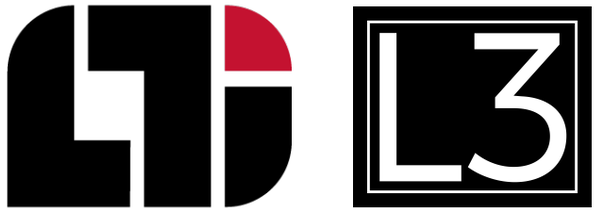
CS11-711 Advanced NLP

# Multimodal Modeling I

Sean Welleck


https://cmu-l3.github.io/anlp-spring2026/

https://github.com/cmu-l3/anlp-spring2026-code

Text

Model

Text

Text-to-text

Today's Lecture

Text

Model

Text Image

Multi-to-text

Image

Model

Text Image

Multi-to-image

Next 2 Lectures

Text Image

Model

Text Image

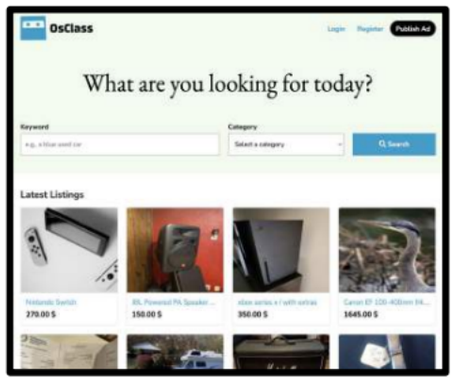
Multi-to-multi

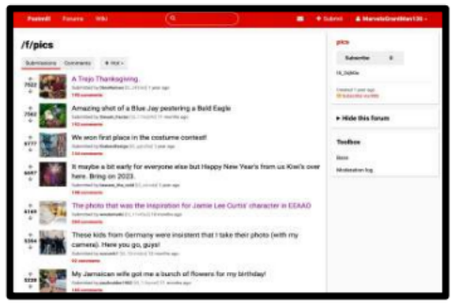
tell me about this image

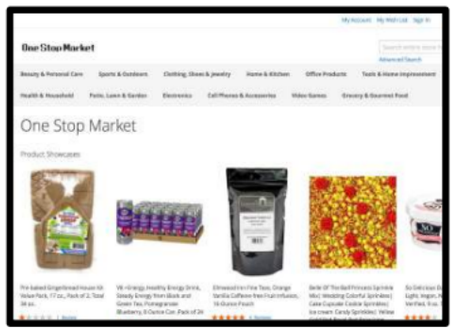
This image showcases a beautiful painting of Pittsburgh, presented in an ornate gold frame. The artwork captures the city's iconic skyline, featuring its distinctive yellow bridges spanning the blue waters below. The cityscape is rendered in soft, impressionistic brushstrokes that create a dreamy, almost ethereal atmosphere.

The painting's background is filled with fluffy white clouds against a light blue sky, adding to the sense of depth and tranquility. The overall color palette is subdued, with yellows from the bridges, blues from the sky and water, and greens from scattered trees, creating a harmonious and serene composition.

# Example: AI2 Molmo interface




**Webpage**

"Help me make a post selling this item and navigate to it. Price it at $10 cheaper than the most similar item on the site."

"Navigate to the comments section of the latest image post in the /f/Art subreddit that contains animals."

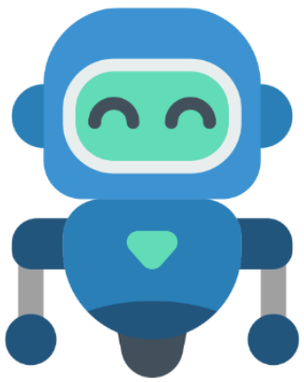
"Buy the cheapest color photo printer and send it to Emily's place (as shown in the image)."

**Task Specification**

**LLM / VLM Agent**

**click [1602]**

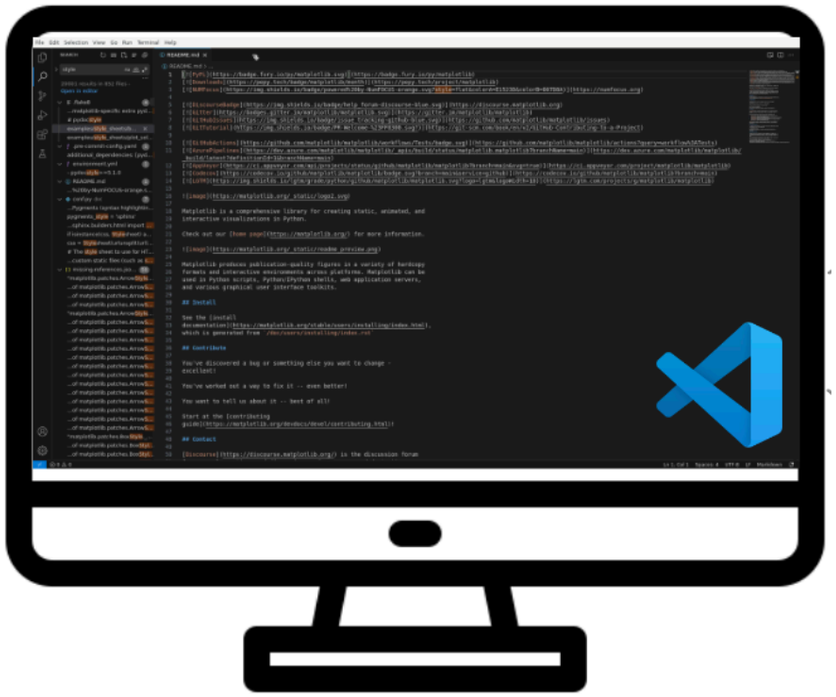
Example: web agents (future Agents lecture!)



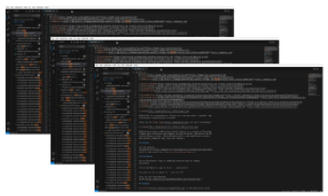
Computer-Use Agent

env.step(action)

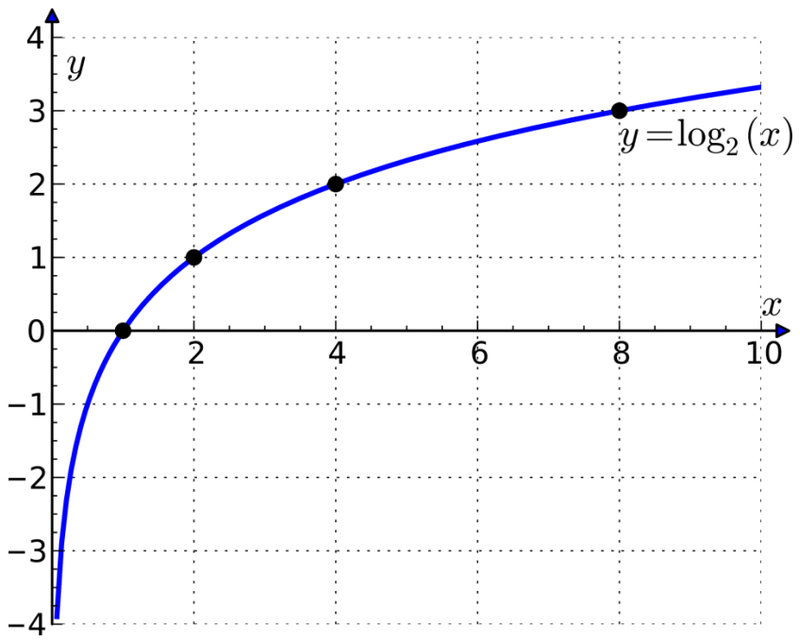
action ∈



observation



Example: computer-use agents (future Agents lecture!)




