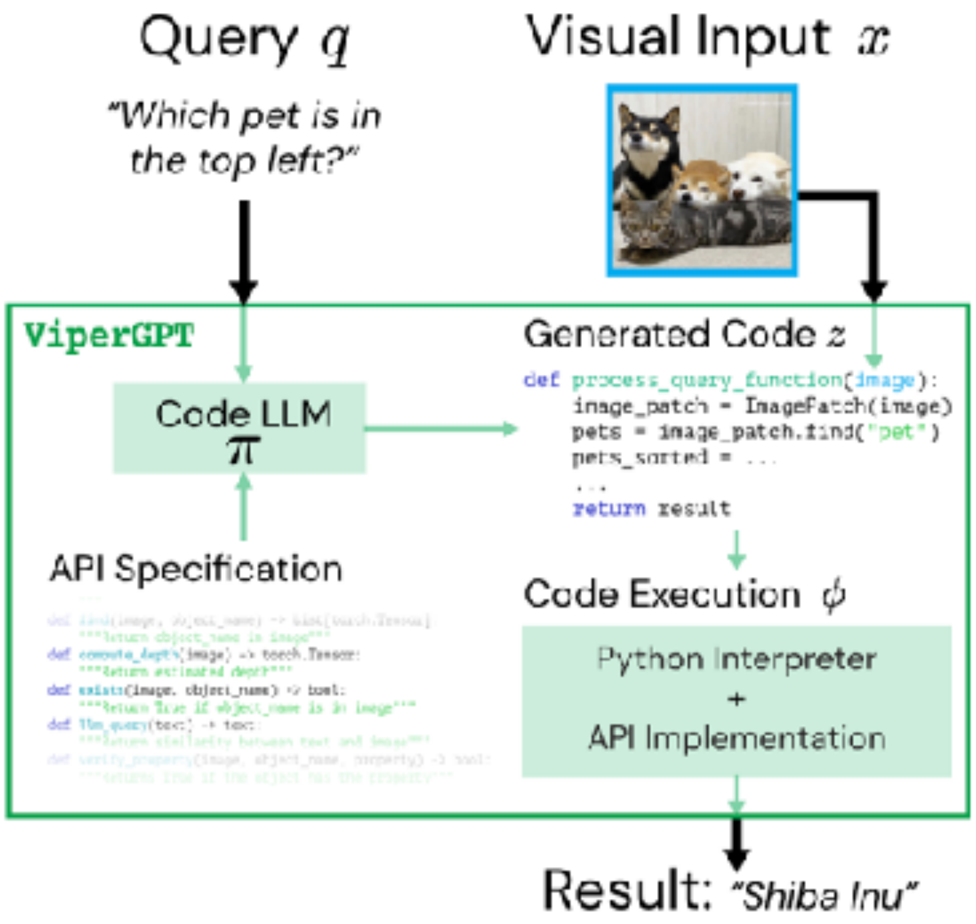
Neuromodular Approaches

- Text representation: executable python code
- Image representation: pixels (also assume access to some computer vision algorithms)
- Grounding: executing python code on image representations



Neuromodular Approaches

With sufficiently powerful code LLMs (e.g., Codex) and access to an API that can operate on top of images (or other modalities), no domain-specific or multimodal training is necessary



Neuromodular Approaches

Query: How many muffins can each kid have for it to be fair?

ViperGPT, Surís et al. 2023

Neuromodular Approaches

Query: How many muffins can each kid have for it to be fair?

Generated Code

```
def execute_command(image):
    image_patch = ImagePatch(image)
    muffin_patches = image_patch.find("muffin")
    kid_patches = image_patch.find("kid")
    return str(len(muffin_patches) // len(kid_patches))
```

ViperGPT, Surís et al. 2023

Neuromodular Approaches

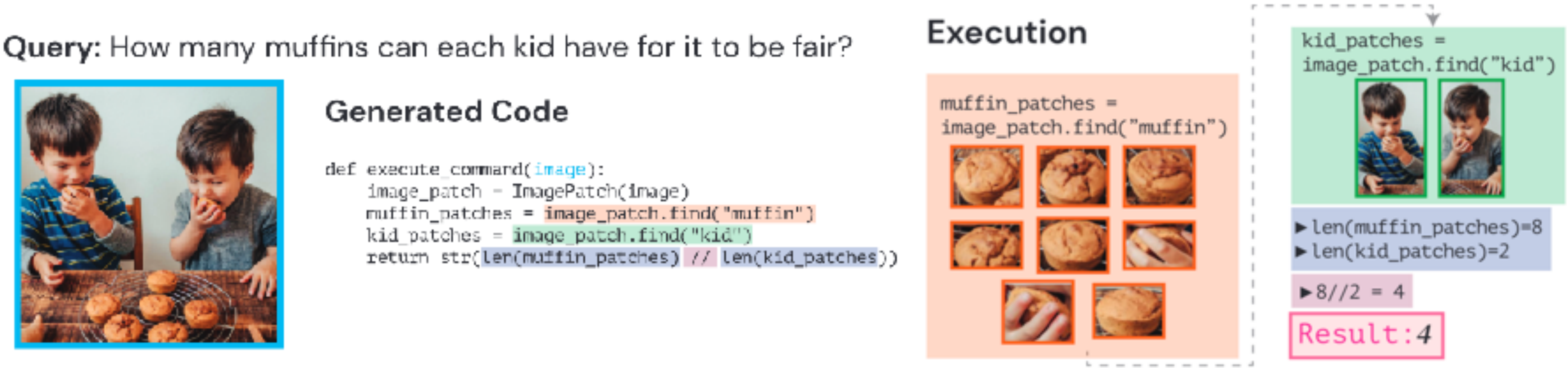
Query: How many muffins can each kid have for it to be fair?

Generated Code

```
def execute_command(image):
    image_patch = ImagePatch(image)
    muffin_patches = image_patch.find("muffin")
    kid_patches = image_patch.find("kid")
    return str(len(muffin_patches) // len(kid_patches))
```

ViperGPT, Surís et al. 2023

Neuromodular Approaches



ViperGPT, Surís et al. 2023

Neuromodular Approaches

Query: How many muffins can each kid have for it to be fair?

Generated Code

```
def execute_command(image):
    image_patch = ImagePatch(image)
    muffin_patches = image_patch.find("muffin")
    kid_patches = image_patch.find("kid")
    return str(len(muffin_patches) // len(kid_patches))
```

Execution

```
muffin_patches =
image_patch.find("muffin")
```

```
kid_patches =
image_patch.find("kid")
```

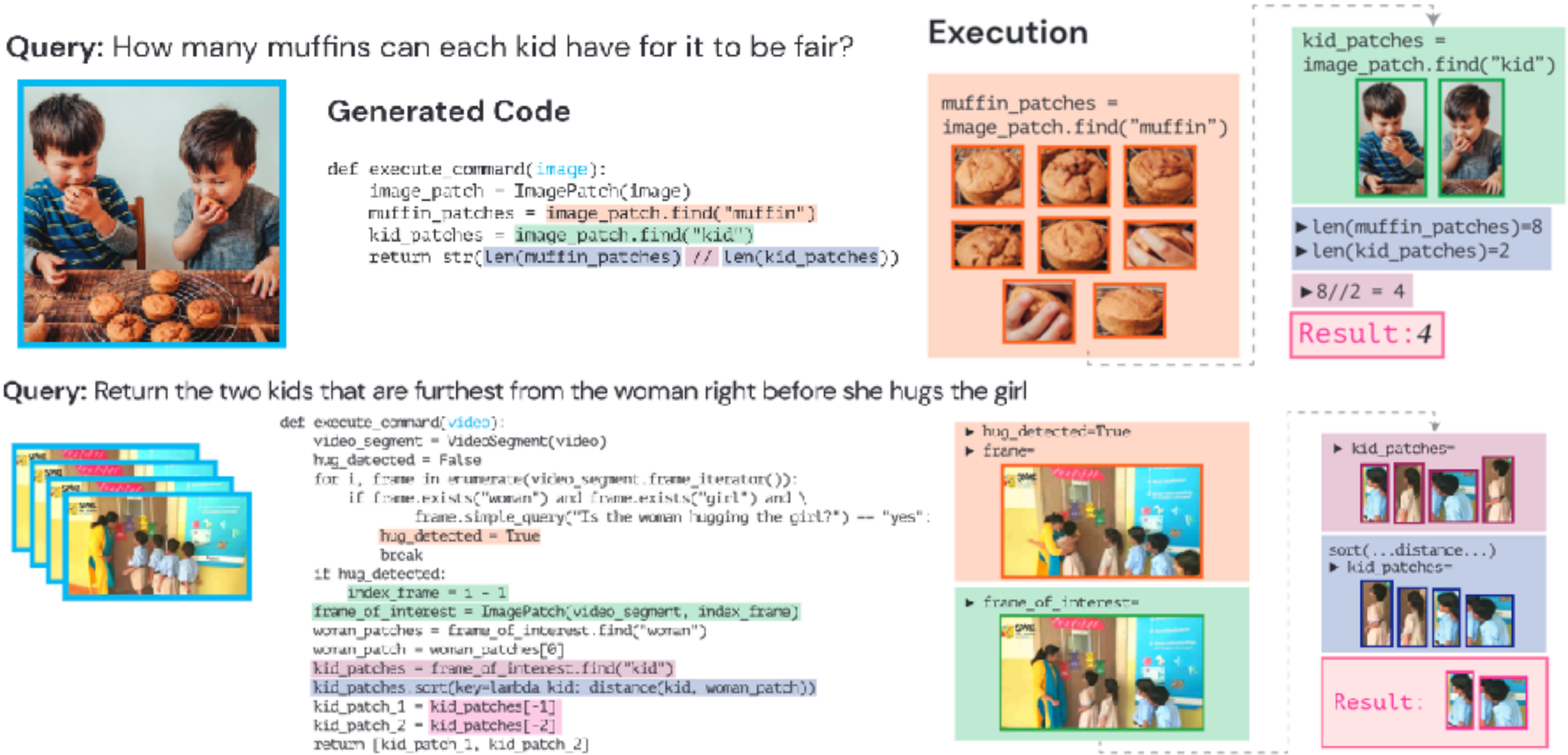
```
▶len(muffin_patches)=8
▶len(kid_patches)=2
▶8//2 = 4
Result:4
```

