CS 577: NATURAL LANGUAGE PROCESSING

Abulhair Saparov

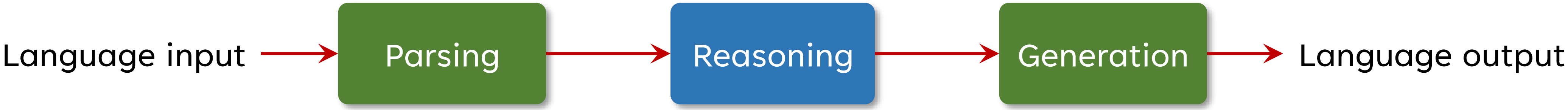
Lecture 27: Multi-modal NLP

[some slide content from: Sean Welleck, 11-711 Lecture Slides, 2025]

MULTI-MODAL NLP

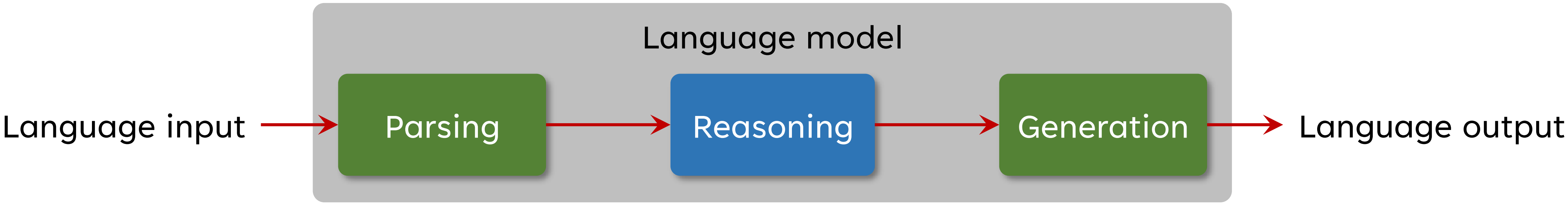
- Real-world tasks and reasoning often utilizes more than one modality.
  - Language
    - Speech, reading, writing, Braille
  - Sound
  - Vision
  - Touch
  - etc…
- We will focus on language + vision,
  - But there is active research in combining other modalities.

MULTI-MODAL NLP



Language input → Parsing → Reasoning → Generation → Language output

MULTI-MODAL NLP


Language model

Language input

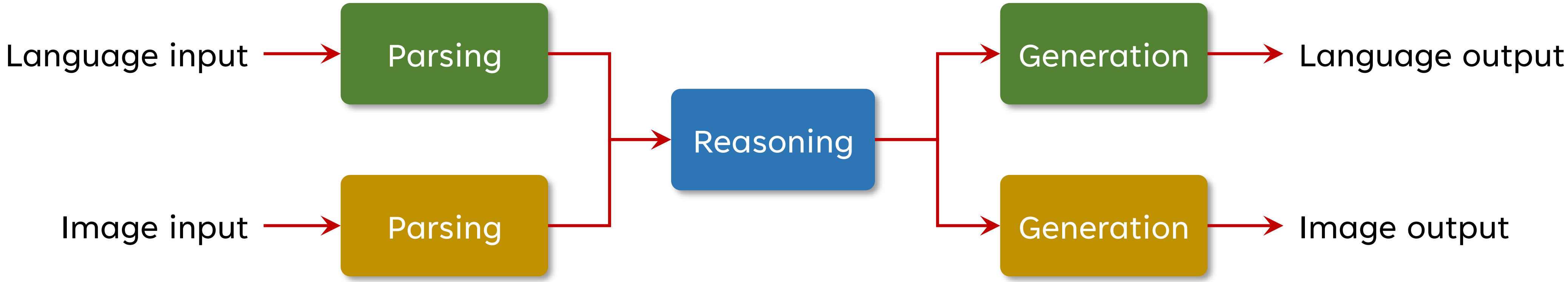
Parsing

Reasoning

Generation

Language output

MULTI-MODAL NLP



Language input

Parsing

Image input

Parsing

Reasoning

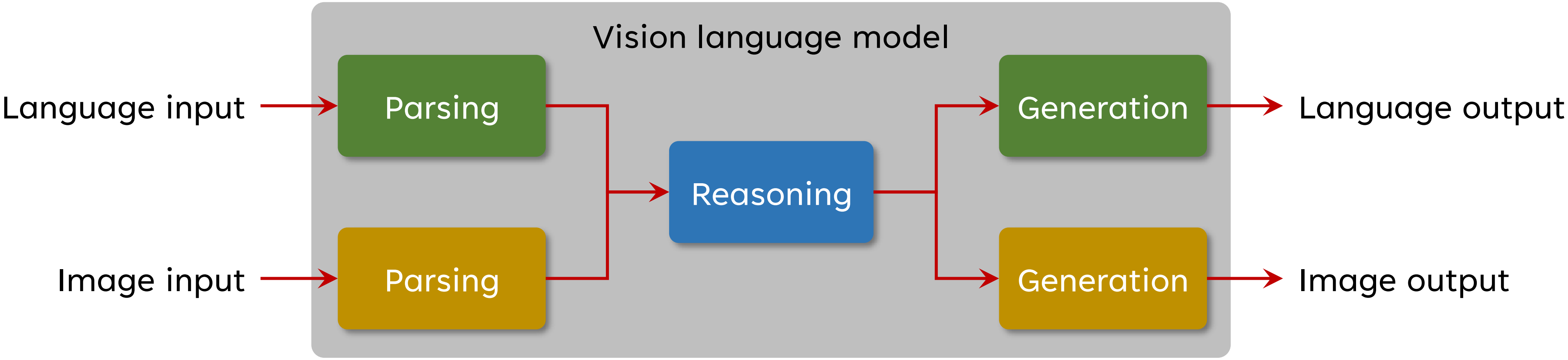
Generation

Language output

Generation

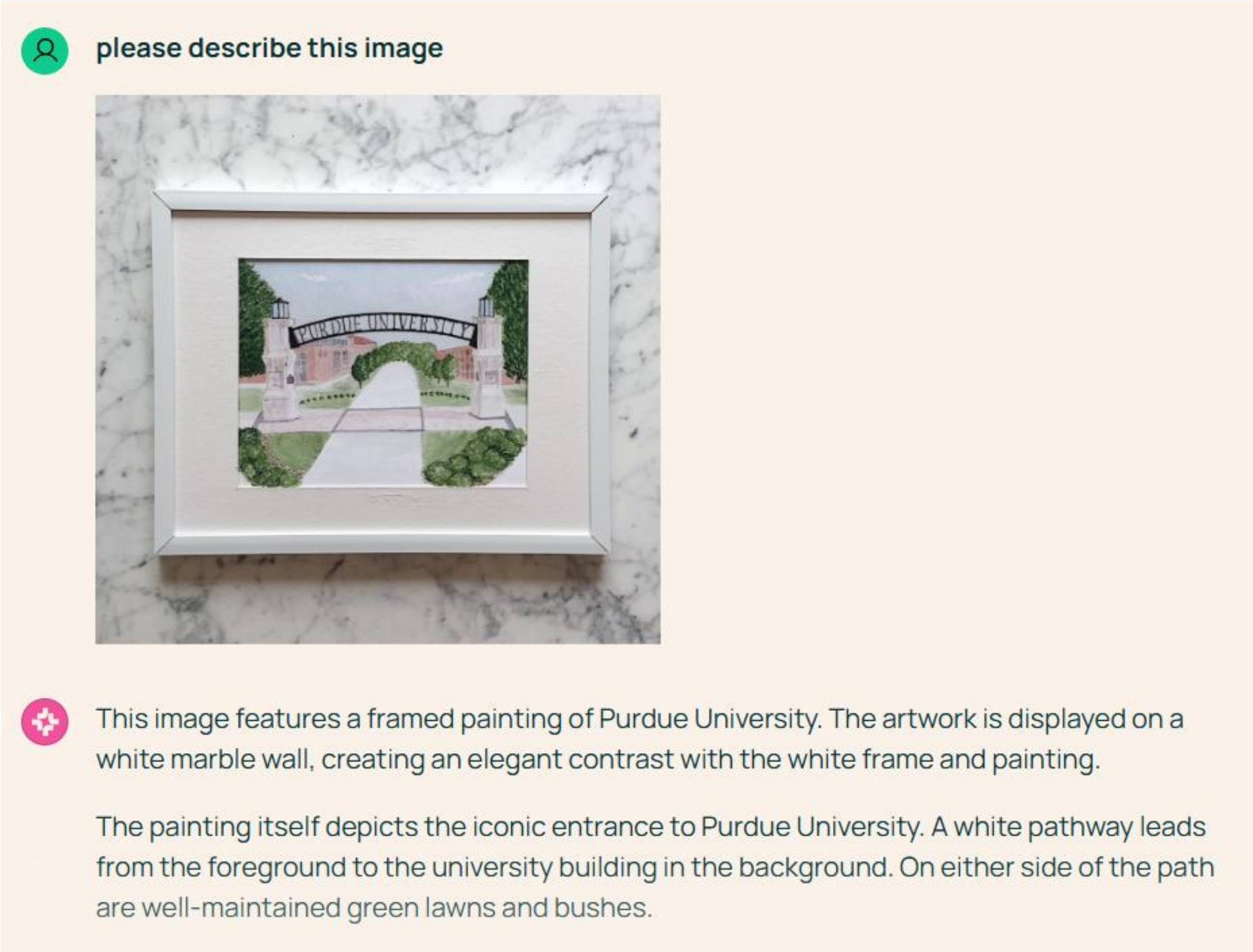
Image output

MULTI-MODAL NLP



Also called multi-modal large language model (MLLM).

MULTI-MODAL NLP EXAMPLES



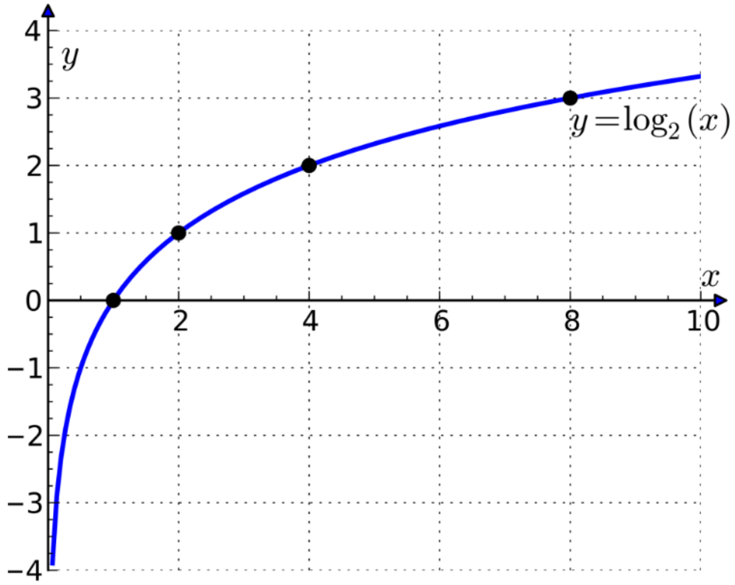
please describe this image

This image features a framed painting of Purdue University. The artwork is displayed on a white marble wall, creating an elegant contrast with the white frame and painting.

The painting itself depicts the iconic entrance to Purdue University. A white pathway leads from the foreground to the university building in the background. On either side of the path are well-maintained green lawns and bushes.

[molmo.allenai.org]

MULTI-MODAL NLP EXAMPLES



Question: The derivative of y at $x = 6$ is ____ that at $x = 8$.
Choices: (A) larger than (B) equal to (C) smaller than
Answer: (A) larger than

Question: How many zeros does this function have?
Answer: 1

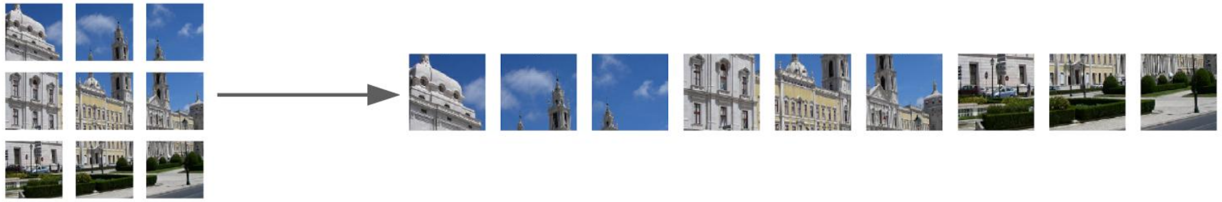
Question: What is the value of y at $x = 1$?
Answer: 0

MULTI-MODAL NLP

- Core problem: How to represent images?
- We represent text as a sequence of tokens.
  - Each token is mapped into an embedding vector.
- Can we apply the same idea to images?
  - Rasterized images are a 2D grid of pixels (typically with RGB values).
- Idea: Divide an image into patches.
  - Map each patch into a vector.
  - Provide these vectors as input to a transformer.

VISION TRANSFORMER

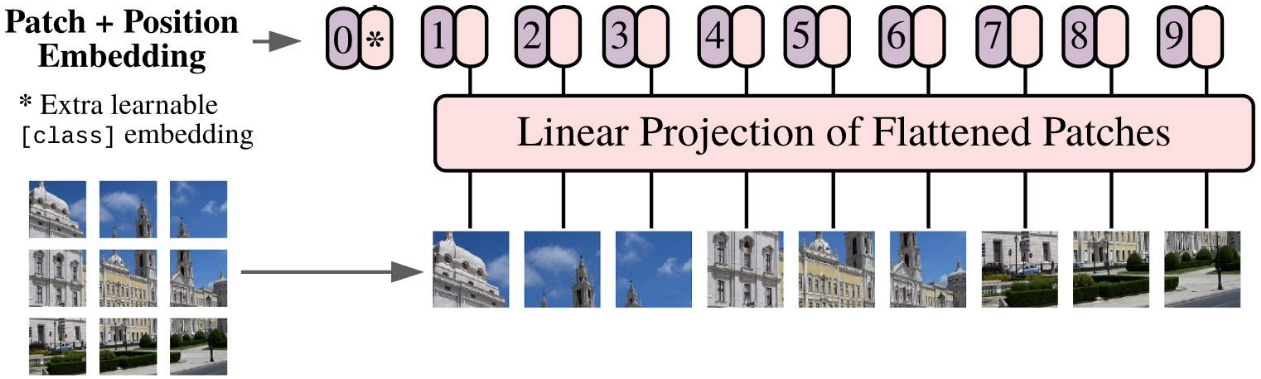
- First divide an image into $n \times n$ pixel patches.
  - Arrange them in a linear order (e.g., row-major).



[Dosovitskiy, Beyer, Kolesnikov, and Weissenborn et al., 2021]

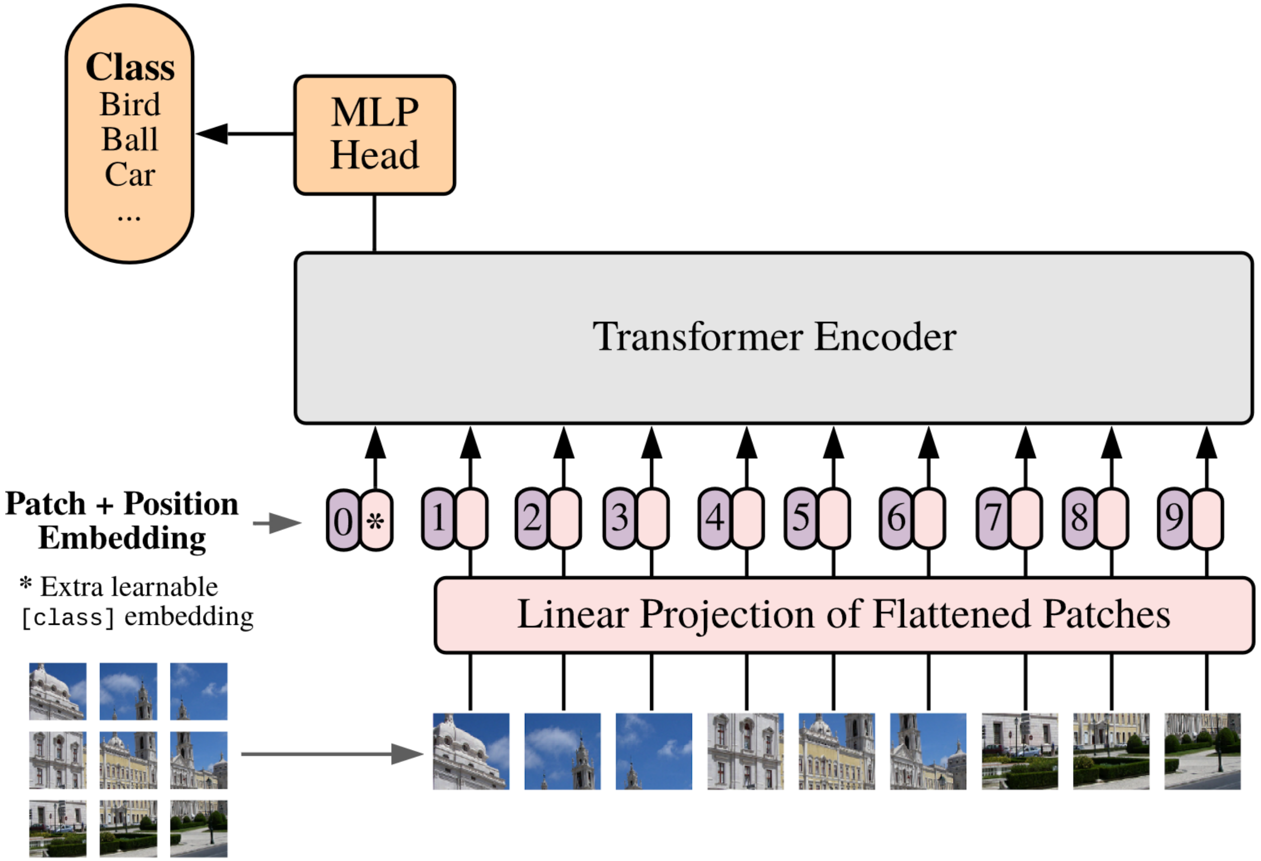
VISION TRANSFORMER

- Each patch is now a $3n^2$-dimensional vector (assuming 3 channels; e.g., RGB).
  - Use a linear layer to map each patch vector into an embedding.
  - Add a position embedding to each patch embedding.
- The first token is a special `[class]` token.


Patch + Position Embedding
* Extra learnable `[class]` embedding
0
*
1
2
3
4
5
6
7
8
9
Linear Projection of Flattened Patches

[Dosovitskiy, Beyer, Kolesnikov, and Weissenborn et al., 2021]

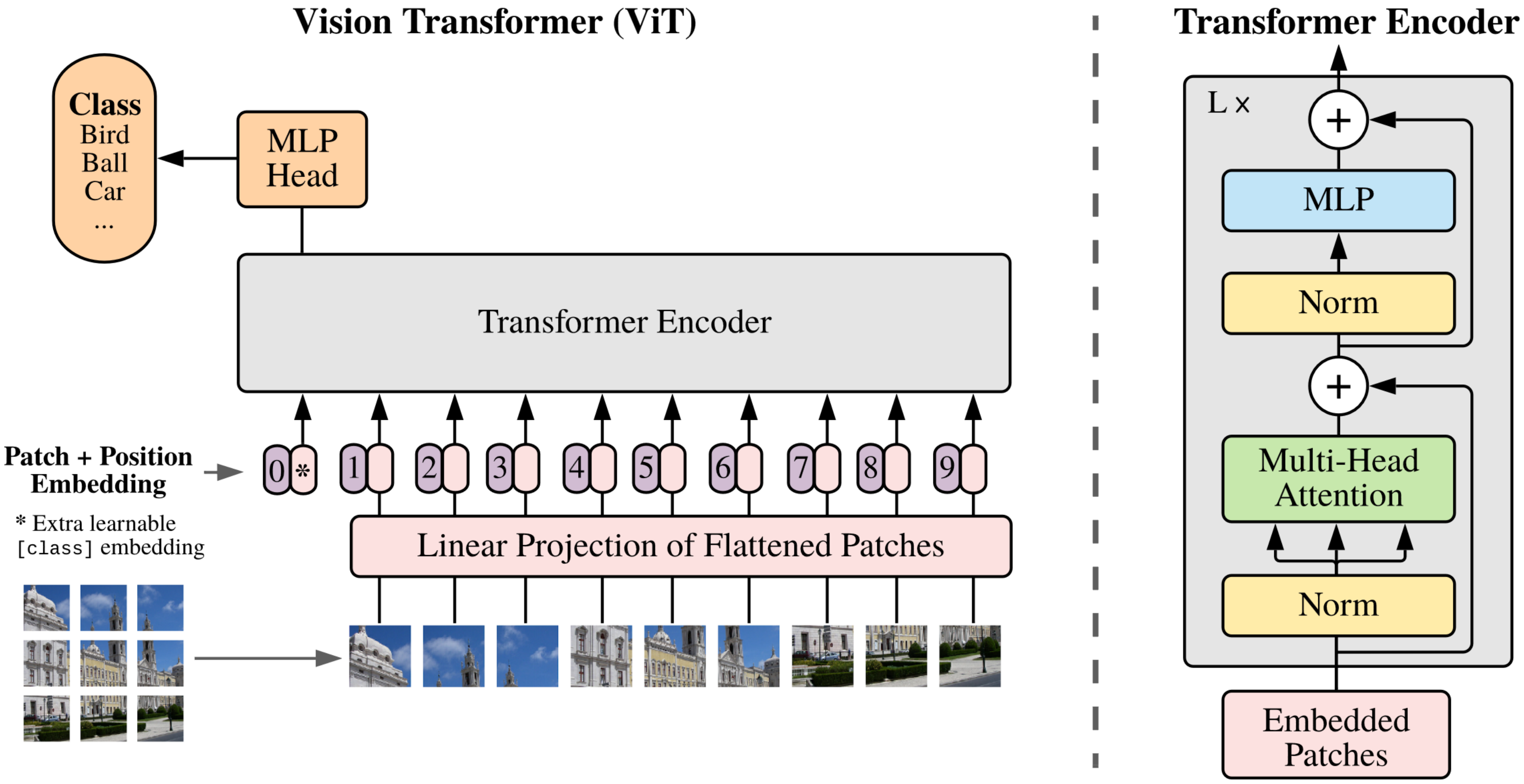
VISION TRANSFORMER


Class
Bird
Ball
Car
...
MLP Head
Transformer Encoder
Patch + Position Embedding
0 *
1
2
3
4
5
6
7
8
9
* Extra learnable [class] embedding
Linear Projection of Flattened Patches

- Run the embeddings through an encoder-only transformer (i.e., no causal mask).
- Take the output embedding of the first token.
  - If the task is multi-class classification:
  - Apply a linear layer and a softmax.