**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

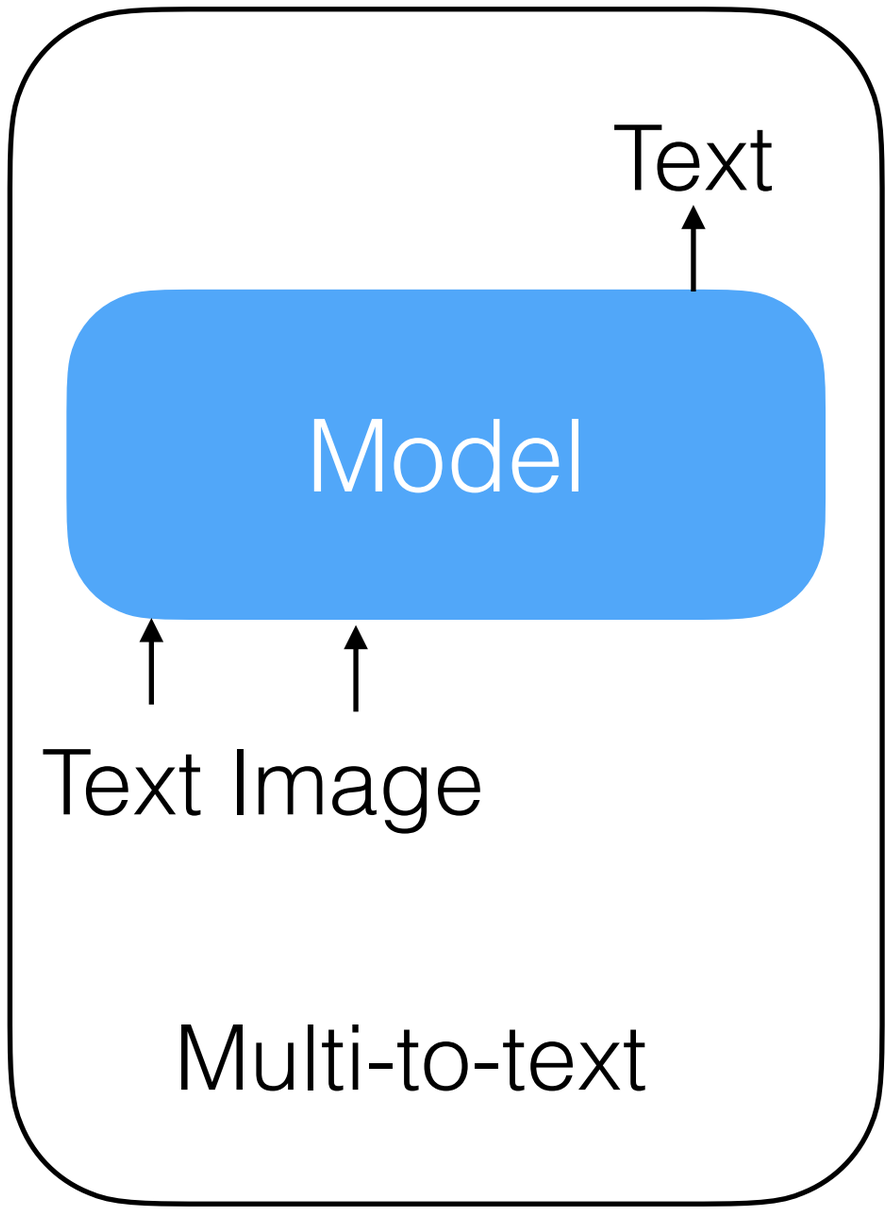
**Question:** How many zeros does this function have?
**Answer:** 1

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

# Example: mathematical reasoning [MathVista, Lu et al 2024]

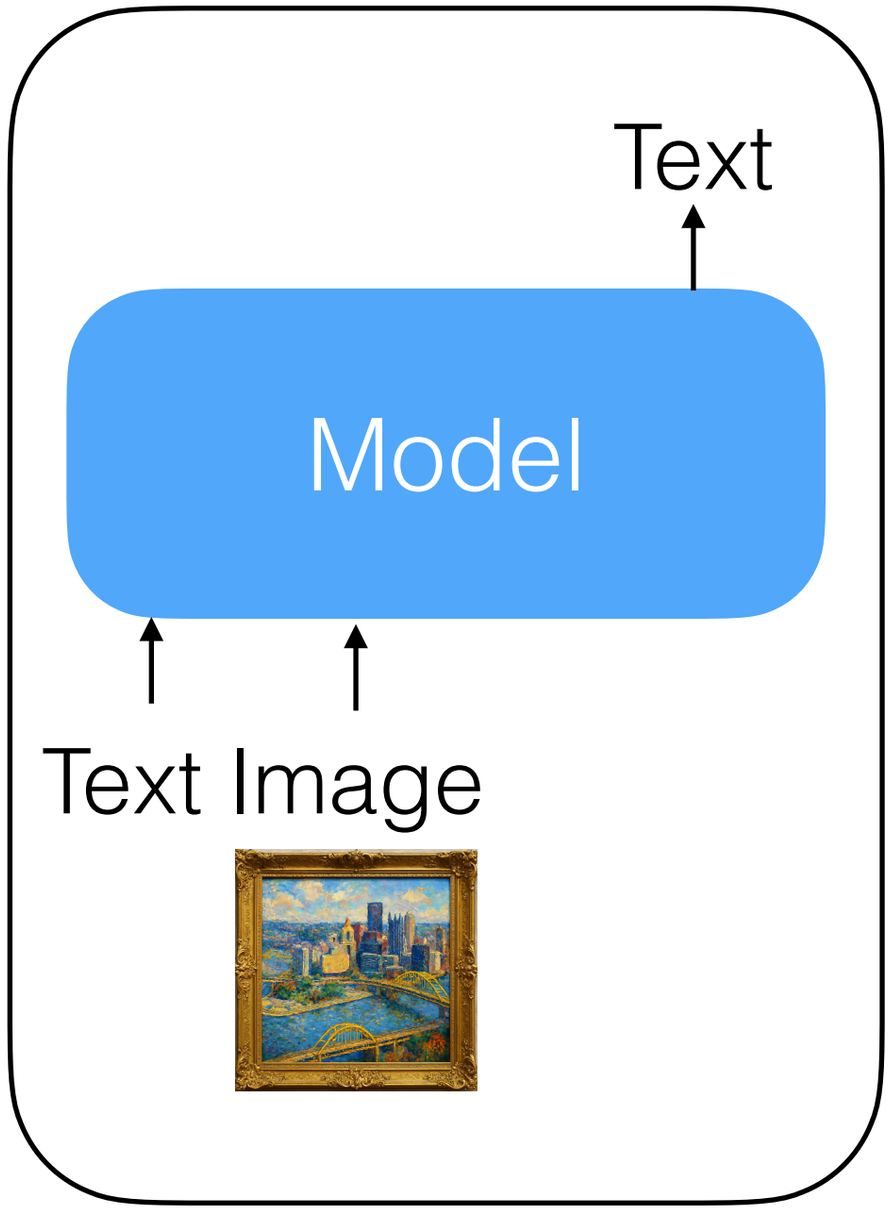
# Today's lecture

- Vision architecture basics
  - ViT [Dosovitskiy et al 2020]
- Learning image representations
  - CLIP [Radford et al 2021]
- Combining with a language model
  - Llava [Liu et al 2023]