Query: Return the two kids that are furthest from the woman right before she hugs the girl

```
def execute_command(video):
    video_segment = VideoSegment(video)
    hug_detected = False
    for i, frame in enumerate(video_segment.frame_iterator()):
        if frame.exists("woman") and frame.exists("girl") and \
                frame.simple_query("Is the woman hugging the girl?") == "yes":
            hug_detected = True
            break
    if hug_detected:
        index_frame = i - 1
    frame_of_interest = ImagePatch(video_segment, index_frame)
    woman_patches = frame_of_interest.find("woman")
    woman_patch = woman_patches[0]
    kid_patches = frame_of_interest.find("kid")
    kid_patches.sort(key=lambda kid: distance(kid, woman_patch))
    kid_patch_1 = kid_patches[-1]
    kid_patch_2 = kid_patches[-2]
    return [kid_patch_1, kid_patch_2]
```

ViperGPT, Surís et al. 2023

Neuromodular Approaches



ViperGPT, Surís et al. 2023

Neuromodular Approaches

Query: What color do you get if you combine the colors of the viper and the flower?

```
def execute_command(image):
    image_patch = ImagePatch(image)
    viper_patches = image_patch.find("viper")
    flower_patches = image_patch.find("flower")
    viper_patch = viper_patches[0]
    flower_patch = flower_patches[0]
    viper_color = viper_patch.simple_query("What color is the viper?")
    flower_color = flower_patch.simple_query("What color is the flower?")
    color = llm_query(f"What color do you get if you combine the colors
                        {viper_color} and {flower_color}?")
    return color
```

▶ viper_patch= 

▶ flower_patch= 

▶ viper_color='blue'

▶ flower_color='red'

▶ color='purple'

Result: *"purple"*

ViperGPT, Surís et al. 2023

Drawback: Context-Dependence

"Is the potted plant to the right of the bench?"

Visual Spatial Reasoning, Liu Fangyu et al. 2023

Drawback: Context-Dependence

"Is the potted plant to the right of the bench?"

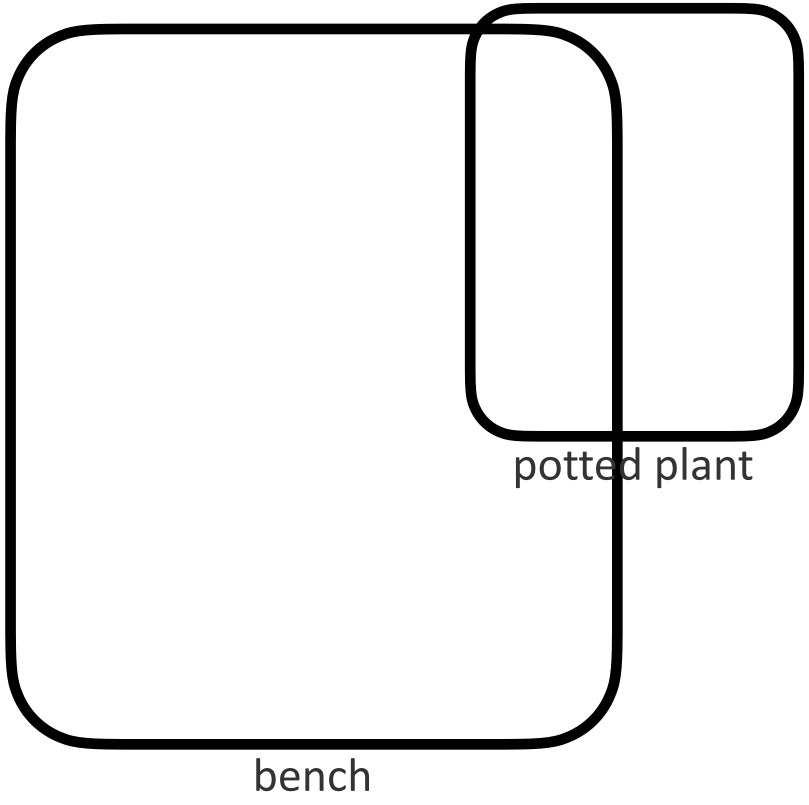
```
bbox_plant = detect(image, "potted plant")
bbox_bench = detect(image, "bench")
return bbox_plant.x > bbox_bench.x
```

Visual Spatial Reasoning, Liu Fangyu et al. 2023

Drawback: Context-Dependence

"Is the potted plant to the right of the bench?"

```
bbox_plant = detect(image, "potted plant")
bbox_bench = detect(image, "bench")
return bbox_plant.x > bbox_bench.x
```



Visual Spatial Reasoning, Liu Fangyu et al. 2023

Drawback: Context-Dependence

“Is the potted plant to the right of the bench?”

```
bbox_plant = detect(image, “potted plant”)
bbox_bench = detect(image, “bench”)
return bbox_plant.x > bbox_bench.x
```

Visual Spatial Reasoning, Liu Fangyu et al. 2023

Drawback: Context-Dependence

"Is the potted plant to the right of the bench?"

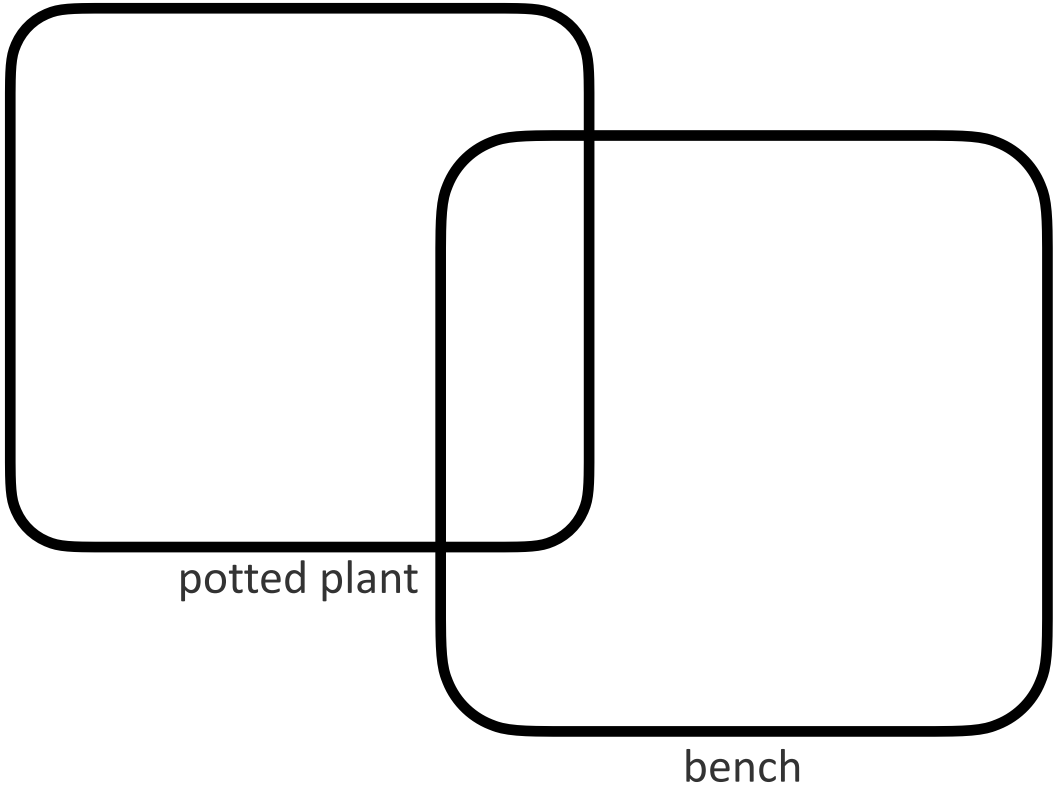
```
bbox_plant = detect(image, "potted plant")
bbox_bench = detect(image, "bench")
return bbox_plant.x > bbox_bench.x
```

Visual Spatial Reasoning, Liu Fangyu et al. 2023

Drawback: Context-Dependence

“Is the potted plant to the right of the bench?”

```
bbox_plant = detect(image, “potted plant”)
bbox_bench = detect(image, “bench”)
return bbox_plant.x > bbox_bench.x
```



Visual Spatial Reasoning, Liu Fangyu et al. 2023

Drawback: Context-Dependence

"Is the potted plant to the right of the bench?"

```
bbox_plant = detect(image, "potted plant")
bbox_bench = detect(image, "bench")
return bbox_plant.x > bbox_bench.x
```

Visual Spatial Reasoning, Liu Fangyu et al. 2023

Diffusion

- Different setting: image is not provided as input
- Instead, want to generate an image from scratch conditioned on some text description
- Problem: evaluation

