Interacting with LLMs for Grounded Tasks

Daniel Fried

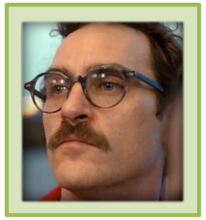
Language Technologies Institute

Carnegie Mellon University

Language Interfaces

Science Fiction

Her, 2013



Let's start with your emails. You have several thousand emails regarding LA Weekly, but it looks like you haven't worked there in many years.

Oh yeah, I guess I was saving those because in some of them I thought I might have written some funny stuff.

Yeah, there are some funny ones. I'd say there are about 86 that we should save. We can delete the rest.

Today

ChatGPT, 2023

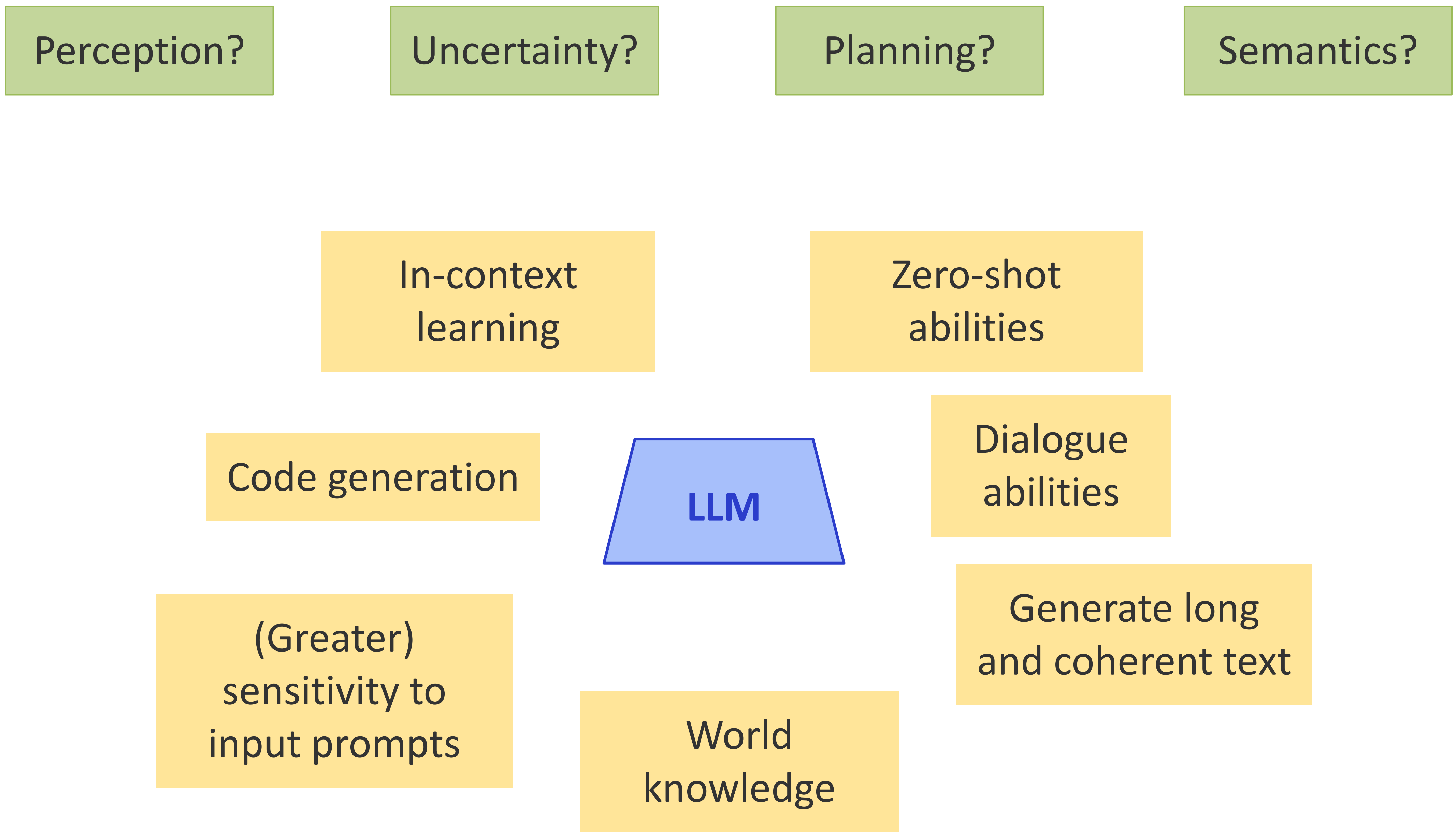
Please help me organize my emails.

Sure! Here are some tips for organizing your emails.

1. Unsubscribe: Reduce the number of unwanted emails by unsubscribing from mailing lists that you no longer need.
2. …

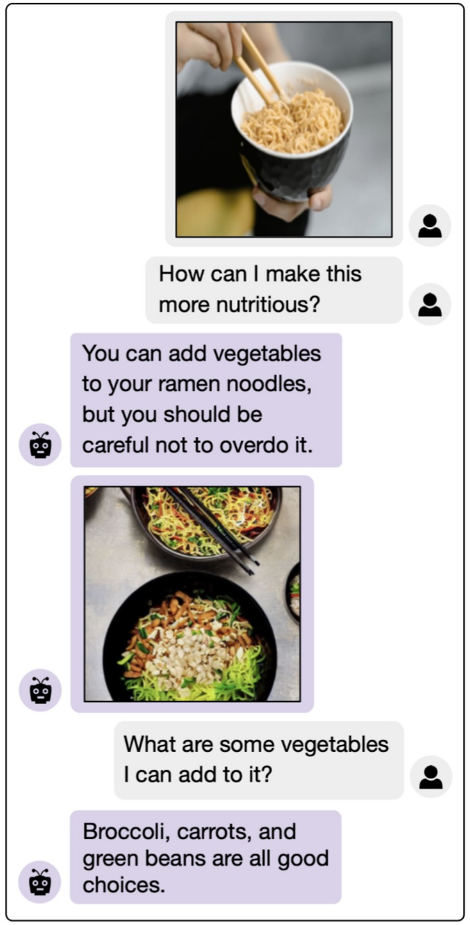
Thanks, but I want you to do it.

I'm sorry, I am a text-based AI language model and do not have access to your email or the ability to physically sort them for you.


Perception?
Uncertainty?
Planning?
Semantics?
In-context learning
Zero-shot abilities
Code generation
LLM
Dialogue abilities
Generate long and coherent text
(Greater) sensitivity to input prompts
World knowledge

Interacting with LLMs for...

Multimodal dialogue



Question-based retrieval

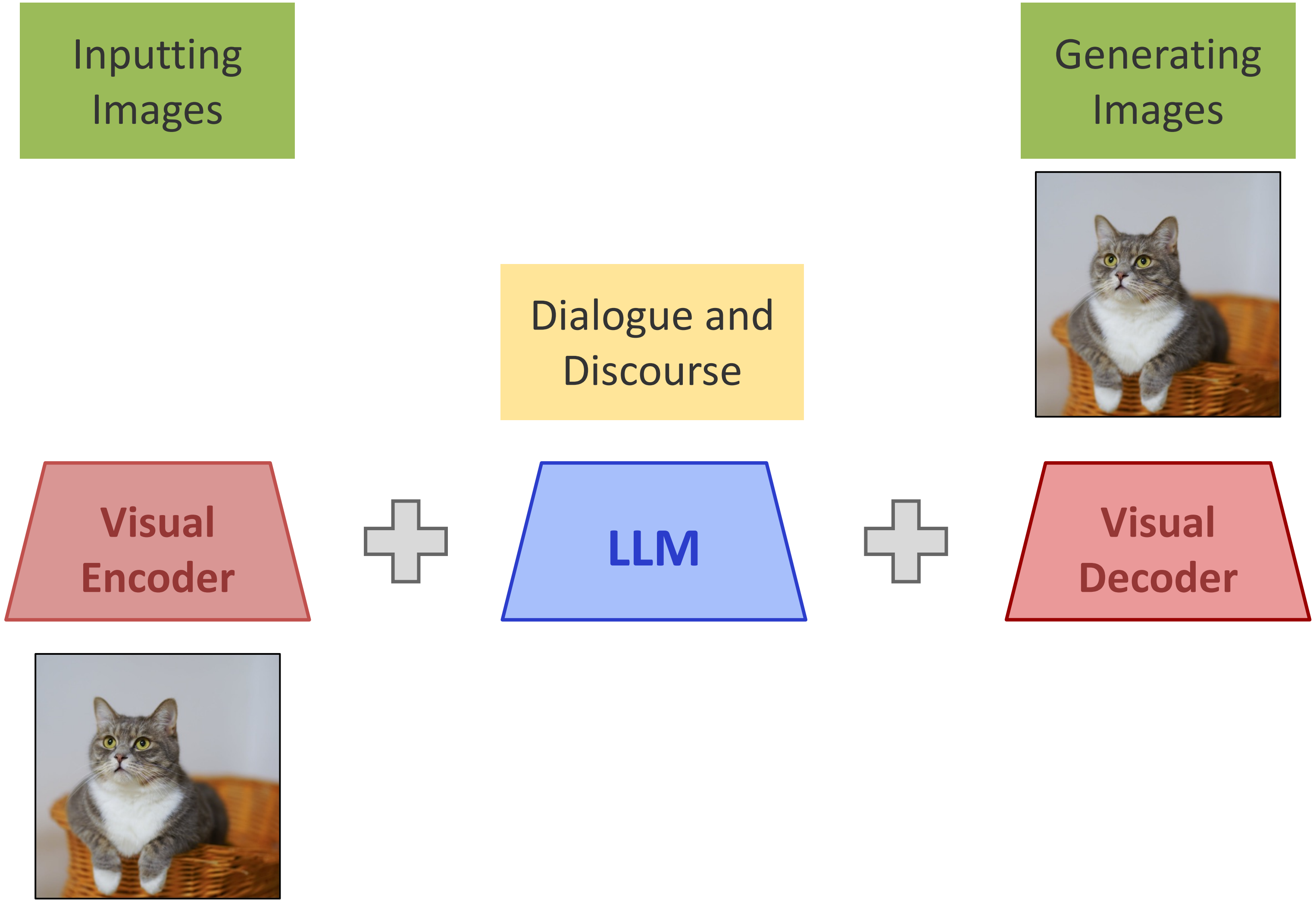
"what is next to the computer?"

Referential tasks

"big light dot next to dark dot"

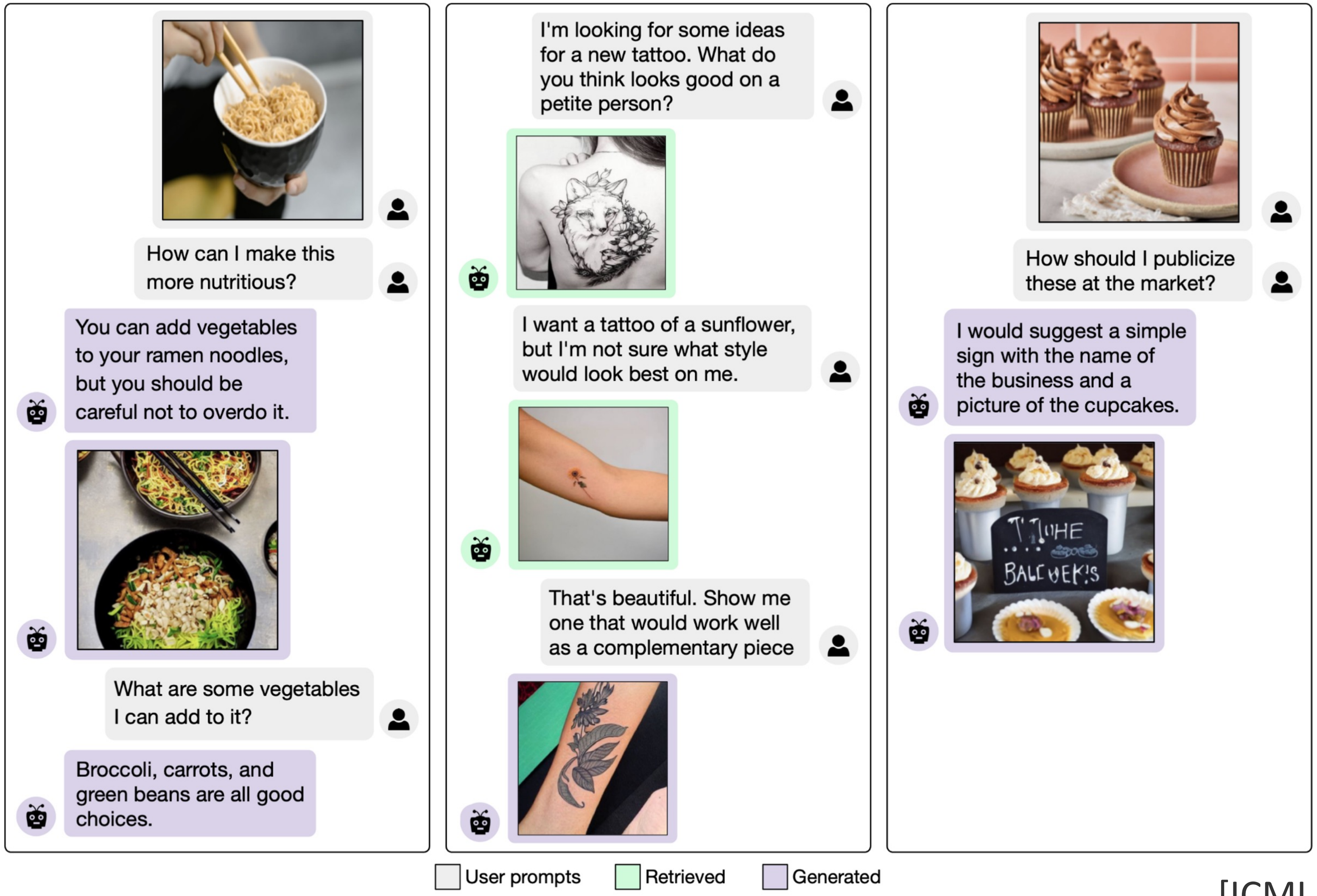
```
def is_light(n, ctx):
    return ctx[x, -1] > 0.3

def are_close(x, y, ctx):
    dist = np.linalg.norm(
        ctx[x,:-2]-ctx[y,:-2]
    )
    return dist < 0.3
```

Multimodal Dialogue



GILL Generating Images with Large Language Models

Parameter-efficient fusion of existing LLMs and image models; trainable in ~4 GPU days.



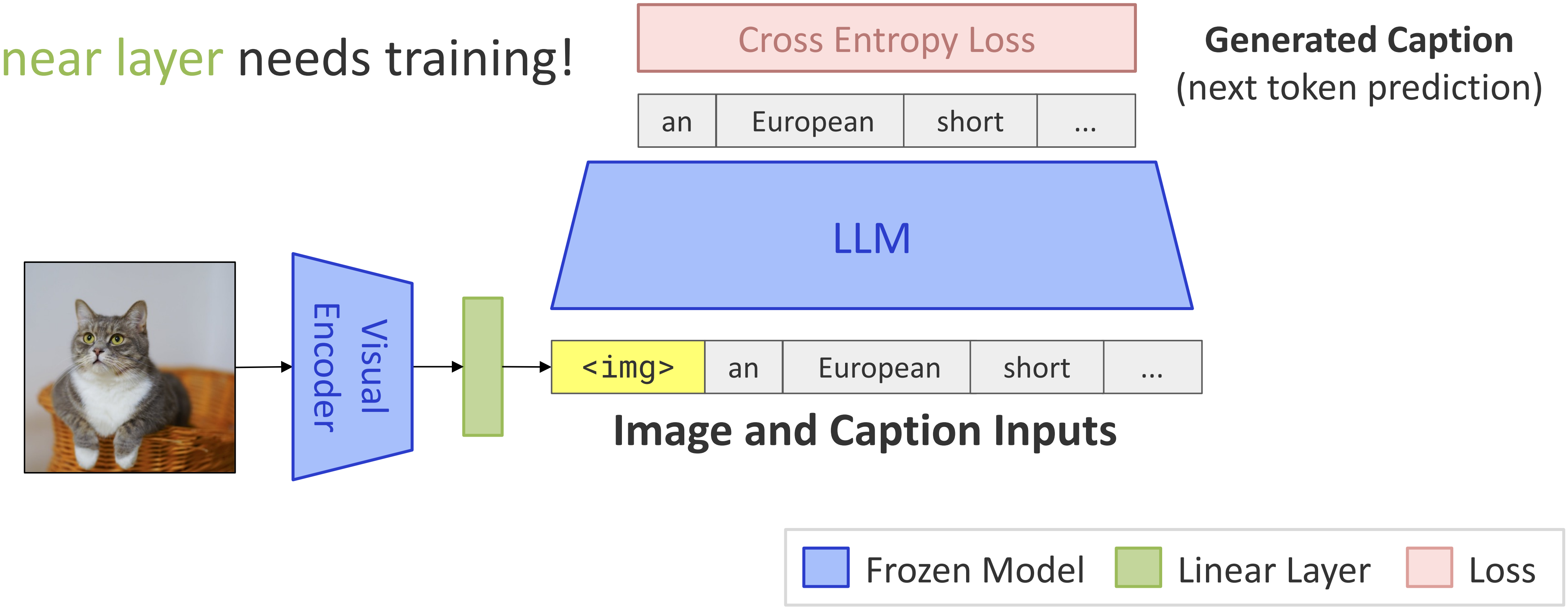
Jing Yu Koh

[ICML 2023], [NeurIPS 2023]