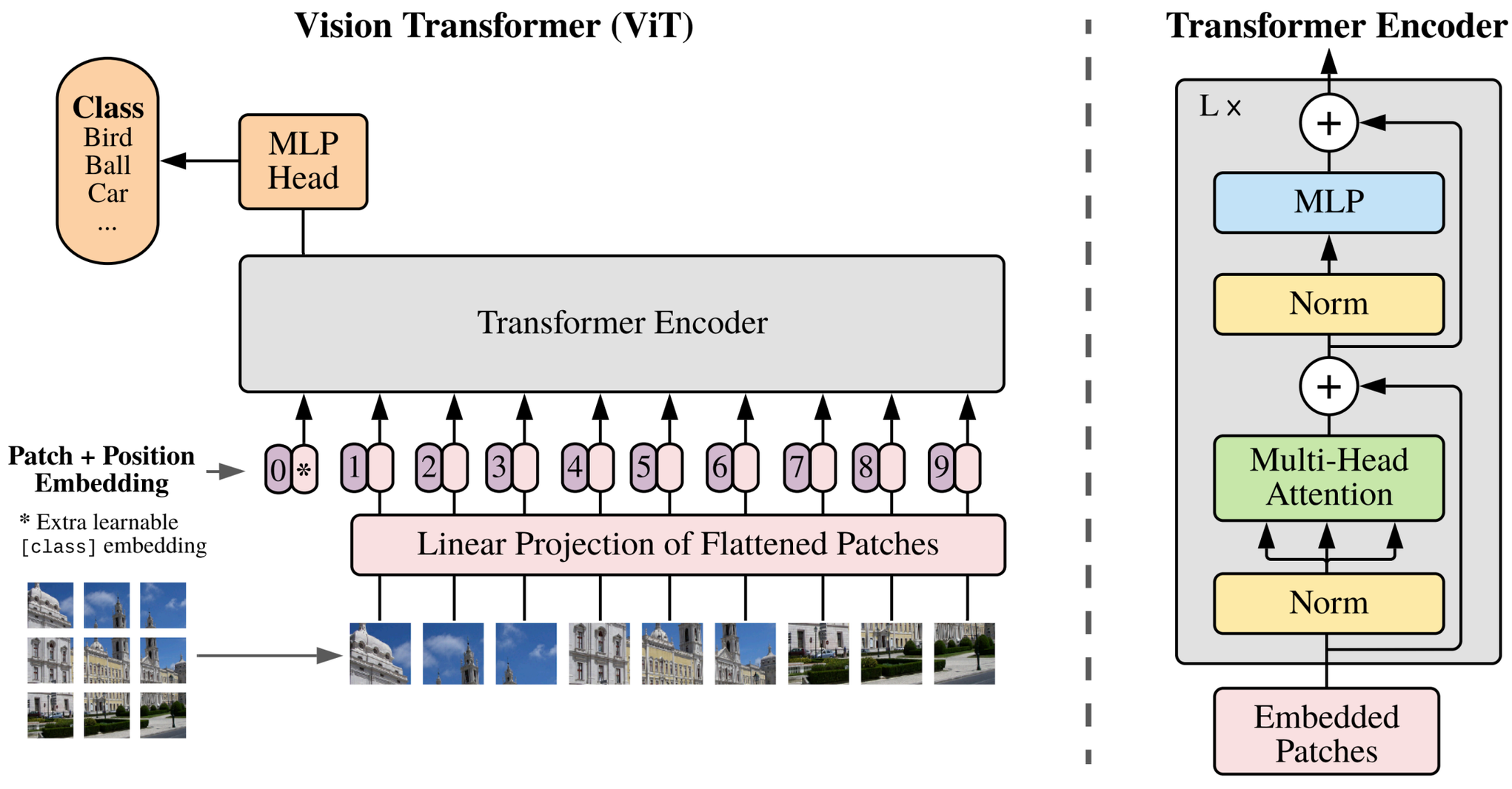
# Key problem: representing images

- We represent text as a sequence of vectors (token embeddings)
- We want to also represent an image as a sequence of vectors
  - $f_{\text{enc}}(x_{\text{image}}) \rightarrow z_1, \ldots, z_L$
- Need:
  - Neural network architecture
  - Algorithm for learning good vectors



# Vision Transformer (ViT)

- Idea: divide an image into patches, flatten the patches into vectors, use a standard transformer



# Vision Transformer (ViT)

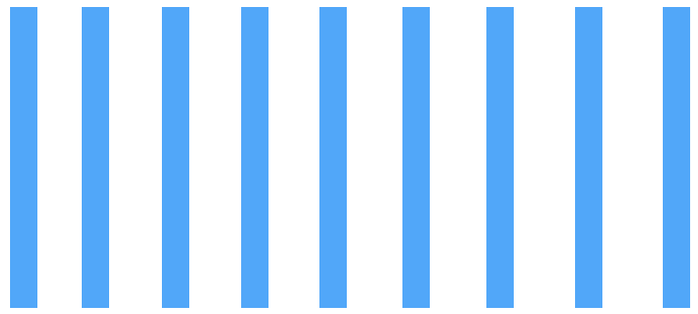
- $x_{\text{image}} \in \mathbb{R}^{H \times W \times C}$

120 x 120 x 3
Patch: 40 x 40

$$N = \frac{(120)(120)}{(40)(40)} = 9$$

patches

- $x_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$

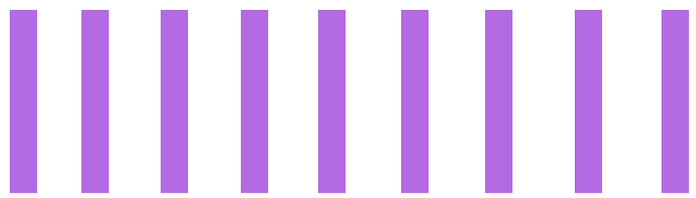


$$P^2 \cdot C = (40)(40)(3)$$

= 4800

$$\implies x_p \in \mathbb{R}^{9 \times 4800}$$

- $x \in \mathbb{R}^{N \times D}$
  - $x = Wx_p$
    $W_e \in \mathbb{R}^{D \times (P^2 \cdot C)}$