Forward Process: Adding Noise

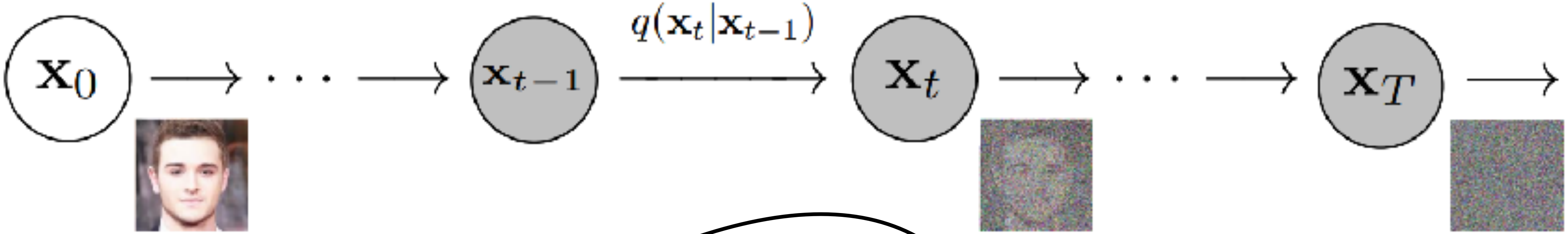


Image from training set

$$q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$$

Markov chain

Diagonal Gaussian distribution

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}\right)$$

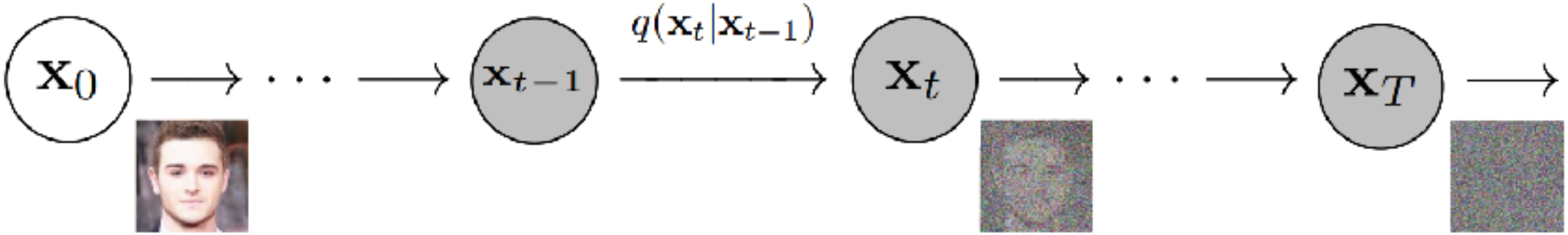
Means: will get closer to zero

Variance

$$\beta_t \in (0,1)$$
$$\beta_1 < \beta_2 < \cdots < \beta_T$$

Slides from Aryan Jain, CS 198-126 Fall 2022, and Ari Seff

Forward Process: Adding Noise



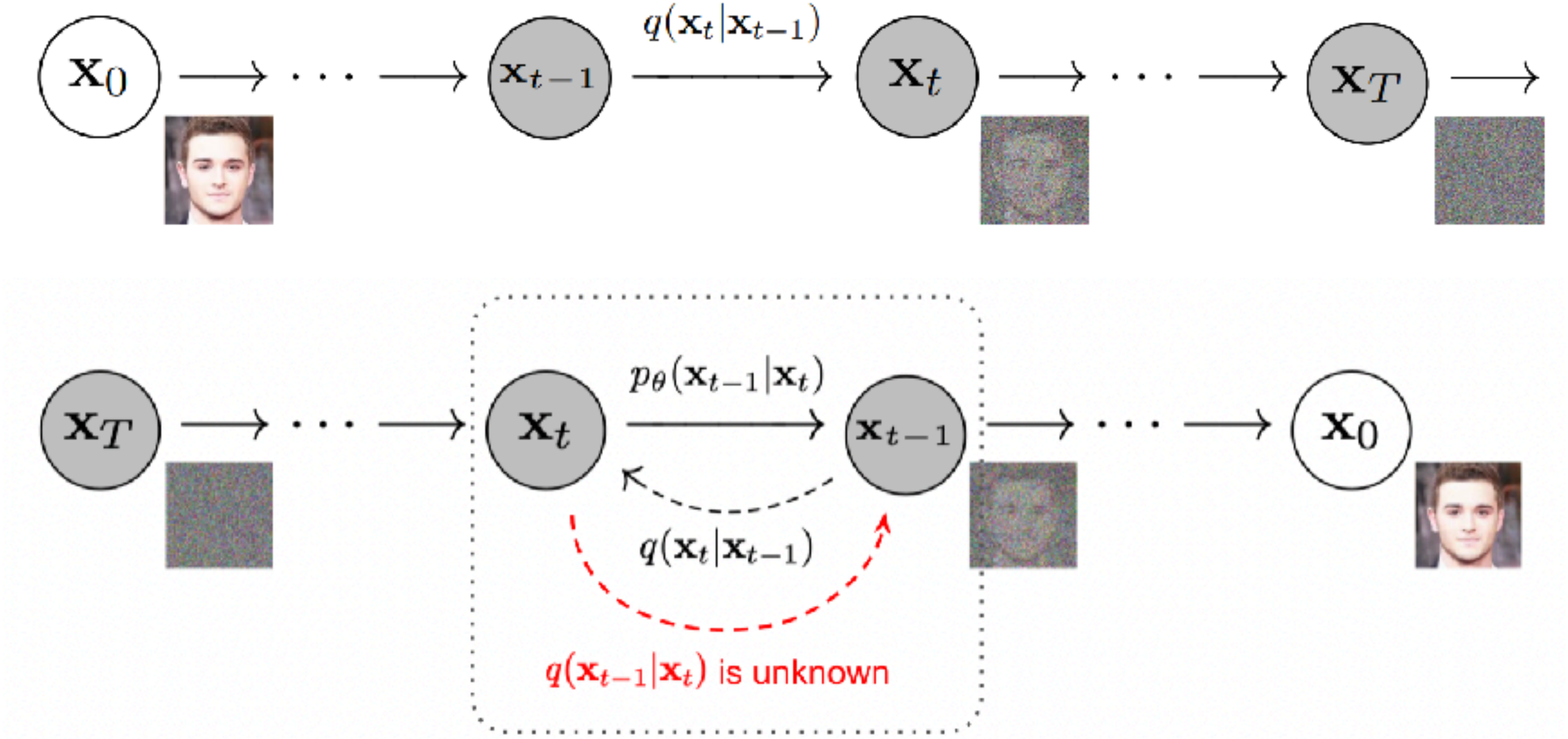
$$q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t \mid \mathbf{x}_{t-1})$$

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}\right)$$

$$q(\mathbf{x}_\infty \mid \mathbf{x}) \approx \mathcal{N}(0, \mathbf{I})$$

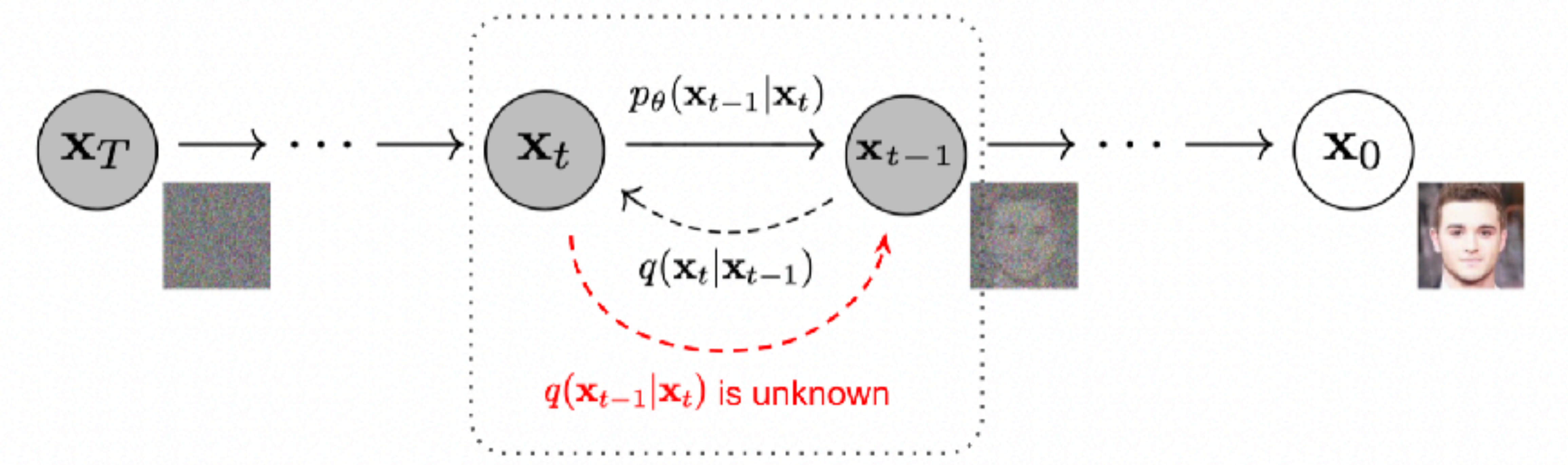
Slides from Aryan Jain, CS 198-126 Fall 2022, and Ari Seff