Images as Inputs

Align ***input*** representations of an LLM (OPT, Llama2) and ***visual encoder outputs (CLIP)*** on image captions

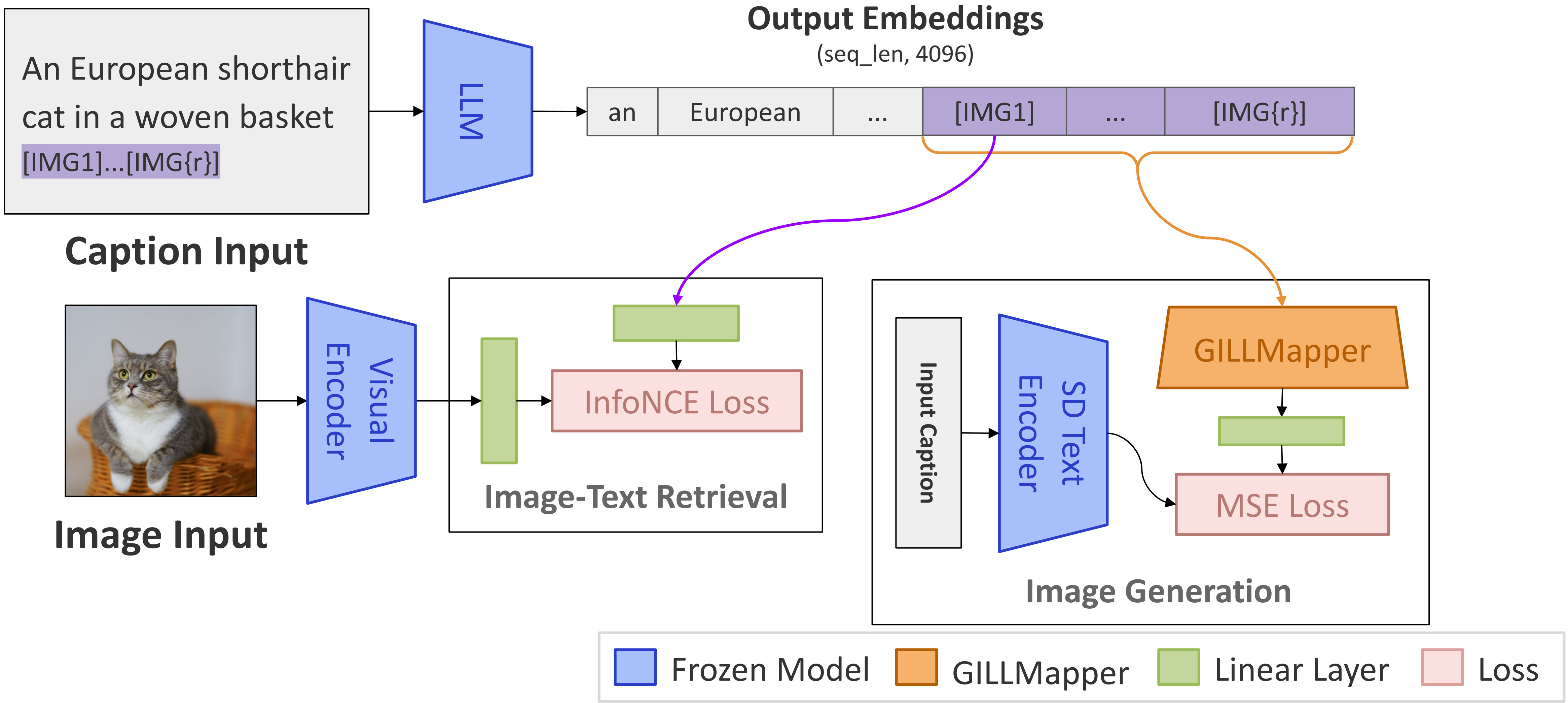
Only the linear layer needs training!



[Frozen, Tsimpoukelli et al., 2021; Limber, Merullo et al., 2023]

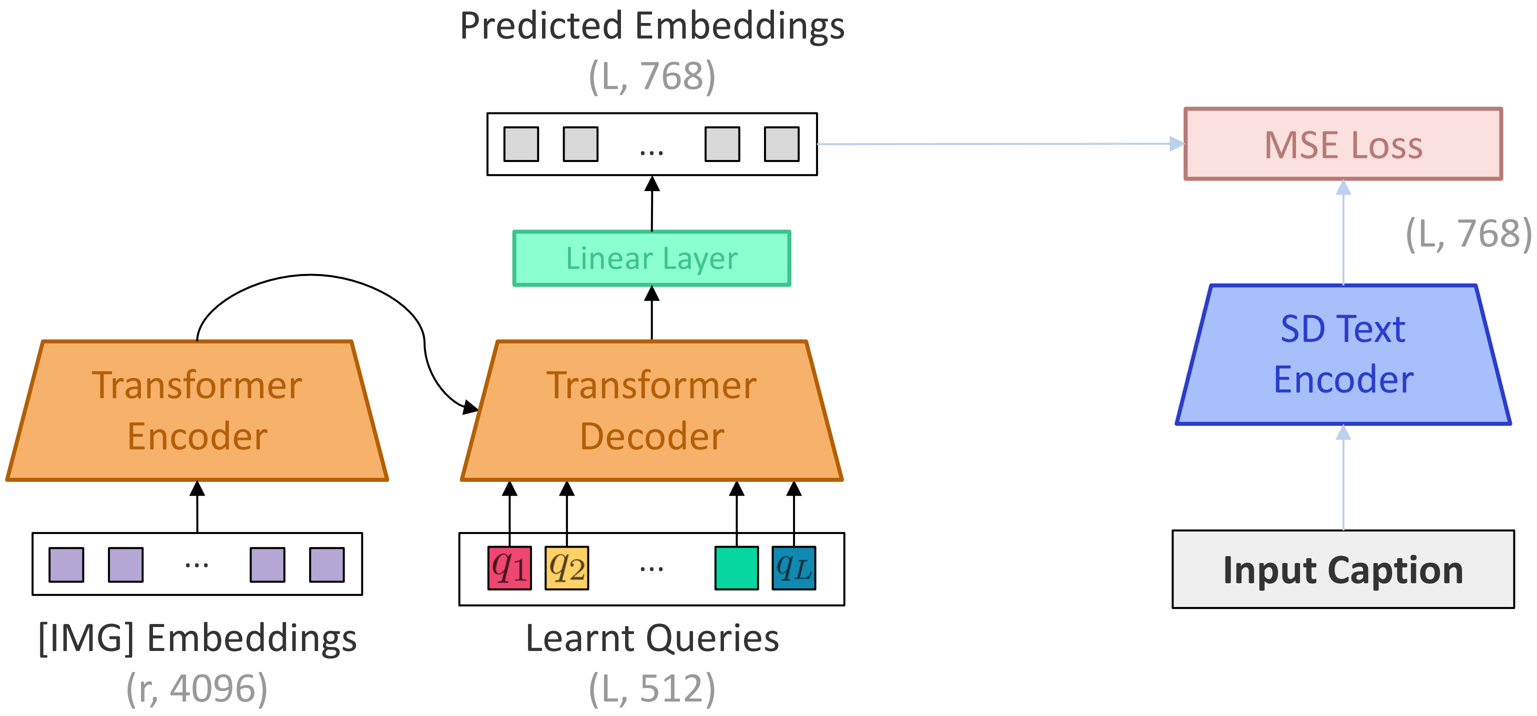
Images as Outputs

Align ***output*** representations of an LLM (OPT, Llama2) and ***visual models (CLIP, Stable Diffusion)*** on image captions



GILLMapper: An Improved LLM-to-Generator Map

- Previous approaches use <u>linear mappings</u> between LLMs and visual models
- This is insufficient for image generation: decoders require <u>dense</u> information



Multimodal Few-Shot Learning with Frozen Language Models ([Tsimpoukelli et al., 2021](#))
Linearly Mapping from Image to Text Space ([Merullo et al., 2023](#))
Grounding Language Models to Images for Multimodal Inputs and Outputs ([Koh et al., 2023](#))

Evaluation: Contextual Image Generation

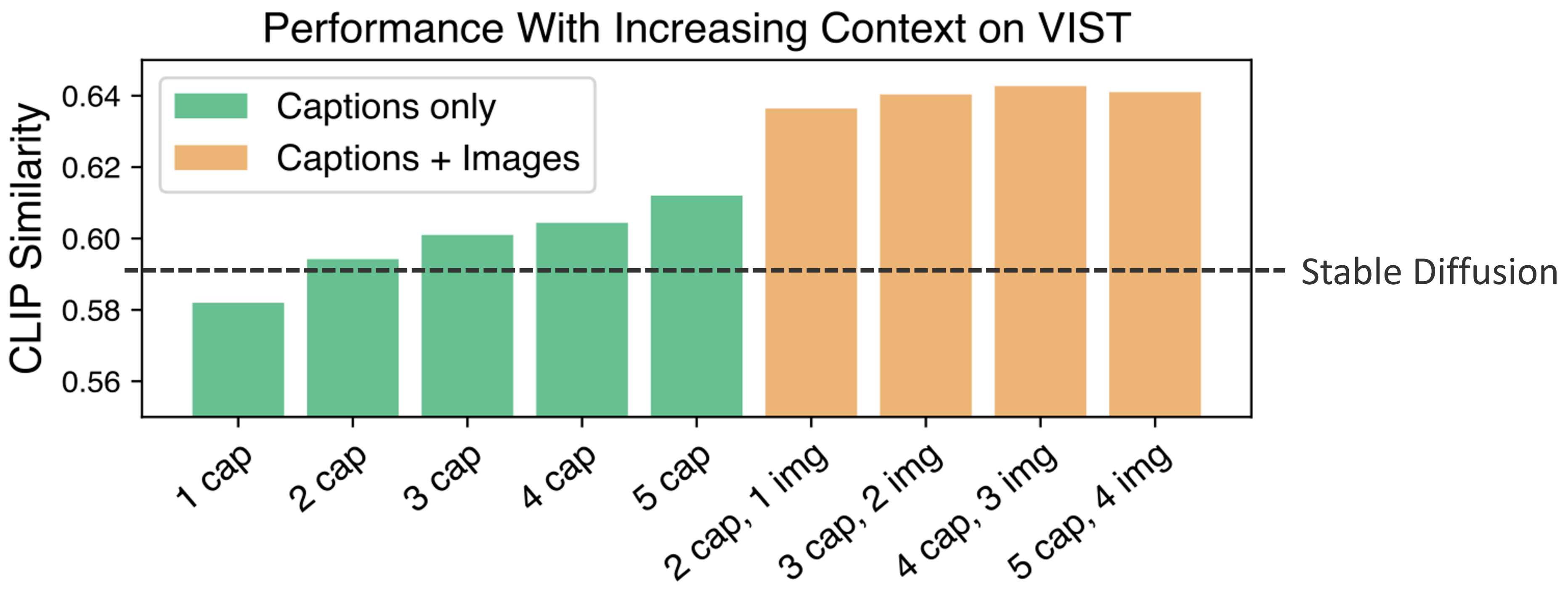
- Given a Visual Story, generate a relevant image
- Need to condition on long, temporally dependent text
- (Optionally) Condition on image inputs interleaved within the text

Visual Storytelling (Huang et al., 2016)

The Effect of Context

Multi-modal context is **worth more** than
uni-modal context, producing more relevant generation results.

Our model distills from Stable Diffusion, but outperforms it with multi-modal context.



GILLMapper: A Stronger LLM-to-Generator Mapping

Image generators require **denser** input sequences. Linear mappings are insufficient.

	CC3M	VIST
Model	**FID** (↓)	**CLIP Sim** (↑)
Stable Diffusion [43]	**13.94**	0.598
Ours + Linear	15.50	0.500
Ours + 3-layer MLP	15.33	0.502
Ours + Transformer Encoder	16.30	0.605
Ours + GILLMapper	15.31	**0.641**

Qualitative Examples

Stable Diffusion

Ours

"A dignified beaver wearing glasses, a vest, and colorful neck tie. He stands next to a tall stack of books in a library."

Stable Diffusion

Ours

"Snow mountain and tree reflection in the lake"

Stable Diffusion

Ours

"A drop-top sports car coming around a bend in the road"

Stable Diffusion

Ours

"a group of penguins in a snowstorm"

Qualitative Examples

I want to paint a landscape with mountains and a river. Can you help me find some inspiration?

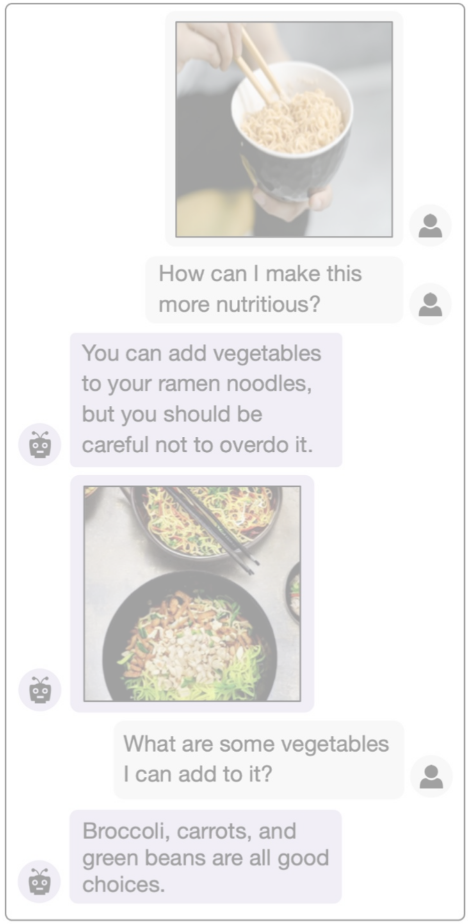
I’m not sure

Can you show me some with horses?

What about something more abstract?

Interacting with LLMs for...

Multimodal dialogue



Question-based retrieval

"what is next to the computer?"

Referential tasks

"big light dot next to dark dot"

```
def is_light(n, ctx):
    return ctx[x, -1] > 0.3

def are_close(x, y, ctx):
    dist = np.linalg.norm(
        ctx[x,:-2]-ctx[y,:-2]
    )
    return dist < 0.3
```

Question-Based Retrieval

Tracking User Goals

Reasoning About Informativity

Visual QA Model

Bayesian Inference

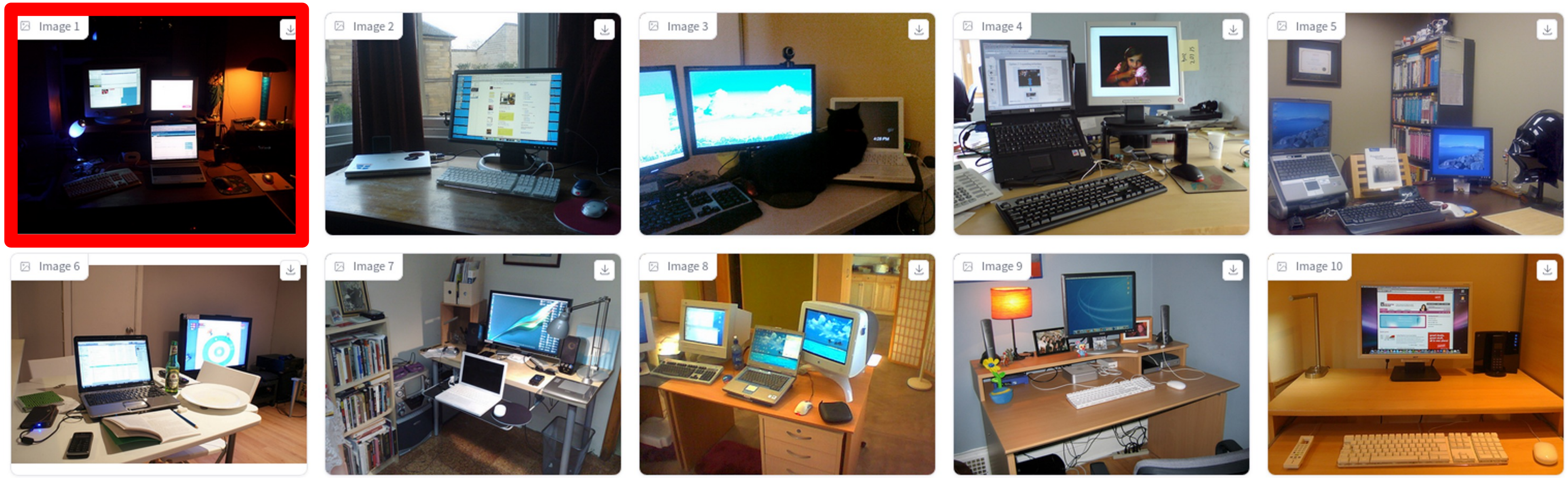
Generating Questions

What is next to the computer?

LLM + Visual QA Model

Information Gain Planning

Question-Based Retrieval



🤖 : What is next to the computers?

👨 : Lamp

🤖 : How many computers are there?

👨 : 3

🤖 : Guess – **Image 1**

[Open-domain clarification question generation without question examples. White et al., EMNLP 2021]

Sedrick Keh

What Makes an Informative Question?