$$W \in \mathbb{R}^{1024 \times 4800}$$

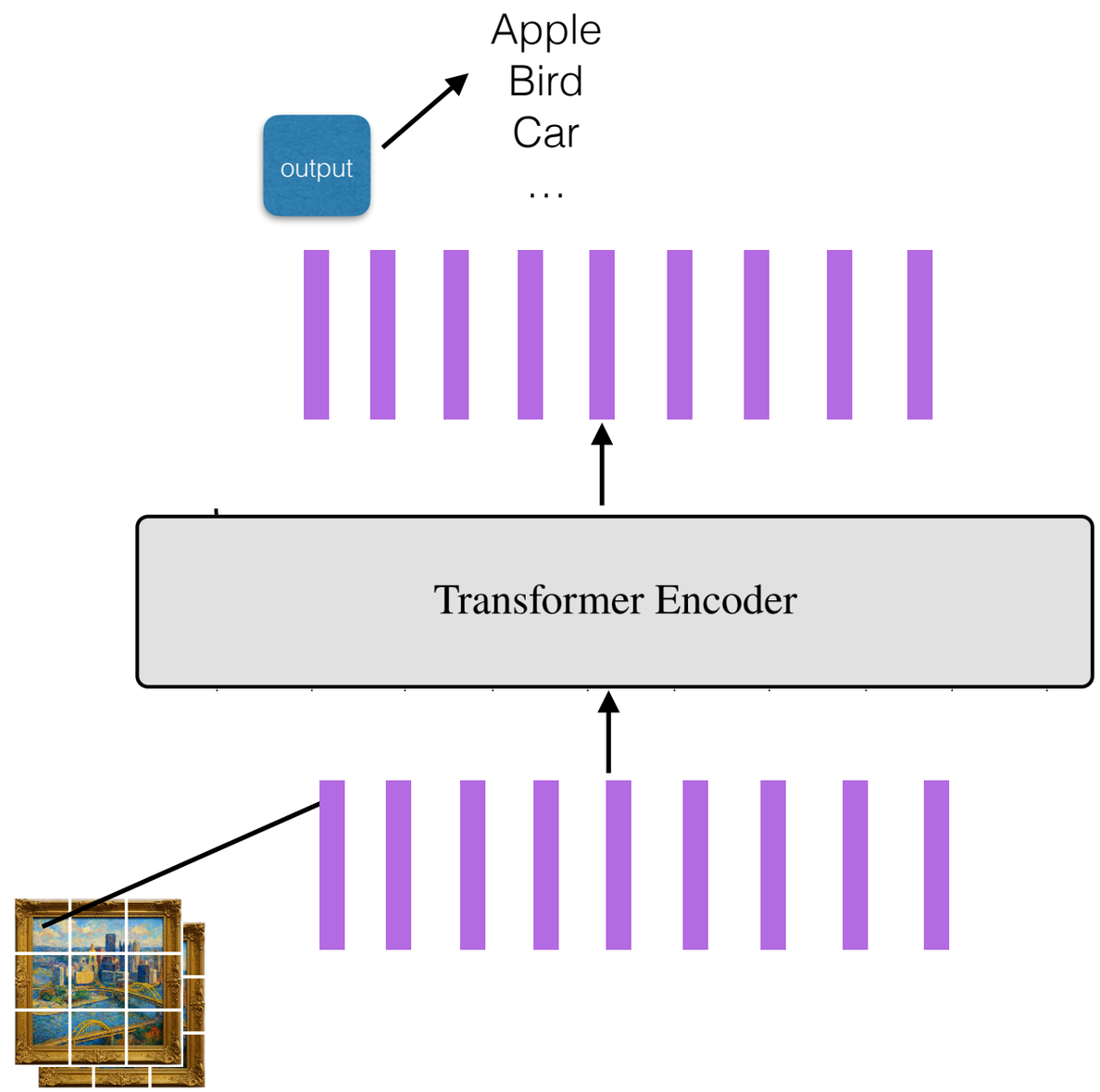
$$\implies x \in \mathbb{R}^{9 \times 1024}$$

"Patch embeddings"

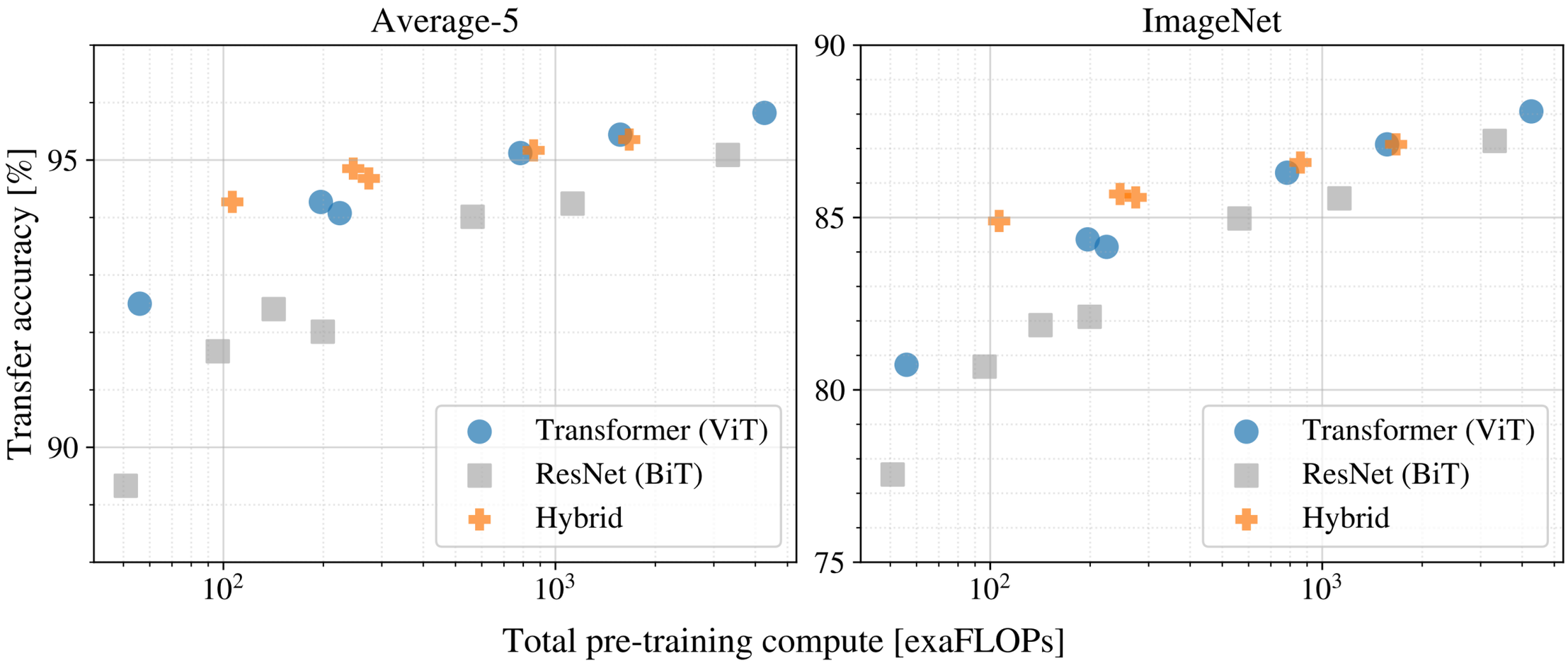
+ position embeddings

# Vision Transformer (ViT)

- The transformer transforms the patch embeddings into vector representations $z_1, \ldots, z_N$
- We can train the model to perform a task such as classification