Reverse Process: Denoising



Slides from Aryan Jain, CS 198-126 Fall 2022, and Ari Seff

Reverse Process: Denoising



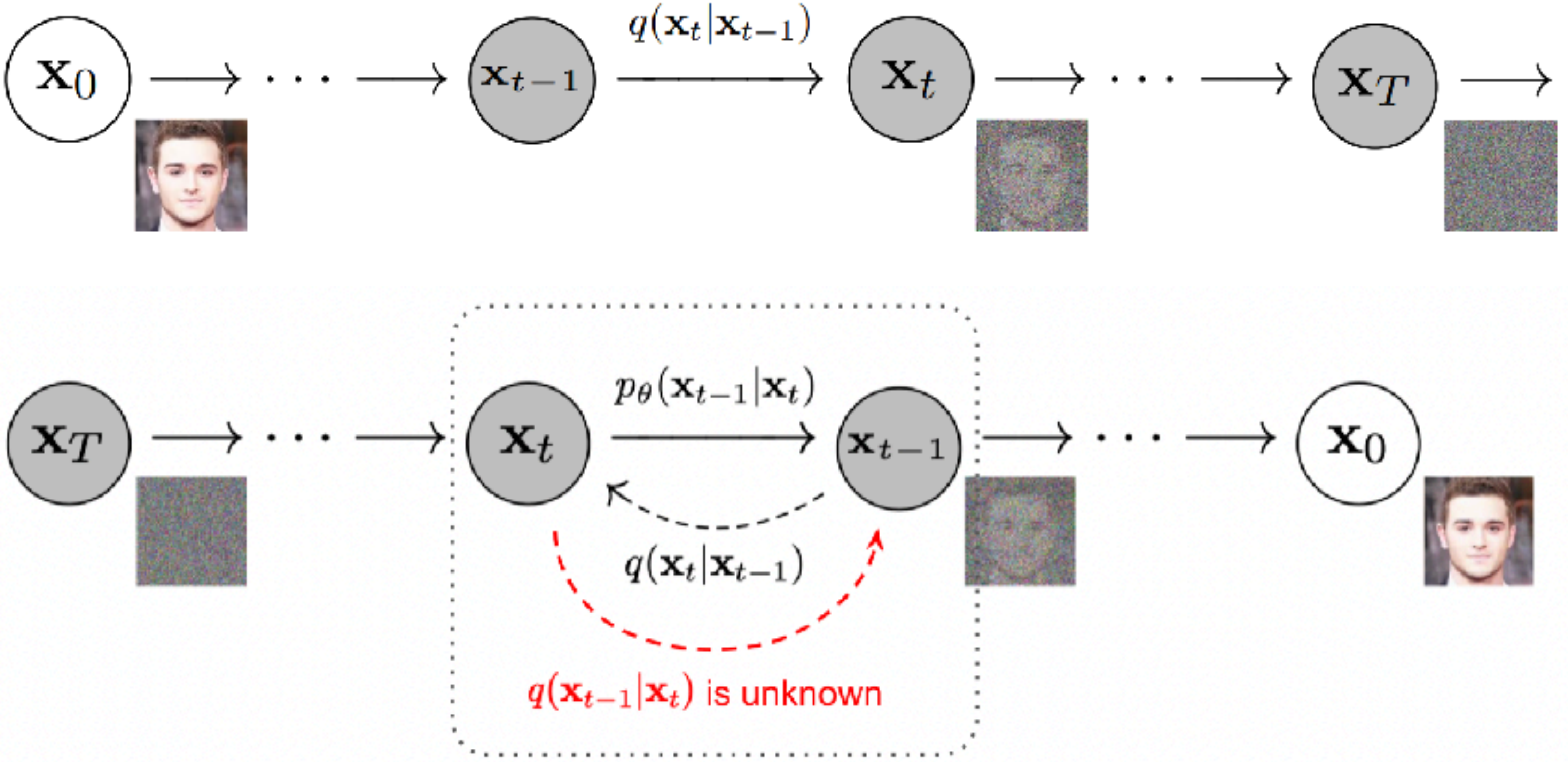
Parameterized denoising process

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)\right)$$

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) \qquad p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; 0, \mathbf{I})$$

Slides from Aryan Jain, CS 198-126 Fall 2022, and Ari Seff

Latent Variable Problem



Training

General variational objective

$$\log p_\theta(x) \geq \mathbb{E}_{q(z|x)}\left[\log p_\theta(x \mid z)\right] - D_{KL}\left(q(z \mid x) \| p_\theta(z)\right)$$

Maximize the likelihood of observed variables *x* over distribution of latent variables *z* given observed variables

Make the posterior distribution of latents *z* given observed variables similar to the prior over latents

$$\log p_\theta(\mathbf{x}_0) \geq \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\log p_\theta(\mathbf{x}_0 \mid \mathbf{x}_{1:T})\right] - D_{KL}\left(q(\mathbf{x}_{1:T} \mid \mathbf{x}_0) \| p_\theta(\mathbf{x}_{1:T})\right)$$

Training

Algorithm 1 Training

1: **repeat**
2: $\quad \mathbf{x}_0 \sim q(\mathbf{x}_0)$ ← Sample an image from training set
3: $\quad t \sim \mathrm{Uniform}(\{1, \ldots, T\})$ ← Sample a random timestep
4: $\quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: $\quad$ Take gradient descent step on

$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$

6: **until** converged

Can sample directly from 0 to timestep *t*!

$$q(\mathbf{x}_t \mid \mathbf{x}_0) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{\overline{\alpha}_t}\mathbf{x}_0, (1-\overline{\alpha}_t)\mathbf{I}\right)$$

$$\alpha_t = 1 - \beta_t$$

$$\overline{\alpha}_t = \prod_{s=1}^{t} \alpha_s$$

Training

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \ldots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on

$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$

6: **until** converged

Sample an image from training set

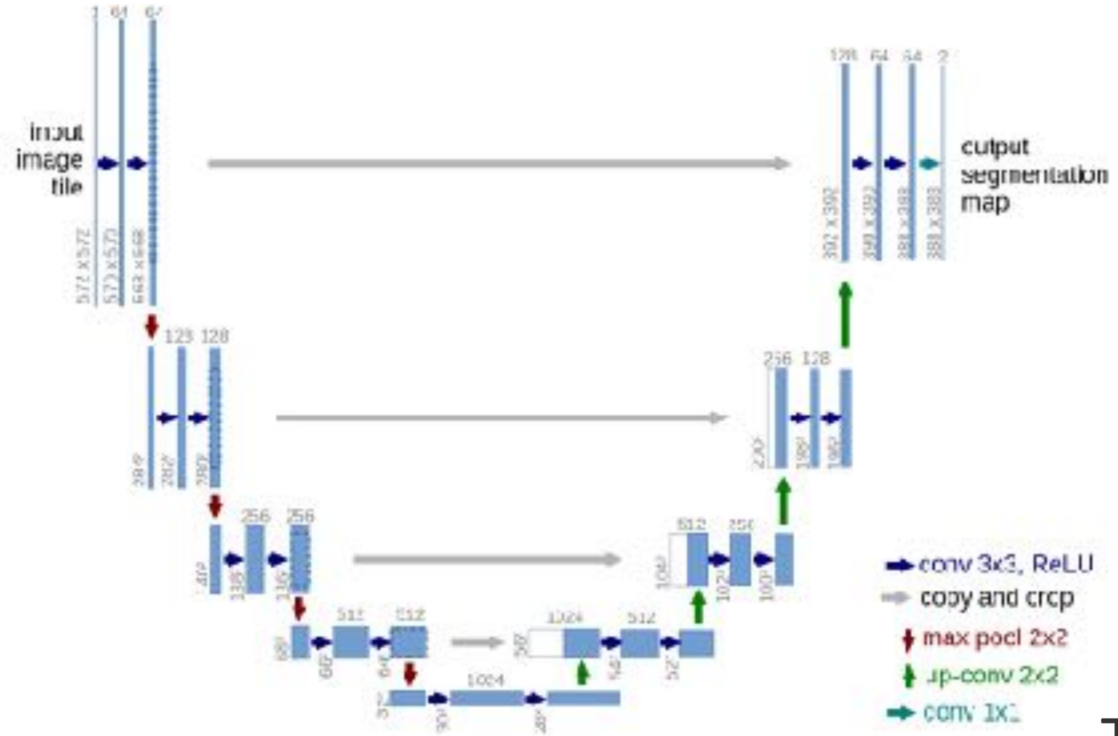
Sample a random timestep

We are actually learning parameters for ϵ_θ

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)\right)$$

$$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right)$$

Training



We are actually learning parameters for ϵ_θ

Typically a U-Net

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)\right)$$

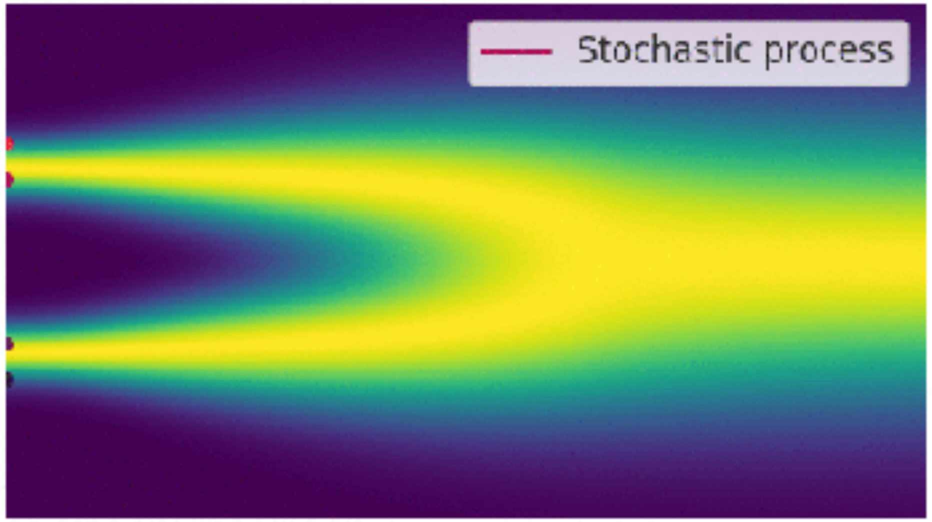
$$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right)$$

Inference

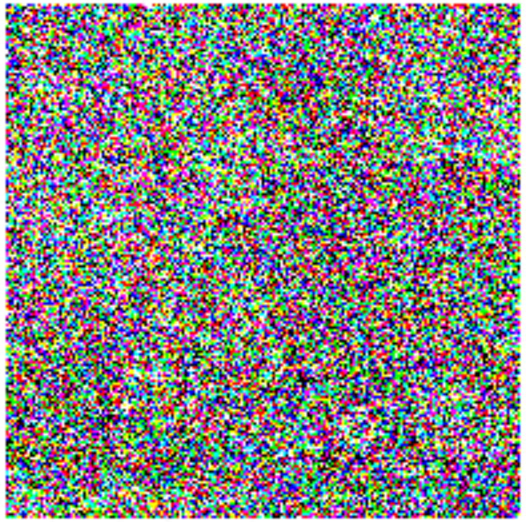
Algorithm 2 Sampling

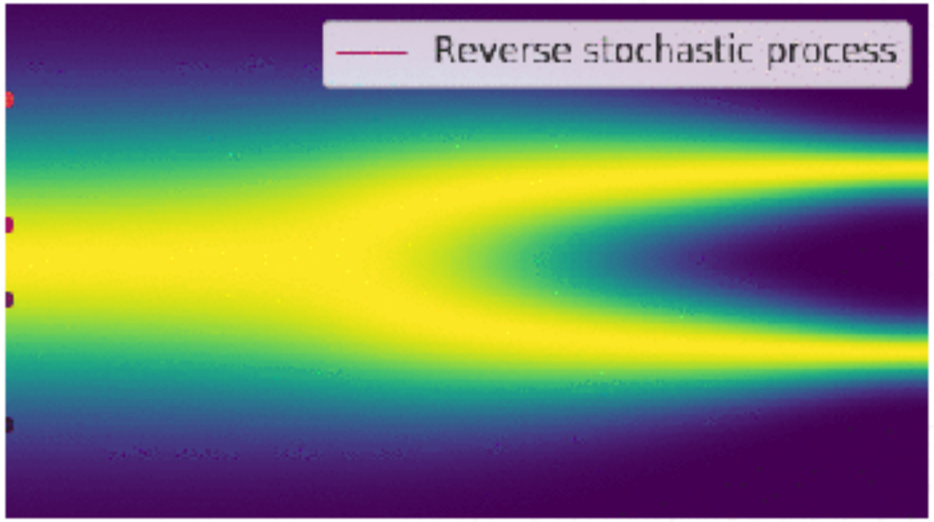
$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ Sample noise to condition upon
for $t = T, \ldots, 1$ **do** Rollout by iteratively sampling
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
  $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
end for
return $\mathbf{x}_0$

Diffusion



Forward process: convert image to noise





Reverse process: sample from the distribution of images, starting with pure noise

Text-Conditioned Diffusion

- Like any latent variable model, we can just add in another observed variable to condition upon
- In this case, it might be an object class or a text description
-

A cute corgi lives in a house made of sushi.