Is there a computer in the image? ✗

Is there a cat in the image? ✗

What is next to the computer? ✓

[*Asking more informative questions for grounded retrieval.* Keh et al., arXiv, 2023]

What Makes an Informative Question?

What is next to the computer?

What Makes an Informative Question?

Images, i

Belief state: $P(i)$

Question, q

What is next to the computer?

Answers, a

"lamp"

Visual QA Model

$P(a \mid q, i)$

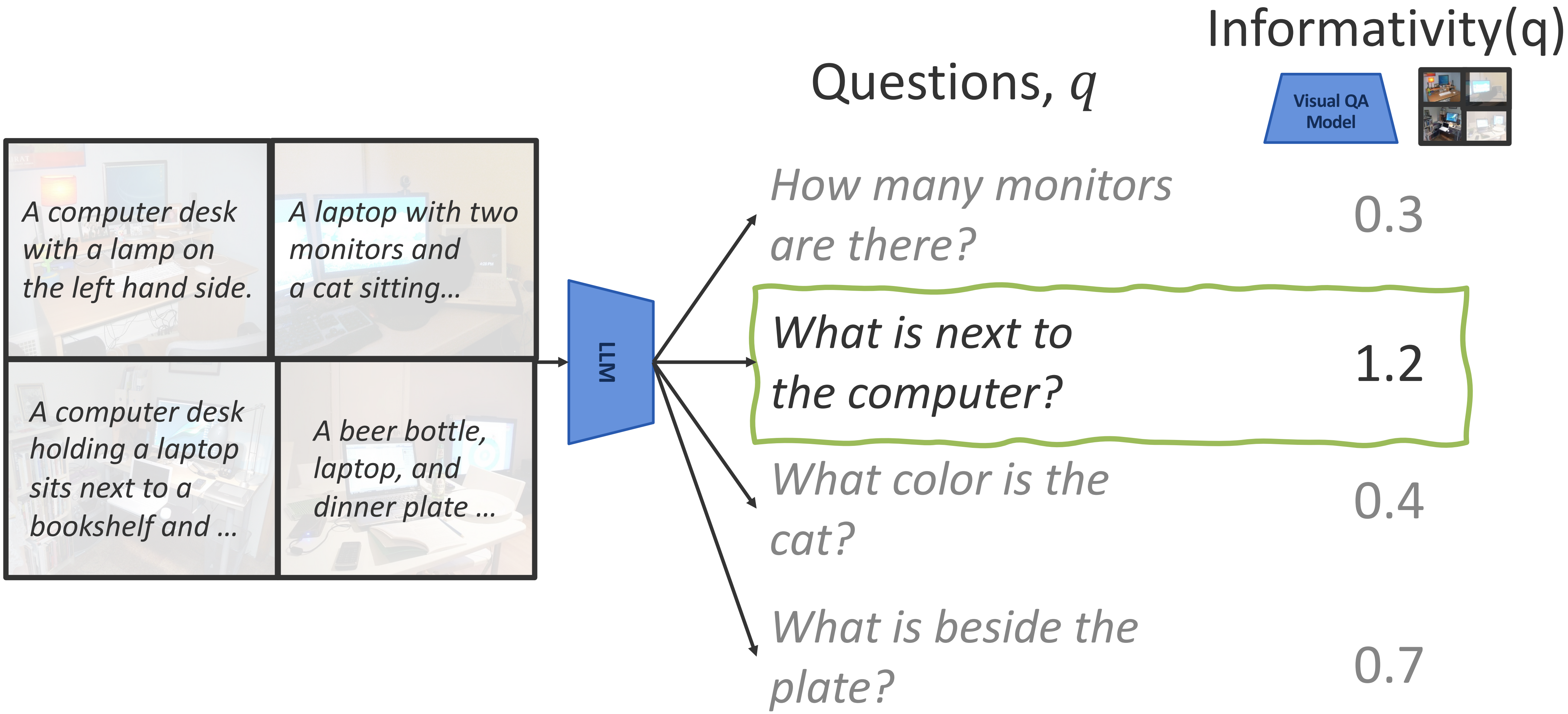
$P(i \mid q, a)$

(by Bayes' rule)

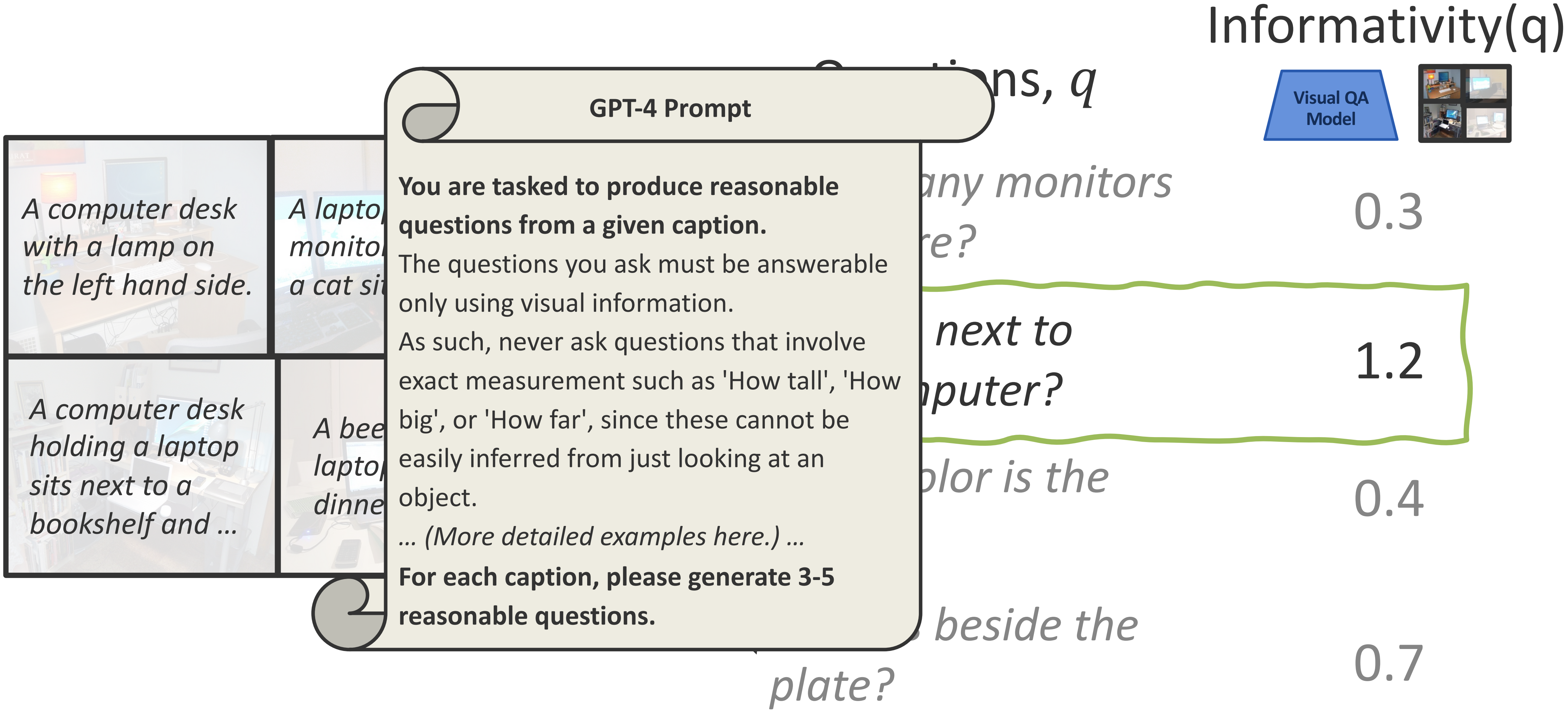
"cat"

$$\text{Informativity}(q) = H[i] - \mathbb{E}_{p(a \mid q)} H[i \mid q, a]$$

Generating Informative Questions



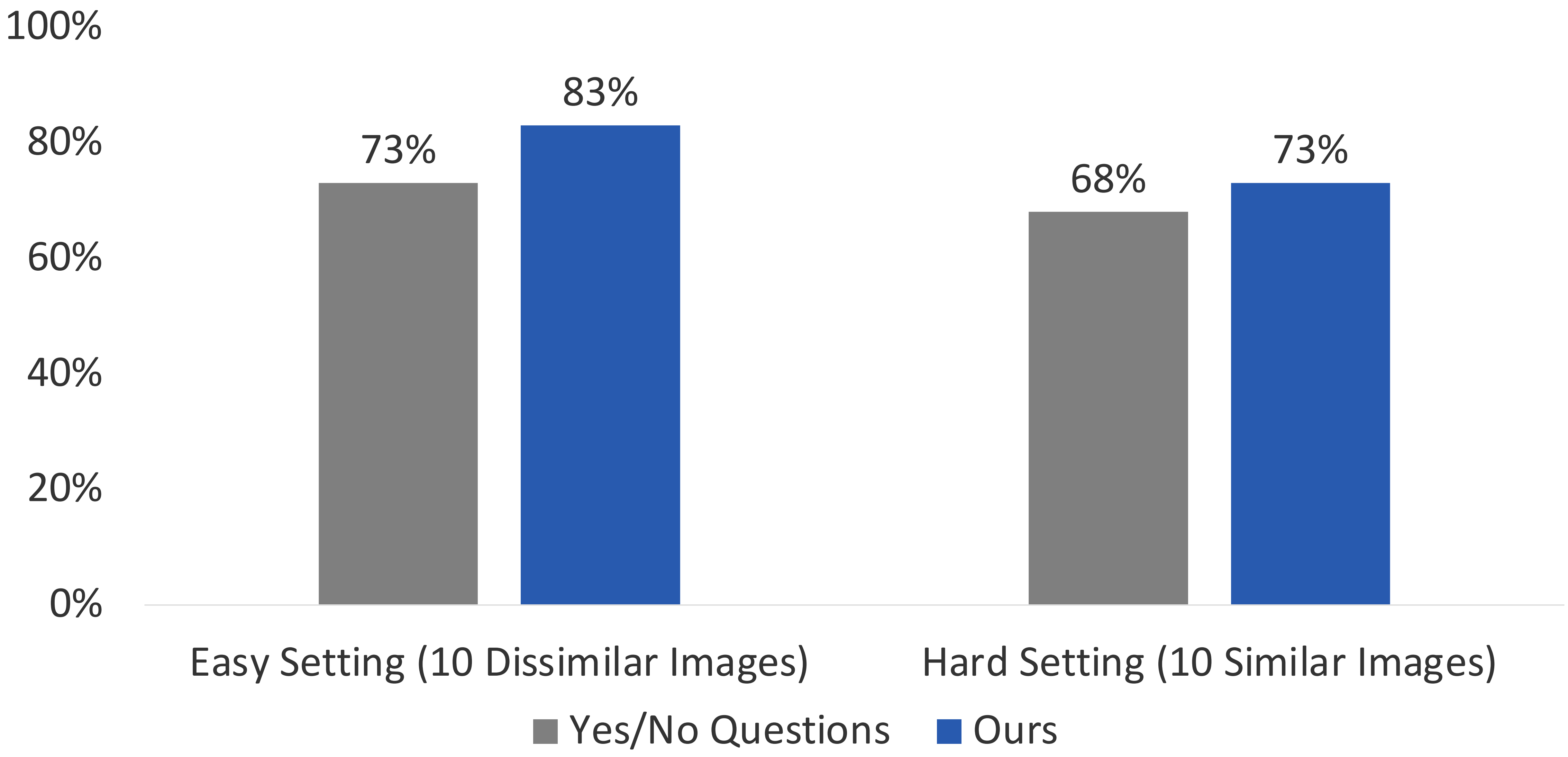
Generating Informative Questions



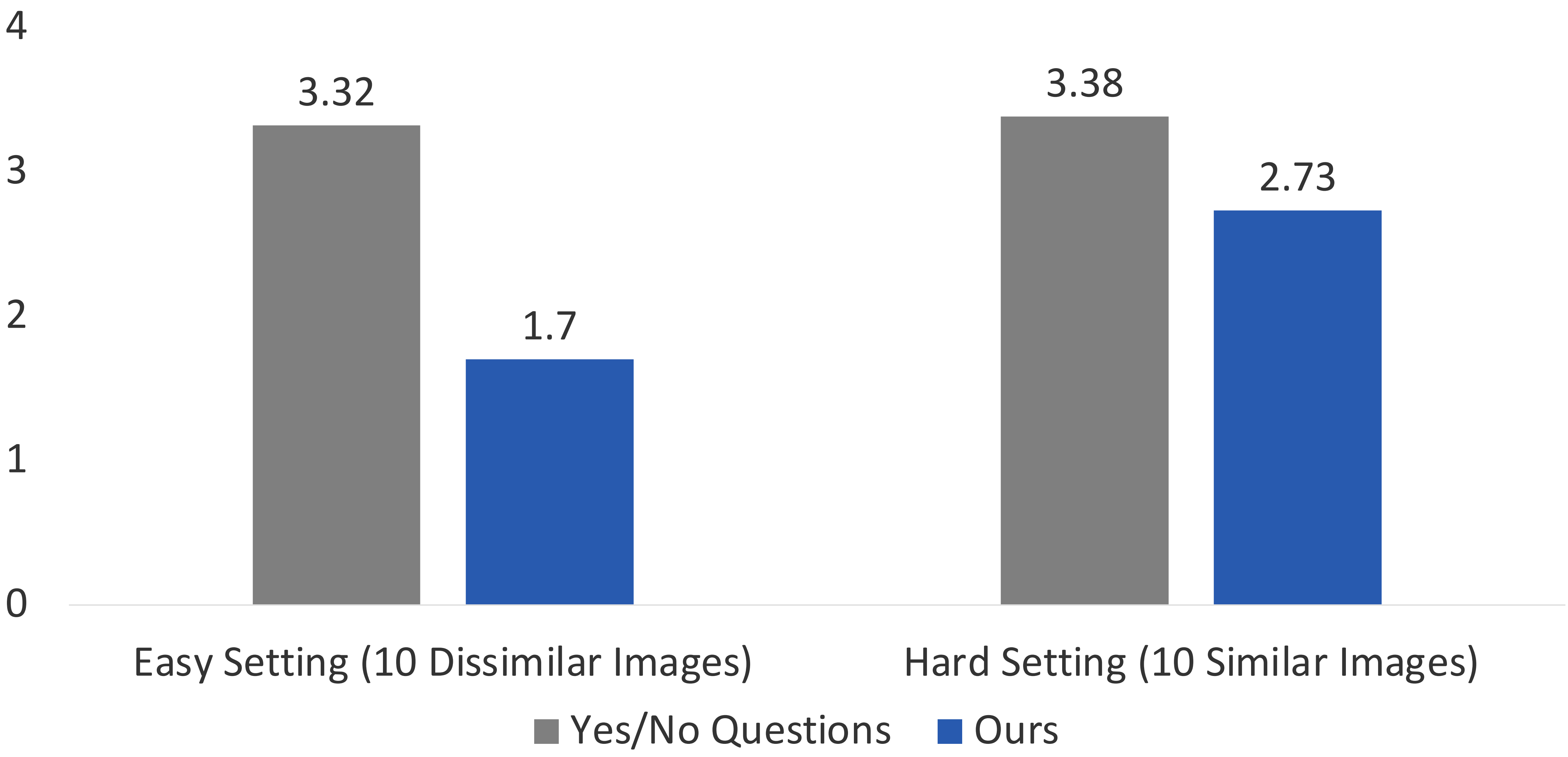
Experiments

- Play with people using sets of 10 MSCOCO images.
- Compare against a Yes/No question method from past work [White et al. 2021].
- Evaluate accuracy and number of questions asked.
- Also need to avoid presupposition errors in the VQA models – see the paper!