[Dosovitskiy, Beyer, Kolesnikov, and Weissenborn et al., 2021]

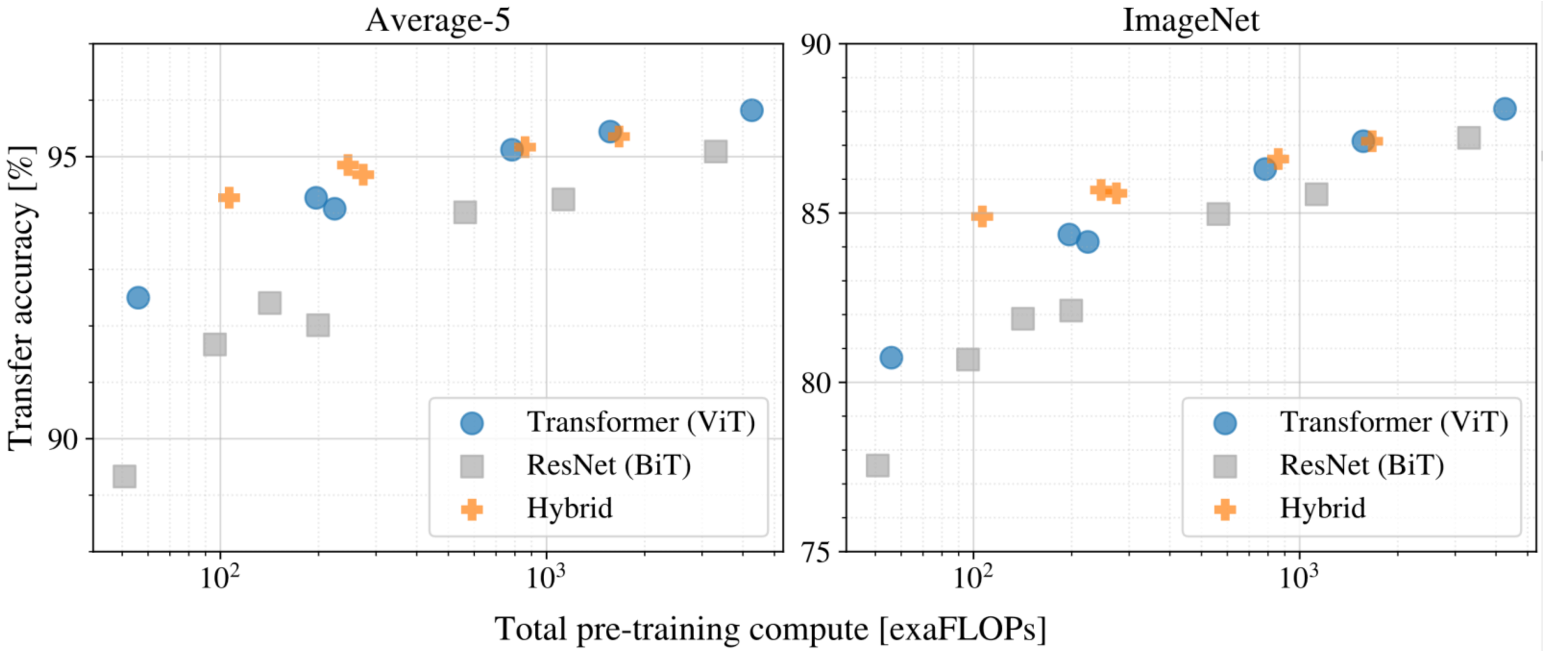
VISION TRANSFORMER



[Dosovitskiy, Beyer, Kolesnikov, and Weissenborn et al., 2021]

VISION TRANSFORMER

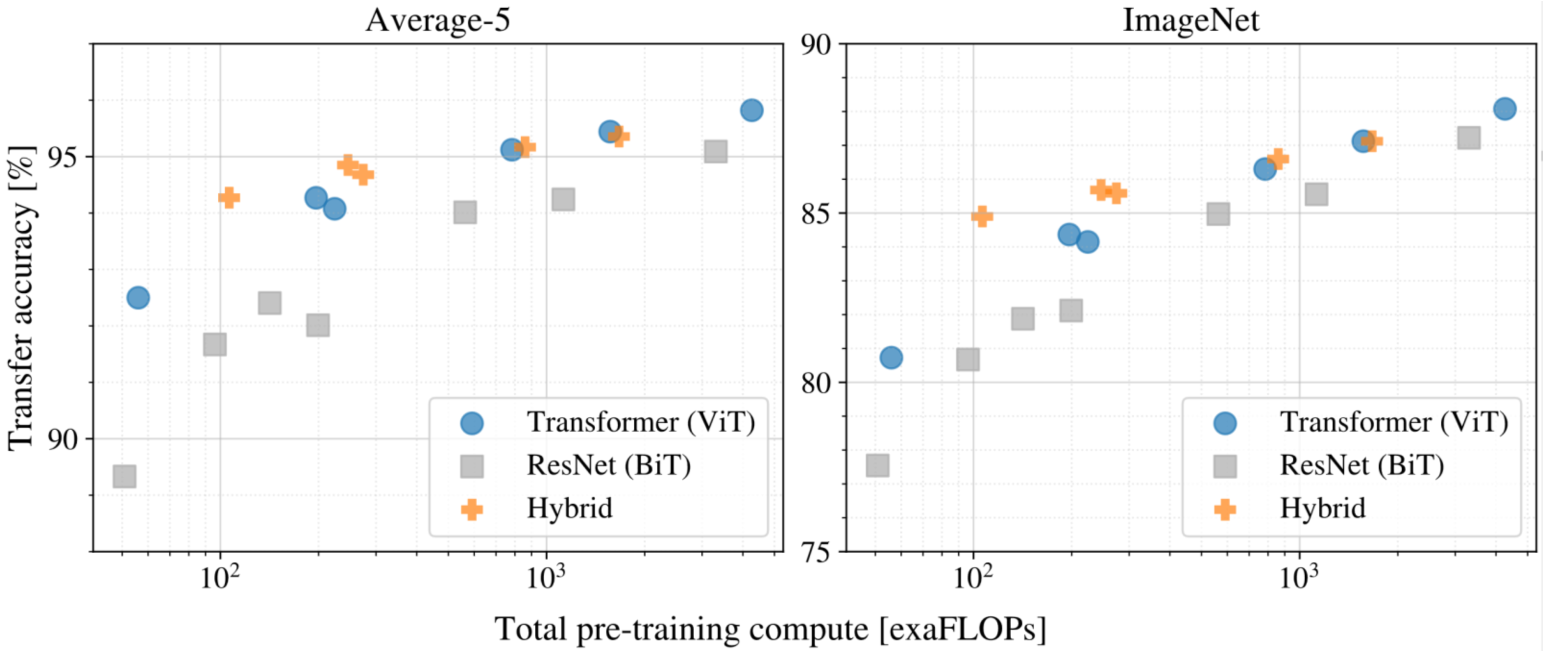
- This approach is called the Vision Transformer (ViT; Dosovitskiy, Beyer, Kolesnikov, and Weissenborn et al., 2021).
- They tested on the image classification task:



[Dosovitskiy, Beyer, Kolesnikov, and Weissenborn et al., 2021]

VISION TRANSFORMER

- Vision transformers have better scaling properties than ResNets.
- Combining ViTs with ResNets leads to better performance at lower compute, but the gap is closed at larger scales.

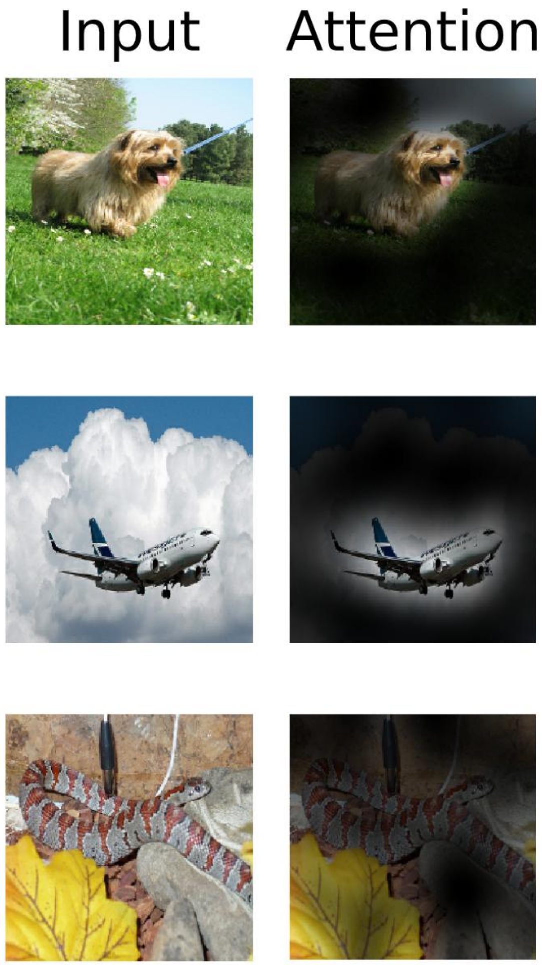


[Dosovitskiy, Beyer, Kolesnikov, and Weissenborn et al., 2021]

VISION TRANSFORMER

- Just as we can inspect the attention weights in transformer layers for a specific text input sequence,
- We can do the same for the vision transformer.
- Shade each image patch according to how much the output token attends to that patch:

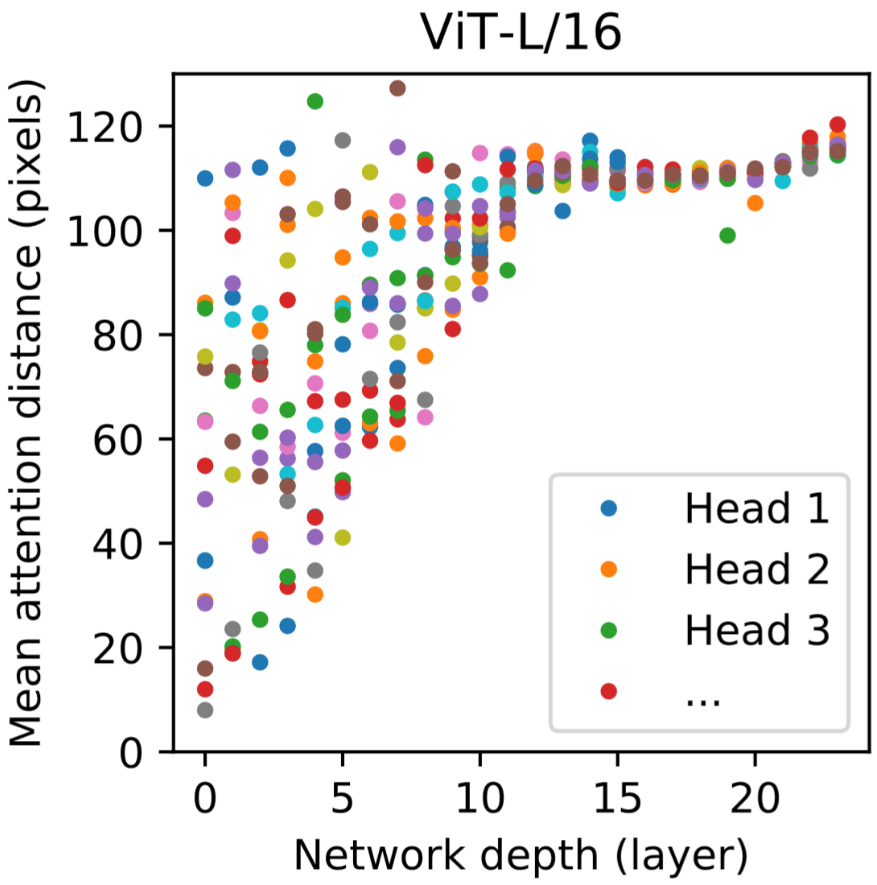
  (brighter areas indicate higher attention)
- The output token tends to attend to the object that specifies the class (i.e., dog, plane, etc).



[Dosovitskiy, Beyer, Kolesnikov, and Weissenborn et al., 2021]

VISION TRANSFORMER

- We can also measure the distance between a patch that attends to another patch.
- If we plot the average distance as a function of the layer:
- Patch tokens in the earlier layers attend to patches that are both near and far.
- But patch tokens in deeper layers tend to attend to patches that are further away.



[Dosovitskiy, Beyer, Kolesnikov, and Weissenborn et al., 2021]

CLIP

- But what if we want a model to take both images and text as input?
  - And then perform tasks as a function of both the image and text?
- We need some sort of shared representation of both images and text.
- The image class label does not provide much information about the image.
  - A text description of the image would be much better.
- Idea:

  Learn image and text representations jointly in a shared embedding space.
  - Learn an image encoder $f_I(x)$ -> z_I.
  - Learn a text encoder $f_T(x)$ -> z_T.

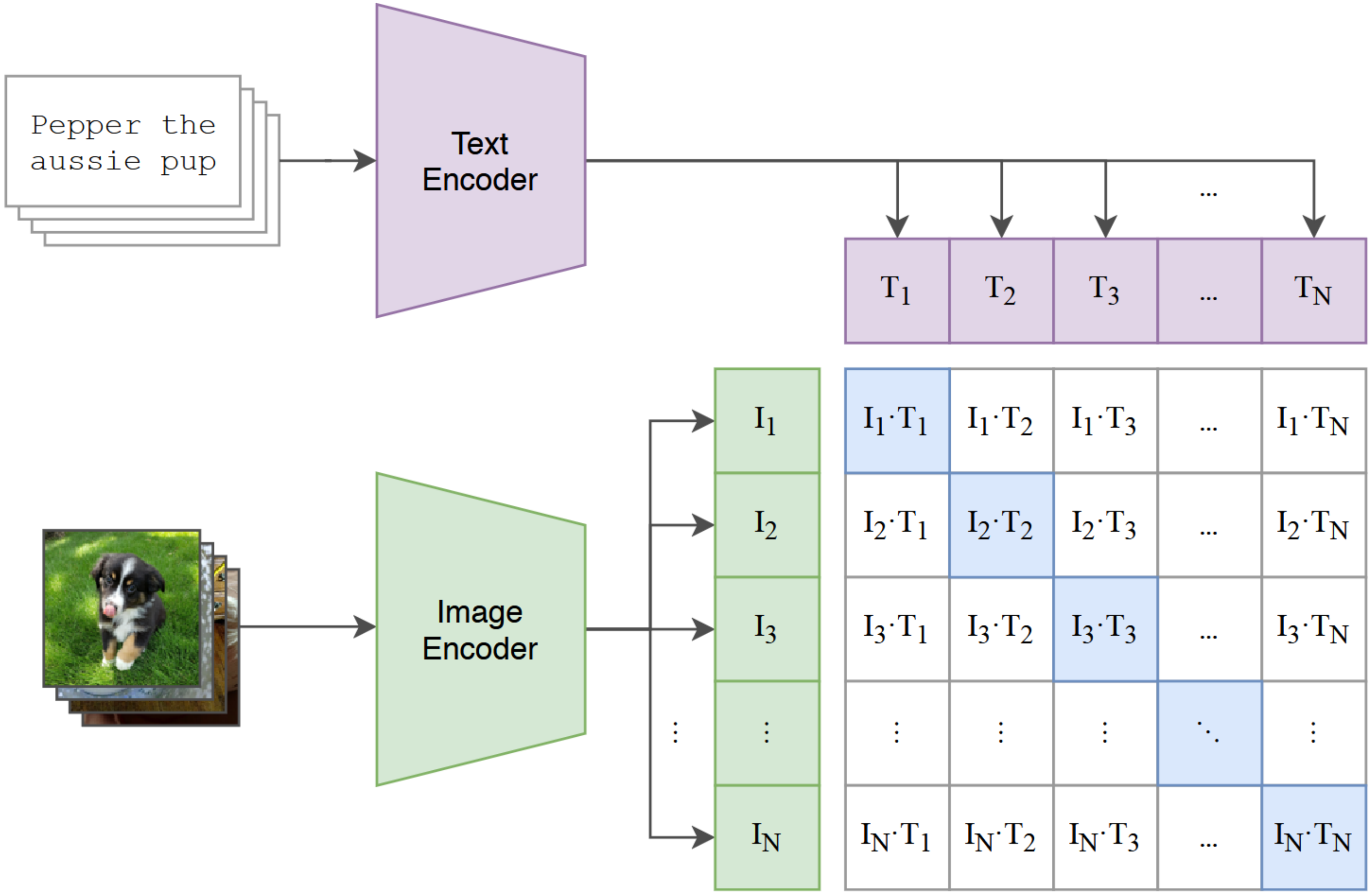
CLIP

- Idea:
    - Learn an image encoder $f_I(\boldsymbol{x})$ -> $\boldsymbol{z}_I$.
        - E.g., vision transformer
    - Learn a text encoder $f_T(\boldsymbol{x})$ -> $\boldsymbol{z}_T$.
        - E.g., transformer
    - Such that for any image with corresponding text description, the representations, $\boldsymbol{z}_I$ and $\boldsymbol{z}_T$, should be close.
    - And for any image and unrelated text, the representations should be far.
- Learn the mappings given a sufficiently large dataset of (image, text description) pairs.

CLIP

- How do we train a model to produce such text and image representations?
    - What would the loss function look like?
- What if we took N image-text pairs,
  encoded the images with the image encoder,
  encoded the text descriptions with the text encoder,
  maximize the cosine similarity of the correct image-text embedding pairs,
  and minimize the cosine similarity of the incorrect image-text pairs.
- This approach is called CLIP (Contrastive Language-Image Pretraining; Radford and Kim et al., 2021).

CLIP



[Radford and Kim et al., 2021]

CLIP

- The loss function is:

$$-\frac{1}{N}\sum_i \ln \frac{e^{v_i \cdot w_i / T}}{\sum_j e^{v_i \cdot w_j / T}} - \frac{1}{N}\sum_j \ln \frac{e^{v_j \cdot w_j / T}}{\sum_i e^{v_i \cdot w_j / T}}$$

- The sum is over the N image-text pairs.
  - v_i is the embedding of the ith image.
  - w_j is the embedding of the jth text description.
- T is a temperature parameter.

[wikipedia.org/wiki/Contrastive_Language-Image_Pre-training]

CLIP

- The loss function is:

$$-\frac{1}{N}\sum_i \ln \frac{\boxed{e^{v_i \cdot w_i / T}}}{\sum_j e^{v_i \cdot w_j / T}} - \frac{1}{N}\sum_j \ln \frac{\boxed{e^{v_j \cdot w_j / T}}}{\sum_i e^{v_i \cdot w_j / T}}$$

- By minimizing the loss, we are maximizing the vector similarity between v_i and w_i (and v_j and w_j).

[wikipedia.org/wiki/Contrastive_Language-Image_Pre-training]

CLIP

- The loss function is:

$$-\frac{1}{N}\sum_i \ln \frac{e^{v_i \cdot w_i / T}}{\sum_j \boxed{e^{v_i \cdot w_j / T}}} - \frac{1}{N}\sum_j \ln \frac{e^{v_j \cdot w_j / T}}{\sum_i \boxed{e^{v_i \cdot w_j / T}}}$$

- And we are simultaneously minimizing the vector similarity between other image-text description pairs: v_i and w_j.
- Each term is actually a cross-entropy loss, where the model's log probability (logit) for the pair `(i,j)` is $v_i^{\mathrm{T}} w_j / T$ and the correct label is `(i,i)`.