Text-Conditioned Diffusion

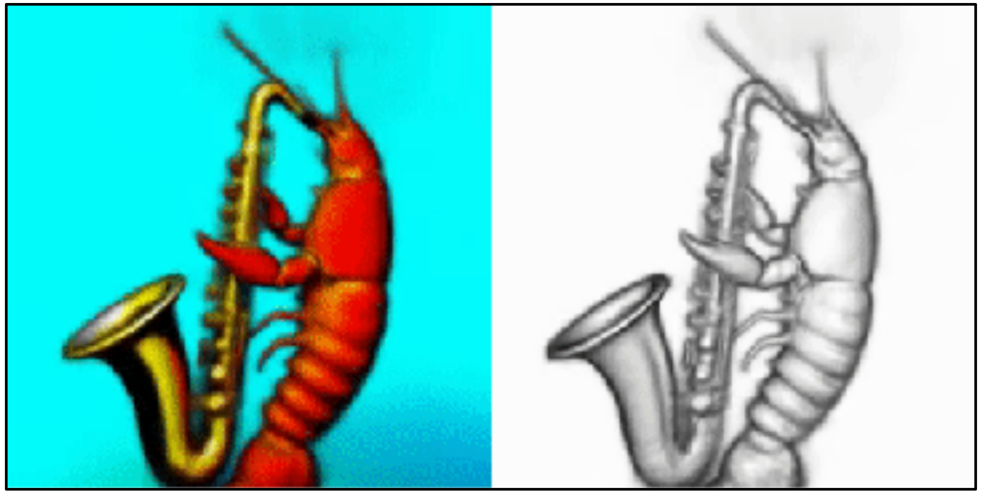
- Like any latent variable model, we can just add in another observed variable to condition upon
- In this case, it might be an object class or a text description
- We can also generate media beyond 2d images…

Horse drinking water

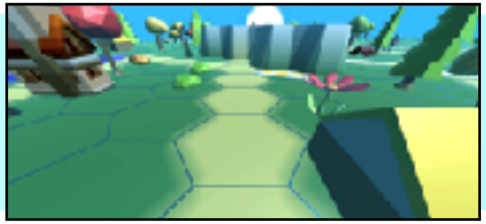
Text-Conditioned Diffusion

- Like any latent variable model, we can just add in another observed variable to condition upon
- In this case, it might be an object class or a text description
- We can also generate media beyond 2d images…

A lobster playing the saxophone



Situated Instruction Following

$f\,(\text{instruction},$  $) \to \texttt{actions}$

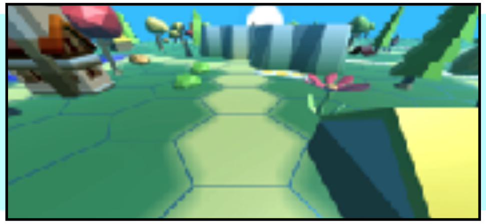
Room to Room, Anderson et al. 2018

Leave the bedroom, and enter the kitchen. Walk forward, and take a left at the couch. Stop in front of the window.

Touchdown, Chen et al. 2018

Orient yourself so that the umbrellas are to the right. Go straight and take a right at the first intersection. At the next intersection there should be an old-fashioned store to the left. There is also a dinosaur mural to the right.

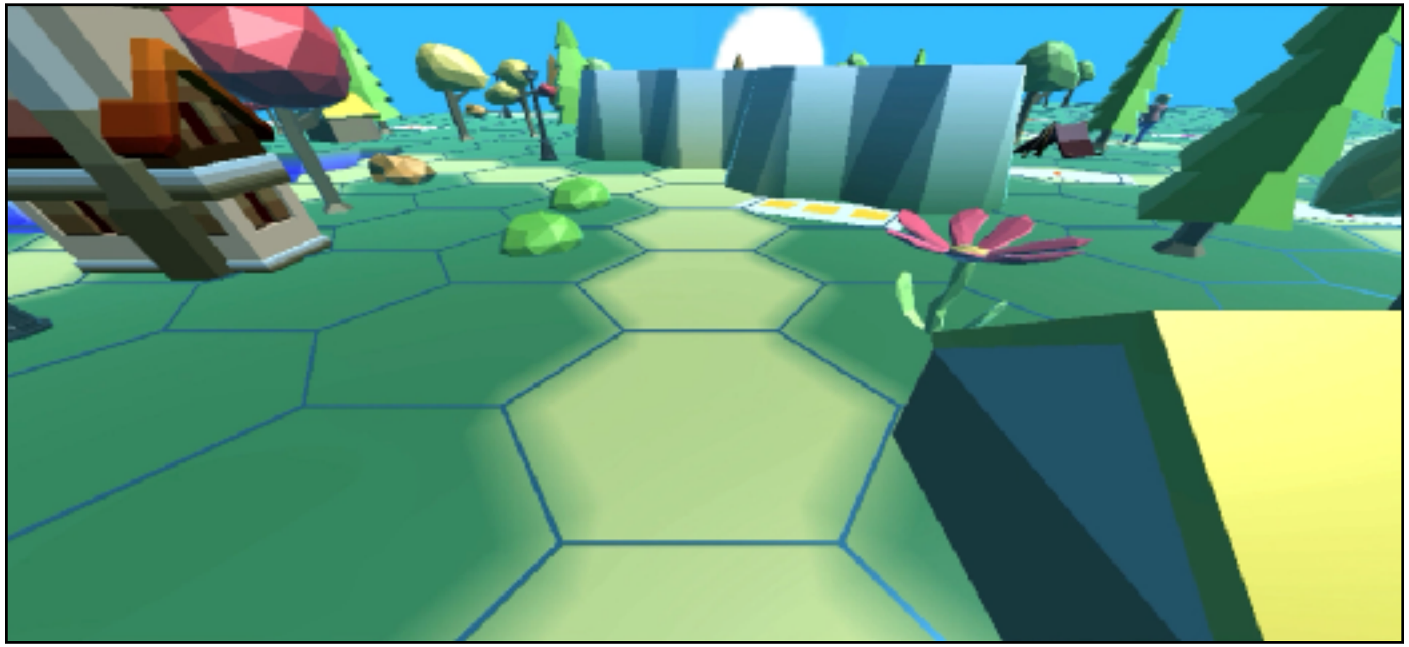
Situated Instruction Following

$f\,(\text{instruction},$  $) \rightarrow \texttt{actions}$

ALFRED, Shridhard et al. 2020

CerealBar, Suhr et al. 2019

Pick up knife, cut potato, put potato in fridge, remove from fridge, place in the microwave

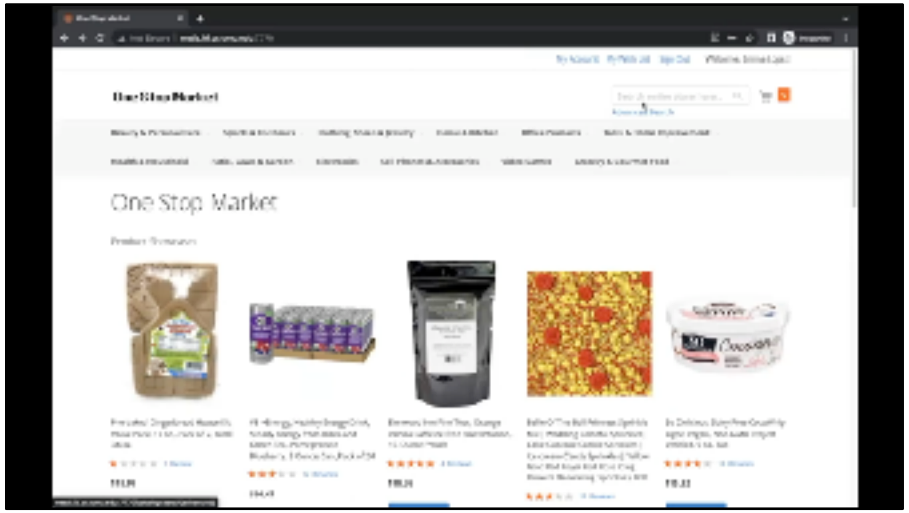


Turn around and get the three red stripes behind you.