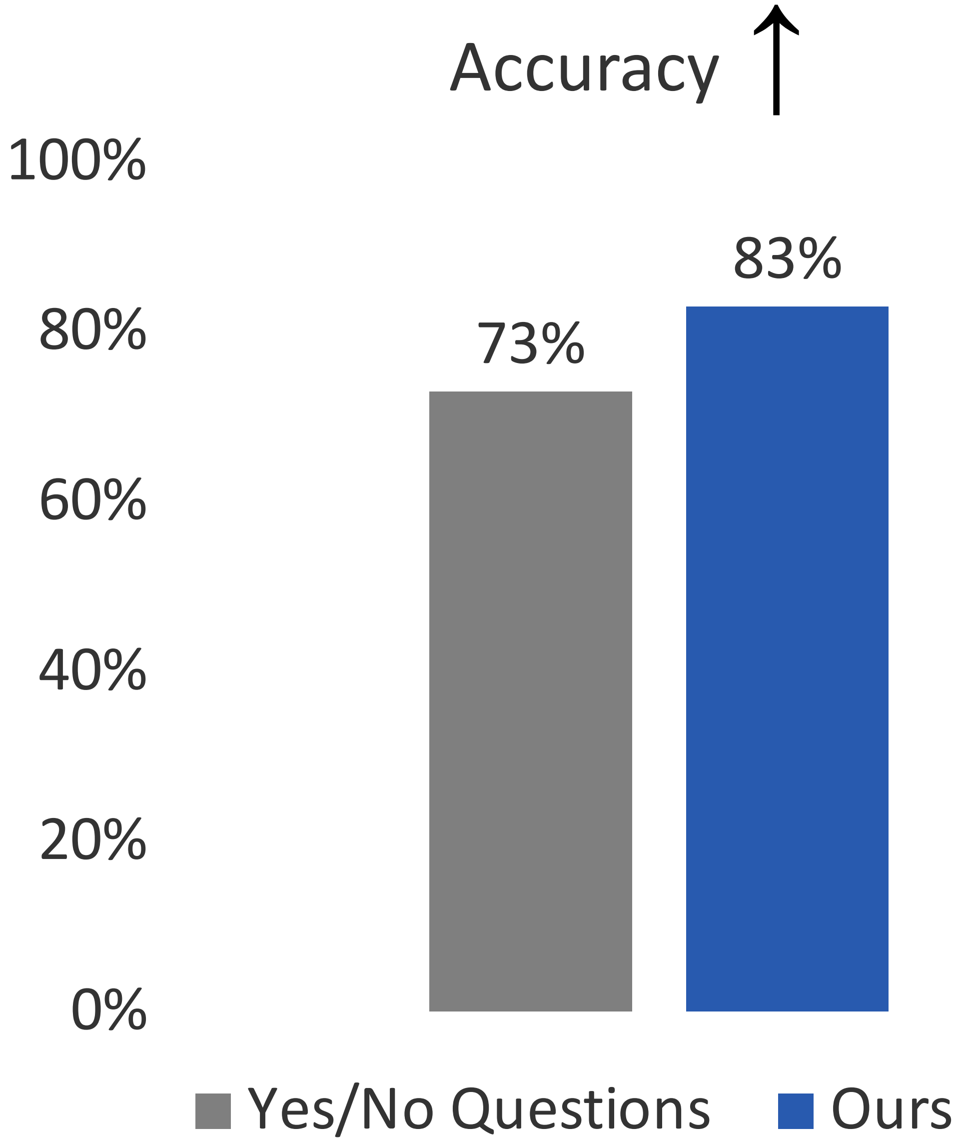
Results: Accuracy ↑

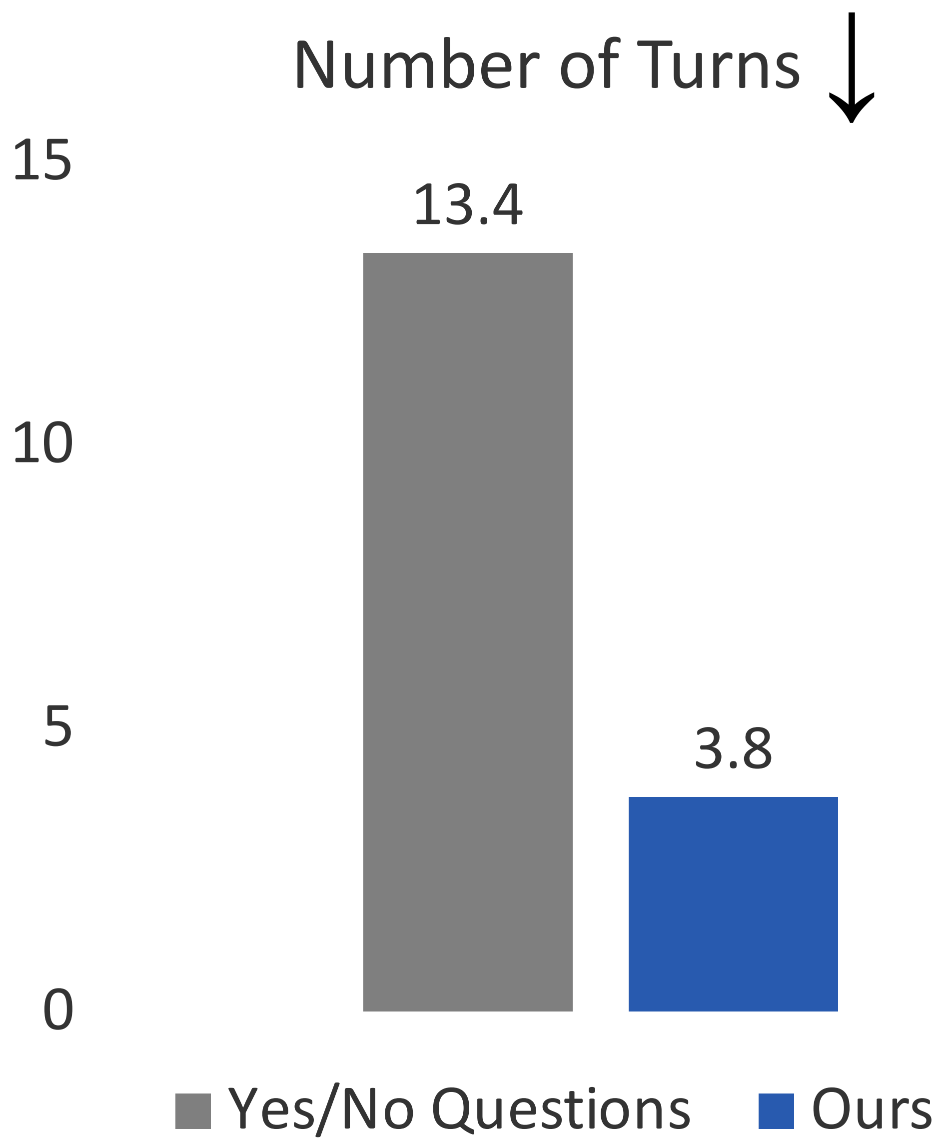


Results: Number of Questions ↓



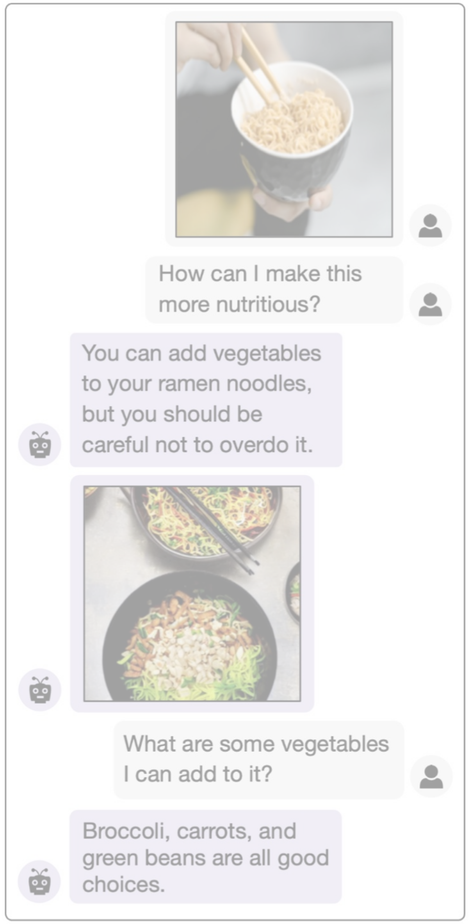
With 100 Images (Automatic Evals)





Interacting with LLMs for…

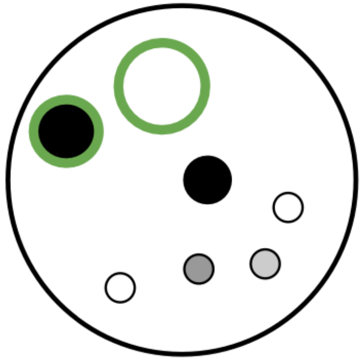
Multimodal dialogue



Question-based retrieval

"what is next to the computer?"

Referential tasks

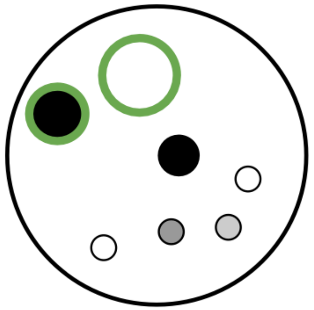


"big light dot next to dark dot"

```
def is_light(n, ctx):
    return ctx[x, -1] > 0.3

def are_close(x, y, ctx):
    dist = np.linalg.norm(
        ctx[x,:-2]-ctx[y,:-2]
    )
    return dist < 0.3
```

Referential Dialogue and QA

Semantic Parsing



Code Generation

```
is_big(x) and is_light(x)
and is_dark(y) and
are_close(x, y)
```

LLM

+

Python Interpreter

”big light dot next to the dark dot”

```
def is_light(n, ctx):
    return ctx[x, -1] > 0.3

def are_close(x, y, ctx):
    dist = np.linalg.norm(
        ctx[x,:-2]-ctx[y,:-2]
    )
    return dist < 0.3
```

Yihan Cao Shuyi Chen Ryan Liu Zora Wang

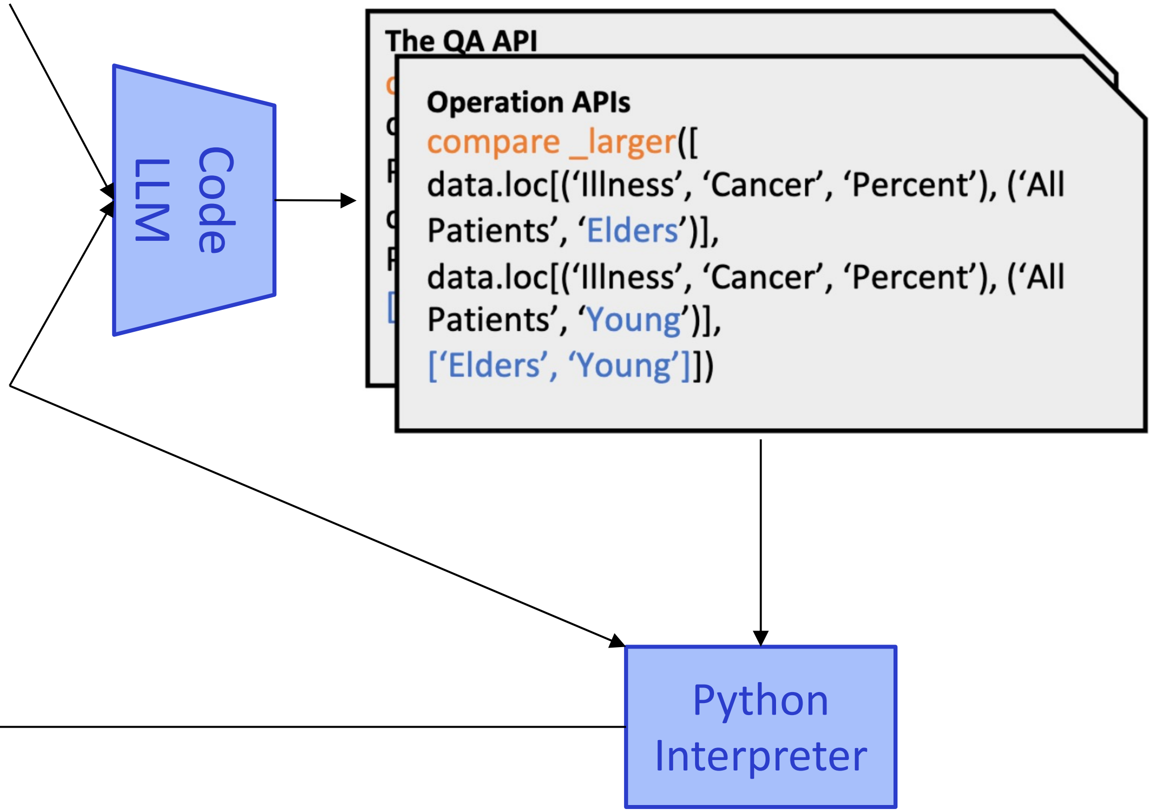
Code for Table QA

Question: Who is more likely to have cancer, *the elder* or *the young*?

Table:

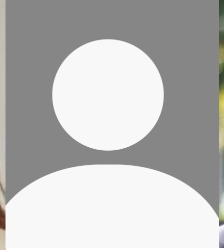
Title: Number and percentage of people who are interviewed who have or have had illnesses.

Illnese	Cold		Cancer	
	total	percent	total	percent
All patients	10,000	2.5%	200	0.3%
Elders	7,400	3.5%	126	0.9%
Young	2,600	1.5%	74	0.2%



Answer: Elder

[API-Assisted Code Generation for Question Answering on Varied Table Structures. Cao et al., EMNLP 2023]



Yihan Cao Shuyi Chen Ryan Liu Zora Wang

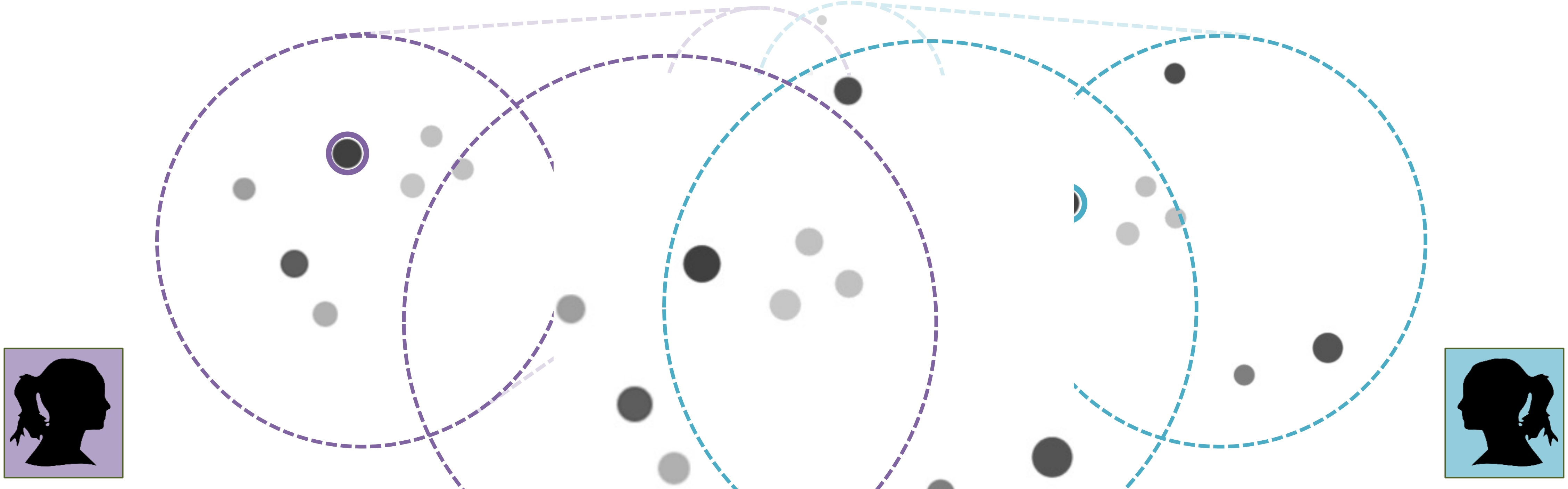
Code for Table QA

Python gives a unified representation across varied table formats and datasets

Dataset	**HiTab**	**Spider**	**AIT-QA**	**WikiTQ**
Baseline	MAPO 40.7	DIN-SQL 61.5	RCI 51.8	BINDER **54.8**
Codex	59.6	61.2	77.8	41.7
w/ API (Ours)	**69.3**	**63.8**	**78.0**	42.4

[*API-Assisted Code Generation for Question Answering on Varied Table Structures.* Cao et al., EMNLP 2023]

Grounded Collaborative Dialogue



A: I have three dots in a line with a dark one in the center.

B: I don’t have that. Do you have a cluster of three grey dots in a triangle?

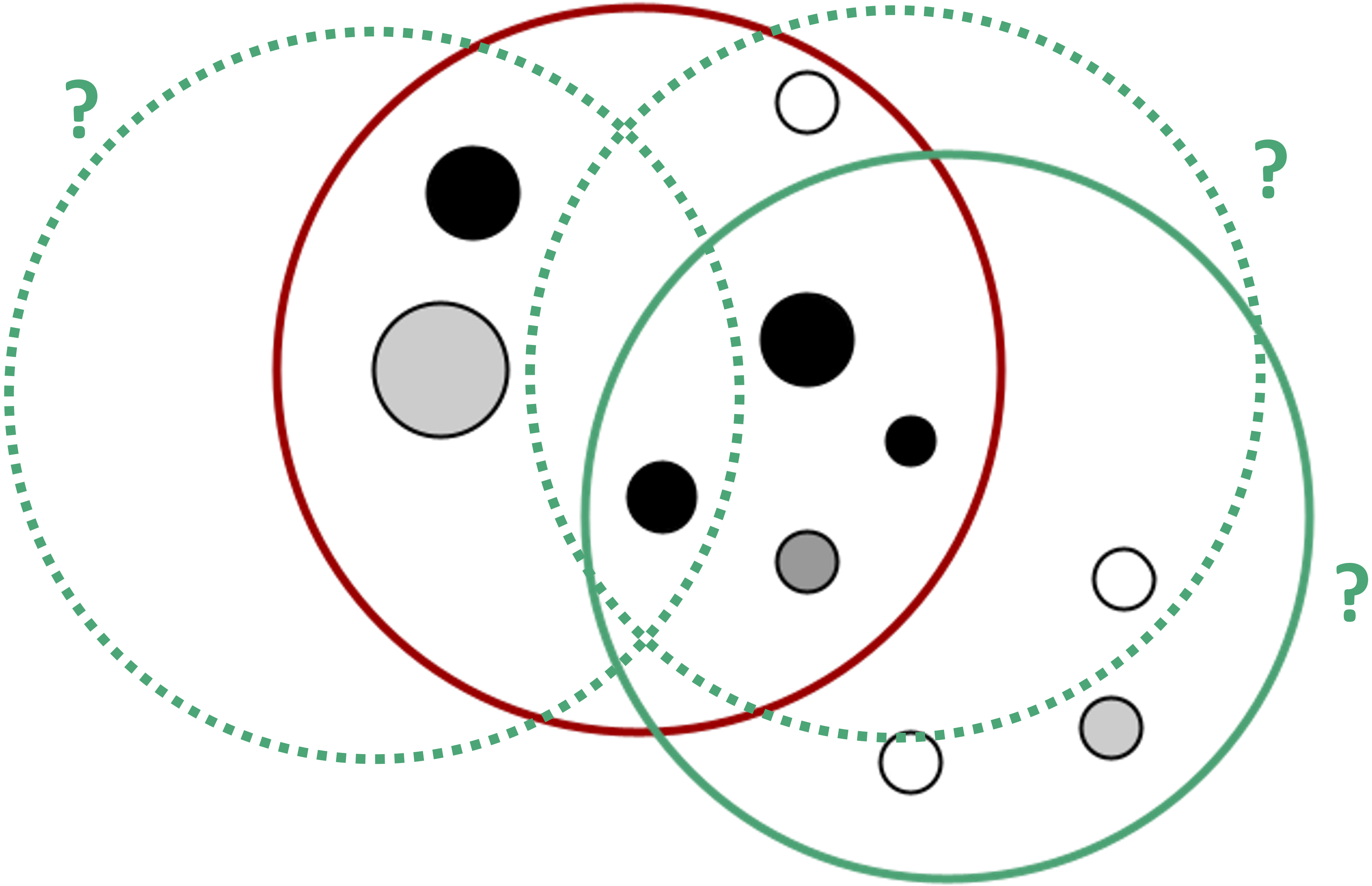
A: Is there a large black dot to the left of the three grey dots?