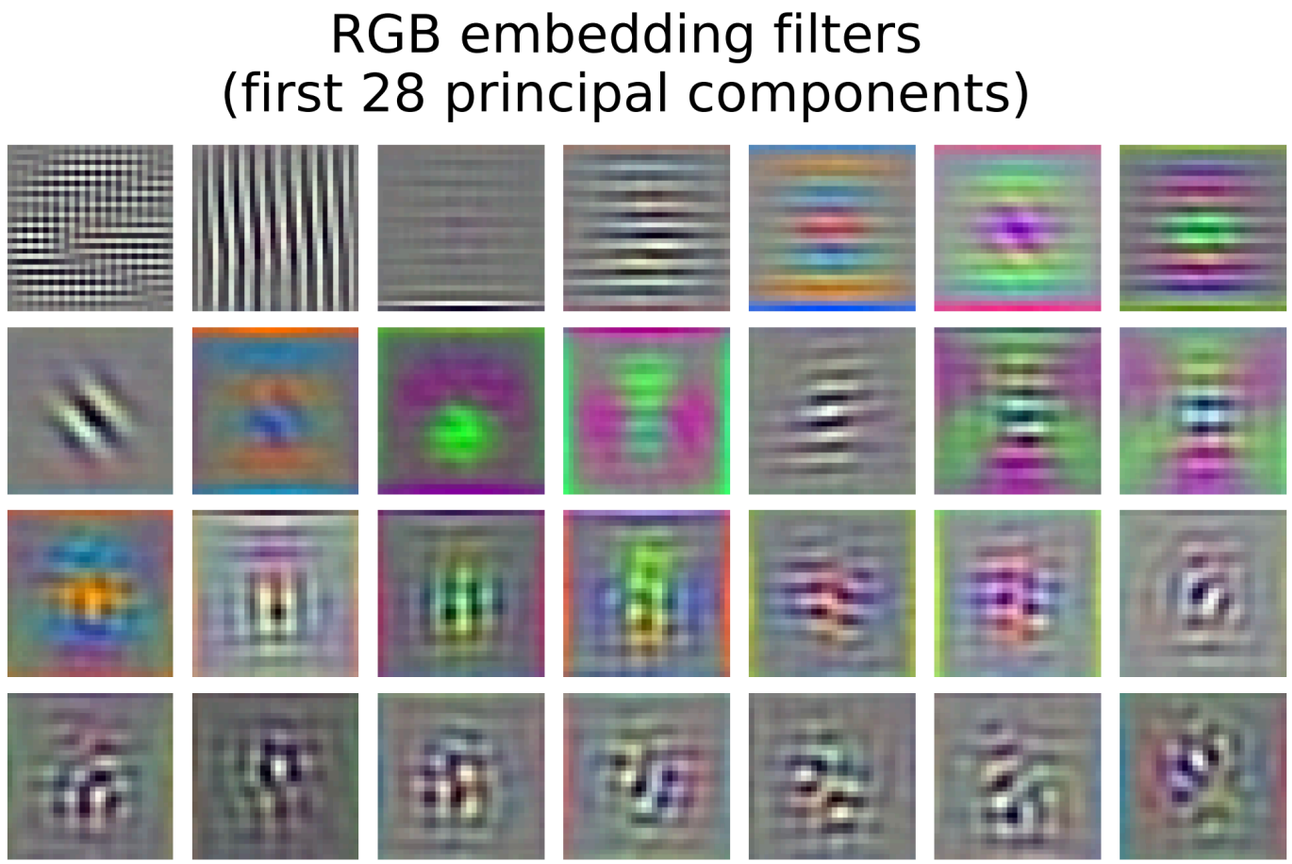


# Vision Transformer (ViT)




Performance versus pre-training compute for different architectures

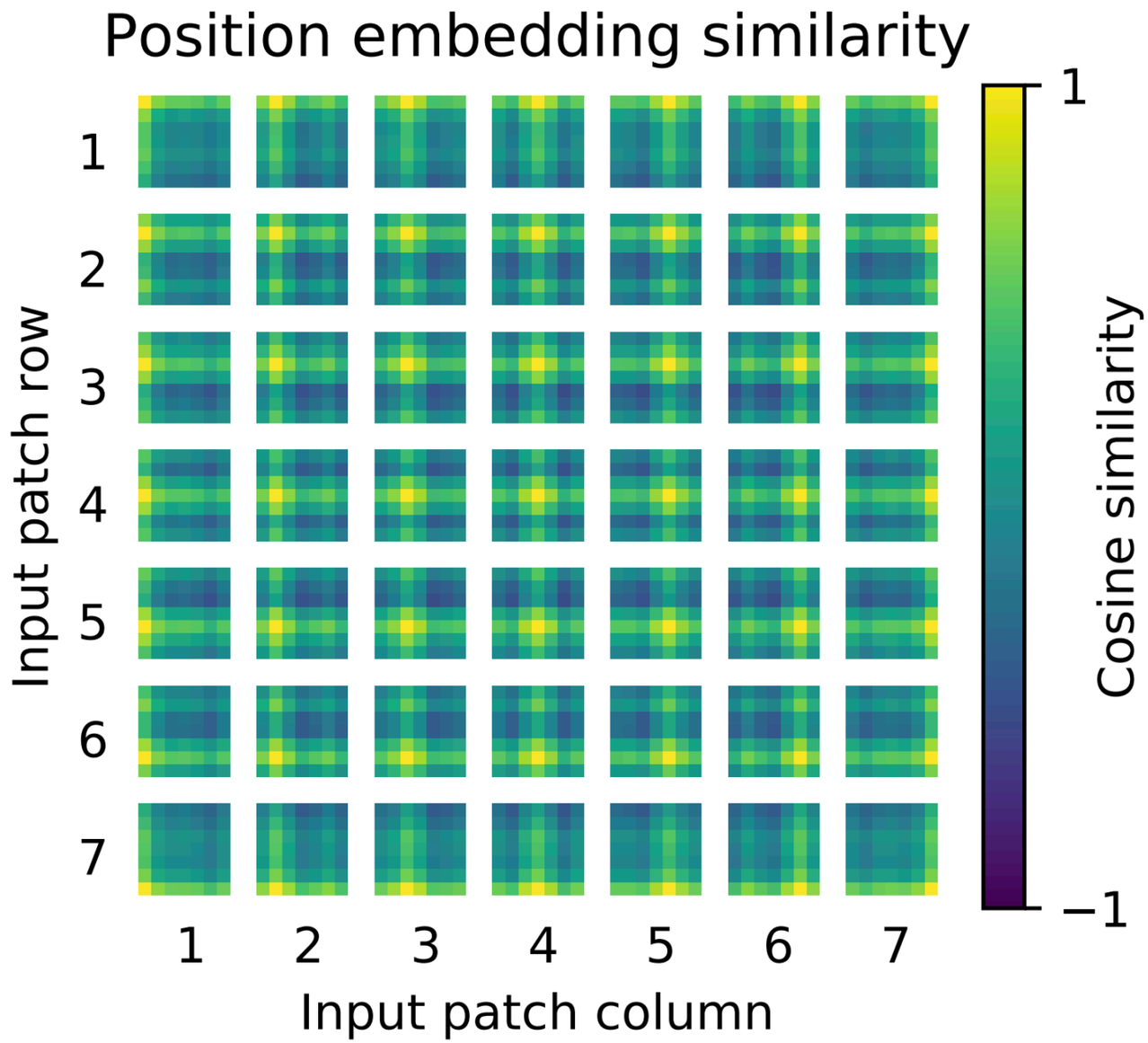
# Vision Transformer (ViT)




$$x = Wx_p,\ W_e \in \mathbb{R}^{D\times(P^2\cdot C)}$$

Reshape rows into P x P, visualize principal components

# Vision Transformer (ViT)




Cosine similarity between the position embedding of the patch with the indicated row and column and the position embeddings of all other patches