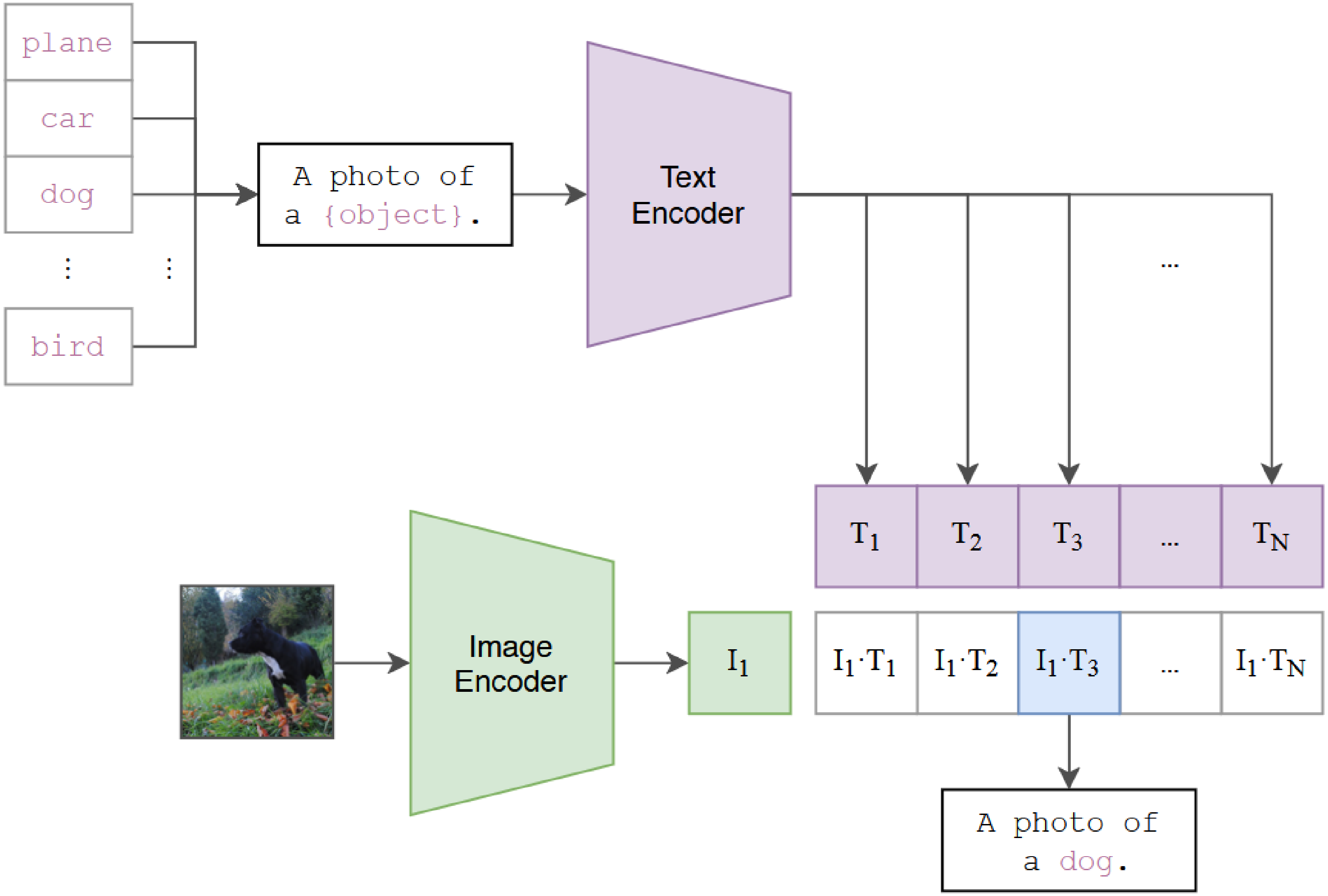
[wikipedia.org/wiki/Contrastive_Language-Image_Pre-training]

CLIP

- We can apply the learned embeddings for other downstream tasks.

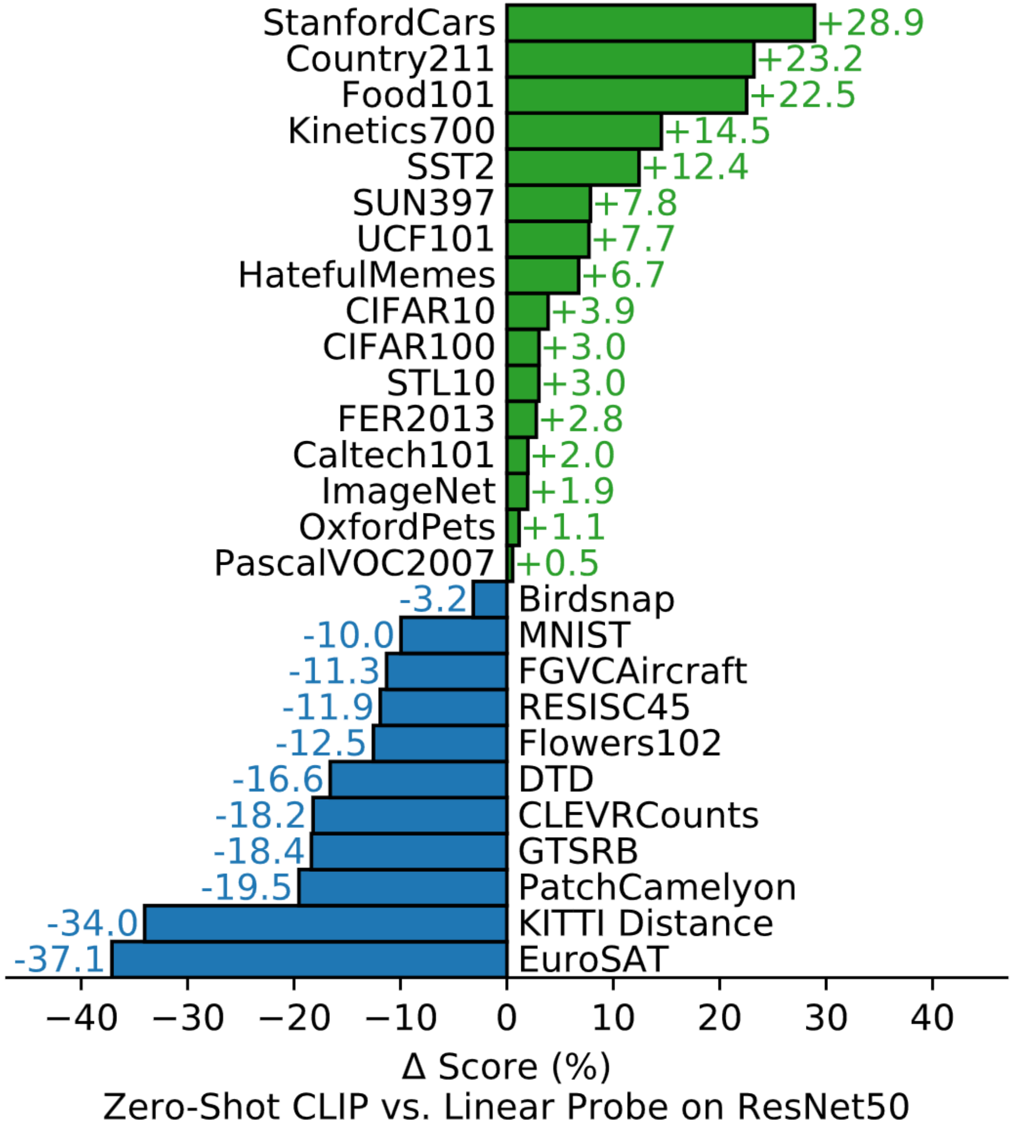
  (akin to how we can use BERT + a linear layer for downstream NLP tasks)

- For example, “zero-shot” image classification:



CLIP

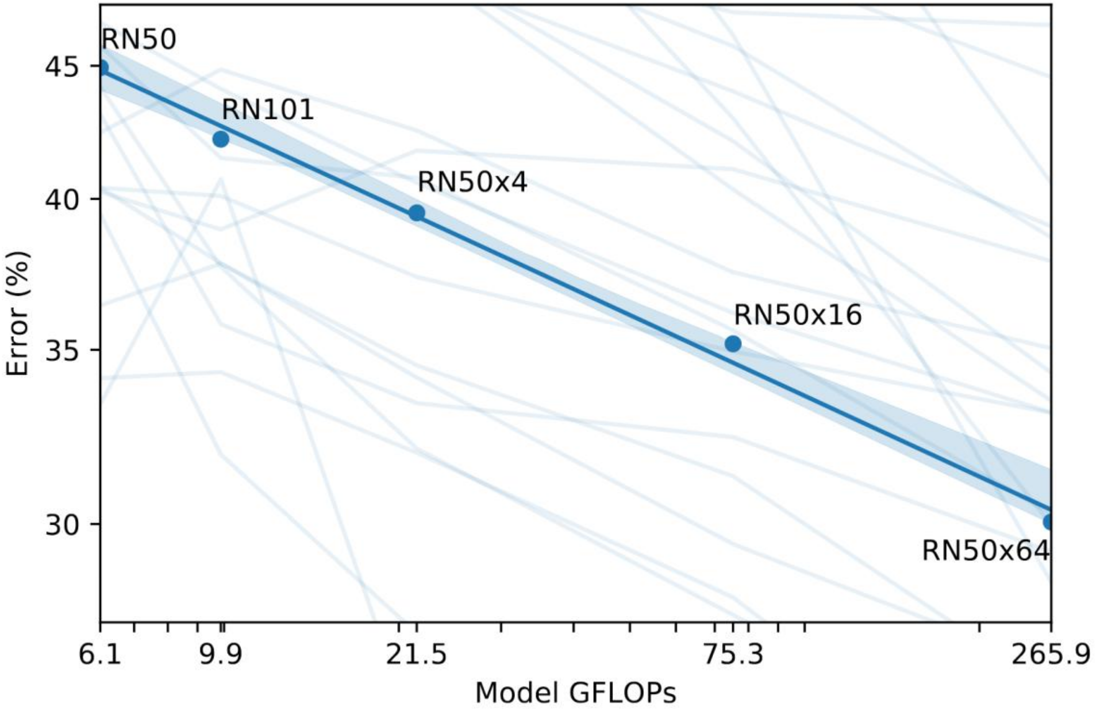
- Compare CLIP zero-shot image classification to a fully-supervised baseline:
  - Test on several different datasets.
- Performs better than supervised baseline on many datasets,
- But also not as well on other datasets.



[Radford and Kim et al., 2021]

CLIP

- Average error rate scales smoothly as a function of training compute.
  - Light blue curves are error rates on individual datasets.
  - Note: y-axis is not loss, unlike LM scaling laws.



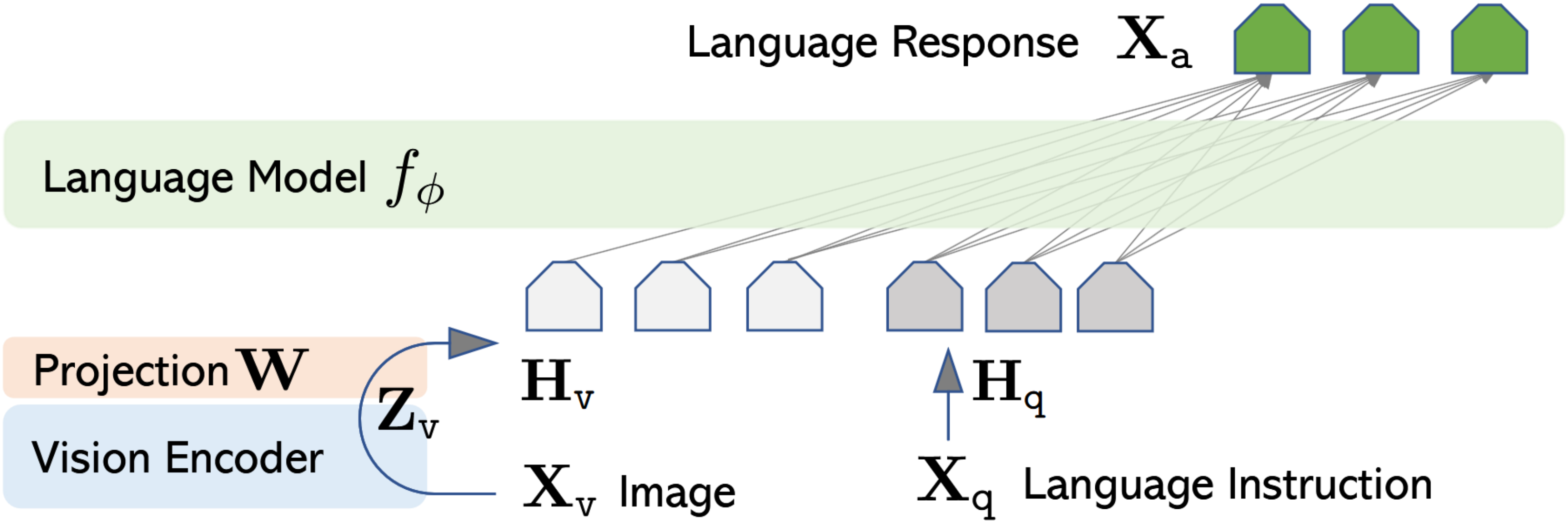
[Radford and Kim et al., 2021]

VISION + LANGUAGE MODELING?

- What if we want to generate text, as in autoregressive language modeling?
  - Where the input is images + text.
- Idea: Start with an instruction-tuned LLM (decoder-only).
  - Take the embeddings from an image encoder.
  - Use a linear layer to project these embeddings into the same dimension as the word embedding space of the LLM.
    - This is often called the connector, and other architectures can be used.
  - Fine-tune the linear layer and LLM (image encoder is frozen).
- This was the idea behind LLaVA (Large Language and Vision Assistant; Liu and Li et al., 2023).

LLAVA

- LLaVA uses Vicuna as the LM and CLIP as the image encoder.
  - The output of CLIP is Z_v,
  - W is the linear transformation that converts CLIP embeddings into Vicuna token embeddings.



[Liu and Li et al., 2023]

LLAVA

- LLaVA is fine-tuned using conversation-formatted data:
  - Each example looks like:

```
User: [image] [question text 1]
Model: [response text 1]
User: [question text 2]
Model: [response text 2]
...
```

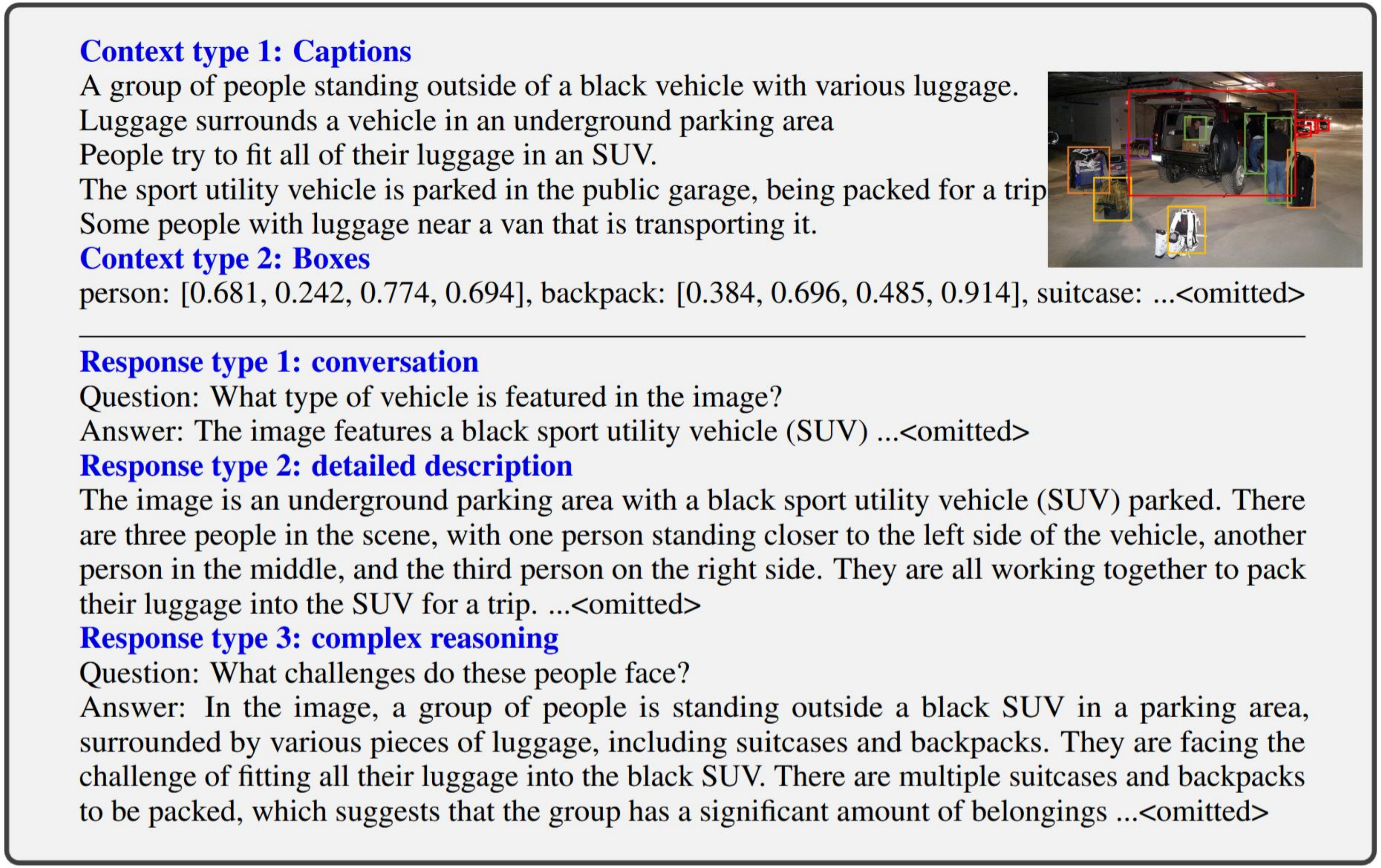
- They convert regular image caption data into this format:

```
User: [image] Describe this image briefly.
Model: [image caption]
```

LLAVA

- They split fine-tuning into two steps:
  - First, they fine-tune only the W projection (between CLIP outputs and Vicuna inputs).
    - LLM and CLIP weights are frozen.
  - Second, they fine-tune the LM and W jointly.
    - They use a larger dataset for this stage.
    - They use text-only GPT4 and ChatGPT to help generate this data.
    - Some image datasets include bounding box annotations:
      - E.g., 'there is a car from position (413px, 677px) to (782px, 1209px).'
    - We can describe the contents of the image entirely in text using these bounding boxes.

DATA GENERATION FOR LLAVA



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

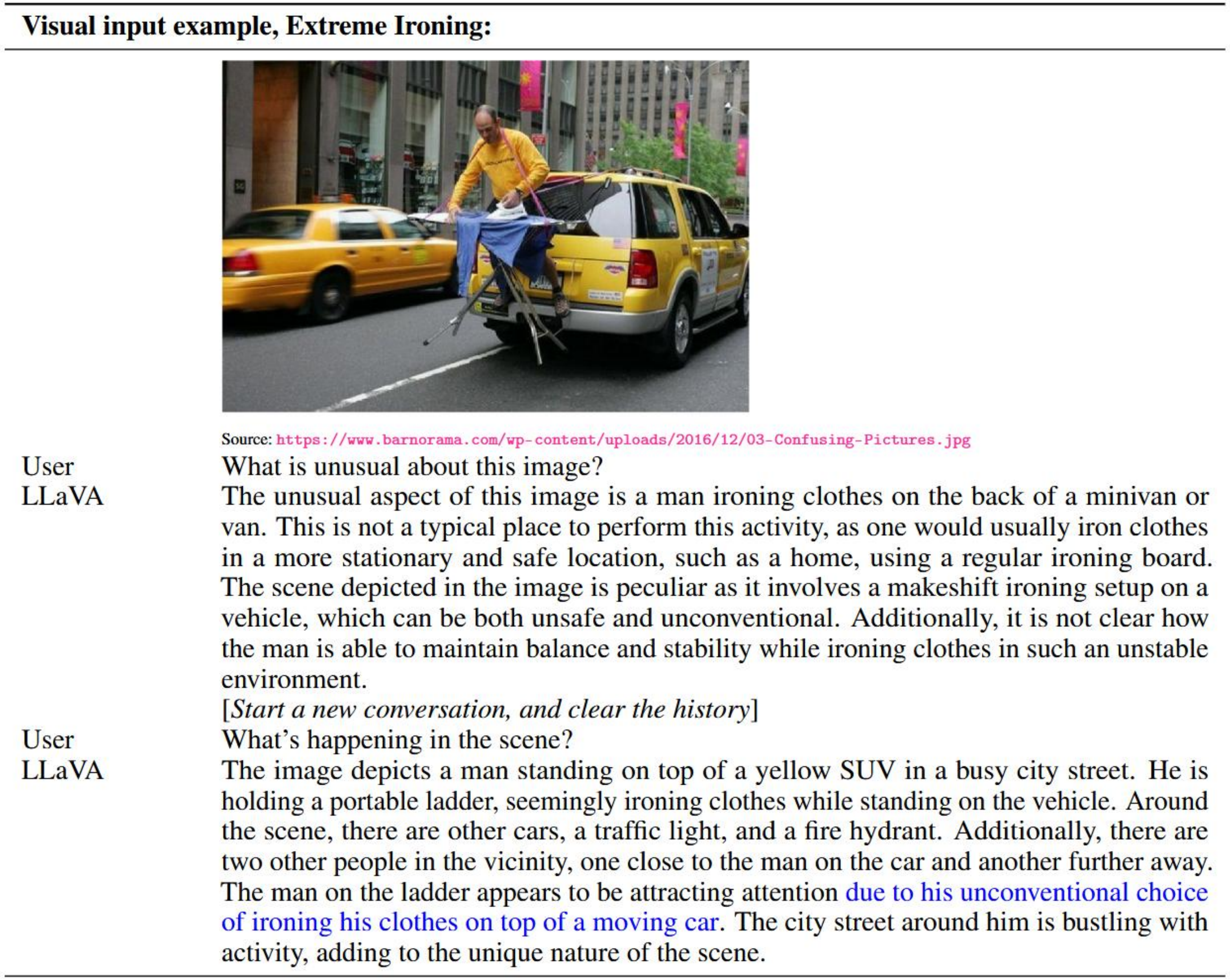
The image is an underground parking area with a black sport utility vehicle (SUV) parked. There are three people in the scene, with one person standing closer to the left side of the vehicle, another person in the middle, and the third person on the right side. They are all working together to pack their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