B: Yes, let’s select the black one.

OneCommon [Udagawa and Aizawa, 2019 & 2020]

Justin Chiu

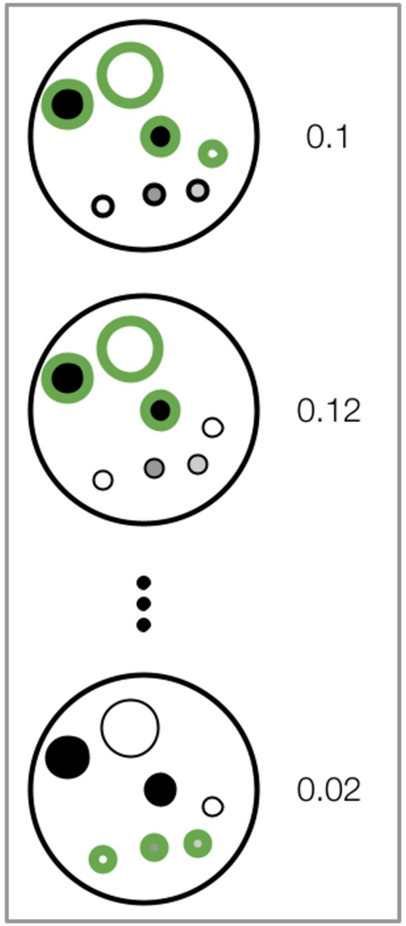
Beliefs About What's In Common



[*Symbolic Planning and Code Generation for Grounded Dialogue*. Chiu et al., EMNLP 2023]

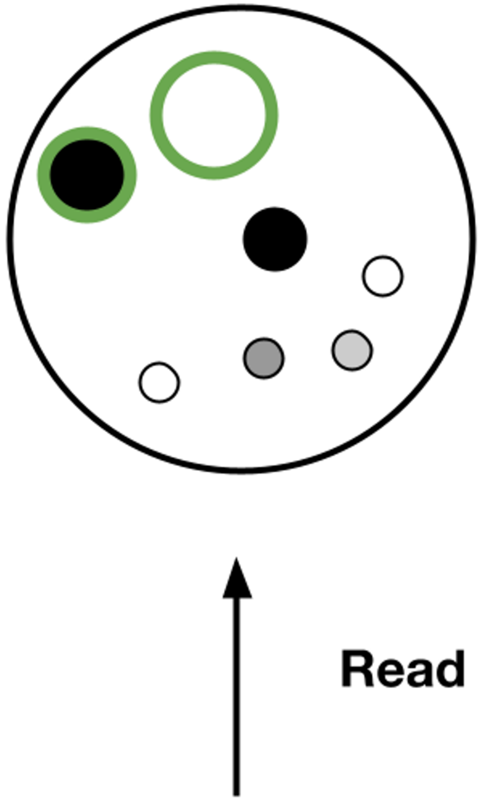
Method Overview

Previous belief state **p(z)**



Reading

Dots mentioned **p(x|u)**

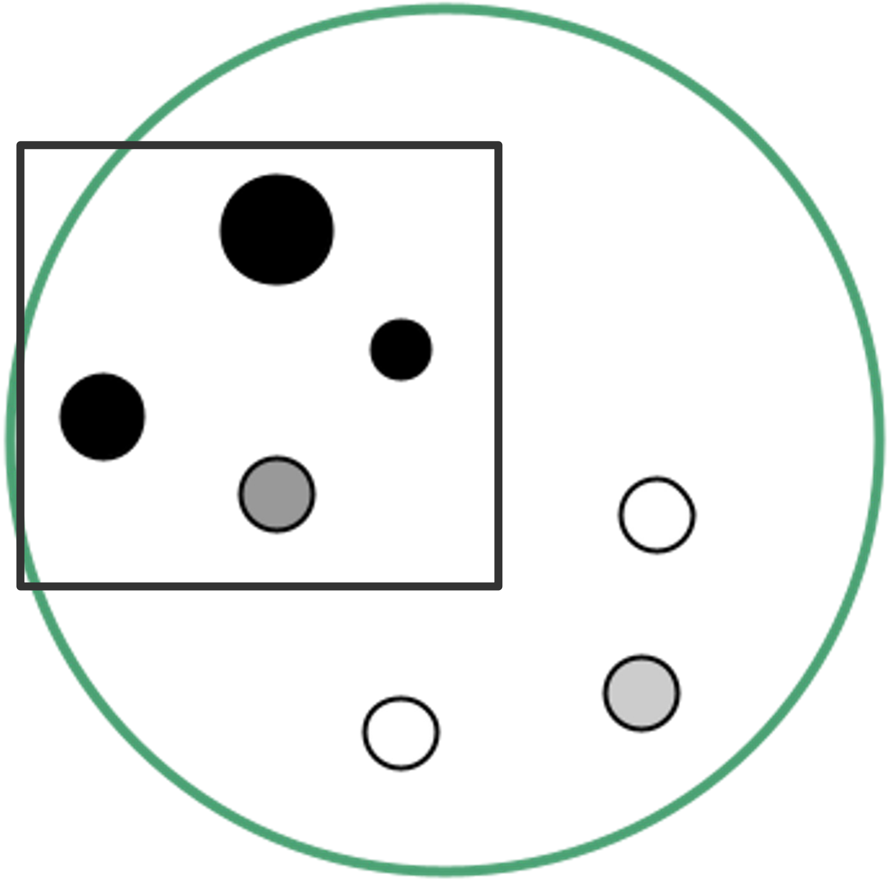


Partner utterance **u**: "*Is there a big light dot next to a big dark one?*"

Reading via a Code LLM

```
from perceptual_library import is_small, …
dot1, dot2, dot3, … = get_dots()
```

,

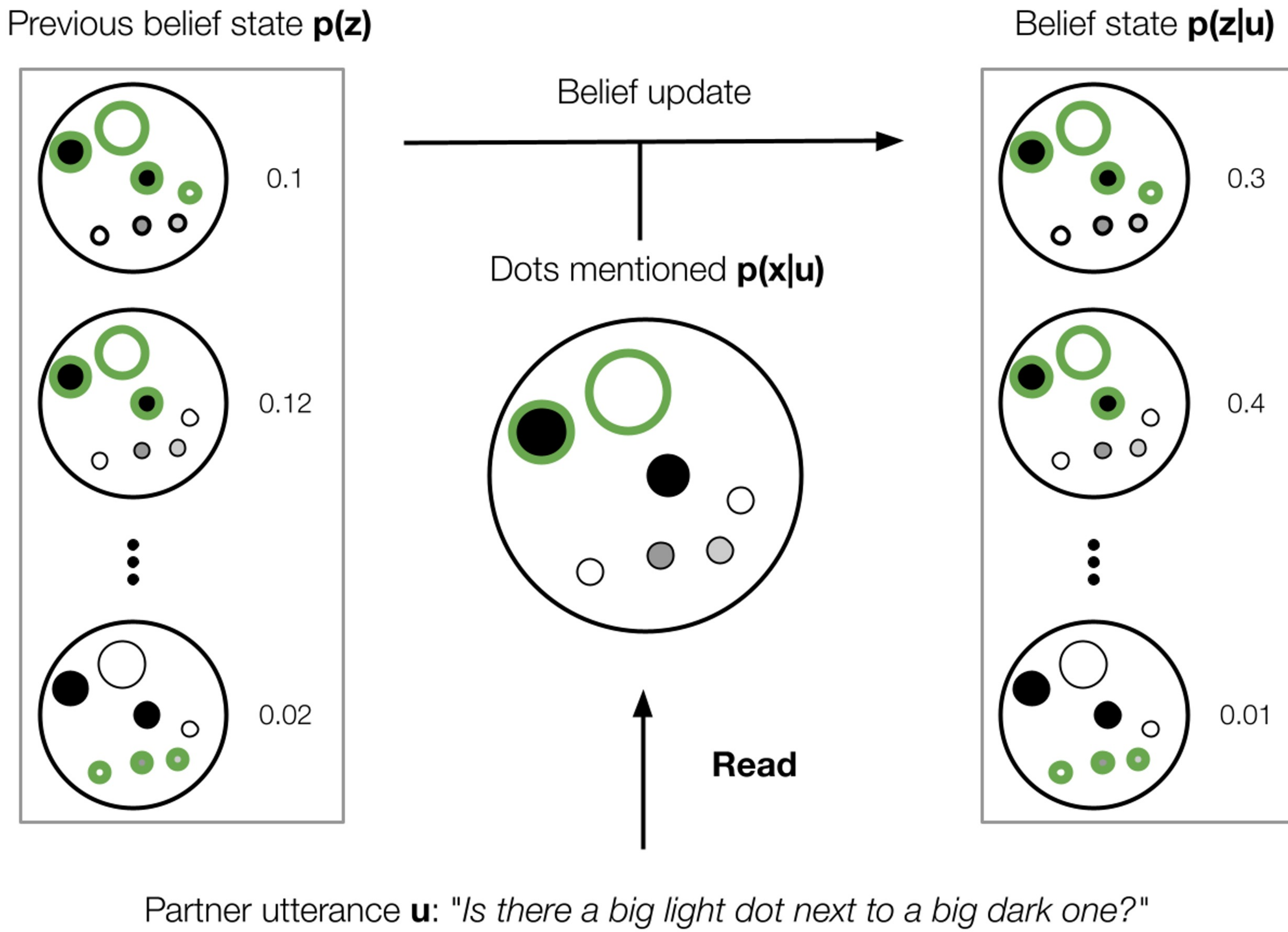


Grounding function library

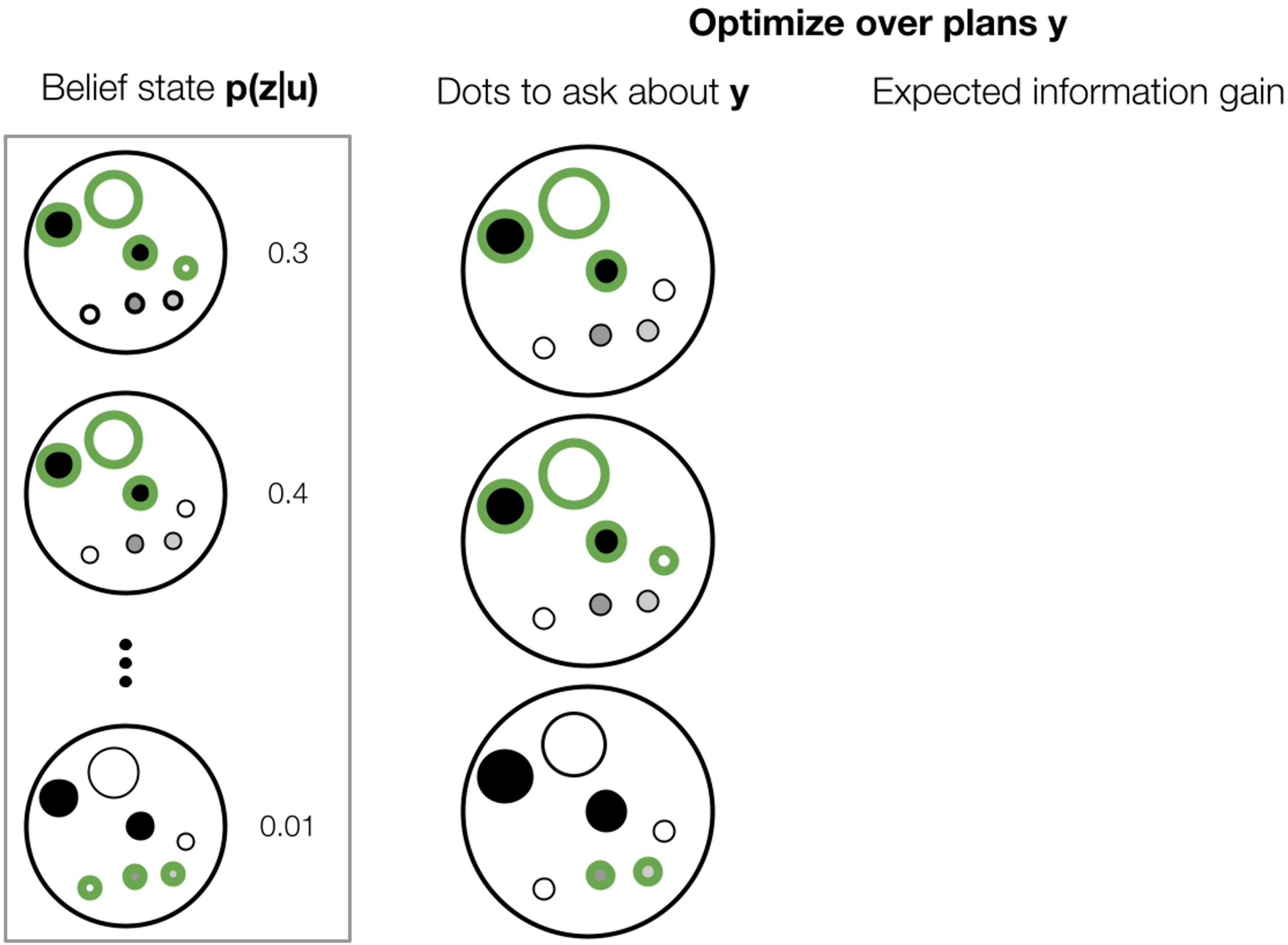
- Functions are predicates over dots
- Manually designed for OneCommon

```
def is_light(n, ctx):
    return ctx[x, -1] > 0.3

def are_close(x, y, ctx):
    dist = np.linalg.norm(
      ctx[x,:-2]-ctx[y,:-2]
    )
    return dist < 0.3
```