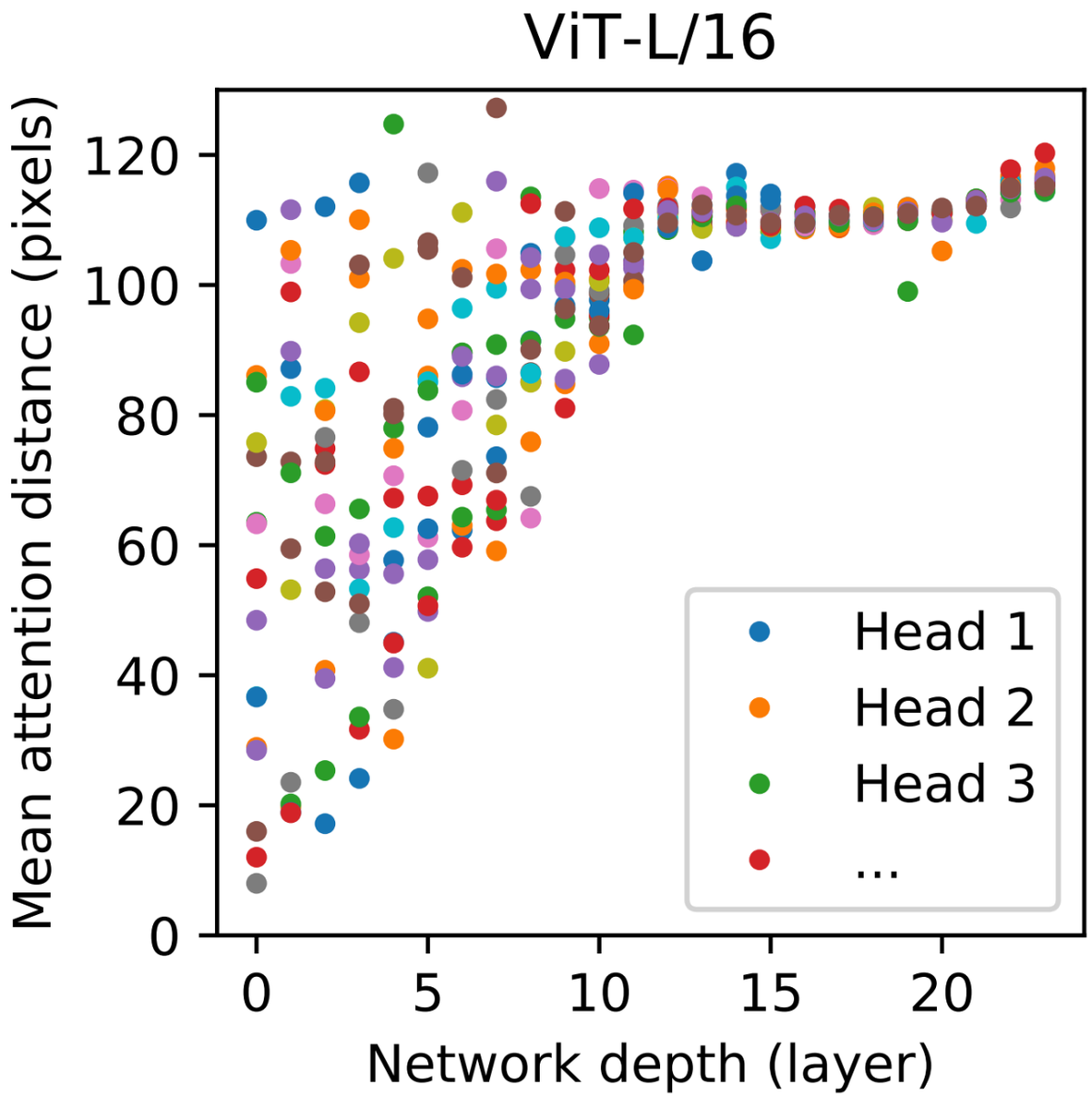
# Vision Transformer (ViT)

Input Attention



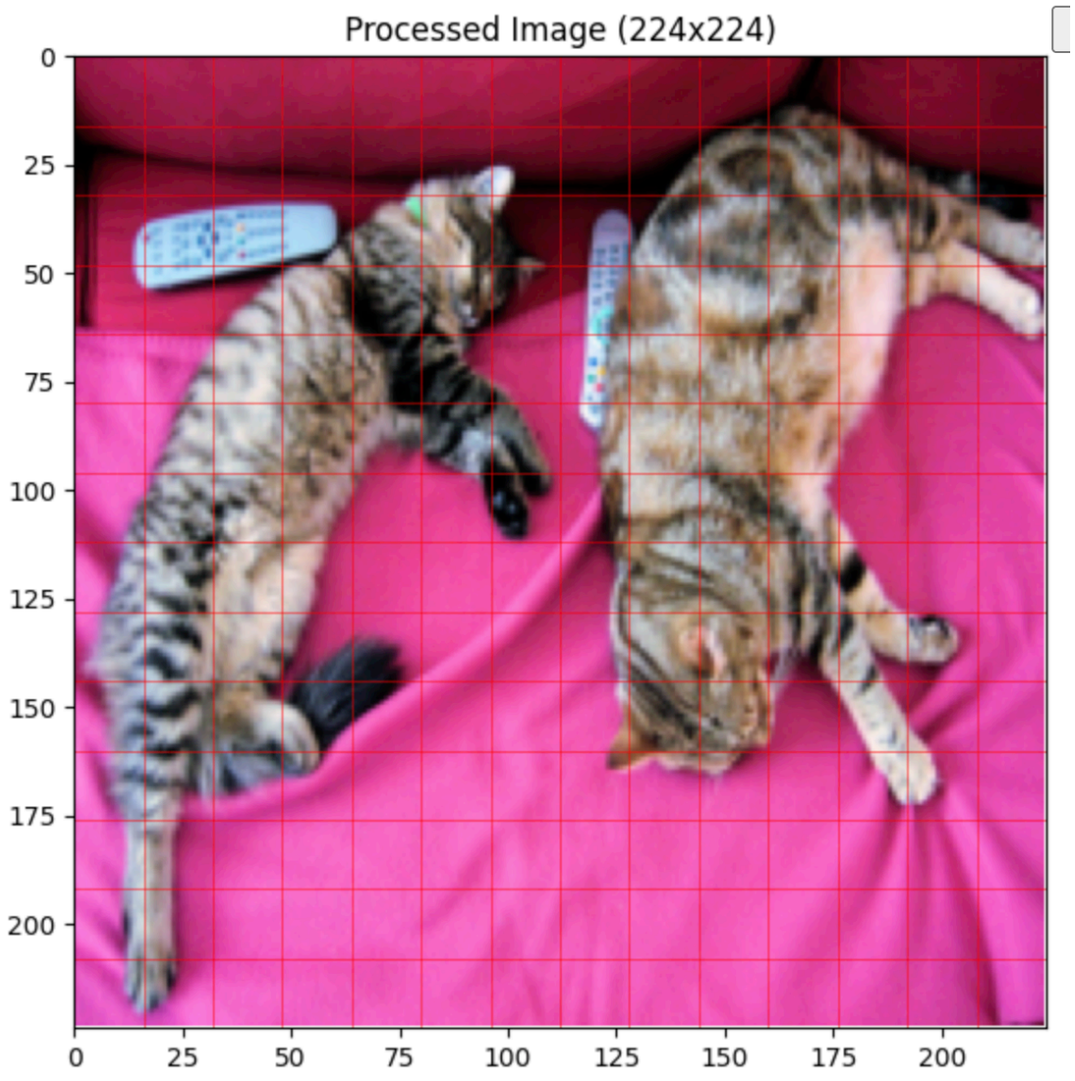

Can attend to regions that are salient for the task (here classification)

# Vision Transformer (ViT)




Early layers either attend to large regions or narrow regions; later layers generally attend to larger regions

# Code example




```
Patch embedding process:
1. Input image: torch.Size([1, 3, 224, 224])
   [batch, channels=3, height=224, width=224]

2. After patch projection: torch.Size([1, 196, 768])
   [batch, num_patches=196, hidden_size=768]

3. Each 16x16x3 patch → 768-dim vector
```