[Liu and Li et al., 2023]

LLAVA EXAMPLE



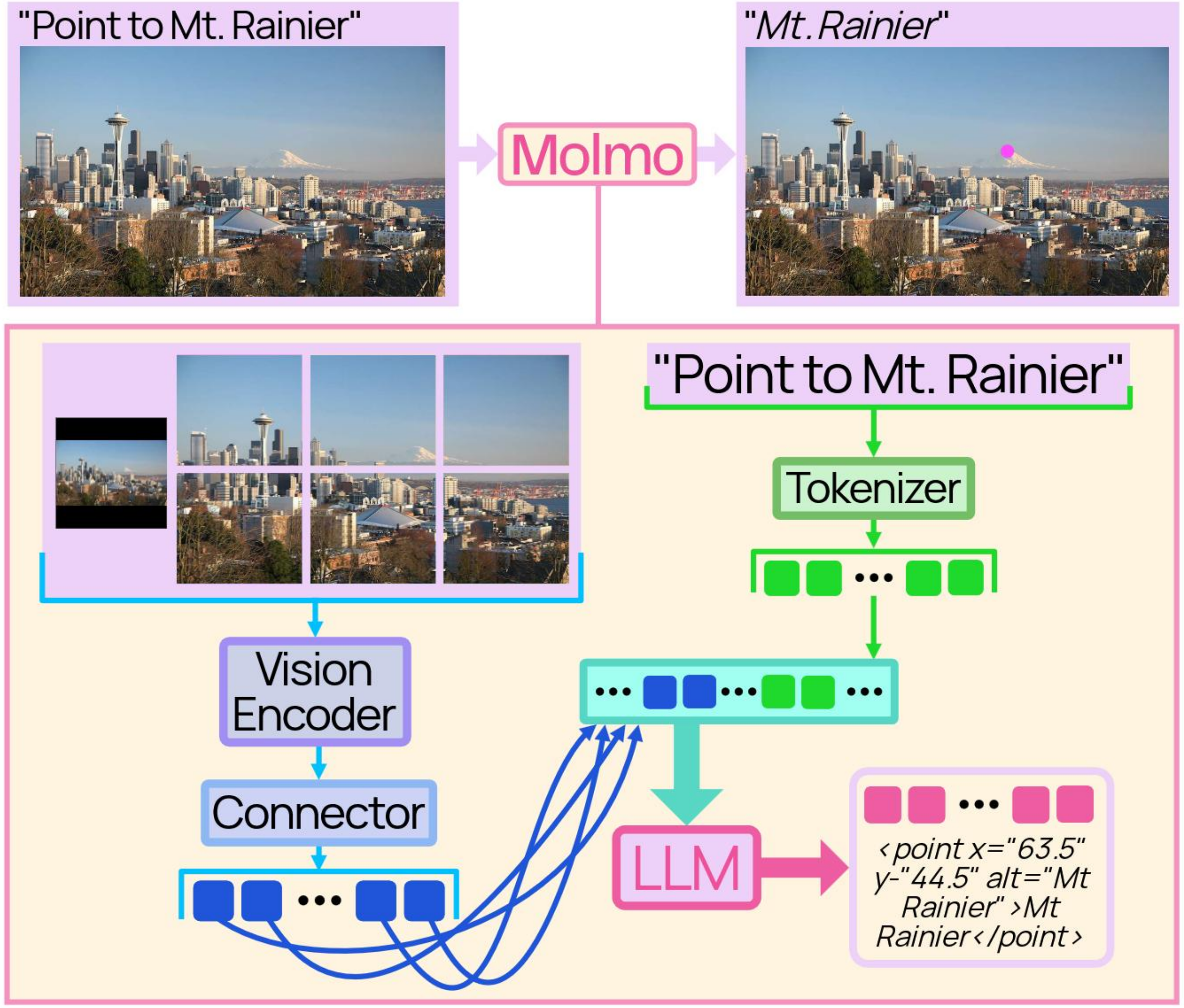
Visual input example, Extreme Ironing:

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

User	What is unusual about this image?
LLaVA	The unusual aspect of this image is a man ironing clothes on the back of a minivan or van. This is not a typical place to perform this activity, as one would usually iron clothes in a more stationary and safe location, such as a home, using a regular ironing board. The scene depicted in the image is peculiar as it involves a makeshift ironing setup on a vehicle, which can be both unsafe and unconventional. Additionally, it is not clear how the man is able to maintain balance and stability while ironing clothes in such an unstable environment.
	[*Start a new conversation, and clear the history*]
User	What's happening in the scene?
LLaVA	The image depicts a man standing on top of a yellow SUV in a busy city street. He is holding a portable ladder, seemingly ironing clothes while standing on the vehicle. Around the scene, there are other cars, a traffic light, and a fire hydrant. Additionally, there are two other people in the vicinity, one close to the man on the car and another further away. The man on the ladder appears to be attracting attention due to his unconventional choice of ironing his clothes on top of a moving car. The city street around him is bustling with activity, adding to the unique nature of the scene.

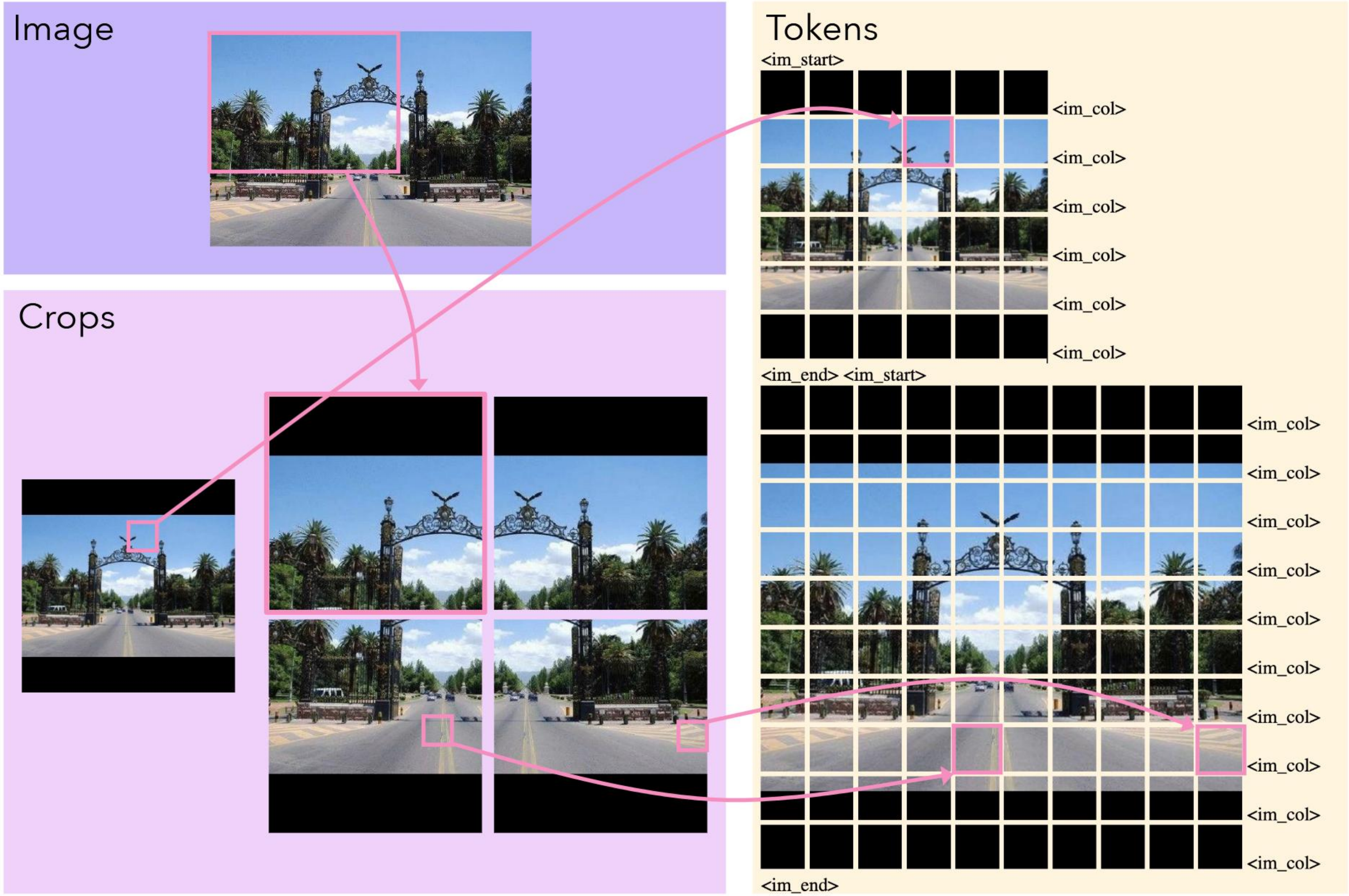
[Liu and Li et al., 2023]

MOLMO



- More recently, Deitke and Clark et al. (2024) developed Molmo (Multimodal Open Language Model).
- They followed the same recipe of connecting an image encoder with an LM.
  - They also use CLIP for the image encoder.
  - They try different LMs:
    - `OLMo-7B`, `OLMoE-1B-7B`
    - `Qwen2 7B`, `Qwen2 72B`
- Open source!

- Image tokenization:
- To enable handling of higher-resolution images with non-1:1 aspect ratio,
- Add black bars to top/bottom/sides so make the aspect ratio 1:1.
- Downscale the image into lower resolution before tokenization.
- Also crop the image (while maintaining higher resolution).



[Deitke and Clark et al., 2024]

MOLMO VISION-LANGUAGE CONNECTOR

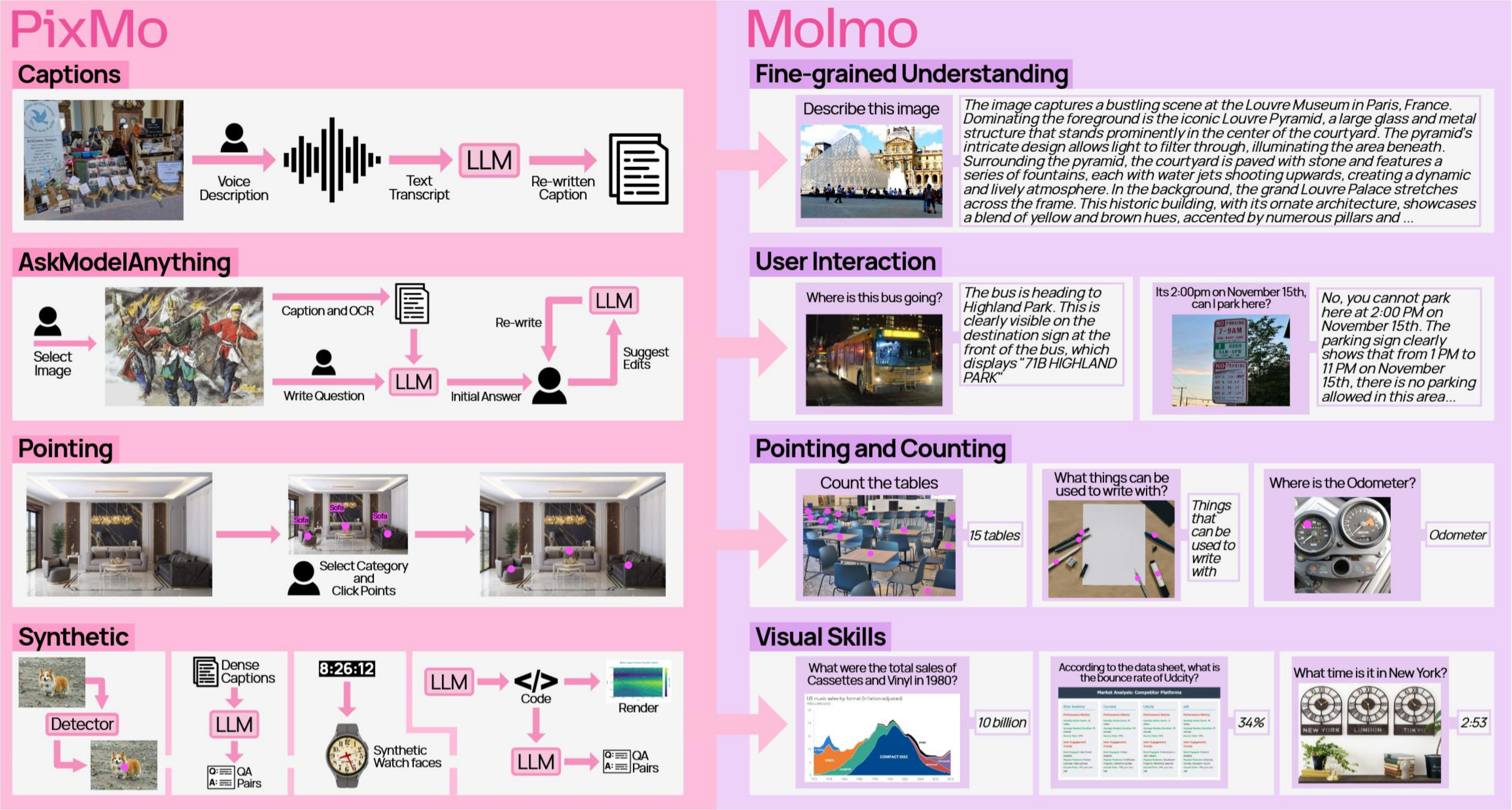
- Their connector architecture is interesting:
  - Instead of taking the patch embeddings from the last layer of CLIP,
  - They use the embeddings from the 3rd-to-last and 10th-to-last layers.
    (empirically, they found this performed better for some reason)
  - For each layer, collect each set of 2x2 patches and apply a multi-head attention layer.
  - The key and value vectors come from the 4 patches,
  - There's only 1 query vector, which is the mean of the 4 patches.
  - Finally use a linear layer to map the output vector into the LLM's model dimension.

MOLMO TRAINING DATA

- To train Molmo, they collected a large high-quality multimodal dataset.
- They called it PixMo (Pixels for Molmo).
- They gathered images from the internet.
  - Ask 3 annotators to provide spoken descriptions (at least 60 seconds).
  - But this was too slow,
    - So later they switch to 1 annotator per image (at least 90 seconds).
  - They prompted annotators with 7 questions to answer per image.
  - The speech was transcribed using speech-to-text, and an LLM summarized the transcripts to produce a caption.
- 712k images, and 1.3M transcripts and captions.
- Also open source!

PIXMO

- They further augmented their data for specific tasks:
    - Image question-answering
    - Pointing (e.g, ‘`point to Mount Rainier`’ in the given image)
    - Caption question-answering
    - Time-telling question-answering (read an image of a clock)
    - Counting (e.g., ‘`how many people are in this image?`’)
    - Reading charts/plots
- Some of these augmentations relied on further human annotation.
    - Many relied on text-only LLMs.



The image captures a bustling scene at the Louvre Museum in Paris, France. Dominating the foreground is the iconic Louvre Pyramid, a large glass and metal structure that stands prominently in the center of the courtyard. The pyramid's intricate design allows light to filter through, illuminating the area beneath. Surrounding the pyramid, the courtyard is paved with stone and features a series of fountains, each with water jets shooting upwards, creating a dynamic and lively atmosphere. In the background, the grand Louvre Palace stretches across the frame. This historic building, with its ornate architecture, showcases a blend of yellow and brown hues, accented by numerous pillars and ...

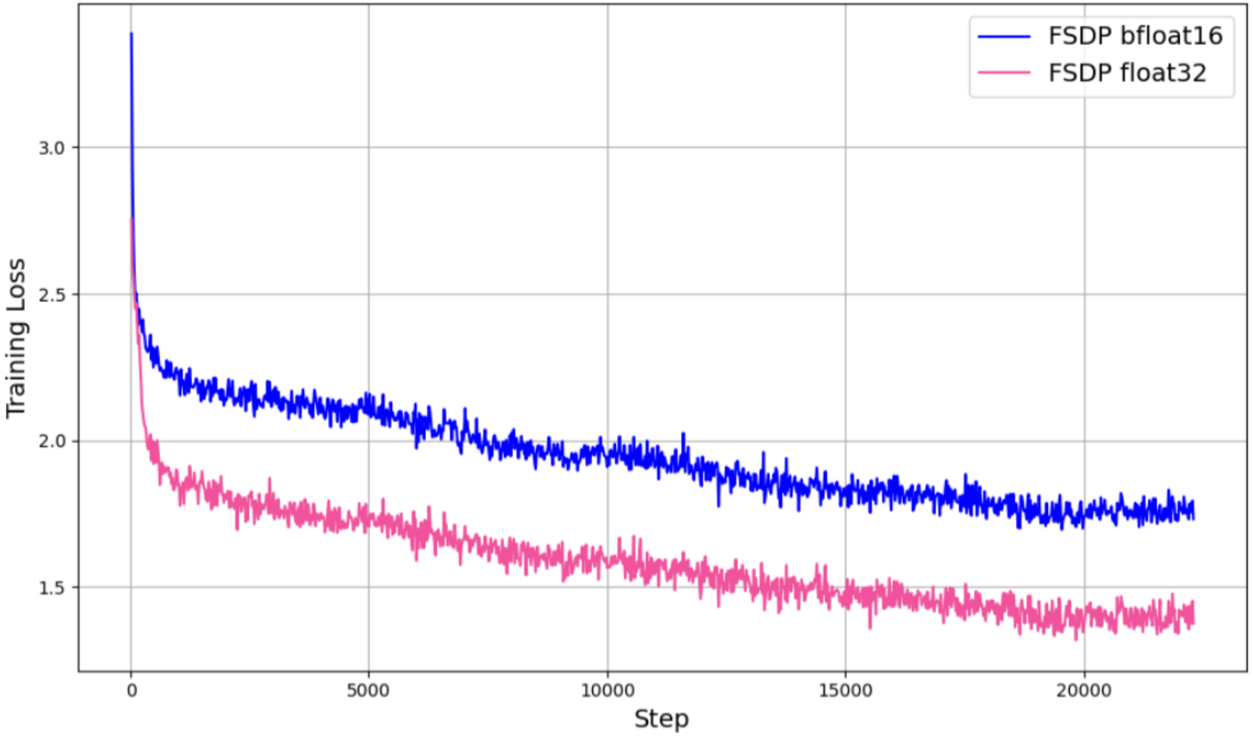
The bus is heading to Highland Park. This is clearly visible on the destination sign at the front of the bus, which displays "71B HIGHLAND PARK"

No, you cannot park here at 2:00 PM on November 15th. The parking sign clearly shows that from 1 PM to 11 PM on November 15th, there is no parking allowed in this area...

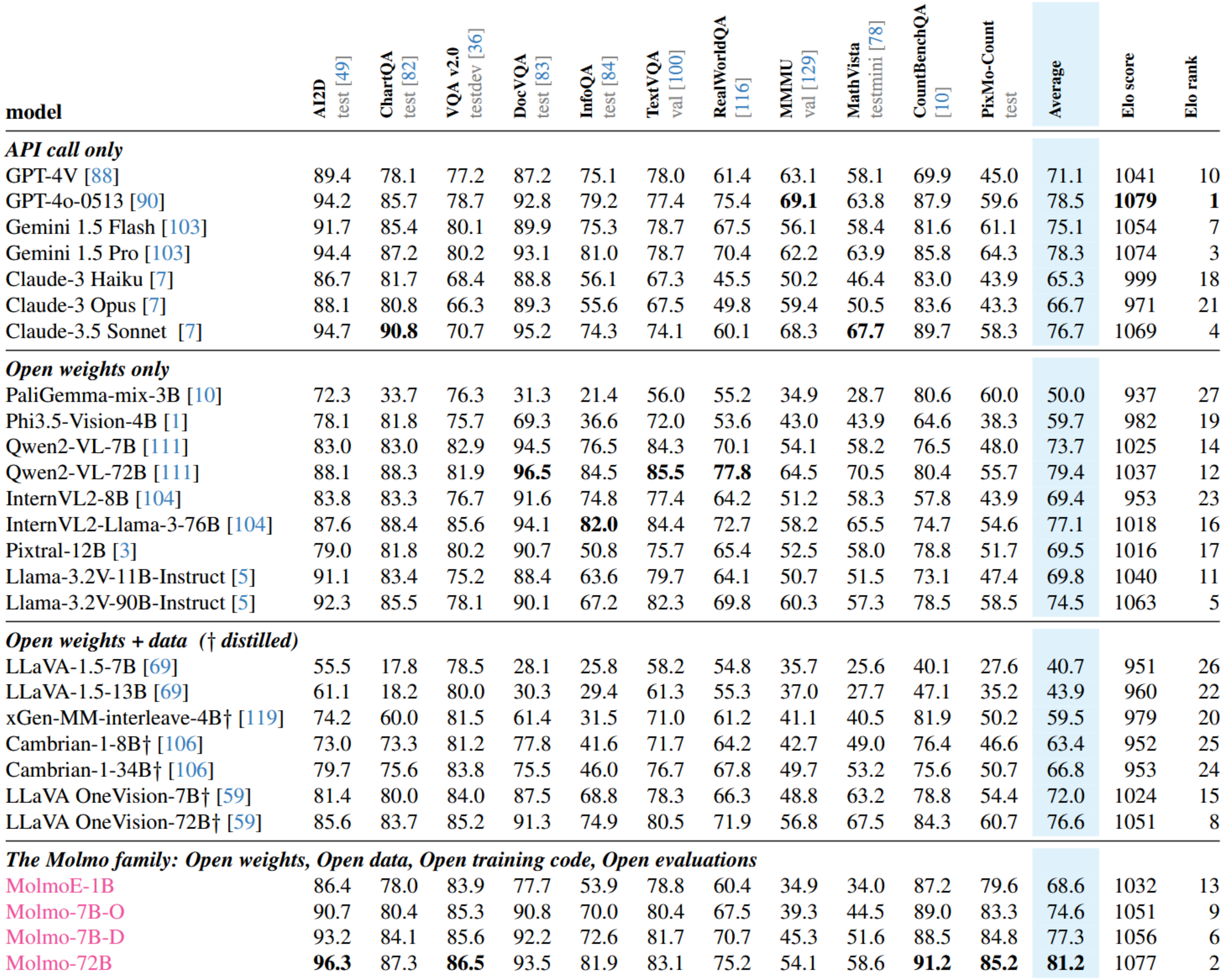
[Deitke and Clark et al., 2024]

TRAINING MOLMO

- Unlike LLaVA, Molmo is pretrained end-to-end.
    - All three components, ViT, connector, and LM, are trained simultaneously.
    - But with different learning rates for each.
    - Pretraining is done only on the image-caption data.
- After pretraining, they fine-tune on the full PixMo dataset + other open-source visual QA datasets.