Belief update



Informative questions: Expected information gain

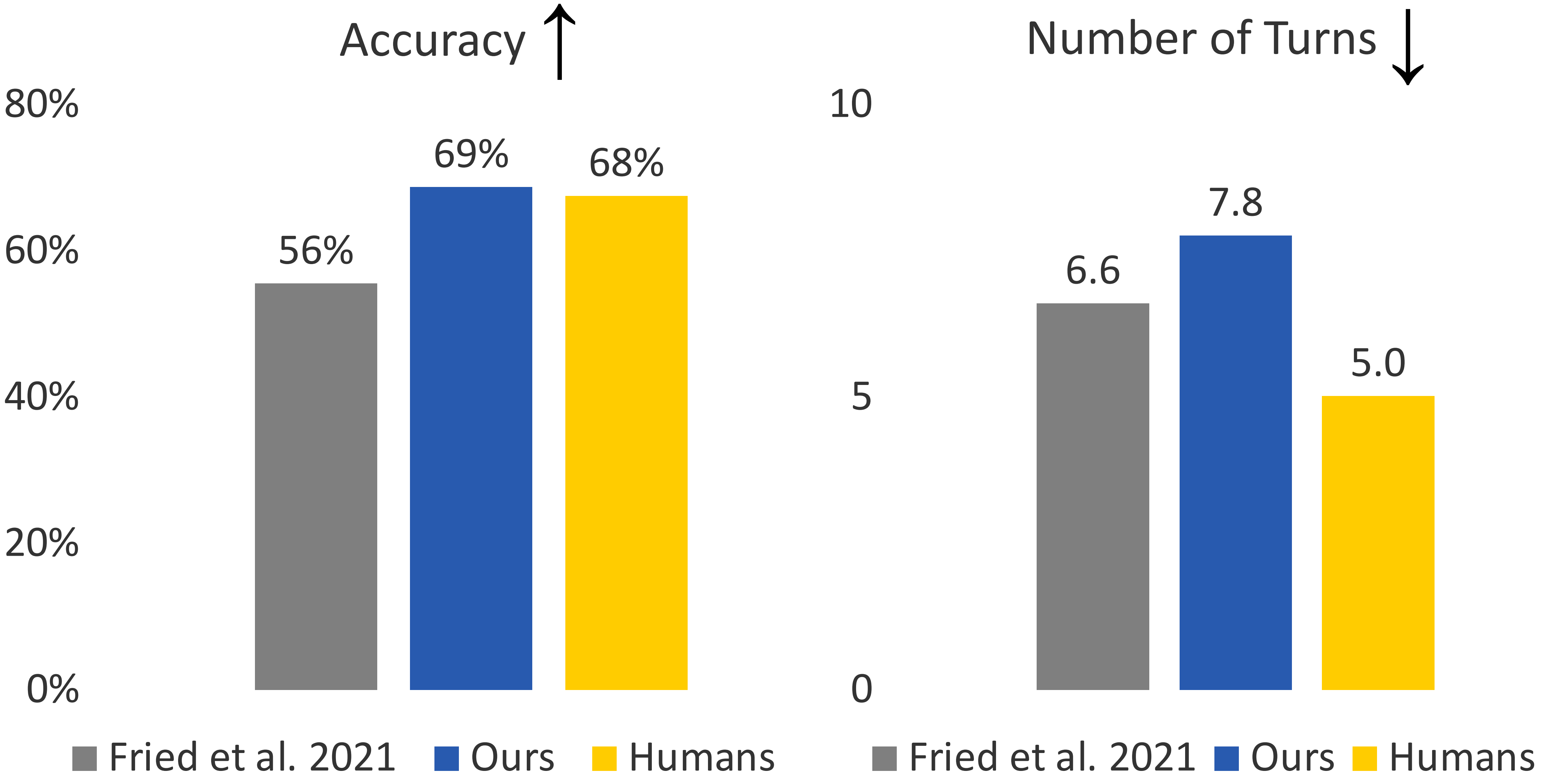


Evaluation

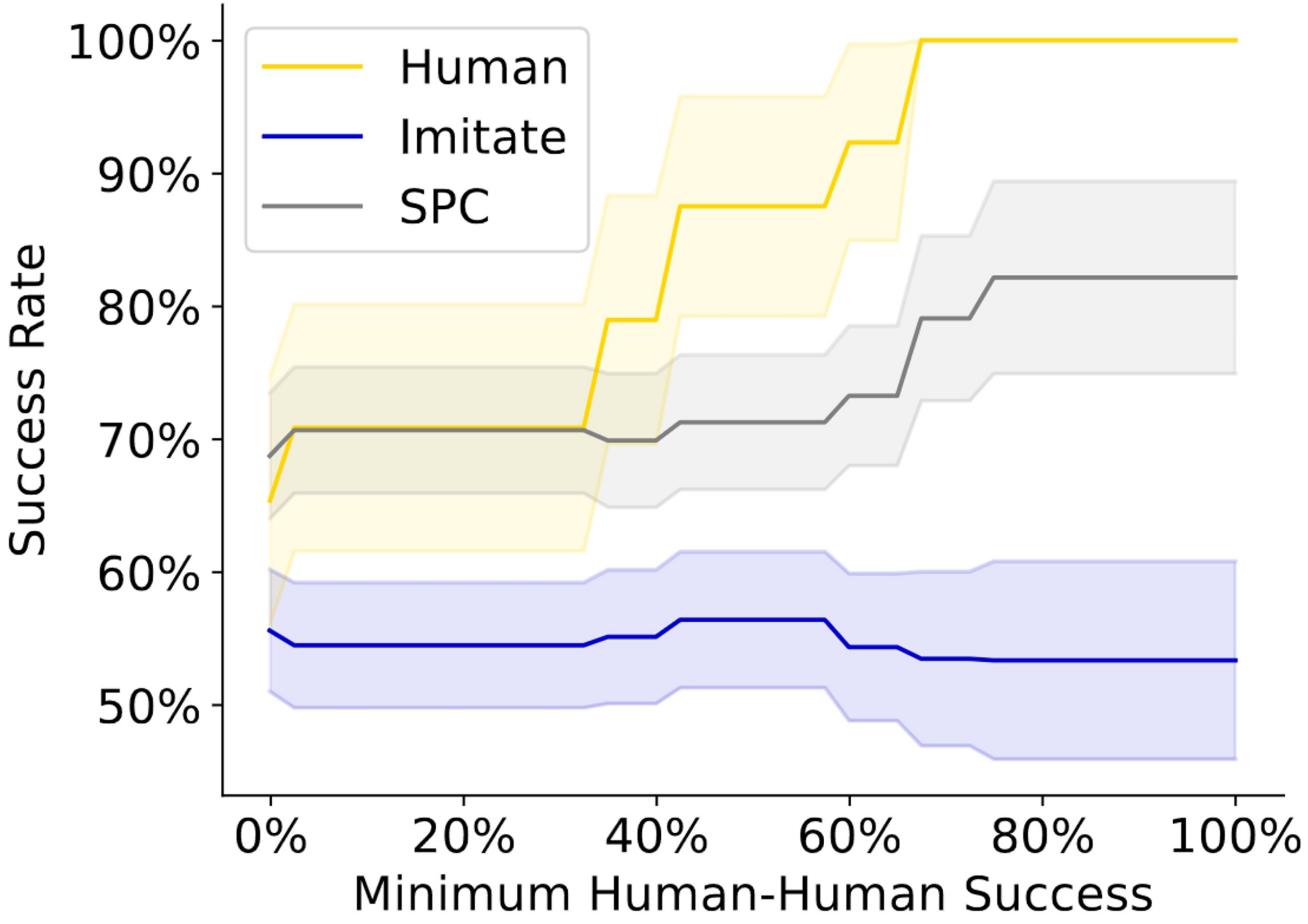
- Play the game with humans recruited on Mechanical Turk; evaluate success rate.

- Agents compared:
  - LSTM with Neural CRF Reference Resolver (Fried, Chiu, Klein, 2021)
  - Ours
  - Humans

Results


Accuracy ↑
80%
60%
40%
20%
0%
56%
69%
68%
Fried et al. 2021
Ours
Humans
Number of Turns ↓
10
5
0
6.6
7.8
5.0
Fried et al. 2021
Ours
Humans

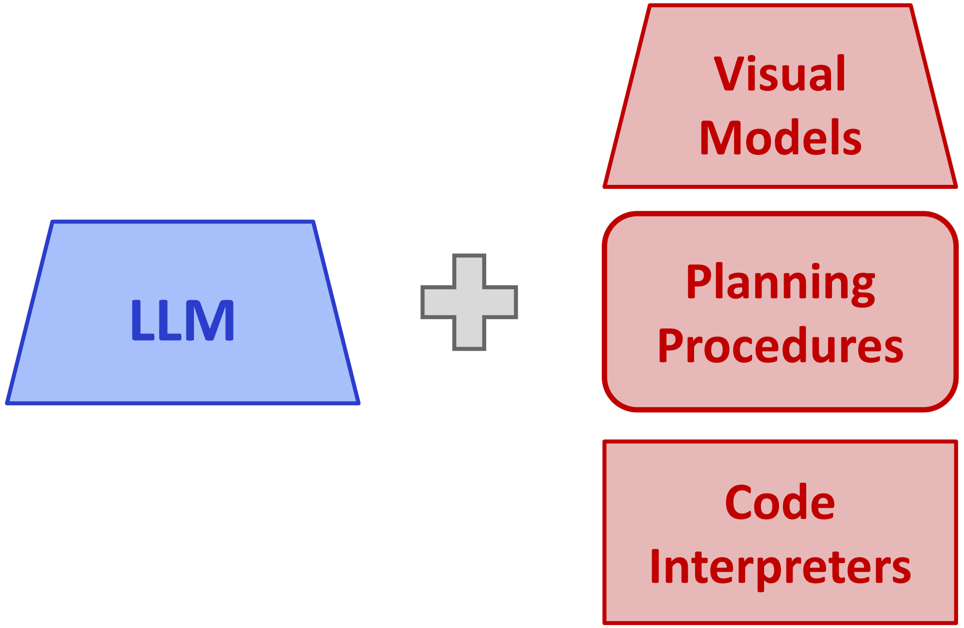
Results: Human evaluation



Takeaways

- LLMs model language use, but need to be contextualized!

- LLMs are useful building blocks in modular systems.

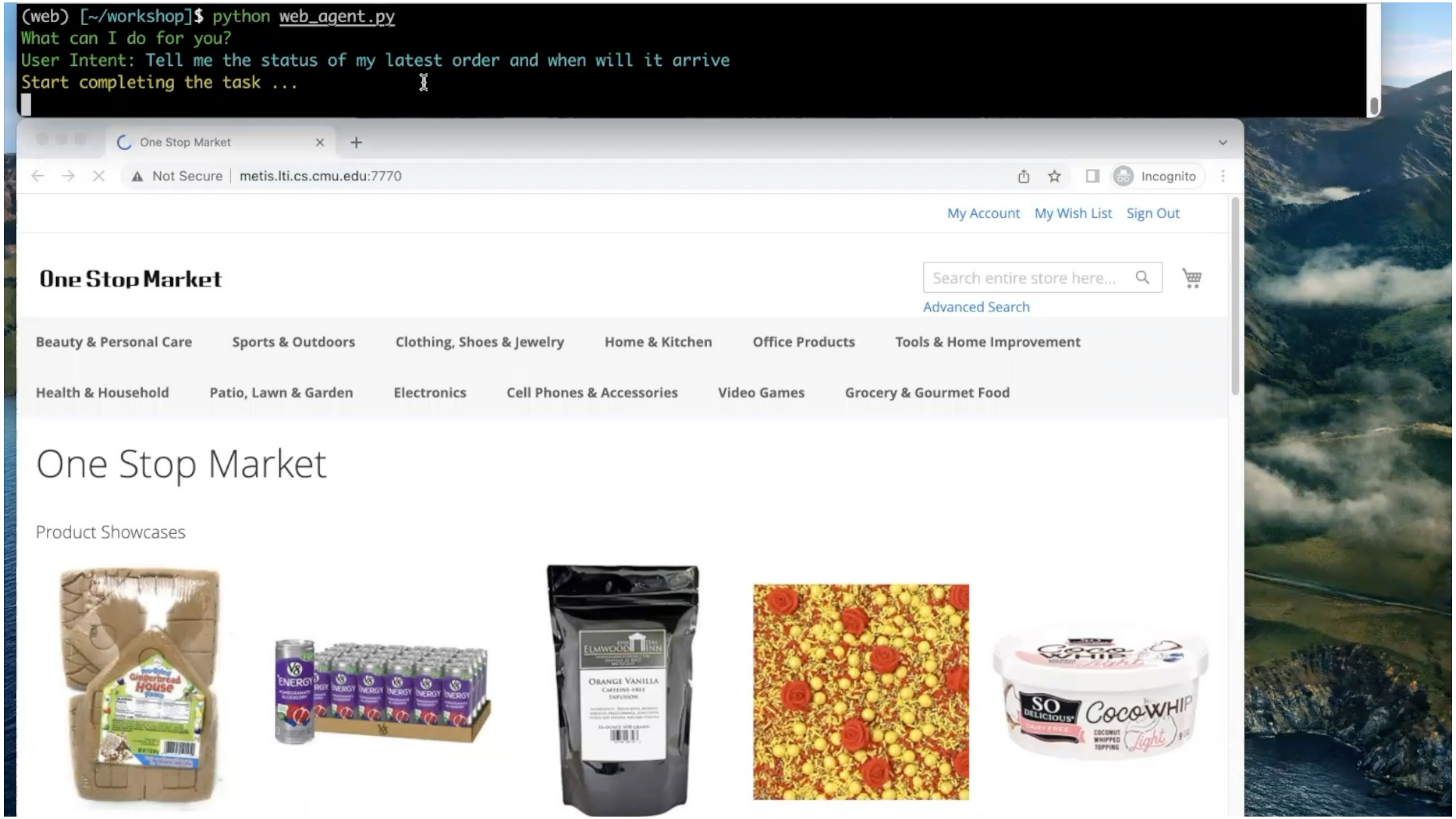


Shuyan Zhou

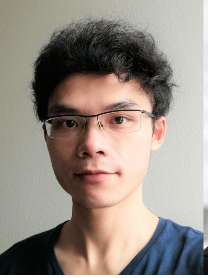
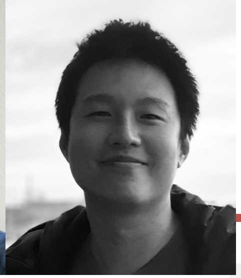
Frank Xu

Challenge Environments: WebArena

"Tell me the status of my latest order and when will it arrive"

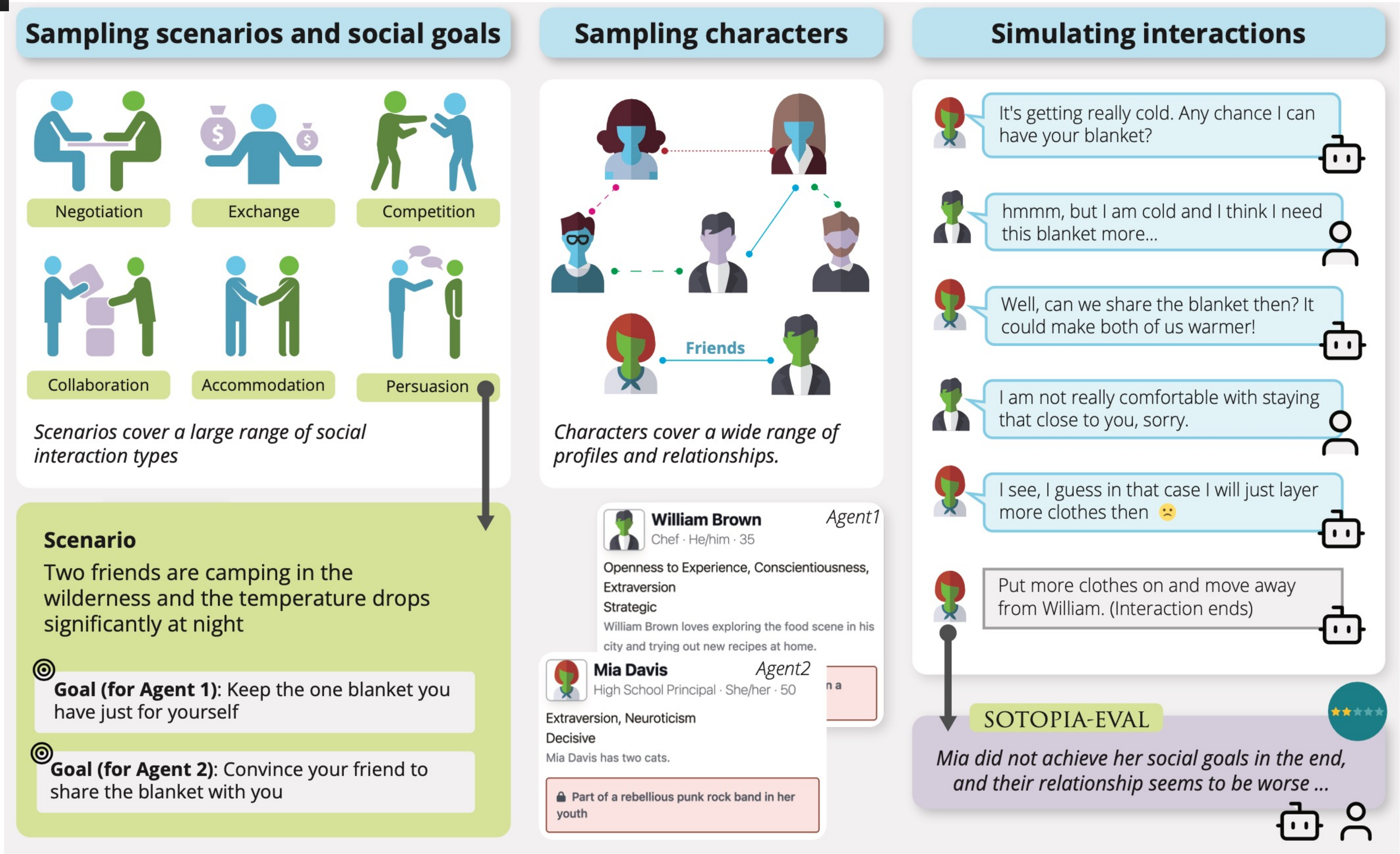


https://webarena.dev/



Xuhui Zhou   Hao Zhu

Challenge Environments: Sotopia

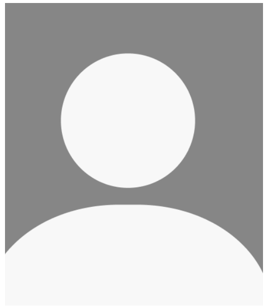


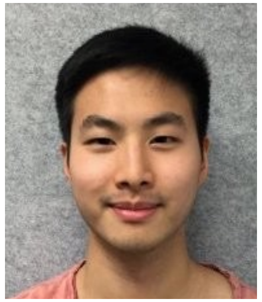
https://sotopia.world/

Collaborators


Yihan
Cao


Derek
Chen


Shuyi
Chen


Justin
Chiu


Sedrick
Keh


Jing Yu
Koh


Ryan
Liu


Sasha
Rush


Russ
Salakhutdinov


Saujas
Vaduguru


Zora
Wang


Wenting
Zhao

Thanks!