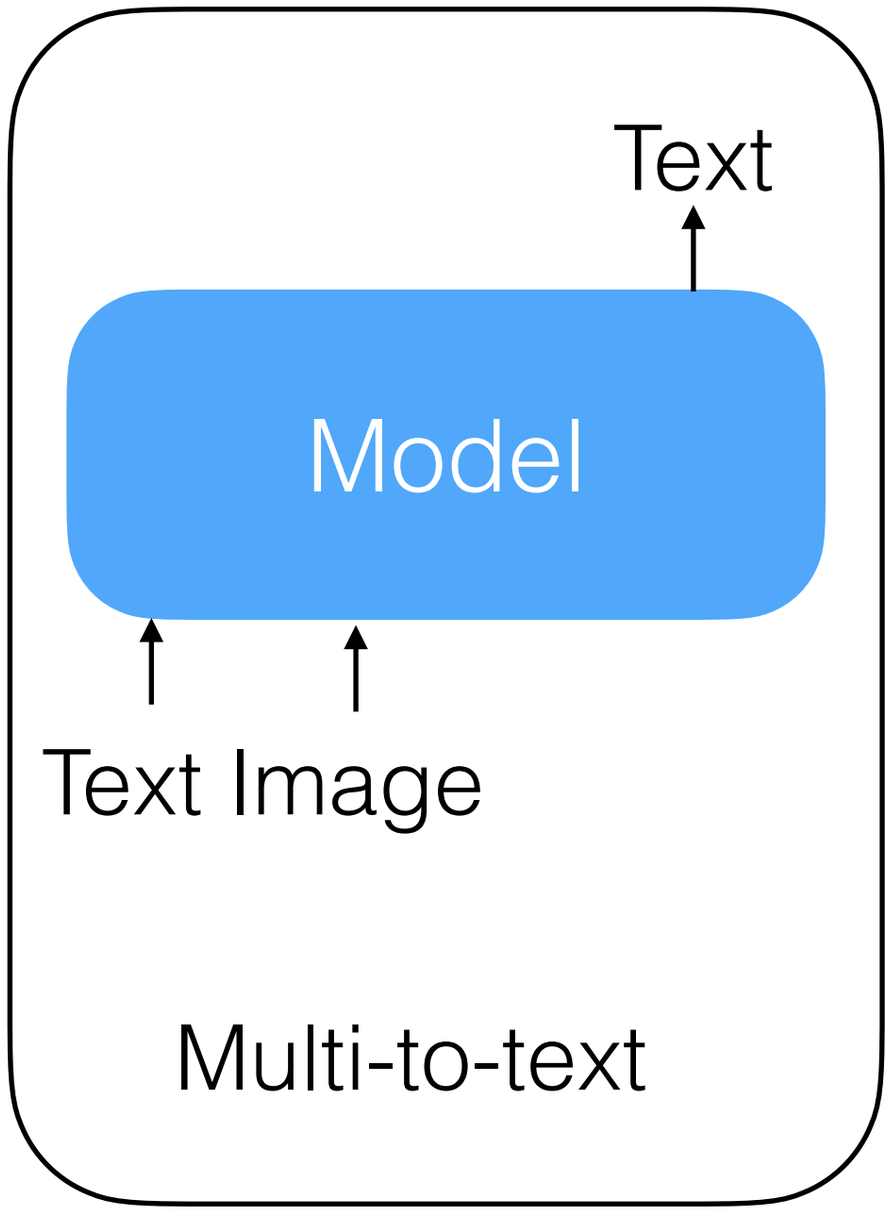
```
Model architecture:
Number of layers: 12
Number of attention heads: 12
Hidden size: 768
Head dimension: 64

Output shape: torch.Size([1, 197, 768])

CLS token representation shape: torch.Size([1, 768])
  (Used for classification tasks)
```

```
Top 5 predictions:
========================================
1. Egyptian cat: 93.74%
2. tabby, tabby cat: 3.84%
3. tiger cat: 1.44%
4. lynx, catamount: 0.33%
5. Siamese cat, Siamese: 0.07%
```

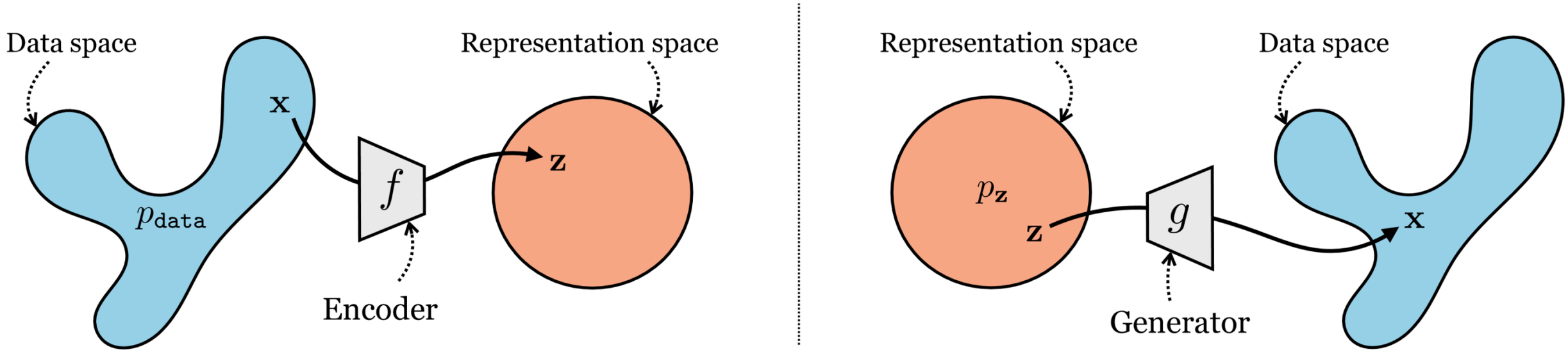
https://github.com/cmu-l3/anlp-spring2026-code

# Today's lecture

- Vision architecture basics
  - ViT
- **Learning image representations**
  - **CLIP**
- Combining with a language model
  - Llava



# Learning image representations




Foundations of Computer Vision, Torralba et al

# Contrastive Language-Image Pre-training (CLIP)

Goal: pre-training objective for learning image representations

- Learn from text
    - At the time, models were pre-trained on classification: the only textual supervision was from the class label.
    - A textual description of an image provides much more information than one class label.
- Scalable
    - At the time, image pre-training was largely limited to hand labeled data.
    - Want to have the property of improving by adding more compute.