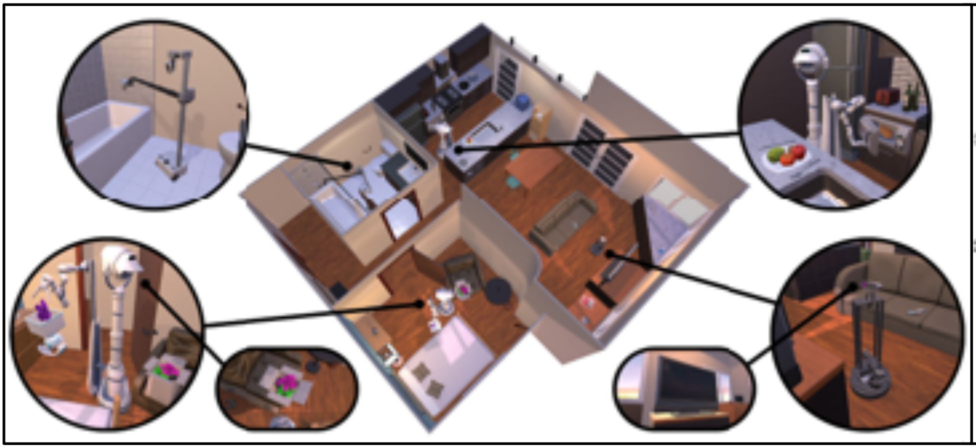
Environments

- 2D or 3D rendered environments
  - Can easily generate new environments on the fly
  - Support manipulable environments
  - Simulation allows for rapid experimentation and evaluation

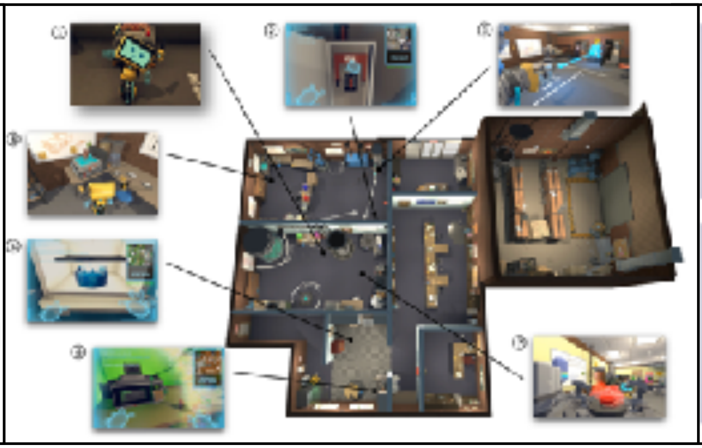
WebArena, Zhou Shuyan et al. 2023



AI2-THOR, Kolve et al. 2022



Alexa Arena, Gao Qiaozi et al. 2023

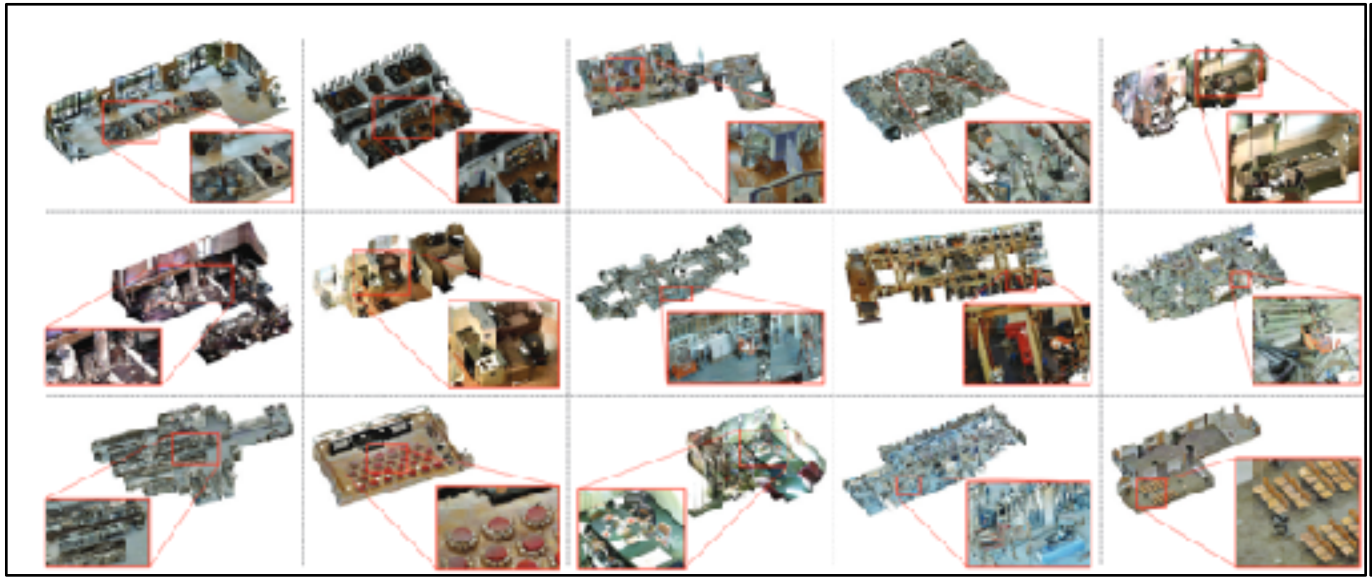


VRKitchen, Gao Xiaofeng et al. 2019

Environments

- 2D or 3D rendered environments
- Photorealistic environments

Gibson Env, Xia Fei et al. 2018

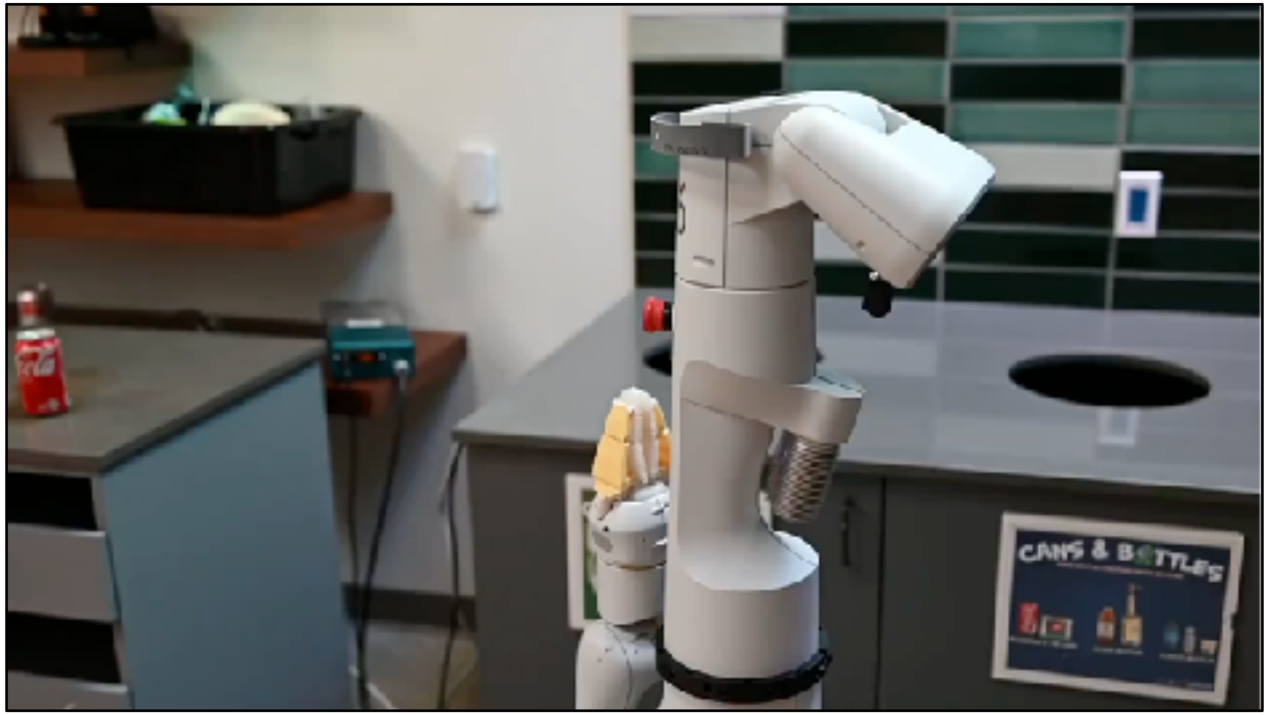


StreetLearn, Mirowski et al. 2019

Environments

- 2D or 3D rendered environments
- Photorealistic environments
- Literal physical embodiment (robotics)

SayCan, Ahn et al. 2022



GRIF, Myers et al. 2023

Place the knife in front of the microwave.

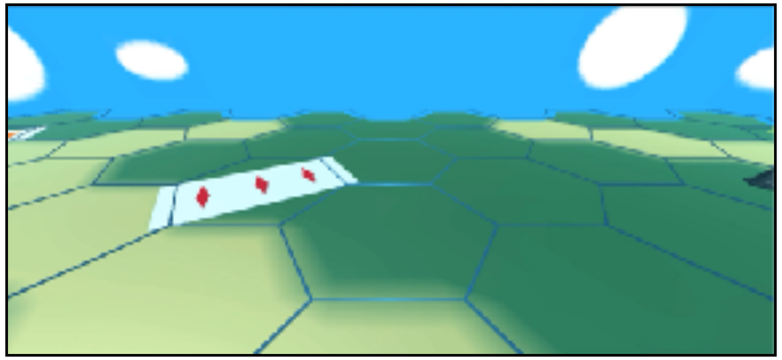
Embodied Agents: Challenges

- Grounding language to perception
- Reasoning about world dynamics
- Grounding language to action
- In collaborative tasks: also reasoning about one’s interlocutor
- Evaluating success

Reasoning about World Dynamics

(Partially observable) Markov decision process formulation of embodied agents

Reasoning about World Dynamics



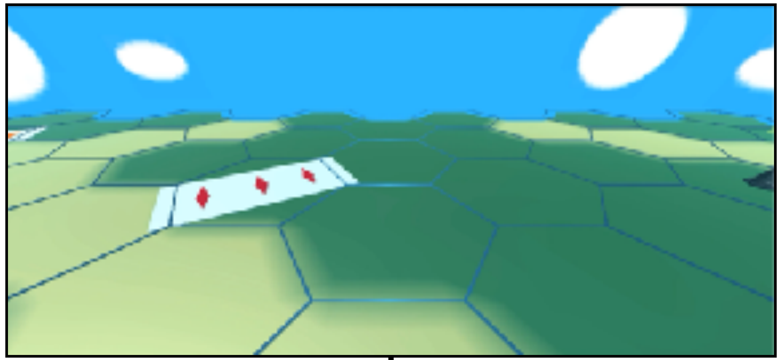
(Partially observable) Markov decision process formulation of embodied agents

- States $\mathcal{S}$ (and observations $\mathcal{O}$)