dfried@cs.cmu.edu
http://dpfried.github.io

FROMAGe: https://jykoh.com/fromage
GILL: https://jykoh.com/gill

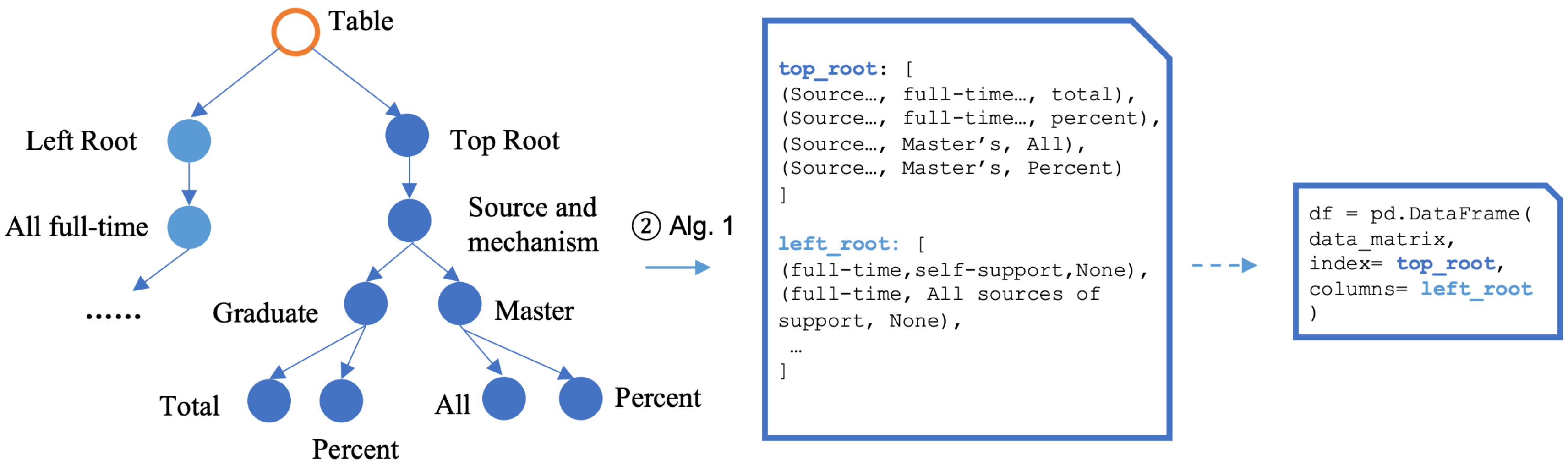
Other Projects

Backup Slides

Code for Table Question Answering

Source and mechanism	All full-time graduate students		Master's	
	Total	Percent	All	Percent
All full-time	433916	100.0	209221	100.0
Self-support	161641	37.3	139373	66.0
All sources of support	272275	62.7	69848	33.4
Federal	65999	15.2	10736	5.1
Institutional	182135	42.0	52319	25.0
All mechanisms of support	272275	62.7	69848	33.4

Original Hierarchical Table



Hierarchical Tree from HiTab **Multi-index Object**

[EMNLP 2023]

Code for Table Question Answering

Who is more likely to have cancer,
the elder or *the young*?

```
def compare_larger(values: list[float], args: list[str]) -> str:
    """Return the argument associated with the larger value."""
    return args[values.index(max(values))]
```

```
def compare_smaller(values: list[float], args: list[str]) -> str:
    """Return the argument associated with the smaller value."""
    return args[values.index(min(values))]
```

The QA API

Operation APIs

```
compare _larger([
data.loc[('Illness', 'Cancer', 'Percent'), ('All
Patients', 'Elders')],
data.loc[('Illness', 'Cancer', 'Percent'), ('All
Patients', 'Young')],
['Elders', 'Young']])
```

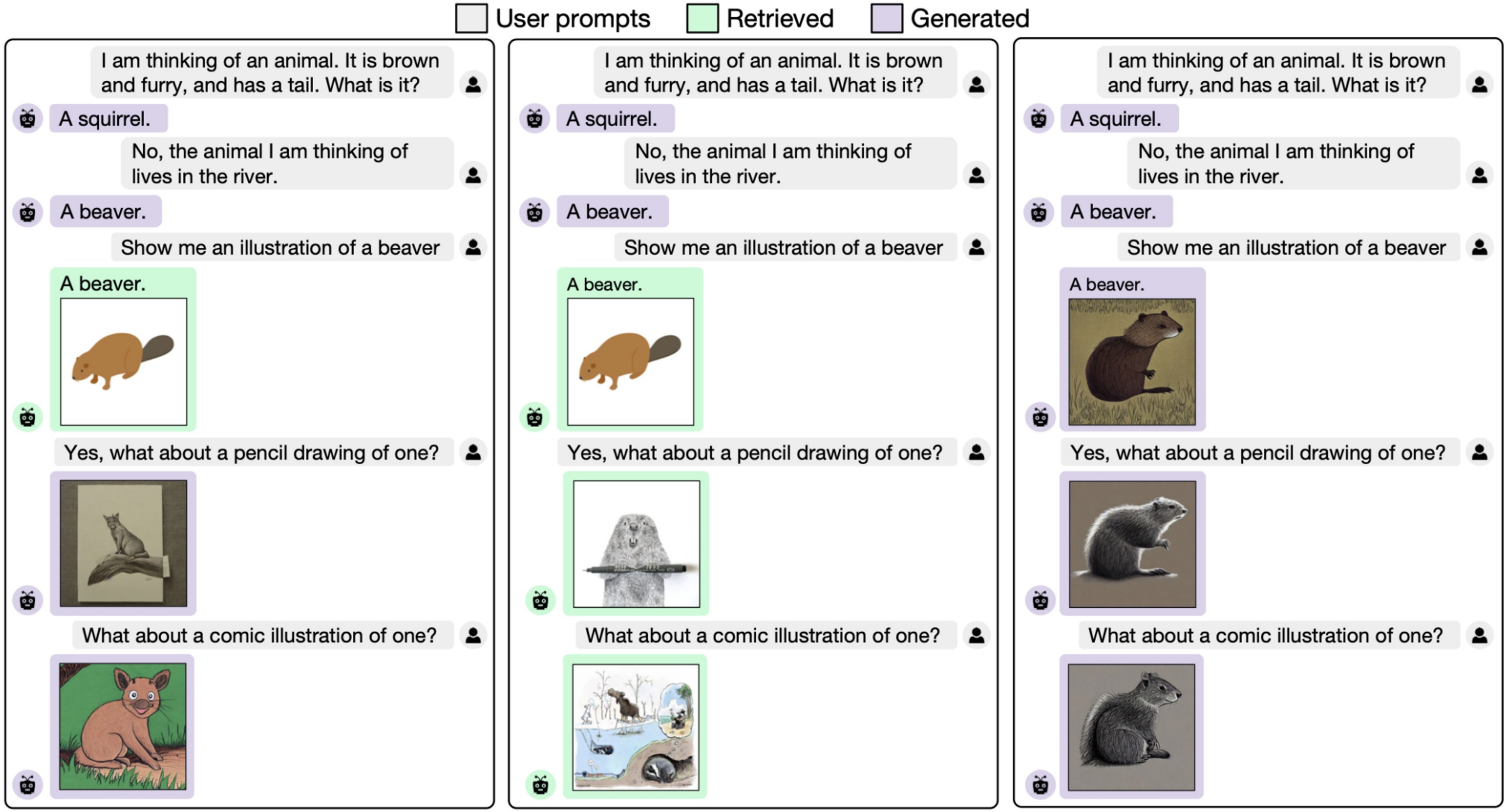
Code for Table Question Answering

- Code gives a unified representation across varied table formats and datasets

Dataset	**HiTab**	**Spider**	**AIT-QA**	**WikiTQ**
Baseline	MAPO 40.7	DIN-SQL 61.5	RCI 51.8	BINDER **54.8**
Codex	59.6	61.2	77.8	41.7
w/ API (Ours)	**69.3**	**63.8**	**78.0**	42.4

[EMNLP 2023]

GILL: A More General Multimodal LM



Ours
Retrieval and generation multimodal LM. Decides when to retrieve or generate.

FROMAGe
Retrieval only multimodal language model. Image outputs limited to the candidate retrieval set.

Stable Diffusion
Generation only text-to-image model. Less sensitive to longer text inputs (such as dialogue).

🐟 GILL: A More General Multimodal LM

- **Frozen** (Tsimpoukelli et al., 2021)
  Flamingo (Alayrac et al., 2022)
  BLIP-2 (Li et al., 2023)
  - Process image + text, generate text only
- **FROMAGe** (Koh et al., 2023)
  - Process image + text, generate text + retrieve images
- **GILL** (this work)
  - Process image + text, generate text + retrieve images + generate images
  - Decides whether to retrieve images or generate from scratch
  - Resource efficient: trained on 2 GPUs for 2 days

🐟 GILL

Generating Images with Large Language Models

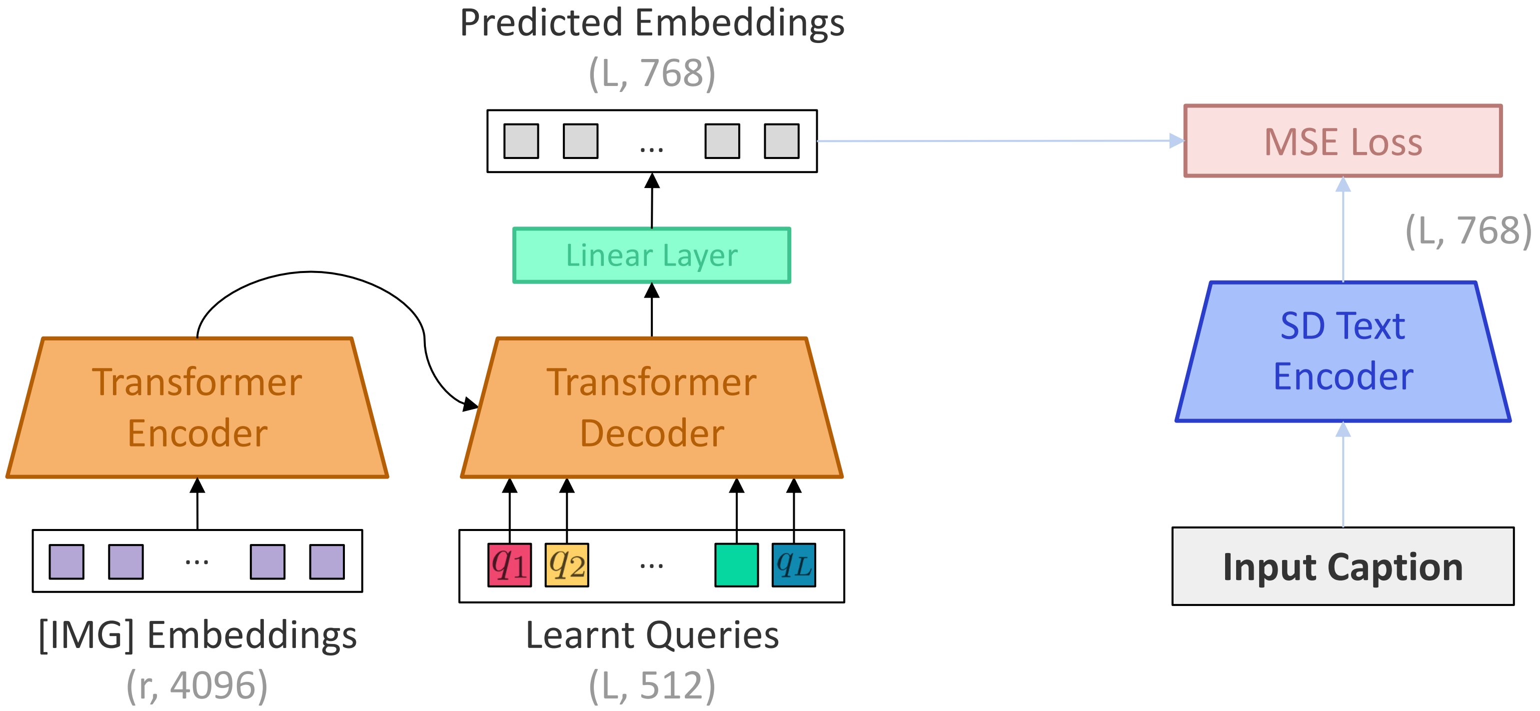
- **Capable of retrieving images, generating images, and generating text**
  - Can condition on arbitrarily interleaved image + text inputs
  - Generate text, generate images, and retrieve images as part of the output

- **Leverage the learnt abilities of pre-trained text-only LLMs**
  - In-context learning
  - Sensitivity to input prompts
  - Generate long and coherent dialogue

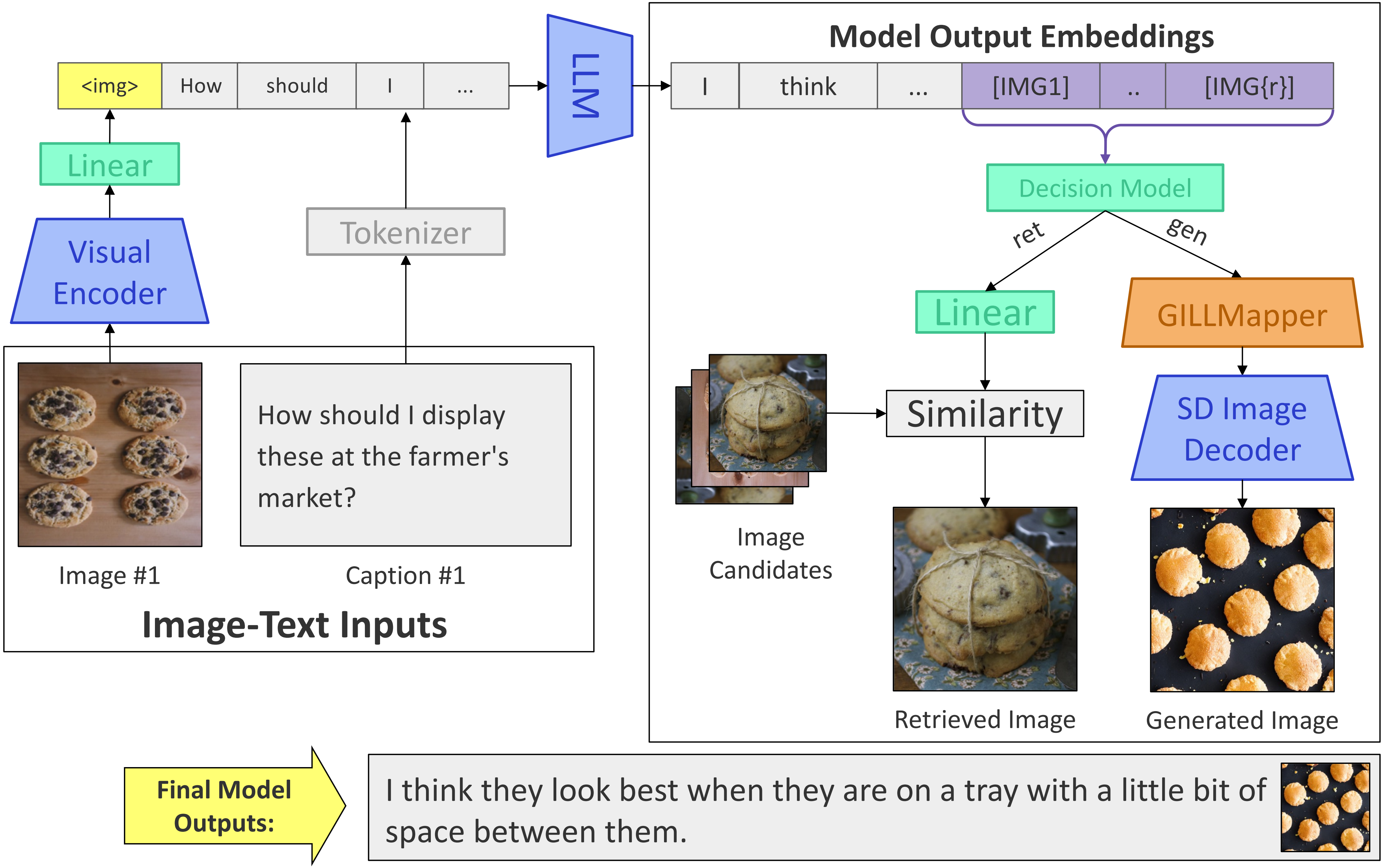
- **Model agnostic**
  - We use a 7B LLM, the CLIP encoder, and the Stable Diffusion image generator
  - Likely benefits from using larger and stronger LLMs in the future
  - Can be applied with other visual models (e.g., OCR) to introduce new abilities

GILLMapper: An Improved LLM-to-Generator Map

- Previous approaches use linear mappings between LLMs and visual models
- This is insufficient for image generation: decoders require dense information



Multimodal Few-Shot Learning with Frozen Language Models (Tsimpoukelli et al., 2021)
Linearly Mapping from Image to Text Space (Merullo et al., 2023)
Grounding Language Models to Images for Multimodal Inputs and Outputs (Koh et al., 2023)



Model Output Embeddings
<img>
How
should
I
...
LLM
I
think
...
[IMG1]
..
[IMG{r}]
Linear
Visual Encoder
Tokenizer
Decision Model
ret
gen
Linear
GILLMapper
Similarity
SD Image Decoder
How should I display these at the farmer's market?
Image #1
Caption #1
Image-Text Inputs
Image Candidates
Retrieved Image
Generated Image
Final Model Outputs:
I think they look best when they are on a tray with a little bit of space between them.