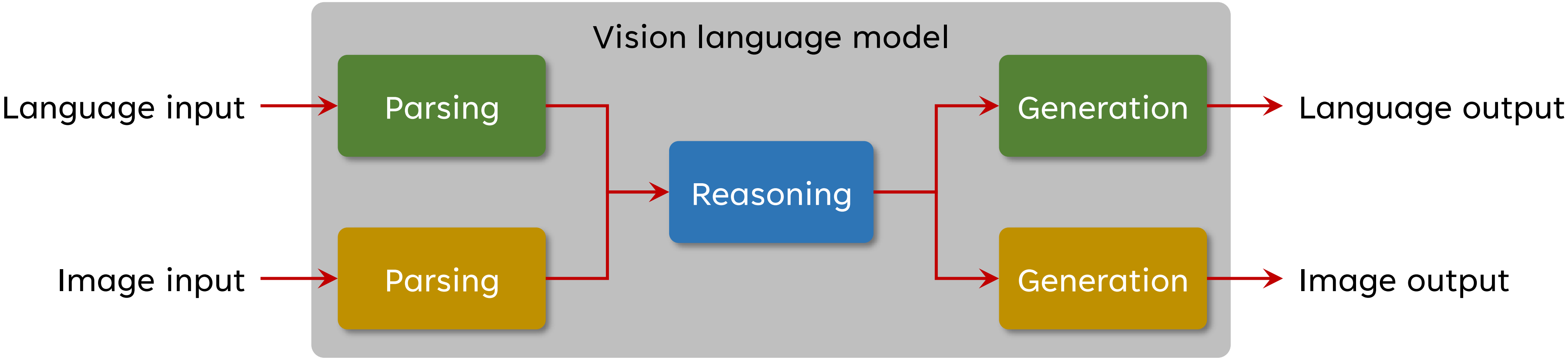


[Deitke and Clark et al., 2024]



model	**AI2D** test [49]	**ChartQA** test [82]	**VQA v2.0** testdev [36]	**DocVQA** test [83]	**InfoQA** test [84]	**TextVQA** val [100]	**RealWorldQA** [116]	**MMMU** val [129]	**MathVista** testmini [78]	**CountBenchQA** [10]	**PixMo-Count** test	**Average**	**Elo score**	**Elo rank**
API call only														
GPT-4V [88]	89.4	78.1	77.2	87.2	75.1	78.0	61.4	63.1	58.1	69.9	45.0	71.1	1041	10
GPT-4o-0513 [90]	94.2	85.7	78.7	92.8	79.2	77.4	75.4	**69.1**	63.8	87.9	59.6	78.5	**1079**	**1**
Gemini 1.5 Flash [103]	91.7	85.4	80.1	89.9	75.3	78.7	67.5	56.1	58.4	81.6	61.1	75.1	1054	7
Gemini 1.5 Pro [103]	94.4	87.2	80.2	93.1	81.0	78.7	70.4	62.2	63.9	85.8	64.3	78.3	1074	3
Claude-3 Haiku [7]	86.7	81.7	68.4	88.8	56.1	67.3	45.5	50.2	46.4	83.0	43.9	65.3	999	18
Claude-3 Opus [7]	88.1	80.8	66.3	89.3	55.6	67.5	49.8	59.4	50.5	83.6	43.3	66.7	971	21
Claude-3.5 Sonnet [7]	94.7	**90.8**	70.7	95.2	74.3	74.1	60.1	68.3	**67.7**	89.7	58.3	76.7	1069	4
Open weights only														
PaliGemma-mix-3B [10]	72.3	33.7	76.3	31.3	21.4	56.0	55.2	34.9	28.7	80.6	60.0	50.0	937	27
Phi3.5-Vision-4B [1]	78.1	81.8	75.7	69.3	36.6	72.0	53.6	43.0	43.9	64.6	38.3	59.7	982	19
Qwen2-VL-7B [111]	83.0	83.0	82.9	94.5	76.5	84.3	70.1	54.1	58.2	76.5	48.0	73.7	1025	14
Qwen2-VL-72B [111]	88.1	88.3	81.9	**96.5**	84.5	**85.5**	**77.8**	64.5	70.5	80.4	55.7	79.4	1037	12
InternVL2-8B [104]	83.8	83.3	76.7	91.6	74.8	77.4	64.2	51.2	58.3	57.8	43.9	69.4	953	23
InternVL2-Llama-3-76B [104]	87.6	88.4	85.6	94.1	**82.0**	84.4	72.7	58.2	65.5	74.7	54.6	77.1	1018	16
Pixtral-12B [3]	79.0	81.8	80.2	90.7	50.8	75.7	65.4	52.5	58.0	78.8	51.7	69.5	1016	17
Llama-3.2V-11B-Instruct [5]	91.1	83.4	75.2	88.4	63.6	79.7	64.1	50.7	51.5	73.1	47.4	69.8	1040	11
Llama-3.2V-90B-Instruct [5]	92.3	85.5	78.1	90.1	67.2	82.3	69.8	60.3	57.3	78.5	58.5	74.5	1063	5
Open weights + data († distilled)														
LLaVA-1.5-7B [69]	55.5	17.8	78.5	28.1	25.8	58.2	54.8	35.7	25.6	40.1	27.6	40.7	951	26
LLaVA-1.5-13B [69]	61.1	18.2	80.0	30.3	29.4	61.3	55.3	37.0	27.7	47.1	35.2	43.9	960	22
xGen-MM-interleave-4B† [119]	74.2	60.0	81.5	61.4	31.5	71.0	61.2	41.1	40.5	81.9	50.2	59.5	979	20
Cambrian-1-8B† [106]	73.0	73.3	81.2	77.8	41.6	71.7	64.2	42.7	49.0	76.4	46.6	63.4	952	25
Cambrian-1-34B† [106]	79.7	75.6	83.8	75.5	46.0	76.7	67.8	49.7	53.2	75.6	50.7	66.8	953	24
LLaVA OneVision-7B† [59]	81.4	80.0	84.0	87.5	68.8	78.3	66.3	48.8	63.2	78.8	54.4	72.0	1024	15
LLaVA OneVision-72B† [59]	85.6	83.7	85.2	91.3	74.9	80.5	71.9	56.8	67.5	84.3	60.7	76.6	1051	8
The Molmo family: Open weights, Open data, Open training code, Open evaluations														
MolmoE-1B	86.4	78.0	83.9	77.7	53.9	78.8	60.4	34.9	34.0	87.2	79.6	68.6	1032	13
Molmo-7B-O	90.7	80.4	85.3	90.8	70.0	80.4	67.5	39.3	44.5	89.0	83.3	74.6	1051	9
Molmo-7B-D	93.2	84.1	85.6	92.2	72.6	81.7	70.7	45.3	51.6	88.5	84.8	77.3	1056	6
Molmo-72B	**96.3**	87.3	**86.5**	93.5	81.9	83.1	75.2	54.1	58.6	**91.2**	**85.2**	**81.2**	1077	2

HOW DO WE GENERATE TEXT+IMAGES?


Vision language model

Language input

Image input

Parsing

Parsing

Reasoning

Generation

Generation

Language output

Image output

IMAGE GENERATION EXAMPLE

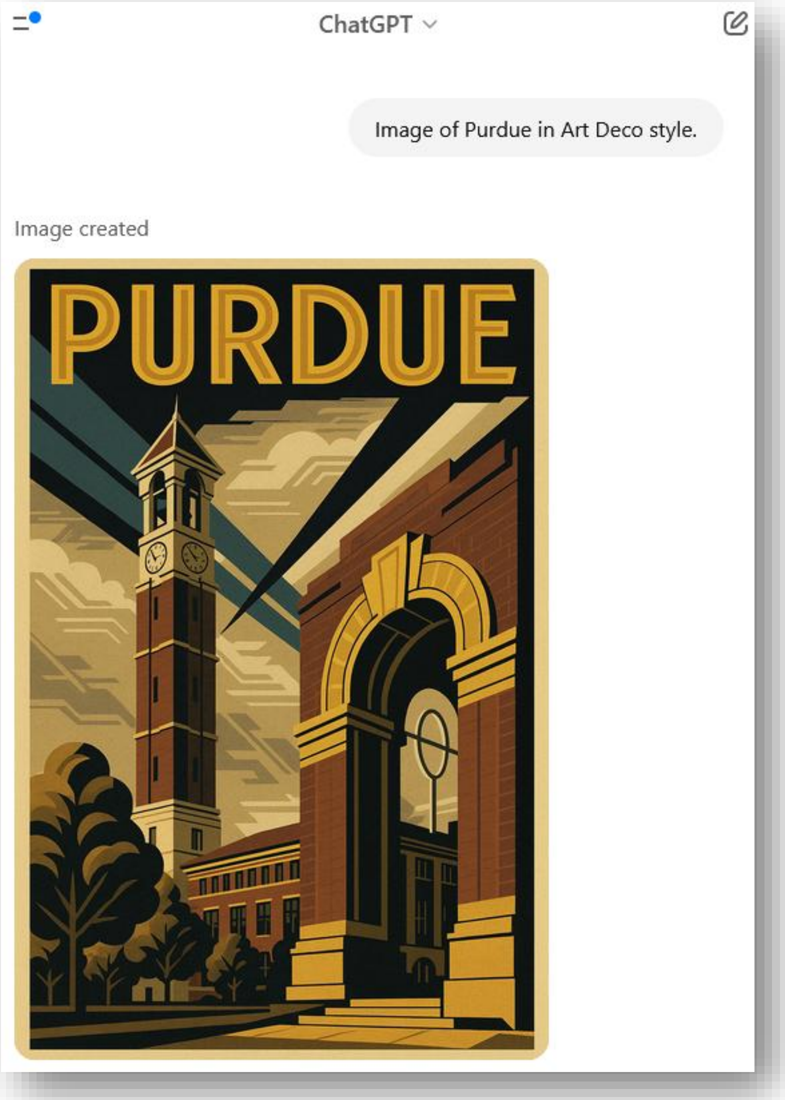
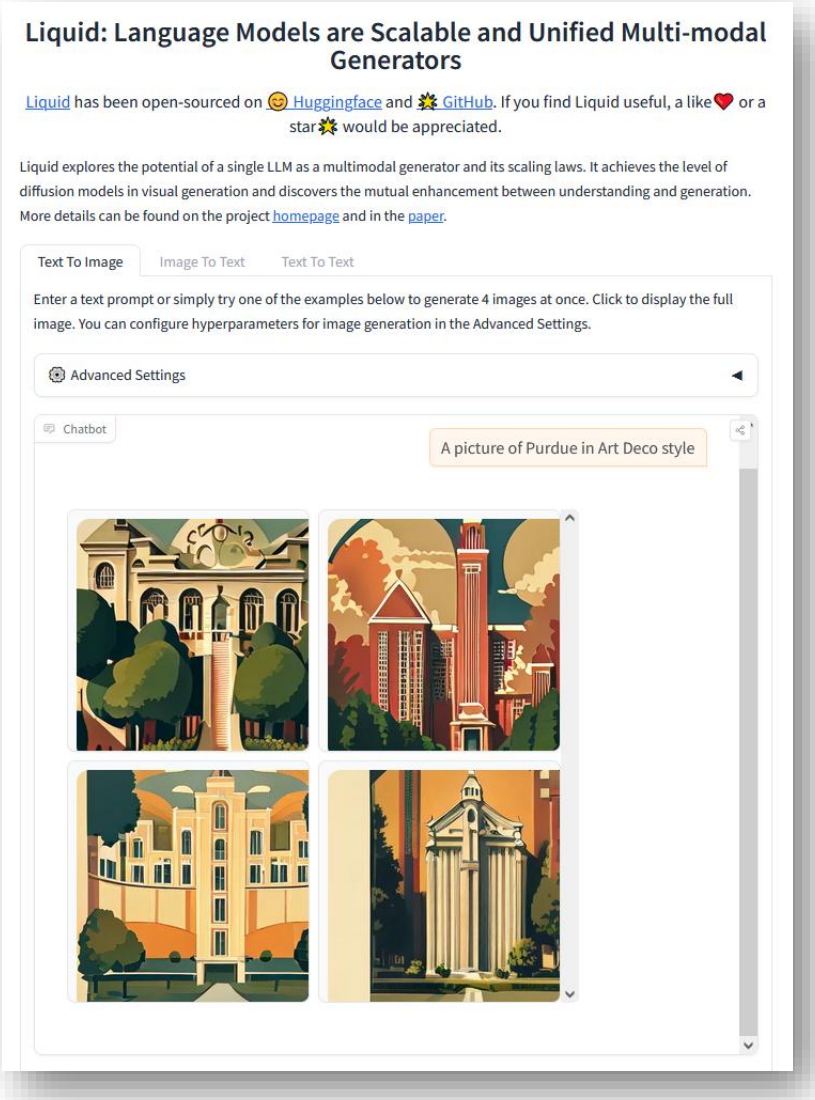


Image of Purdue in Art Deco style.

Image created

[https://chatgpt.com]

IMAGE GENERATION EXAMPLE



Liquid: Language Models are Scalable and Unified Multi-modal Generators

Liquid has been open-sourced on 😊 Huggingface and 🌟 GitHub. If you find Liquid useful, a like❤️ or a star🌟 would be appreciated.

Liquid explores the potential of a single LLM as a multimodal generator and its scaling laws. It achieves the level of diffusion models in visual generation and discovers the mutual enhancement between understanding and generation. More details can be found on the project homepage and in the paper.

Enter a text prompt or simply try one of the examples below to generate 4 images at once. Click to display the full image. You can configure hyperparameters for image generation in the Advanced Settings.

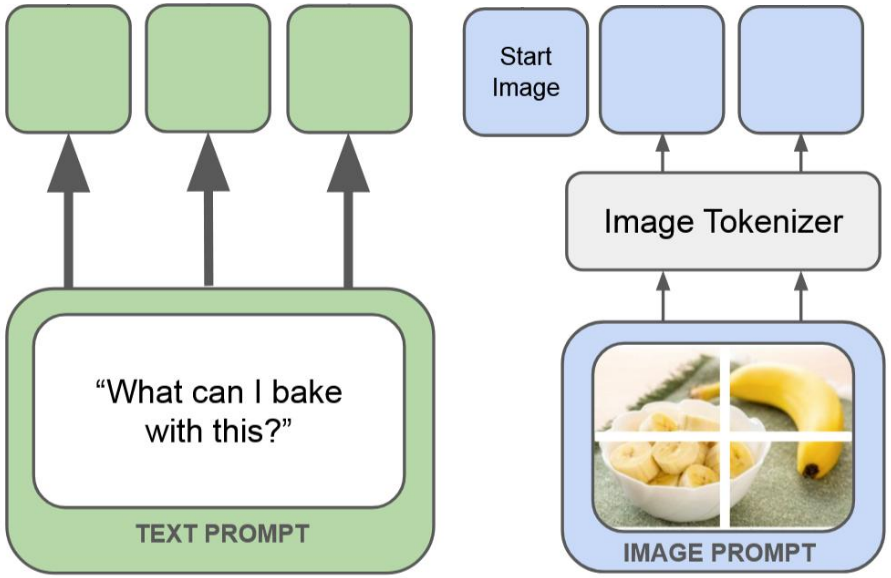
A picture of Purdue in Art Deco style

[huggingface.co/spaces/Junfeng5/Liquid_demo]

GENERATING IMAGES

- Simple idea:
  - Use an image tokenizer and detokenizer.
- We can use the tokenizer to convert multi-modal input (containing images and text)

  Into a sequence containing text and image tokens.

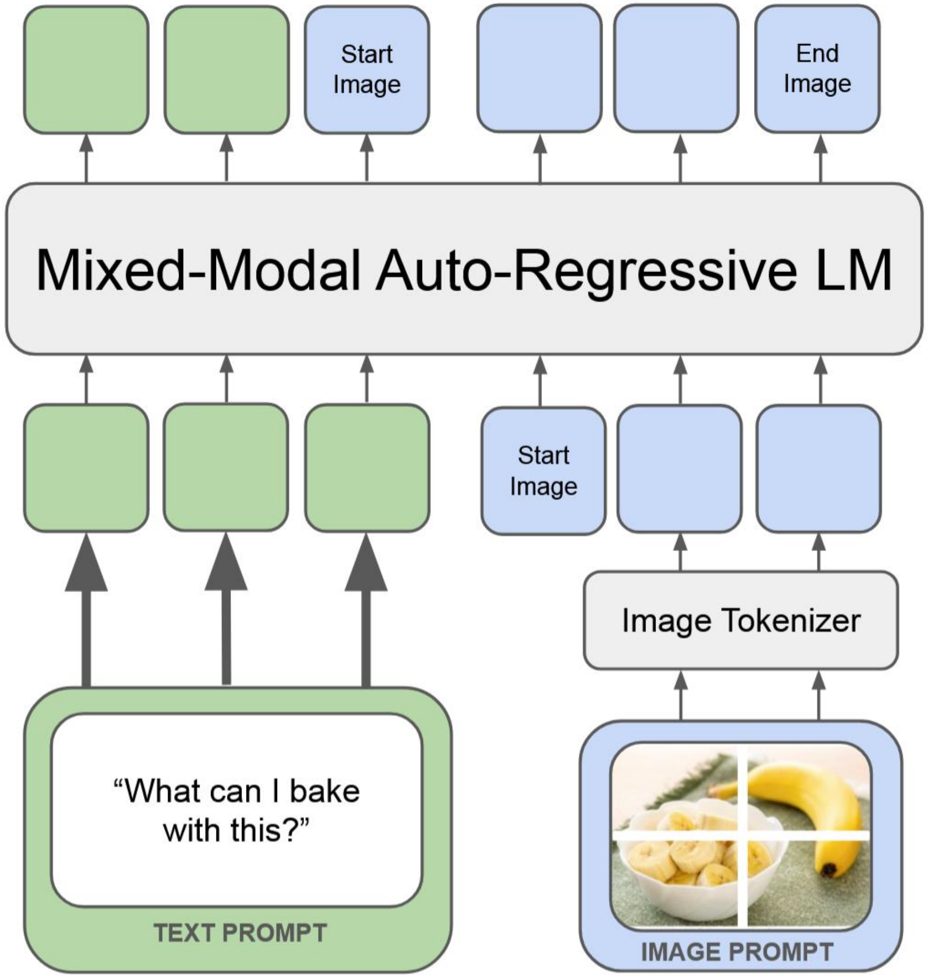


[Chameleon Team, 2025]

GENERATING IMAGES

- Simple idea:
  - Use an image tokenizer and detokenizer.
- We can use the tokenizer to convert multi-modal input (containing images and text)

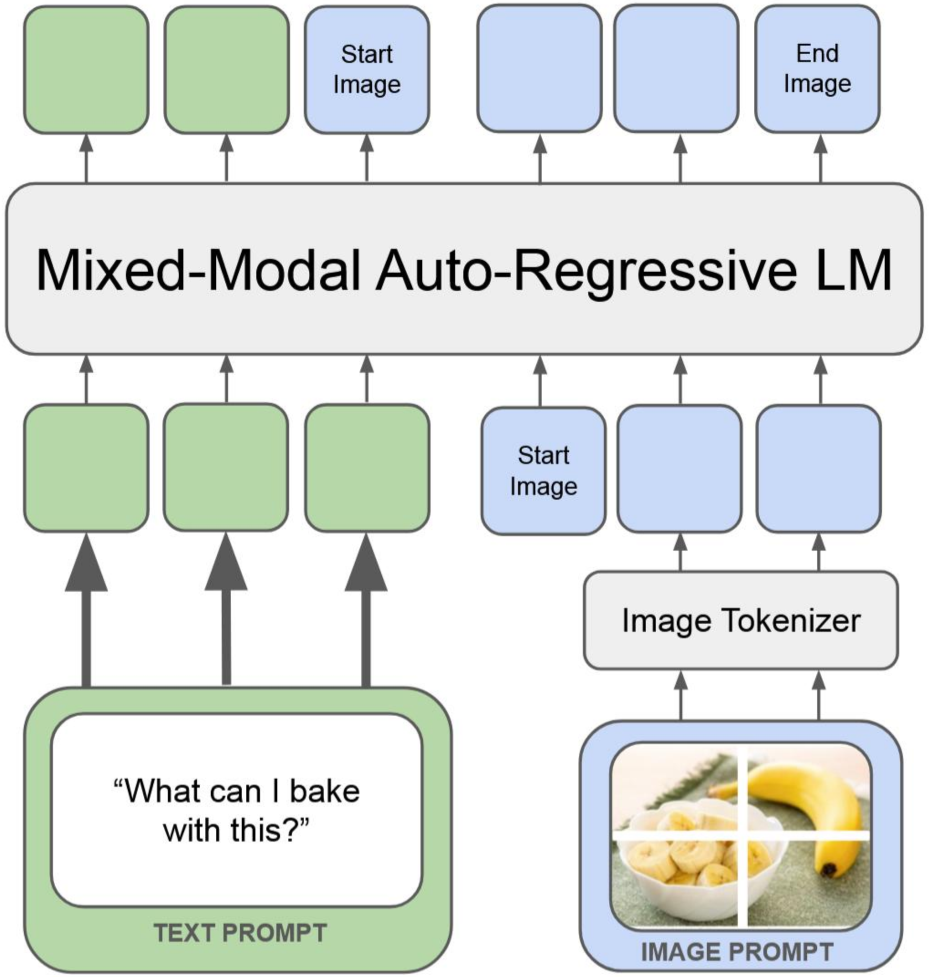
  Into a sequence containing text and image tokens.
- Then we can apply any model for auto-regressive token prediction.
  - E.g., large-scale transformer models.