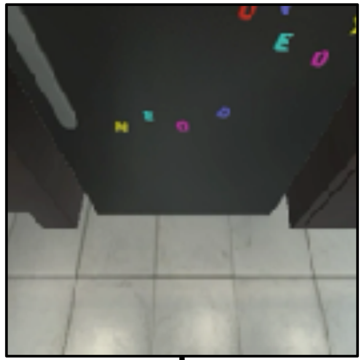
Reasoning about World Dynamics

(Partially observable) Markov decision process formulation of embodied agents

- States $\mathcal{S}$ (and observations $\mathcal{O}$)
- Actions $\mathcal{A}$



LEFT

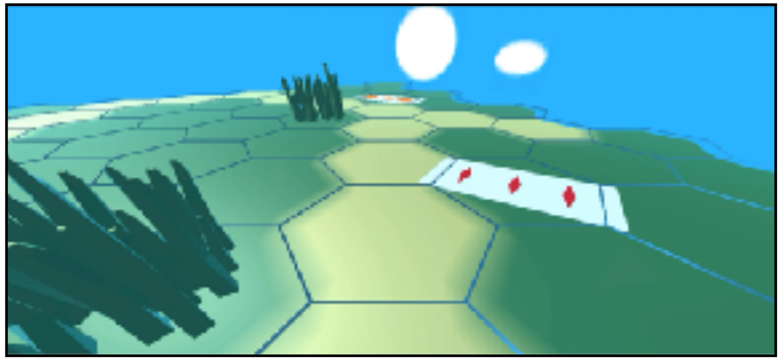


OPEN(FRIDGE)

Reasoning about World Dynamics

(Partially observable) Markov decision process formulation of embodied agents

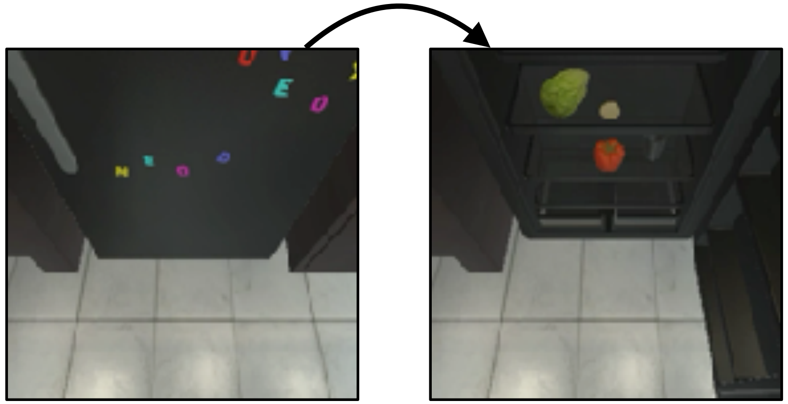
- States $\mathcal{S}$ (and observations $\mathcal{O}$)
- Actions $\mathcal{A}$
- Transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \Delta^{\mathcal{S}}$



Reasoning about World Dynamics

(Partially observable) Markov decision process formulation of embodied agents

- States $\mathcal{S}$ (and observations $\mathcal{O}$)
- Actions $\mathcal{A}$
- Transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta^{\mathcal{S}}$
- Reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$



Reasoning about World Dynamics

(Partially observable) Markov decision process formulation of embodied agents

- States $\mathcal{S}$ (and observations $\mathcal{O}$)
- Actions $\mathcal{A}$
- Transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \Delta^{\mathcal{S}}$
- Reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R}$

$$\pi : \mathcal{O} \rightarrow \Delta^{\mathcal{A}}$$

Reasoning about World Dynamics

- What is your state space?
  - Does it include all information about the environment?
  - Does it include information about the trajectory so far, e.g., previous states and actions?
  - Does it include a natural language instruction?
- Is the environment partially observable?
- What is the action space?
  - Lowest level action space: continuous control
  - Higher level action space: sufficient for simulated environments
- How is the policy implemented?

Embodied Agent Policies

Observation space:

- Previous and current visual observations
- Previous actions
- Instruction

Policy: whatever neural implementation you want

π

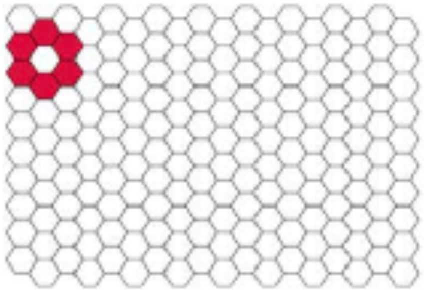
Turn around and get the three red stripes behind you.

Action	Probability
LEFT	64%
RIGHT	2%
FORWARD	28%
BACKWARD	3%
STOP	3%

Grounding Language to Action

- How do we define our action space?
- In many cases, language provides a decent set of abstractions that help us define meaningful higher-level action spaces
- Language can also allude to structured action spaces

1. Make a *red flower*, by coloring in red *all tiles adjacent* to the 2nd tile from the top in the 2nd column from the left.

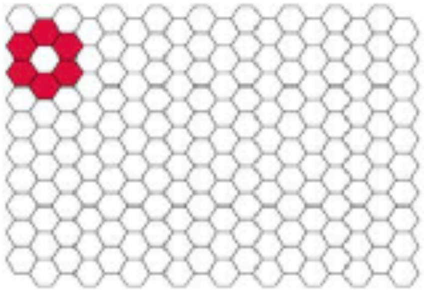


Hexagons, Lachmy et al. 2022

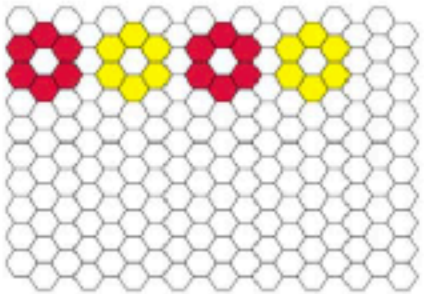
Grounding Language to Action

- How do we define our action space?
- In many cases, language provides a decent set of abstractions that help us define meaningful higher-level action spaces
- Language can also allude to structured action spaces

1. Make a *red flower*, by coloring in red *all tiles adjacent* to the 2nd tile from the top in the 2nd column from the left.



2. *Repeat* this *flower* pattern *across the board* to the right, *alternating* yellow and red, leaving a blank column *between every* 2 flowers.

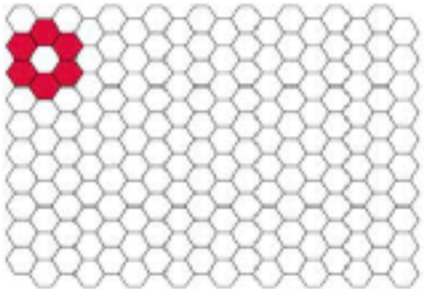


Hexagons, Lachmy et al. 2022

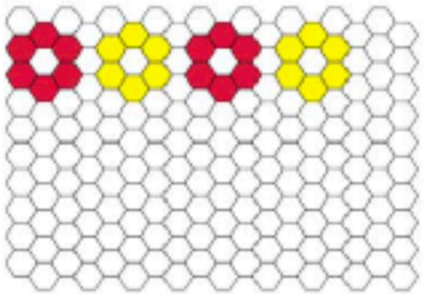
Grounding Language to Action

- How do we define our action space?
- In many cases, language provides a decent set of abstractions that help us define meaningful higher-level action spaces
- Language can also allude to structured action spaces

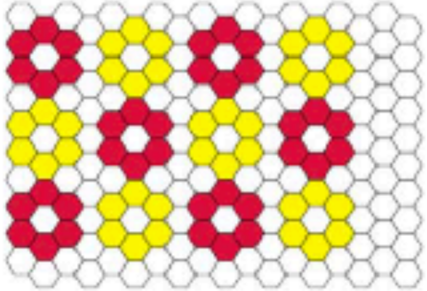
1. Make a *red flower*, by coloring in red *all tiles adjacent* to the 2nd tile from the top in the 2nd column from the left.



2. *Repeat* this *flower* pattern *across the board* to the right, *alternating* yellow and red, leaving a blank column *between every 2* flowers.



3. *Repeat* this *row of flowers* 2 more times, but *reverse* the colors *in each new row*. You should get 6 red flowers and 6 yellow flowers *in total*.



Hexagons, Lachmy et al. 2022

Reasoning about an Interlocutor

- Single instruction following — still could require pragmatic reasoning

Room to Room, Anderson et al. 2018

Leave the bedroom, and enter the kitchen. Walk forward, and take a left at the couch. Stop in front of the window.

Reasoning about an Interlocutor

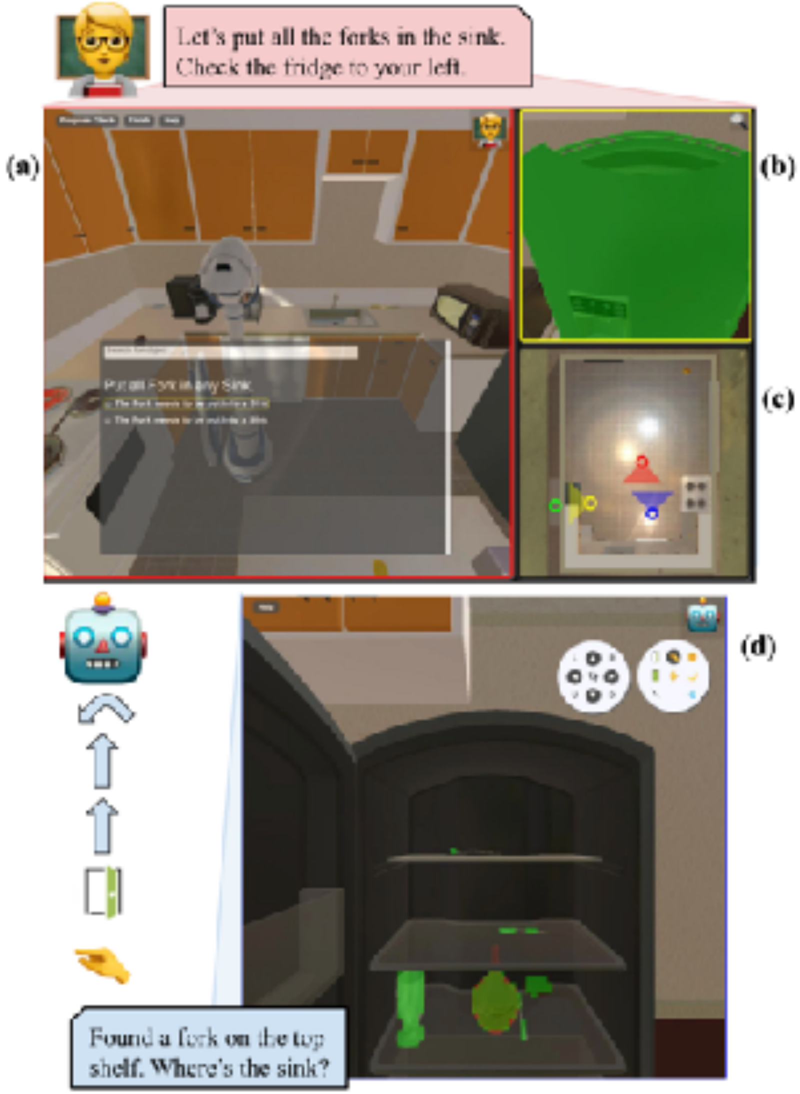
- Single instruction following — still could require pragmatic reasoning
- Following sequences of instructions — user can dynamically instruct the agent according to its current behavior