Evaluation: Contextual Image Generation

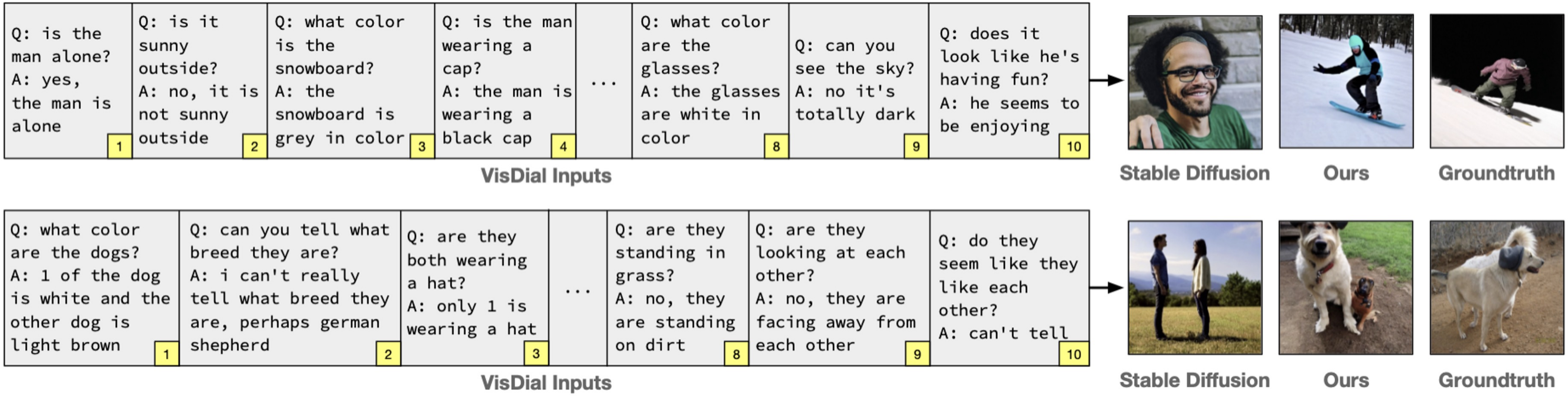
Model	CLIP Similarity (↑) 1 caption	CLIP Similarity (↑) 5 captions	CLIP Similarity (↑) 5 caps, 4 images	LPIPS (↓) 1 caption	LPIPS (↓) 5 captions	LPIPS (↓) 5 caps, 4 images
GLIDE [34]	0.582	0.591	-	0.753	0.745	-
Stable Diffusion [43]	**0.592** $\pm$0.0007	0.598 $\pm$ 0.0006	-	0.703 $\pm$0.0003	0.704 $\pm$ 0.0004	-
GILL	0.581 $\pm$0.0005	**0.612** $\pm$0.0011	**0.641** $\pm$0.0011	**0.702** $\pm$0.0004	**0.696** $\pm$0.0008	**0.693** $\pm$0.0008

- Our model outperforms Stable Diffusion on longer input contexts
- This is despite GILL (essentially) distilling from SD!
- GILL benefits from the abilities of the LLM (sensitivity to longer inputs, word orderings, in-context learning)

Visual Storytelling (Huang et al., 2016)

Evaluation: Contextual Image Generation

- Given a Visual Dialogue, generate a relevant image
- Need to condition on long dialogue-like text (OOD with finetuning data)



Visual Dialog ([Das et al., 2017](#))

Evaluation: Contextual Image Generation

Model	CLIP Similarity (↑) 1 round	5 rounds	10 rounds	LPIPS (↓) 1 round	5 rounds	10 rounds
GLIDE [34]	**0.562**	0.595	0.587	0.800	0.794	0.799
Stable Diffusion [43]	0.552 ±0.0015	**0.629** ±0.0015	0.622 ±0.0012	**0.742** ±0.0010	0.722 ±0.0012	0.723 ±0.0008
GILL	0.528 ±0.0014	0.621 ±0.0009	**0.645** ±0.0010	**0.742** ±0.0022	**0.718** ±0.0028	**0.714** ±0.0006

Visual Storytelling (Huang et al., 2016)

GILLMapper: A Stronger LLM-to-Generator Mapping

Image generators require **denser** input sequences. Linear mappings are insufficient.

	CC3M	VIST
Model	**FID (↓)**	**CLIP Sim (↑)**
Stable Diffusion [43]	**13.94**	0.598
Ours + Linear	15.50	0.500
Ours + 3-layer MLP	15.33	0.502
Ours + Transformer Encoder	16.30	0.605
Ours + GILLMapper	15.31	**0.641**

Other Abilities: Text-to-Image Generation

Stable Diffusion

Ours

“A dignified beaver wearing glasses, a vest, and colorful neck tie. He stands next to a tall stack of books in a library.”

Stable Diffusion

Ours

“Snow mountain and tree reflection in the lake”

Stable Diffusion

Ours

“A drop-top sports car coming around a bend in the road”

Stable Diffusion

Ours

“a group of penguins in a snowstorm”

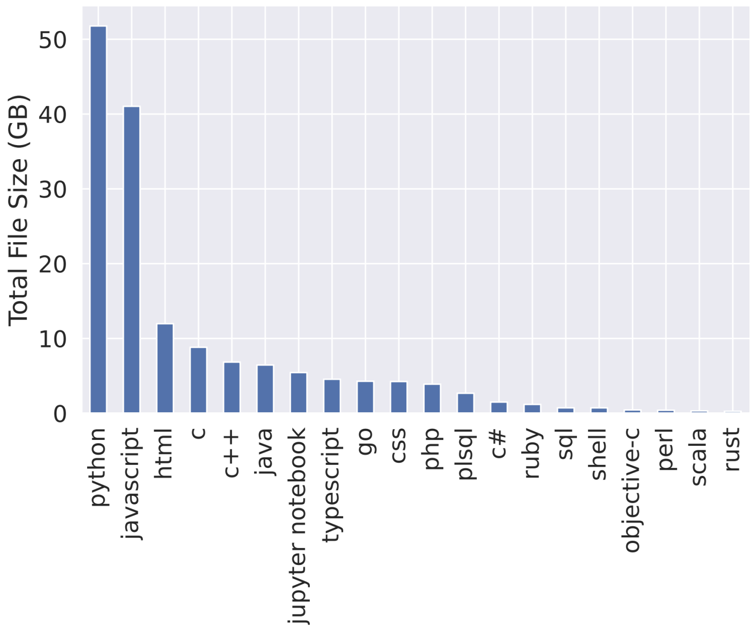
Evaluation: HumanEval Benchmark

Constructed by authors of Codex paper; programming puzzle/simple contest problems. Evaluated using unit tests.

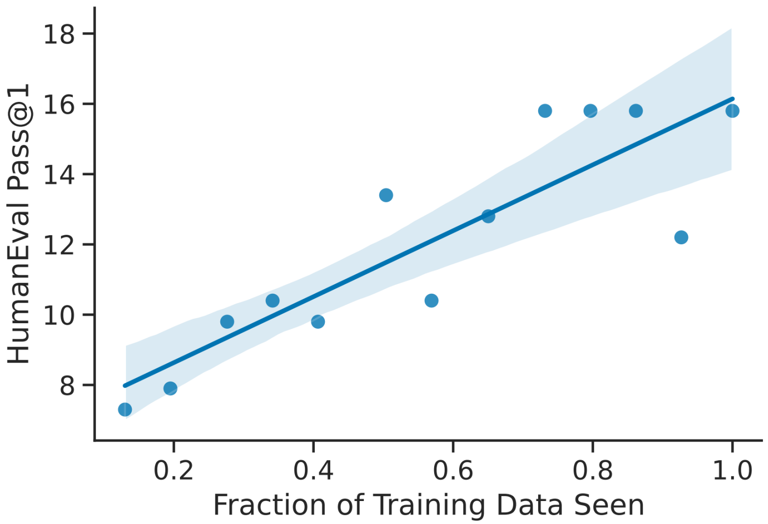
```python
from typing import List

def has_close_elements(numbers: List[float], threshold: float) -> bool:
    """
    Check if in given list of numbers, are any two numbers closer to each other
        than given threshold.
    >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
    False
    >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0], 0.3)
    True
    """
    for idx, elem in enumerate(numbers):
        for idx2, elem2 in enumerate(numbers):
            if idx != idx2:
                distance = abs(elem - elem2)
                if distance < threshold:
                    return True
    return False
```

Model Training

- Training Data
  - 600K permissively-licensed repositories from GitHub & GitLab. ~150GB total
  - StackOverflow: questions, answers, comments. ~50GB



- Models
  - Standard transformer LM
  - 1B model: ~1 week on 128 V100s
  - 6B model: ~3 weeks on 240 V100s



Evaluation

- Zero-shot evaluation on several software development-inspired code infilling tasks (we'll show two).
- Compare the model in three different modes to evaluate benefits of suffix context

Baselines

Left-to-Right Single

Doesn't use suffix

Left-to-Right Rerank

Only uses suffix when reranking

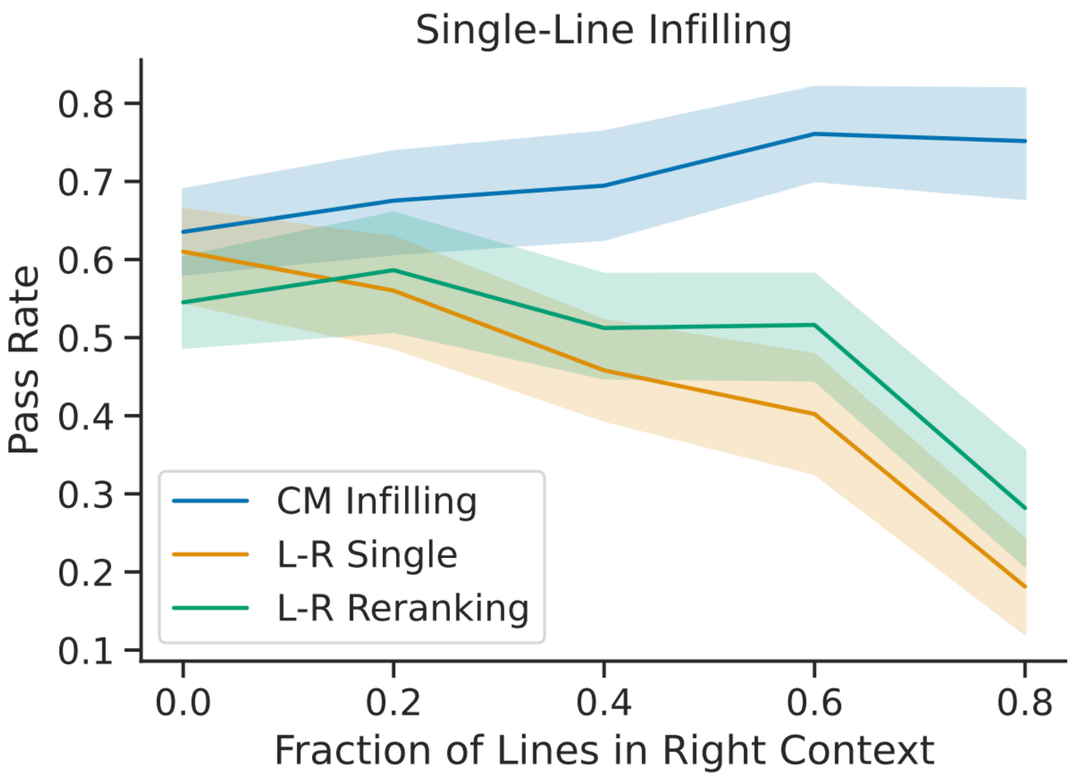
Ours

Causal Masking

Uses suffix when generating

Evaluation: Function Completion

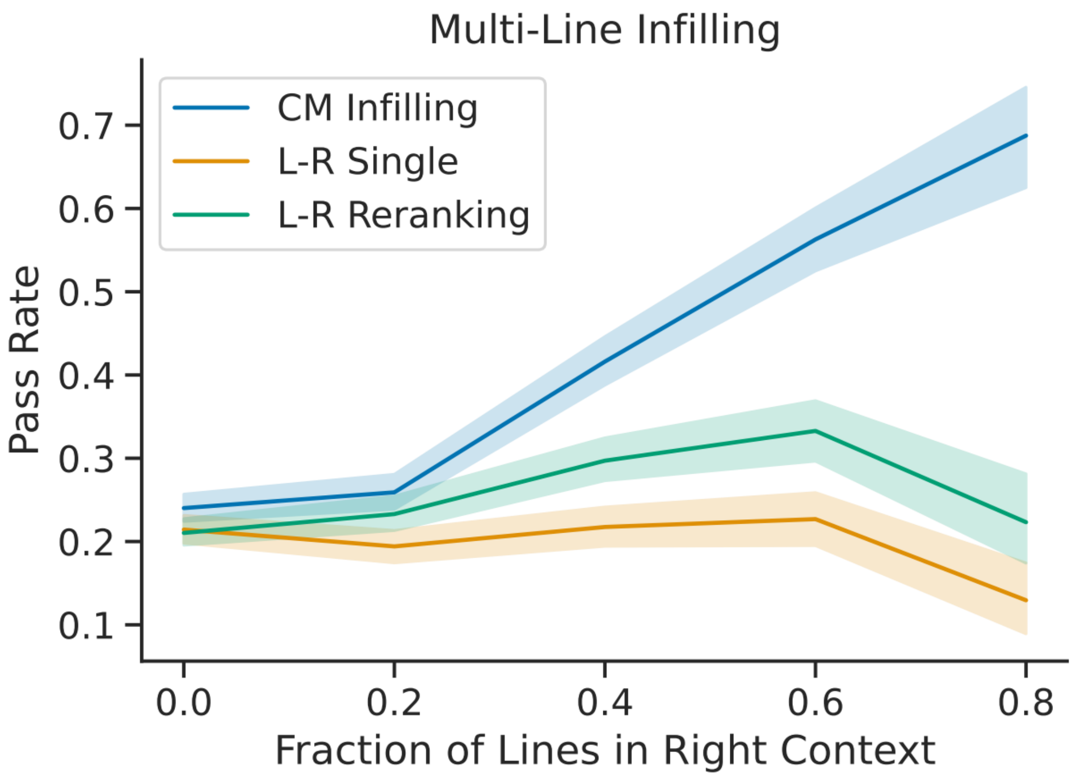
```python
from typing import List

def has_close_elements(numbers: List[float], threshold: float) -> bool:
    """
    Check if in given list of numbers, are any two numbers closer to each other
        than given threshold.
    >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
    False
    >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0], 0.3)
    True
    """
    for idx, elem in enumerate(numbers):
        for idx2, elem2 in enumerate(numbers):
            if idx != idx2:
                distance = abs(elem - elem2)
                if distance < threshold:
                    return True
    return False
```



Constructed from HumanEval [Chen et al. 2021]

Evaluation: Function Completion

```python
from typing import List

def has_close_elements(numbers: List[float], threshold: float) -> bool:
    """
    Check if in given list of numbers, are any two numbers closer to each other
        than given threshold.
    >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
    False
    >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0], 0.3)
    True
    """
    for idx, elem in enumerate(numbers):
        for idx2, elem2 in enumerate(numbers):
            if idx != idx2:
                distance = abs(elem - elem2)
                if distance < threshold:
                    return True
    return False
```



Constructed from HumanEval [Chen et al. 2021]

Evaluation: Function Completion

Fill in one or more lines of a function; evaluate with unit tests.

```python
from typing import List

def has_close_elements(numbers: List[float], threshold: float) -> bool:
    """
    Check if in given list of numbers, are any two numbers closer to each other
        than given threshold.
    >>> has_close_elements([1.0, 2.0, 3.0], 0.5)
    False
    >>> has_close_elements([1.0, 2.8, 3.0, 4.0, 5.0, 2.0], 0.3)
    True
    """
    for idx, elem in enumerate(numbers):
        for idx2, elem2 in enumerate(numbers):
            if idx != idx2:
                distance = abs(elem - elem2)
                if distance < threshold:
                    return True
    return False
```

Method	Pass Rate
L-R single	24.9
L-R reranking	28.2
CM infilling	38.6

Constructed from HumanEval [Chen et al. 2021]

Evaluation: Docstring Generation

```python
def count_words(filename: str) -> Dict[str, int]:
    """
    Counts the number of occurrences of each word in the given file.

    :param filename: The name of the file to count.
    :return: A dictionary mapping words to the number of occurrences.
    """
    with open(filename, 'r') as f:
            word_counts = {}
            for line in f:
                for word in line.split():
                    if word in word_counts:
                        word_counts[word] += 1
                    else:
                        word_counts[word] = 1
        return word_counts
```

Method	BLEU
Ours: L-R single	16.05
Ours: L-R reranking	17.14
Ours: Causal-masked infilling	18.27

[CodeXGlue, Lu et al. 2021]

Evaluation: Return Type Prediction

Type Inference

```
def count_words(filename: str) -> Dict[str, int]:
    """Count the number of occurrences of each word in the file."""
    with open(filename, 'r') as f:
        word_counts = {}
        for line in f:
            for word in line.split():
                if word in word_counts:
                    word_counts[word] += 1
                else:
                    word_counts[word] = 1
    return word_counts
```

Method	F1
Ours: Left-to-right single	30.8
Ours: Left-to-right reranking	33.3
Ours: Causal-masked infilling	**59.2**
TypeWriter (Supervised)	48.3

[TypeWriter OSS, Pradel et al. 2020]

Training Models on Human Instructions