[Chameleon Team, 2025]

GENERATING IMAGES

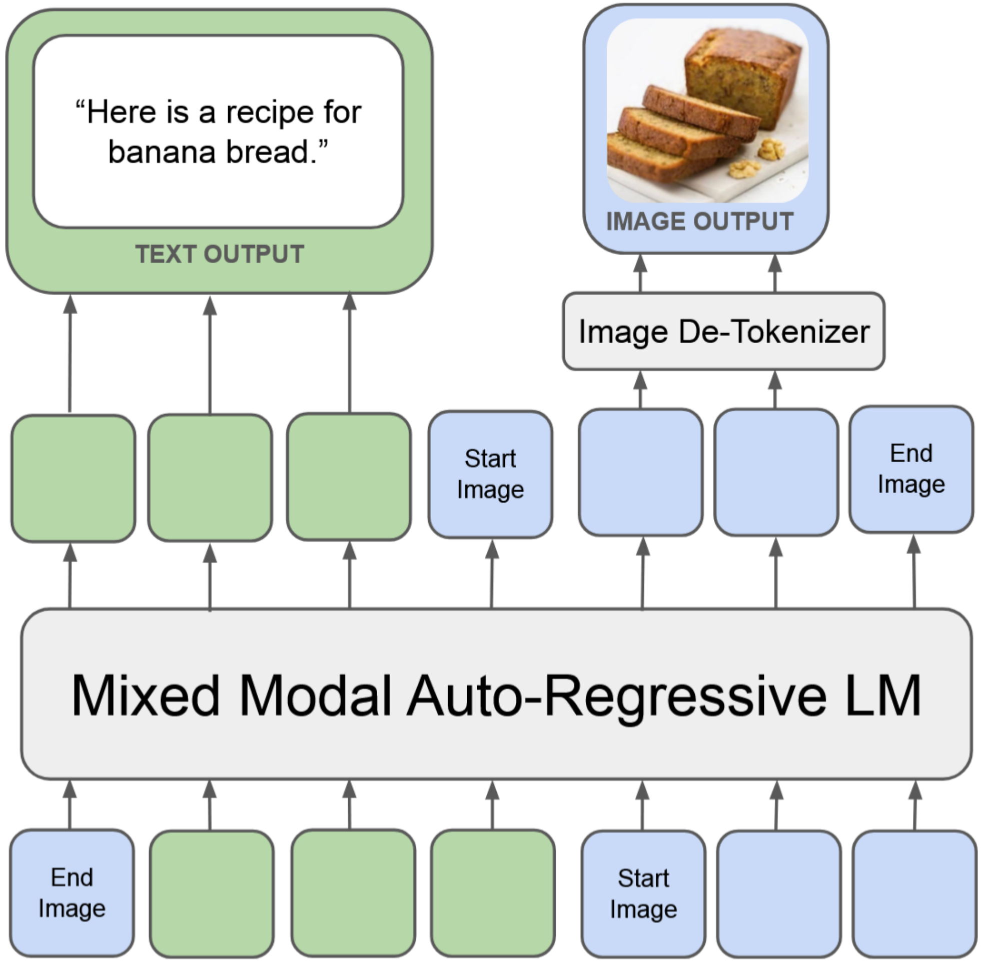
- Simple idea:
  - Use an image tokenizer and detokenizer.
- The right figure shows the setup during pretraining where the image is tokenized into 2 image tokens.
- We have special tokens denoting the beginning and end of an image.
  - `[boi]` and `[eoi]` are analogous to `[bos]` and `[eos]`.



[Chameleon Team, 2025]

GENERATING IMAGES

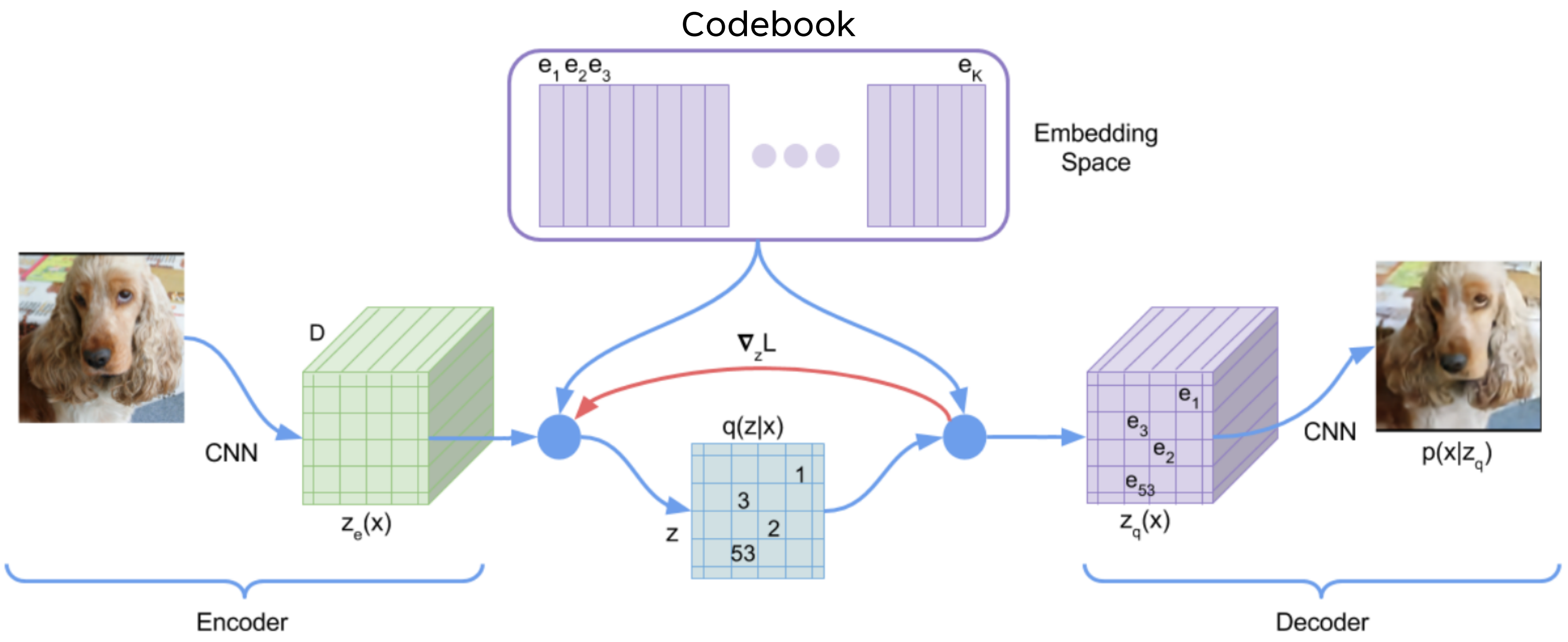
- Simple idea:
  - Use an image tokenizer and detokenizer.
- During inference, after generating the `[eoi]`, we use an image detokenizer to convert the image tokens into the output image.
- Note: This approach is distinct from using an image encoder such as CLIP.
  - Image encoders have continuous outputs.
  - Image tokenizers have discrete outputs.
- LMs produce discrete output tokens.



[Chameleon Team, 2025]

TOKENIZING IMAGES WITH AUTO-ENCODERS

- Train a model to convert (encode) an image patch into a discrete token, and then convert it back (decode) into the same image patch.
  - E.g., VQ-VAE



[Van den Oord et al., 2017]

VQ-VAE

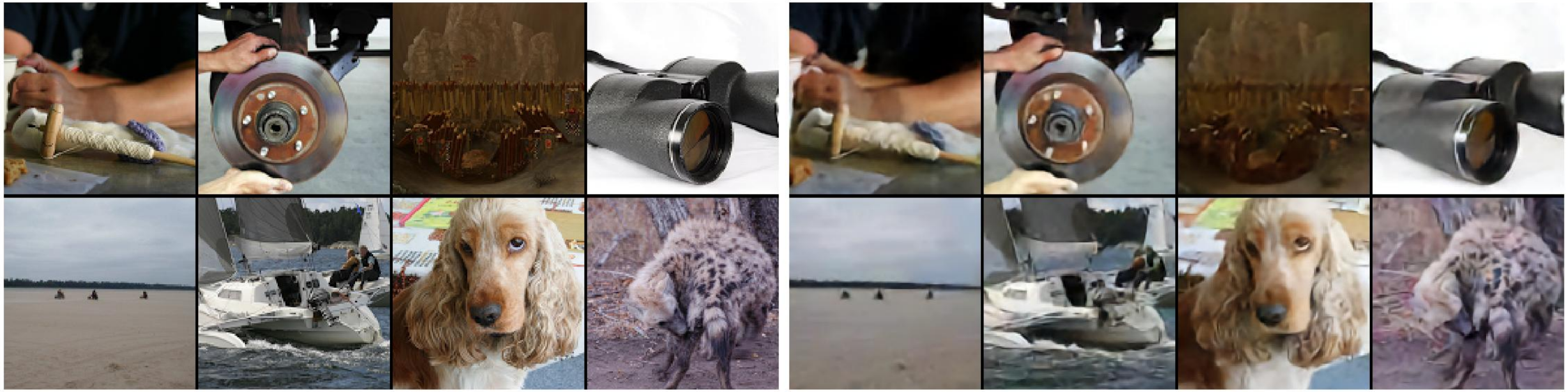
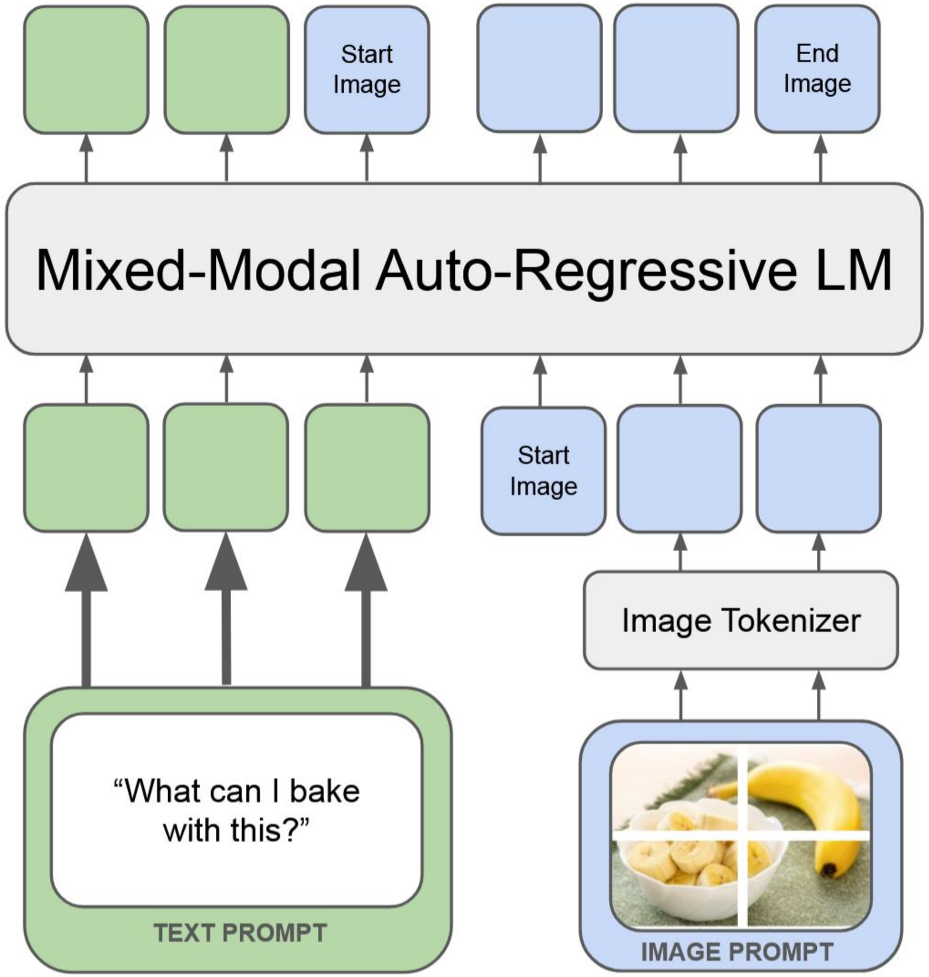


Figure 2: Left: ImageNet 128x128x3 images, right: reconstructions from a VQ-VAE with a 32x32x1 latent space, with K=512.

[Van den Oord et al., 2017]

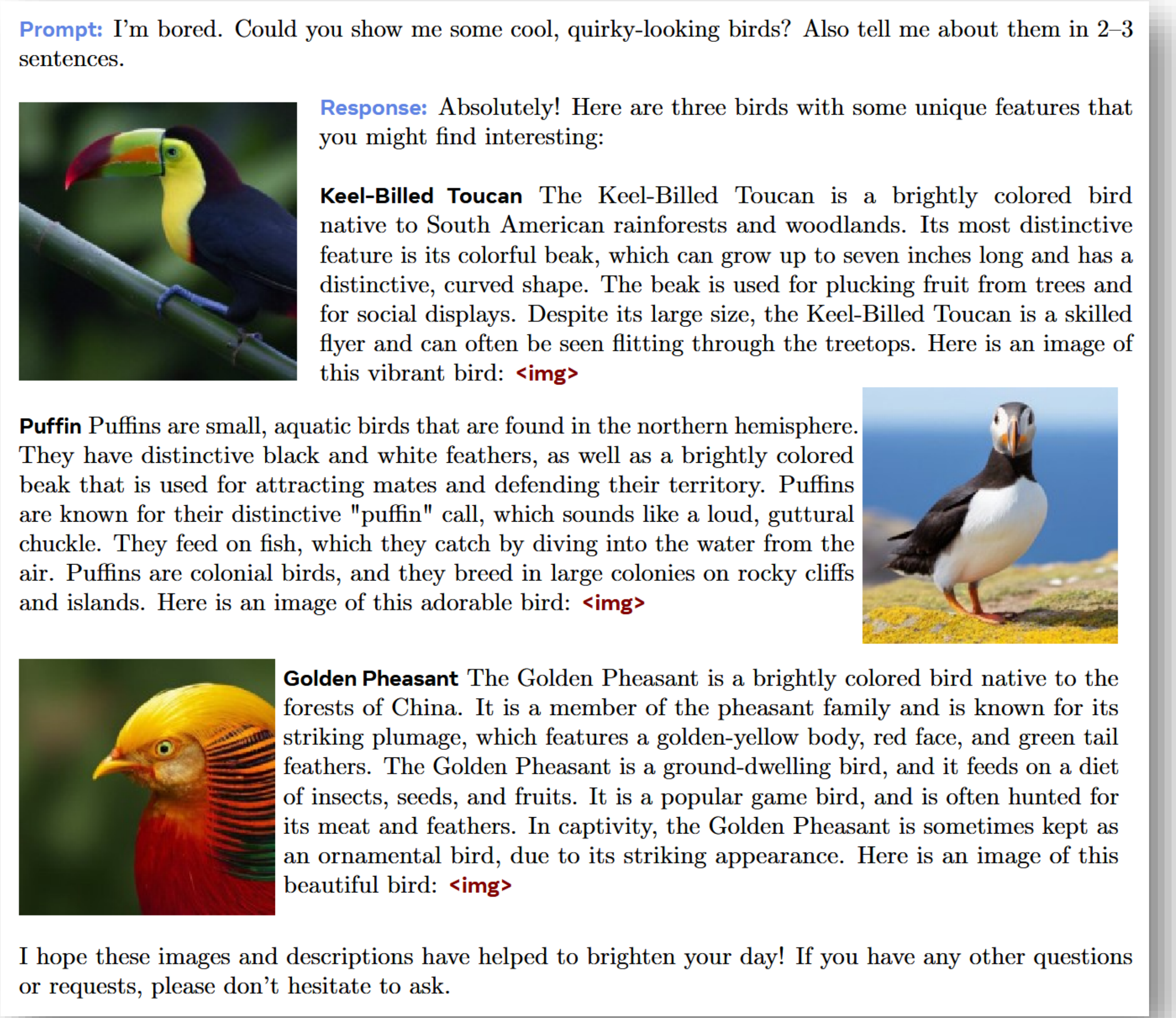
MLLM EXAMPLES

- Chameleon (2024) is an MLLM/VLM that uses an image tokenizer and detokenizer.
- First, they trained the image (de)tokenizer.
  - Image tokenizer encodes 512×512 images into 1024 tokens using a codebook of size 8192.
- Next, they trained the LM on 4.4T tokens (consisting of both image and text tokens).
  - They use Llama-2 as the LM.



[Chameleon Team, 2024]

MLLM EXAMPLES: CHAMELEON



Prompt: I'm bored. Could you show me some cool, quirky-looking birds? Also tell me about them in 2–3 sentences.

Response: Absolutely! Here are three birds with some unique features that you might find interesting:

Keel-Billed Toucan The Keel-Billed Toucan is a brightly colored bird native to South American rainforests and woodlands. Its most distinctive feature is its colorful beak, which can grow up to seven inches long and has a distinctive, curved shape. The beak is used for plucking fruit from trees and for social displays. Despite its large size, the Keel-Billed Toucan is a skilled flyer and can often be seen flitting through the treetops. Here is an image of this vibrant bird: <img>

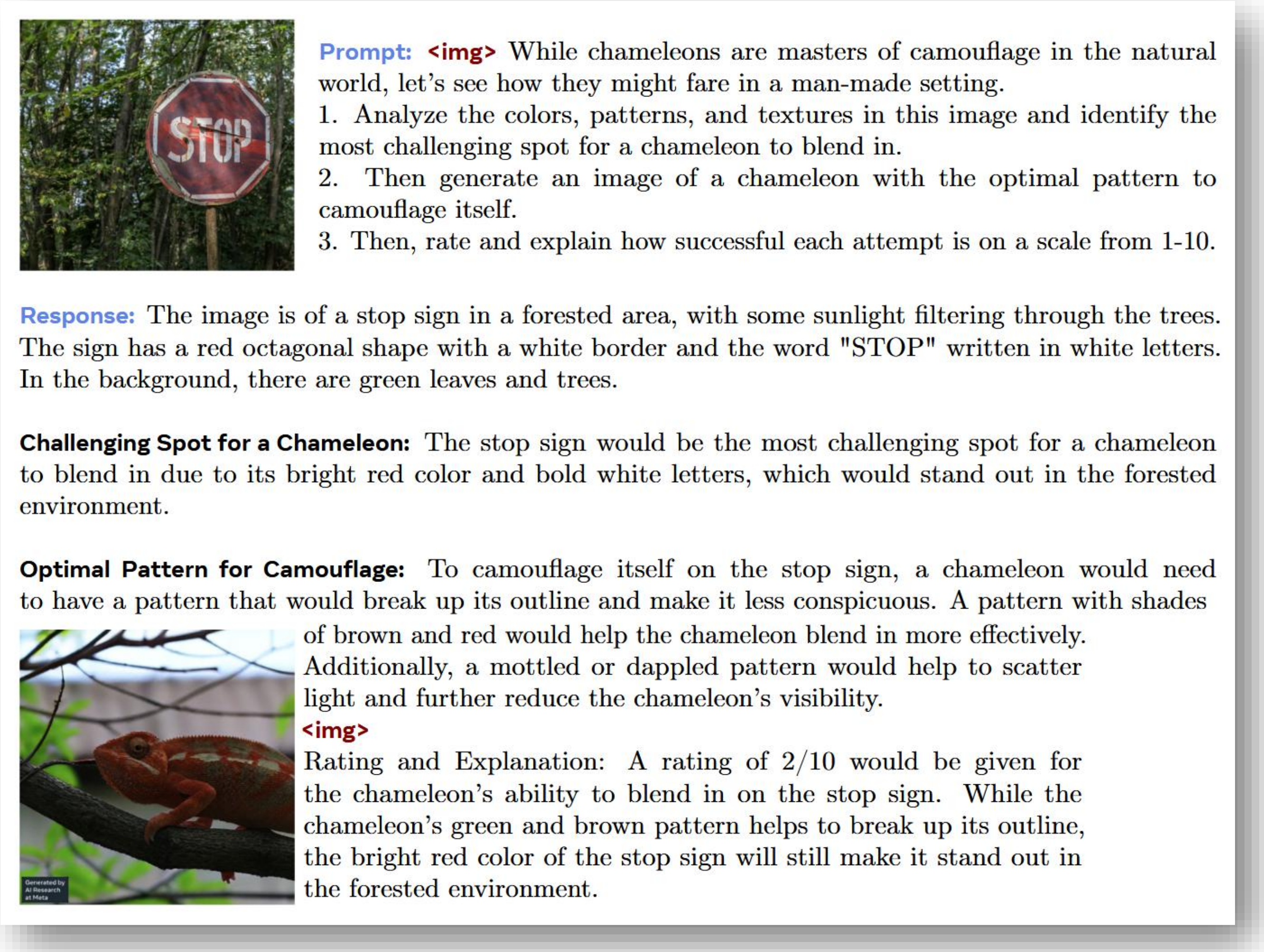
Puffin Puffins are small, aquatic birds that are found in the northern hemisphere. They have distinctive black and white feathers, as well as a brightly colored beak that is used for attracting mates and defending their territory. Puffins are known for their distinctive "puffin" call, which sounds like a loud, guttural chuckle. They feed on fish, which they catch by diving into the water from the air. Puffins are colonial birds, and they breed in large colonies on rocky cliffs and islands. Here is an image of this adorable bird: <img>

Golden Pheasant The Golden Pheasant is a brightly colored bird native to the forests of China. It is a member of the pheasant family and is known for its striking plumage, which features a golden-yellow body, red face, and green tail feathers. The Golden Pheasant is a ground-dwelling bird, and it feeds on a diet of insects, seeds, and fruits. It is a popular game bird, and is often hunted for its meat and feathers. In captivity, the Golden Pheasant is sometimes kept as an ornamental bird, due to its striking appearance. Here is an image of this beautiful bird: <img>

I hope these images and descriptions have helped to brighten your day! If you have any other questions or requests, please don't hesitate to ask.