CerealBar, Suhr et al. 2019

Reasoning about an Interlocutor

- Single instruction following — still could require pragmatic reasoning
- Following sequences of instructions — user can dynamically instruct the agent according to its current behavior
- Bidirectional conversation — agent can ask for clarification or help

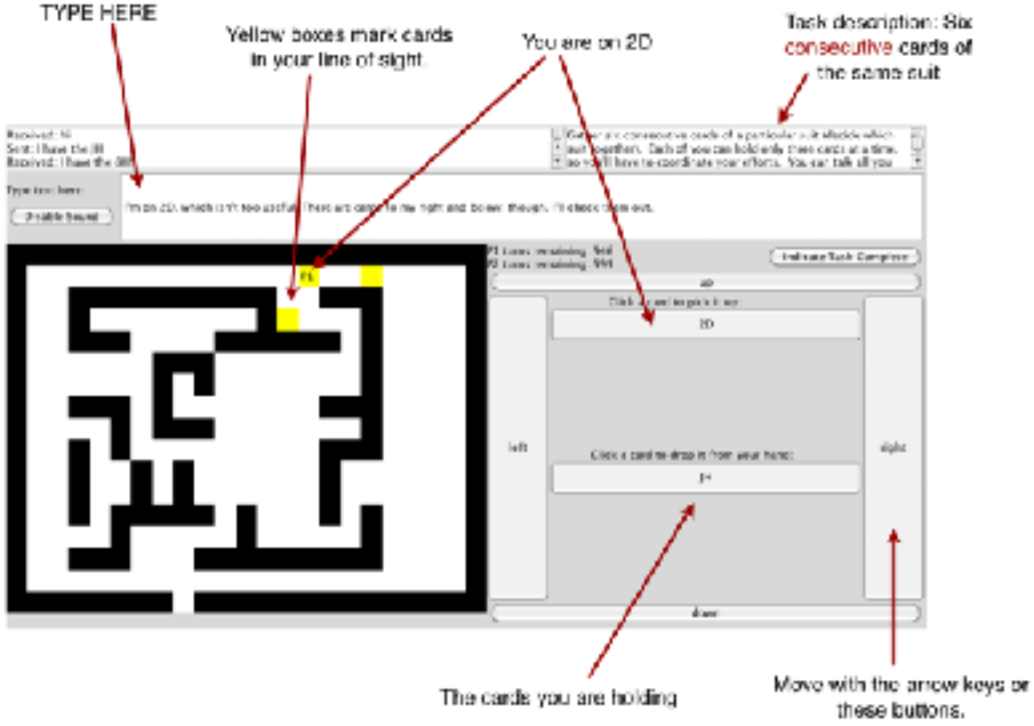
TEACh, Padmakumar et al. 2021



Reasoning about an Interlocutor

- Single instruction following — still could require pragmatic reasoning
- Following sequences of instructions — user can dynamically instruct the agent according to its current behavior
- Bidirectional conversation — agent can ask for clarification or help
- Fully embodied multi-agent conversation — agents can form conventions, negotiate how to solve the task, perform joint planning, etc.

CARDS, Djalali et al. 2011



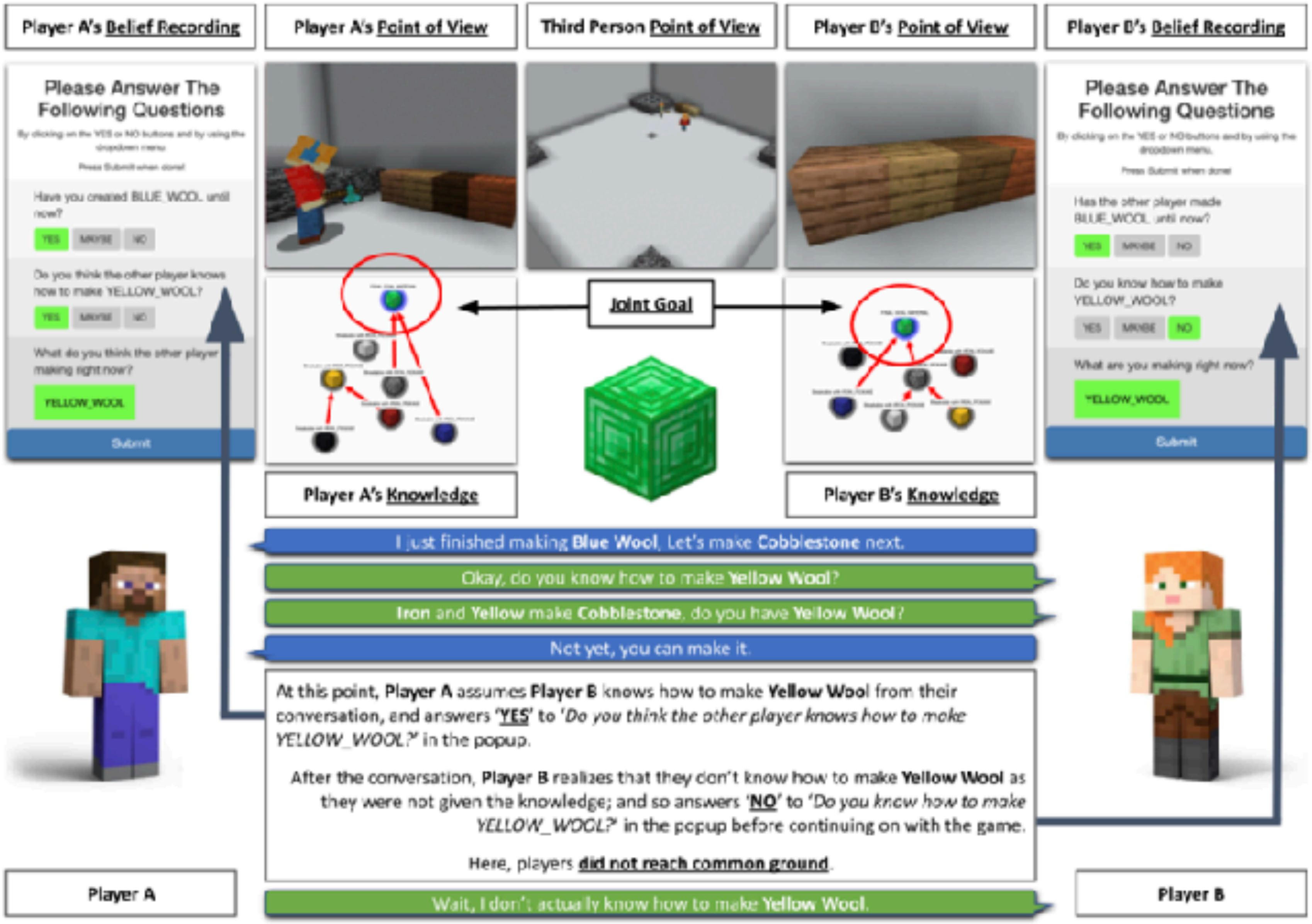
Reasoning about an Interlocutor

- Single instruction following — still could require pragmatic reasoning
- Following sequences of instructions — user can dynamically instruct the agent according to its current behavior
- Bidirectional conversation — agent can ask for clarification or help
- Fully embodied multi-agent conversation — agents can form conventions, negotiate how to solve the task, perform joint planning, etc.

Portal 2 Dialogues

Reasoning about an Interlocutor

- Pragmatic reasoning
- In collaborative tasks: agents need to use language to achieve a shared goal
- Need to model other agent's:
  - Beliefs
  - Goals
  - Observations
  - Knowledge
  - Affordances



MindCraft, Bara et al. 2021

Evaluating Success

- High-level desideratum of language agents: **assist a human user in accomplishing their goal as efficiently as possible.**
- Automatic evaluation
  - Low-level metrics: matching human demonstrations
    - Entire action sequence
    - Action-level accuracy, conditioned on oracle prefix
  - Higher-level metrics: success rate
  - Difficult to define for multi-turn conversation
- Human evaluation
  - When deployed with real users, how effective is the agent?
  - Challenge: human adaptation of expectations, behavior, and language

Learning

- Imitation learning

$$\underbrace{\arg\max_{\theta}}_{\text{Maximum likelihood objective}} \underbrace{\mathbb{E}_{(o,a)\in\mathcal{D}}}_{\text{Expectation over demonstrations}} \underbrace{\pi(a \mid o;\theta)}_{\text{Policy parameterized with }\theta}$$

Maximum likelihood objective

Expectation over demonstrations

Policy parameterized with θ

Essentially supervised learning on a dataset of instructions and observations paired with human demonstrations.

Learning

- Imitation learning
- Reinforcement learning

$$\arg\max_{\theta} \mathbb{E}_{\tau \sim \pi_\theta} \mathcal{R}(\tau)$$

$$a_i \sim \pi_\theta(\cdot \mid s_{i-1})$$
$$s_i \sim \mathcal{T}(\cdot \mid s_{i-1}, a_i)$$

Expectation over trajectories sampled from π

Reward achieved by trajectory

$$\mathcal{R}(\tau) = \sum_{i=0}^{|\tau|} \mathcal{R}(s_i, a_i)\gamma^i$$

Learning

- Imitation learning
- Reinforcement learning
- LLM planning methods

SayCan, Ahn et al. 2022