[Chameleon Team, 2024]

MLLM EXAMPLES: CHAMELEON



Prompt: <img> While chameleons are masters of camouflage in the natural world, let's see how they might fare in a man-made setting.
1. Analyze the colors, patterns, and textures in this image and identify the most challenging spot for a chameleon to blend in.
2. Then generate an image of a chameleon with the optimal pattern to camouflage itself.
3. Then, rate and explain how successful each attempt is on a scale from 1-10.

Response: The image is of a stop sign in a forested area, with some sunlight filtering through the trees. The sign has a red octagonal shape with a white border and the word "STOP" written in white letters. In the background, there are green leaves and trees.

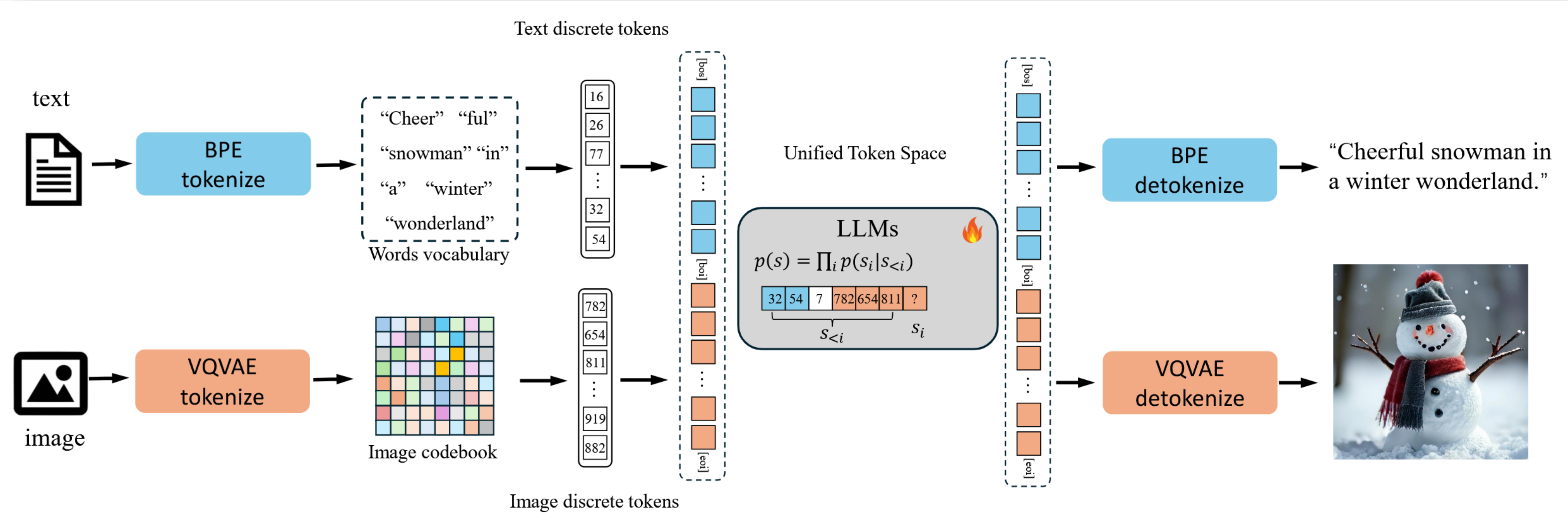
Challenging Spot for a Chameleon: The stop sign would be the most challenging spot for a chameleon to blend in due to its bright red color and bold white letters, which would stand out in the forested environment.

Optimal Pattern for Camouflage: To camouflage itself on the stop sign, a chameleon would need to have a pattern that would break up its outline and make it less conspicuous. A pattern with shades of brown and red would help the chameleon blend in more effectively. Additionally, a mottled or dappled pattern would help to scatter light and further reduce the chameleon's visibility.
<img>
Rating and Explanation: A rating of 2/10 would be given for the chameleon's ability to blend in on the stop sign. While the chameleon's green and brown pattern helps to break up its outline, the bright red color of the stop sign will still make it stand out in the forested environment.

[Chameleon Team, 2024]

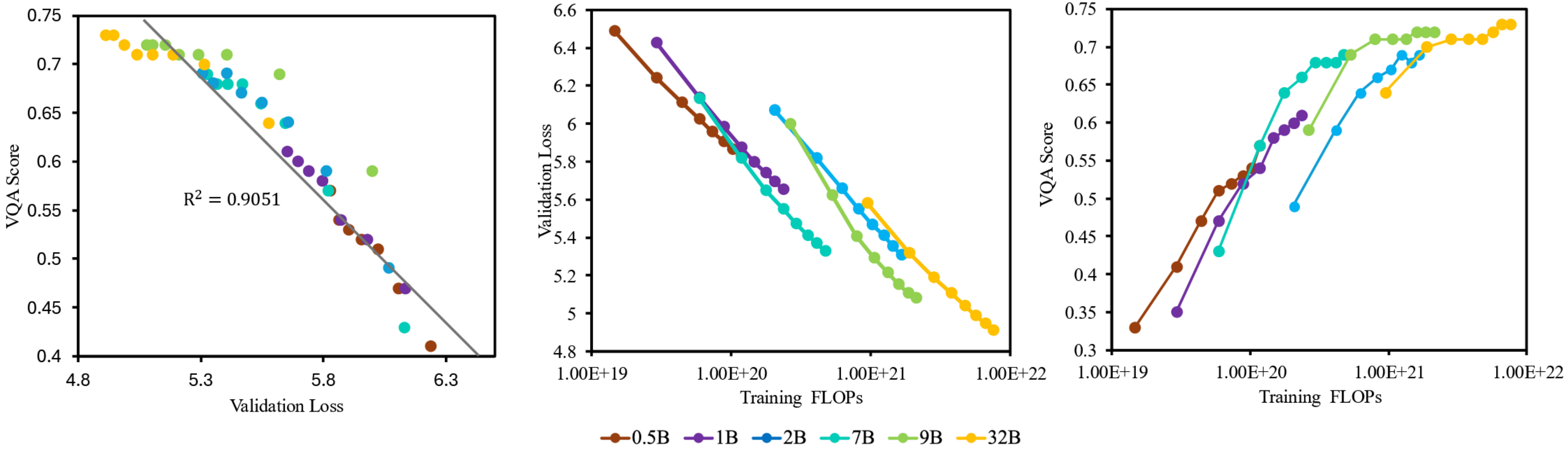
MLLM EXAMPLES: LIQUID

- Wu et al. (2024) developed Liquid.



MLLM EXAMPLES: LIQUID

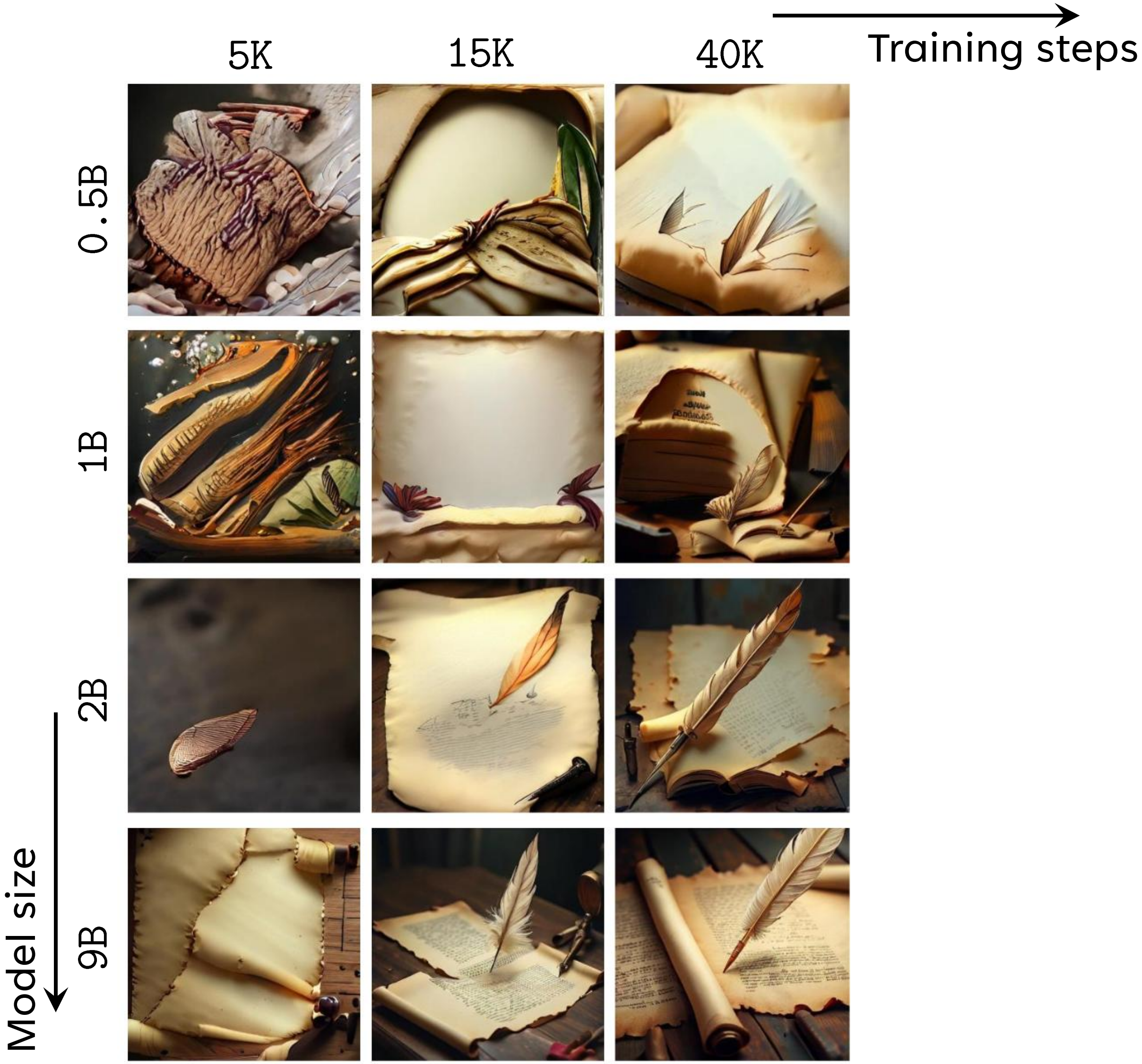
- Wu et al. (2024) developed Liquid.
- They trained models of several sizes, using a different LM for each:
  - Qwen 2.5 0.5B, 7B, 32B, GEMMA-2 2B, 9B, Llama-3 1B.



MLLM EXAMPLES: LIQUID

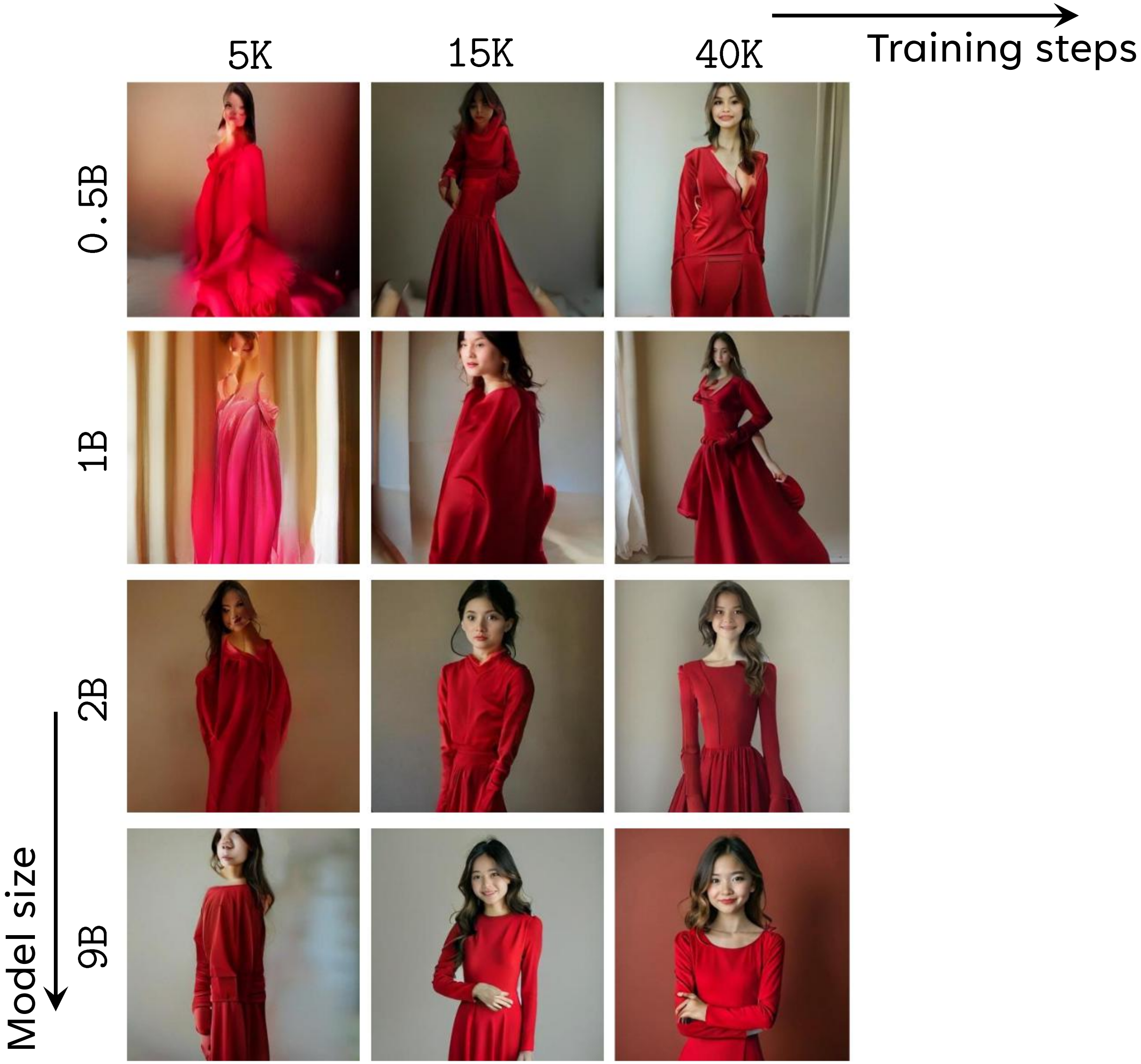
- Wu et al. (2024) developed Liquid.
- They trained models of several sizes, using a different LM for each:
  - Qwen 2.5 `0.5B, 7B, 32B`, GEMMA-2 `2B, 9B`, Llama-3 `1B`.
- Unlike Chameleon, Liquid was not pretrained from scratch.
  - They performed continued pretraining on existing text-only LMs
  - On `60M` tokens (as compared to `4.4T` for Chameleon).

MLLM EXAMPLES: LIQUID



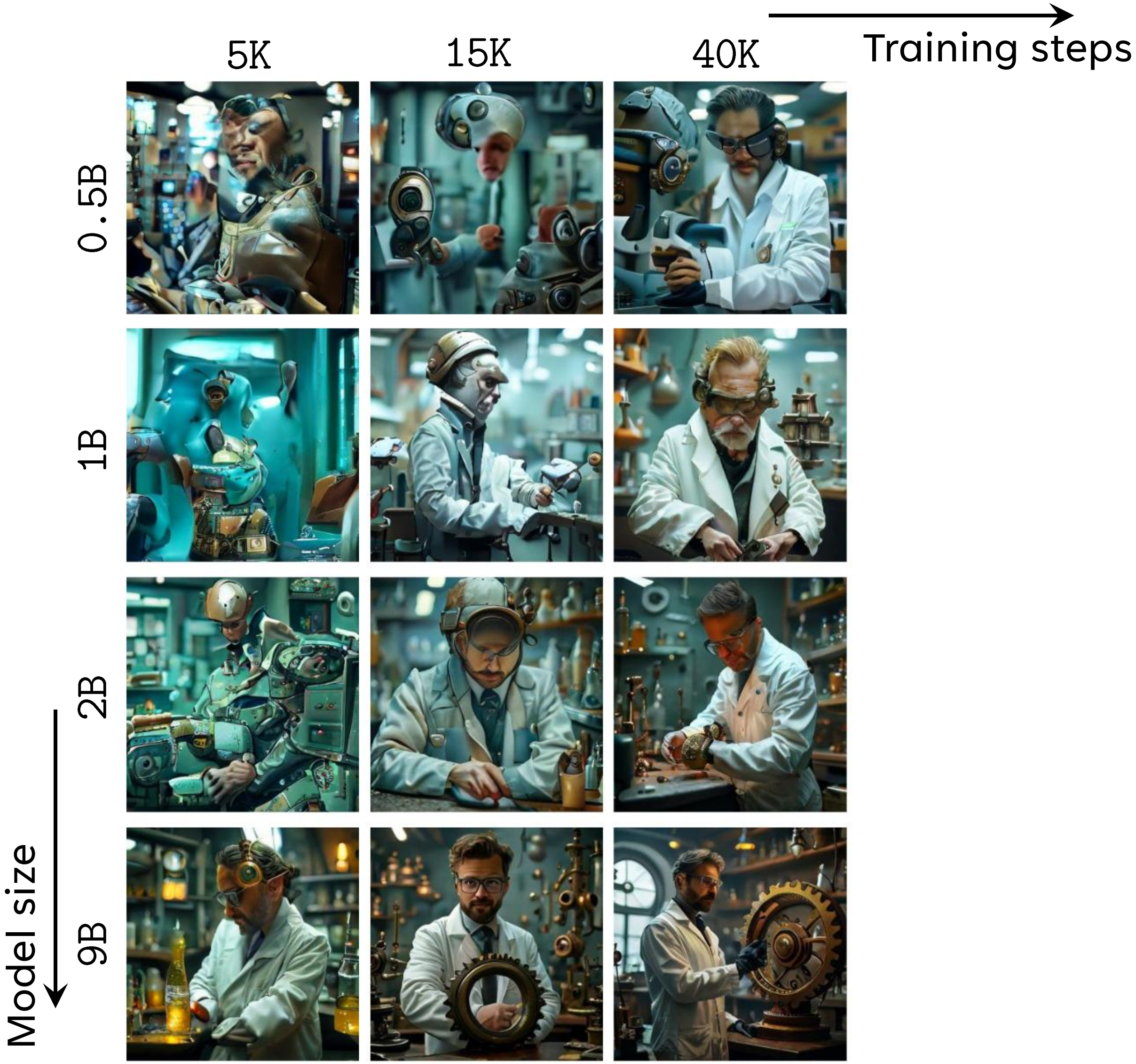
[Wu et al., 2024]

MLLM EXAMPLES: LIQUID



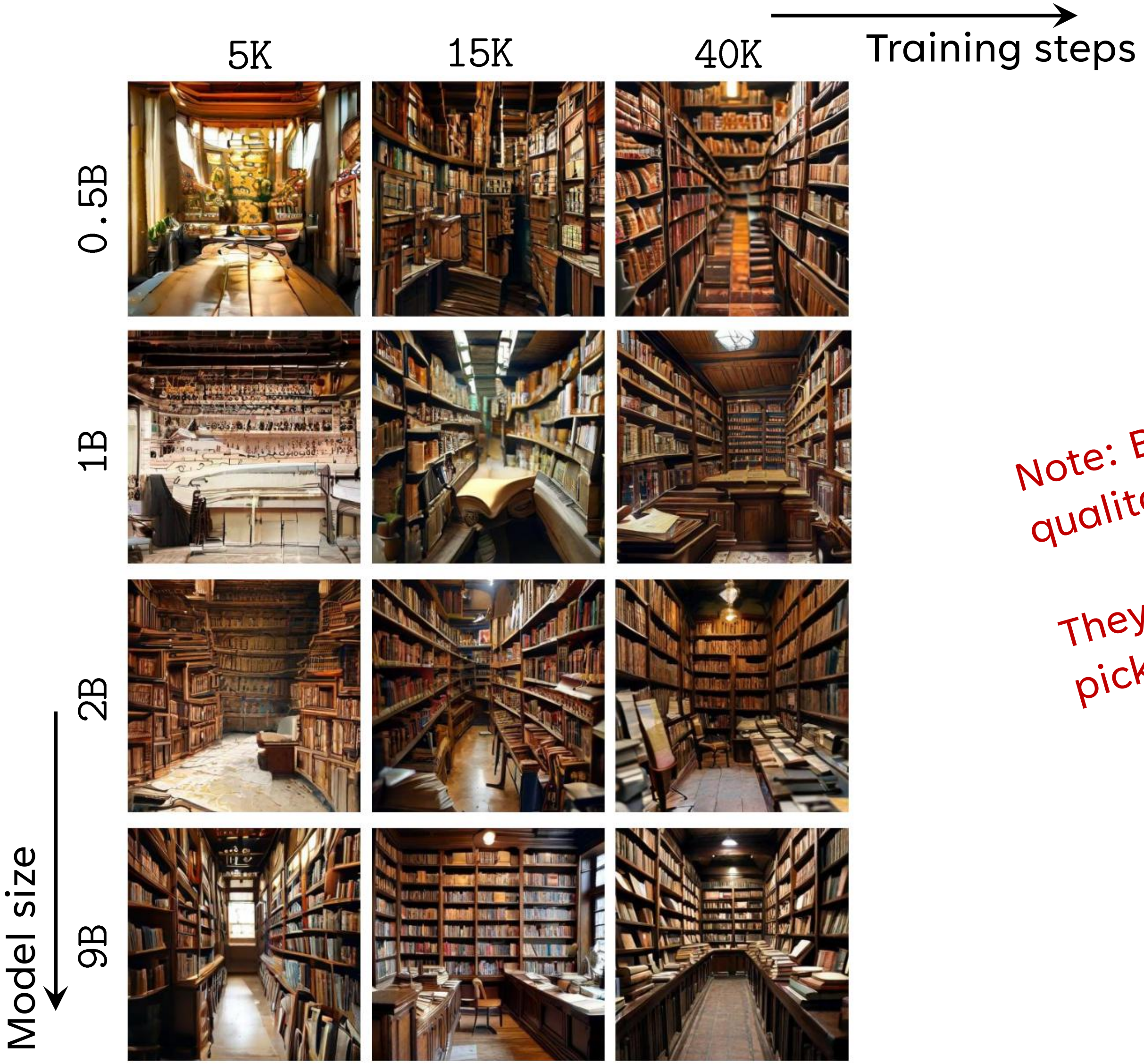
[Wu et al., 2024]

MLLM EXAMPLES: LIQUID



[Wu et al., 2024]

MLLM EXAMPLES: LIQUID



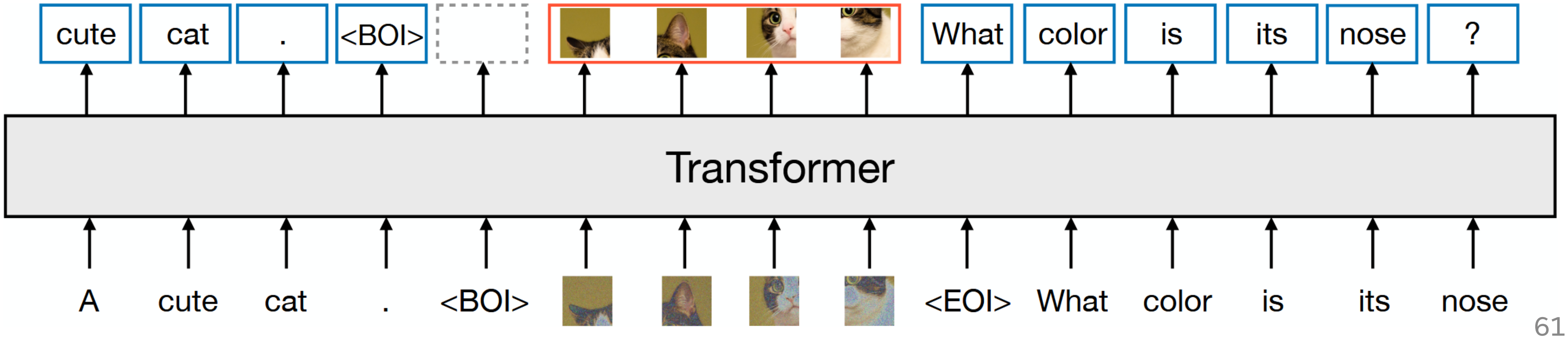
Note: Be wary of qualitative examples.

They may be cherry-picked by the authors.

[Wu et al., 2024]

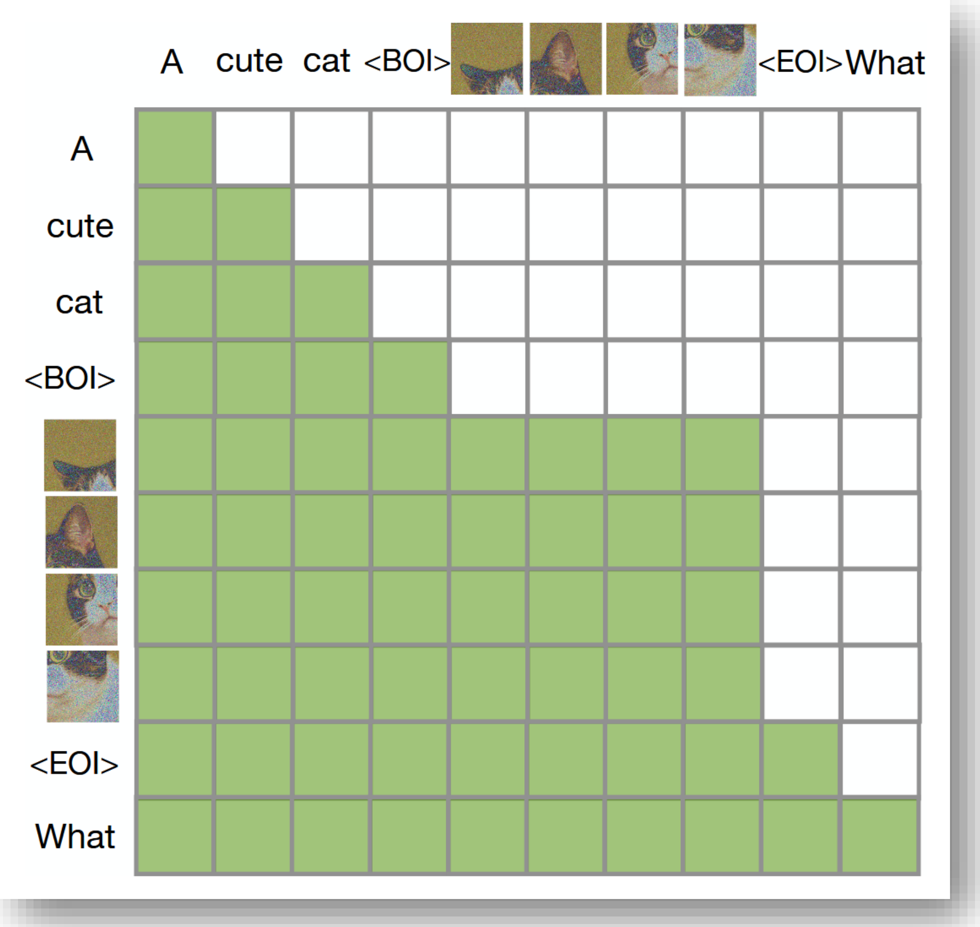
CONTINUOUS IMAGE TOKENS

- Another idea: Instead of using discrete image tokens,
  - Why not use continuous image tokens?
- This was the idea suggested by Zhou and Yu et al. (2024) when they developed their MLLM called Transfusion.
  - Transfusion was pretrained on a multi-modal dataset of 2T tokens.



CONTINUOUS IMAGE TOKENS

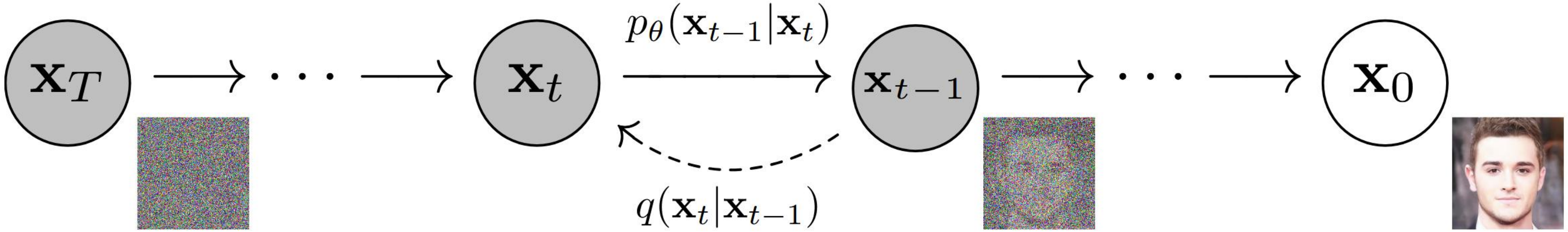
- They use a VAE (not VQ-VAE) to convert image patches into continuous image tokens.
- These are taken together with discrete text tokens to form a single sequence of tokens (some discrete, some continuous).
- This is input into a transformer LM.
- The causal mask in the attention mechanism is modified:
    - Text tokens can only attend to previous text/image tokens.
    - Image tokens can attend to any image token, or previous text tokens.



[Zhou and Yu et al., 2024]

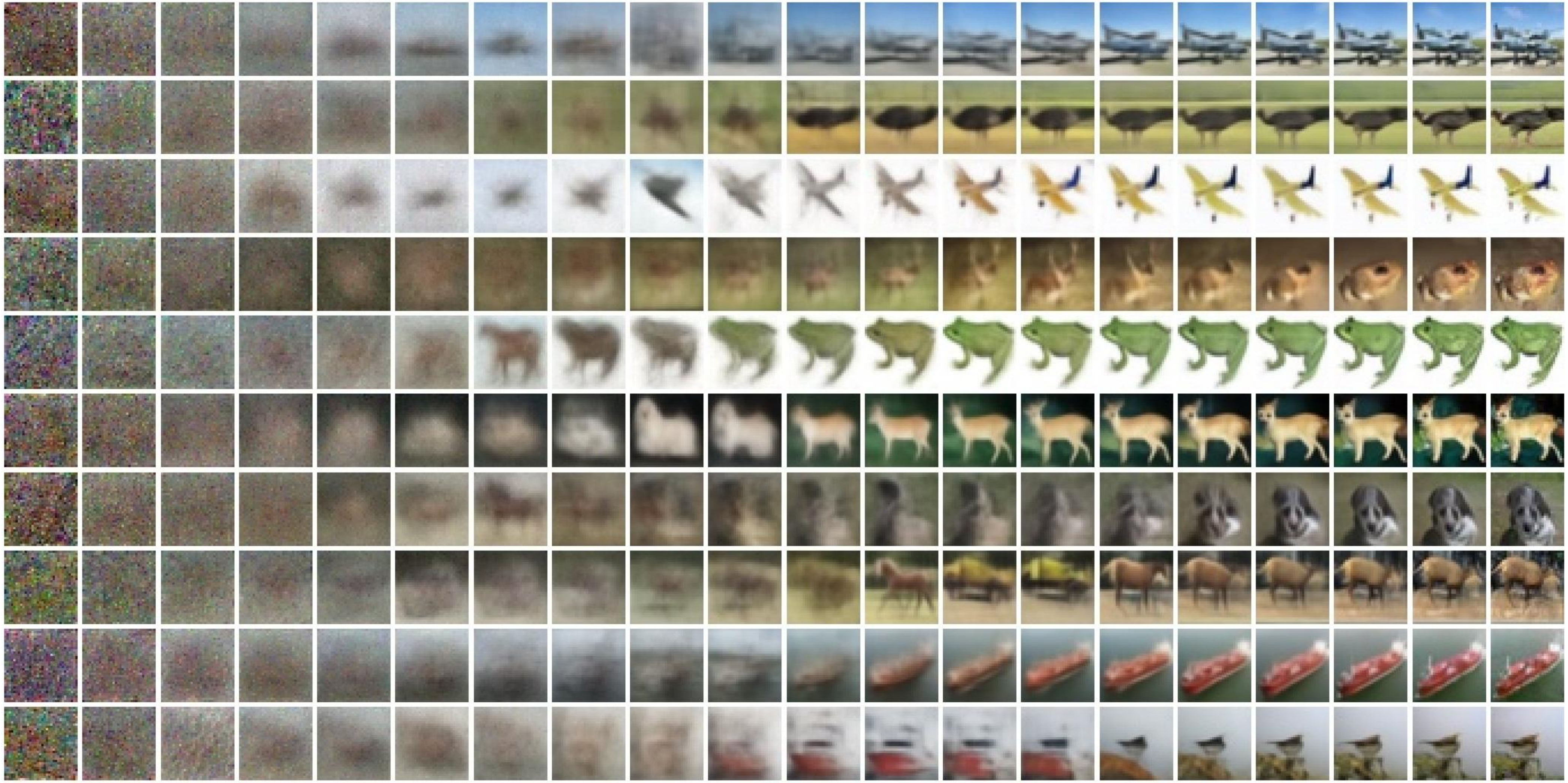
DIFFUSION

- How do they learn to generate the image?
  - Noise is added to the input image patches.
  - During training, the model learns to denoise the image patches.
    - Intuitively, imagine each layer progressively denoising the image.
  - The L2 loss is computed between the predicted output image patches and the ground truth image patches.
- This idea is called diffusion (Ho et al., 2020).


$\mathbf{x}_T$
$\mathbf{x}_t$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$
$\mathbf{x}_{t-1}$
$\mathbf{x}_0$

[Ho et al., 2020]

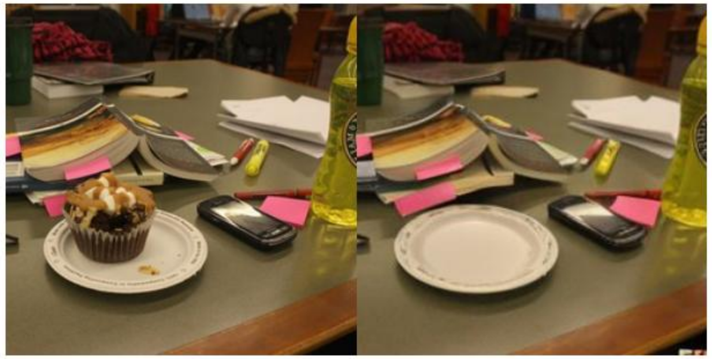
DIFFUSION



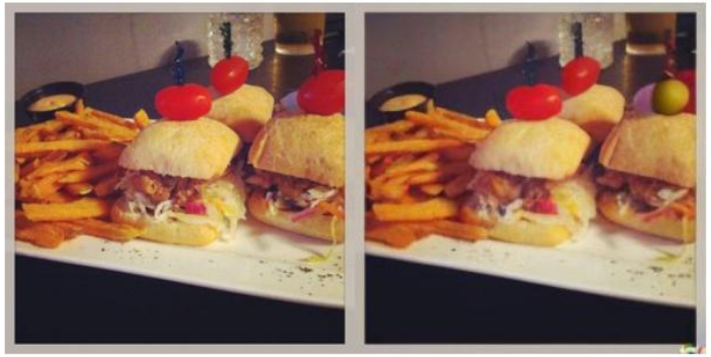
[Ho et al., 2020]

MLLM EXAMPLES: TRANSFUSION

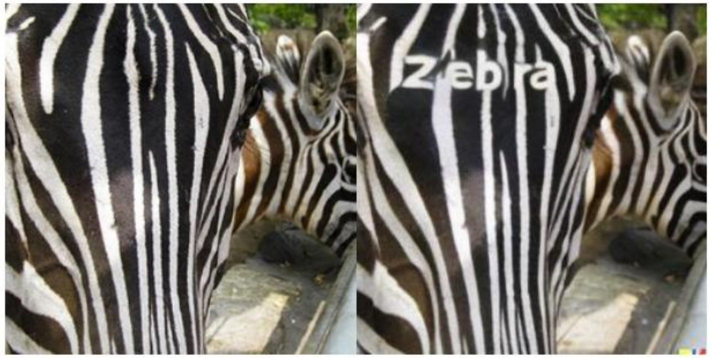
- Transfusion uses continuous image tokens and diffusion to generate images.



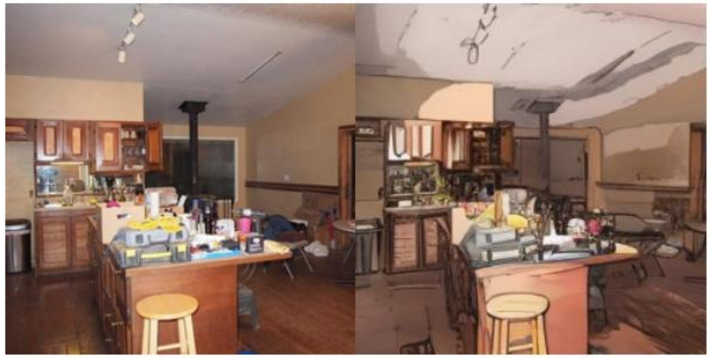
Remove the cupcake on the plate.



Change the tomato on the right to a green olive.



Write the word "Zebra" in Arial bold.

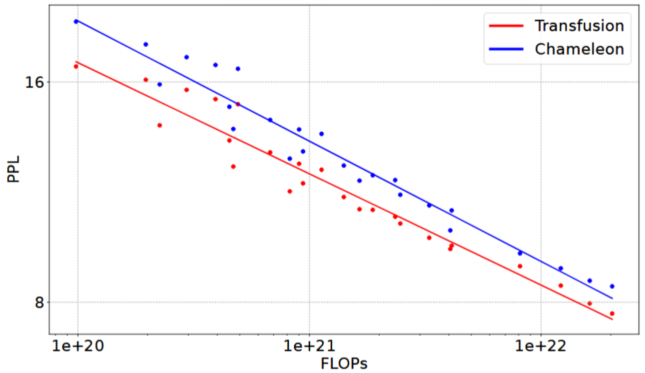


Change this to cartoon style.

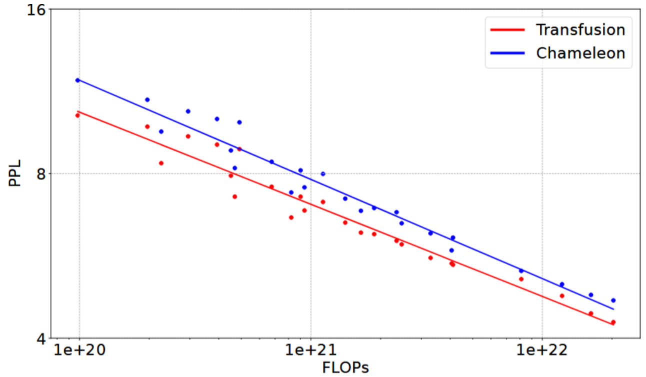
[Zhou and Yu et al., 2024]

MLLM EXAMPLES: TRANSFUSION

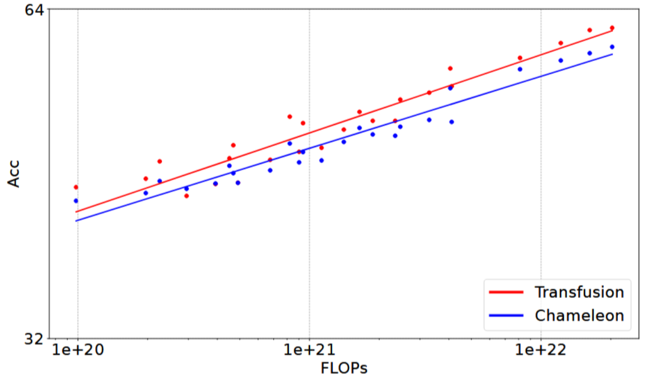
- They claim better performance vs Chameleon.



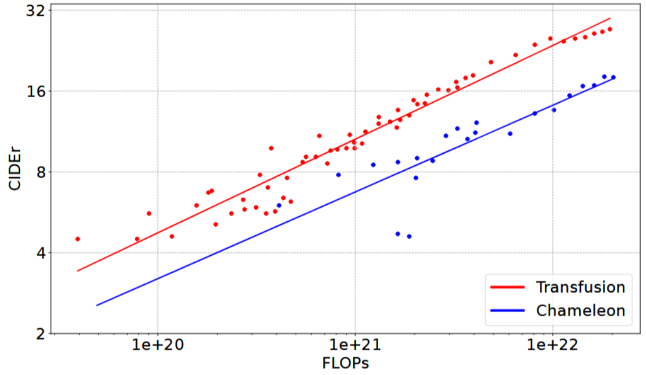
C4 Perplexity



Wikipedia Perplexity



Llama 2 Eval Suite Accuracy



MS-COCO 5k CIDEr

[Zhou and Yu et al., 2024]

SUMMARY

- This was a whirlwind tour of MLLMs/VLMs.
- A lot of details/history from vision were left out.
  - The study of diffusion and score-based models is deep.

- But this was intended to give high-level ideas on how to combine vision and text in multi-modal NLP models.
- Thank you for listening!

QUESTIONS?