# CLIP

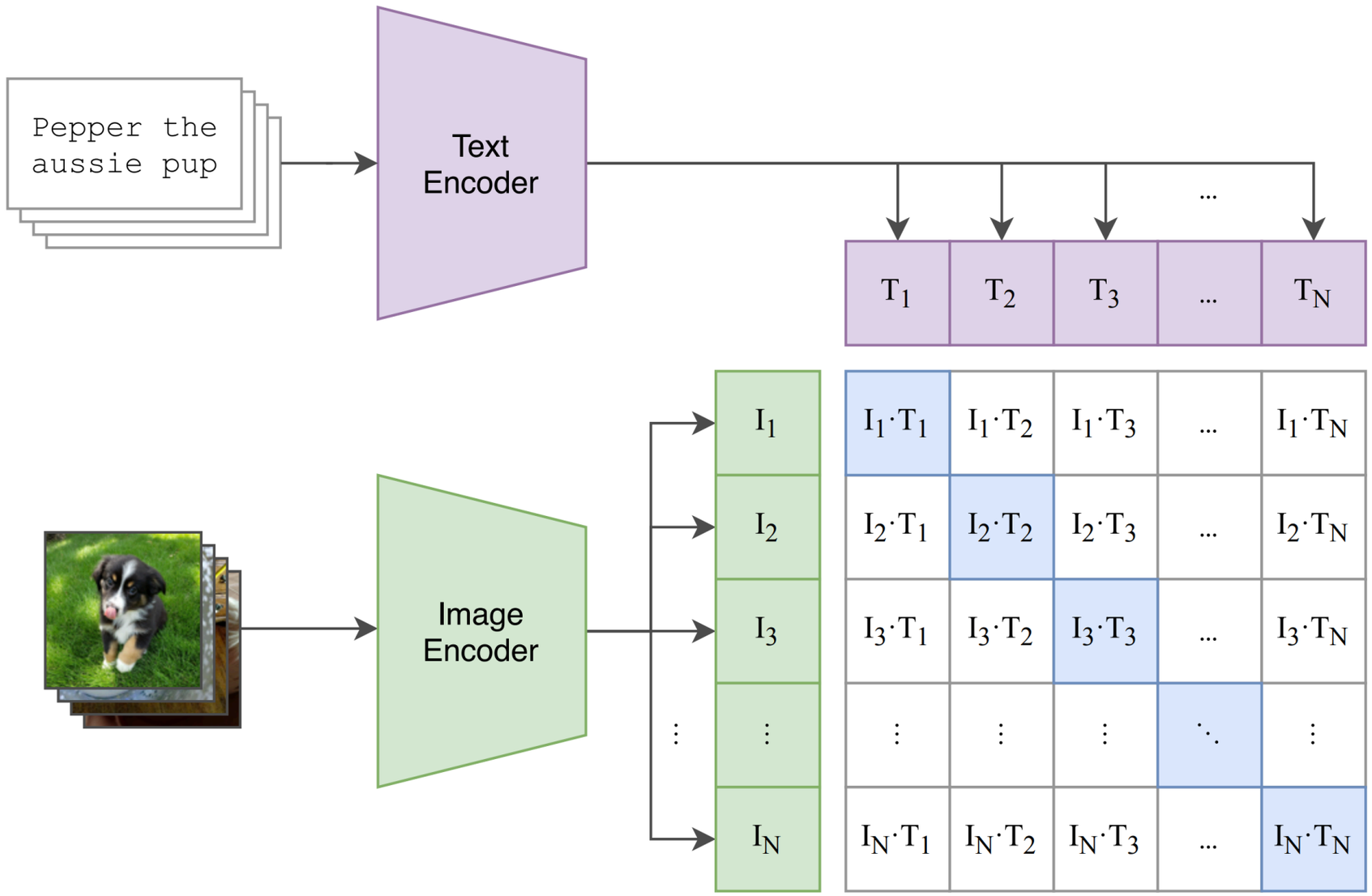
- Idea: learn image and text representations jointly in a shared embedding space
  - Learn an image encoder $f_I(x) \rightarrow z_I$
  - Learn a text encoder $f_T(y) \rightarrow z_T$
  - The representations for a paired image and its text should be close together.
  - The representations for an unpaired image and text should be far apart.
- Apply the method over a large dataset of (image, text) pairs

# CLIP

- Data: pairs of (image, text)
  - E.g. 400 million web images with their text descriptions
- Image encoder $f_I(x) \rightarrow z_I$
  - E.g. vision transformer
- Text encoder $f_T(y) \rightarrow z_T$
  - E.g. transformer

# CLIP

- Basic idea: Given *N* (image, text) pairs, classify which image is paired with which text



# CLIP

$$L((x_1, y_1), \ldots, (x_N, y_N)) =$$

Push up dot product of the correct pair

$$-\frac{1}{2}\sum_{n=1}^{N}\left[\log\frac{\exp\left(f_I(x_n)^\top f_T(y_n)\right)}{\sum_j \exp\left(f_I(x_j)^\top f_T(y_n)\right)} + \log\frac{\exp\left(f_I(x_n)^\top f_T(y_n)\right)}{\sum_j \exp\left(f_I(x_n)^\top f_T(y_j)\right)}\right]$$

Push down dot product of other pairs

Softmax over images

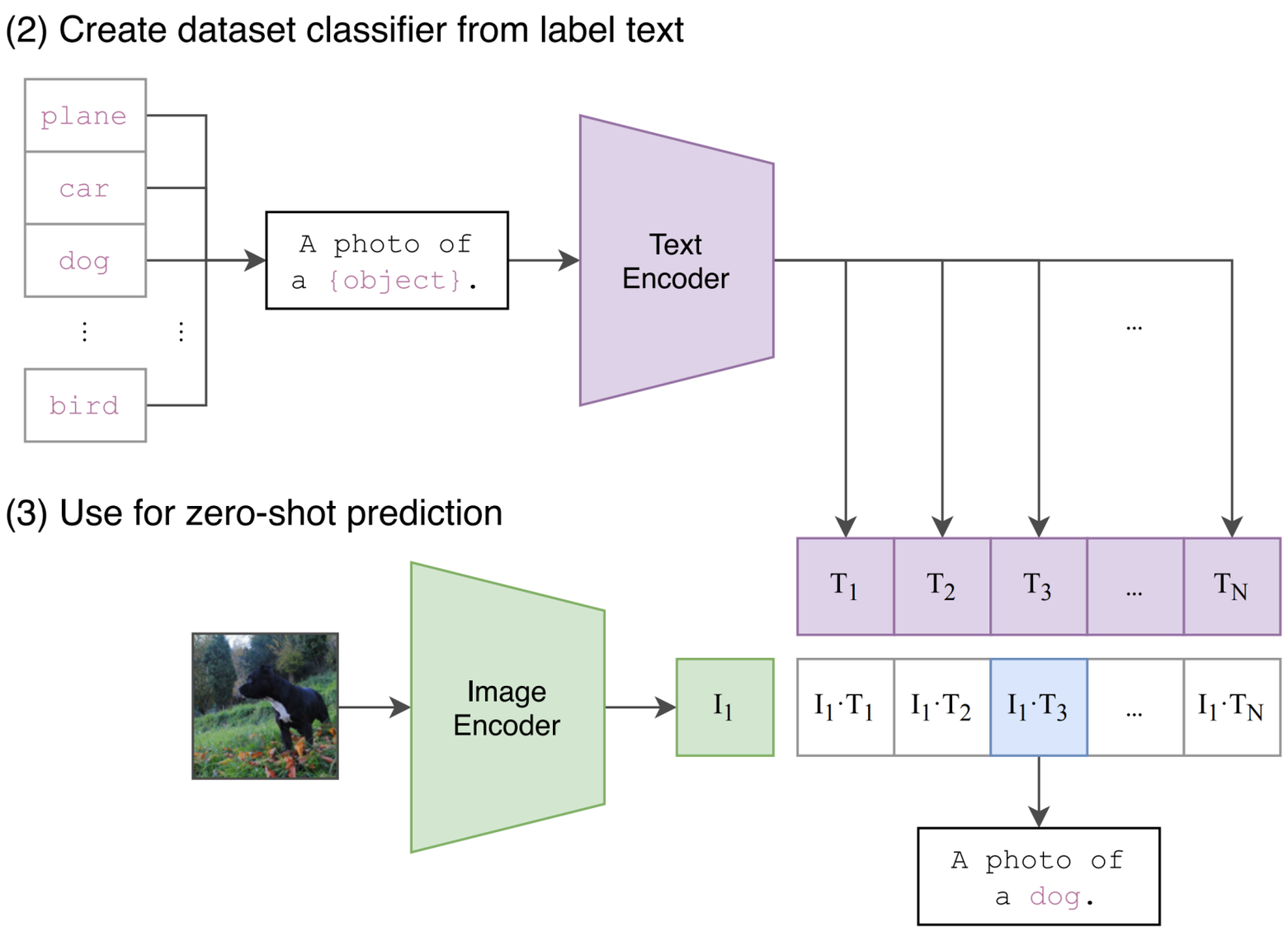
Softmax over text

Each term is a cross-entropy loss, where $p_\theta \propto f_I(x)^\top f_T(y)$ and $p_*$ puts all of its mass on the correct pair

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

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

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

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

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

# CLIP

- Example “zero-shot” usage



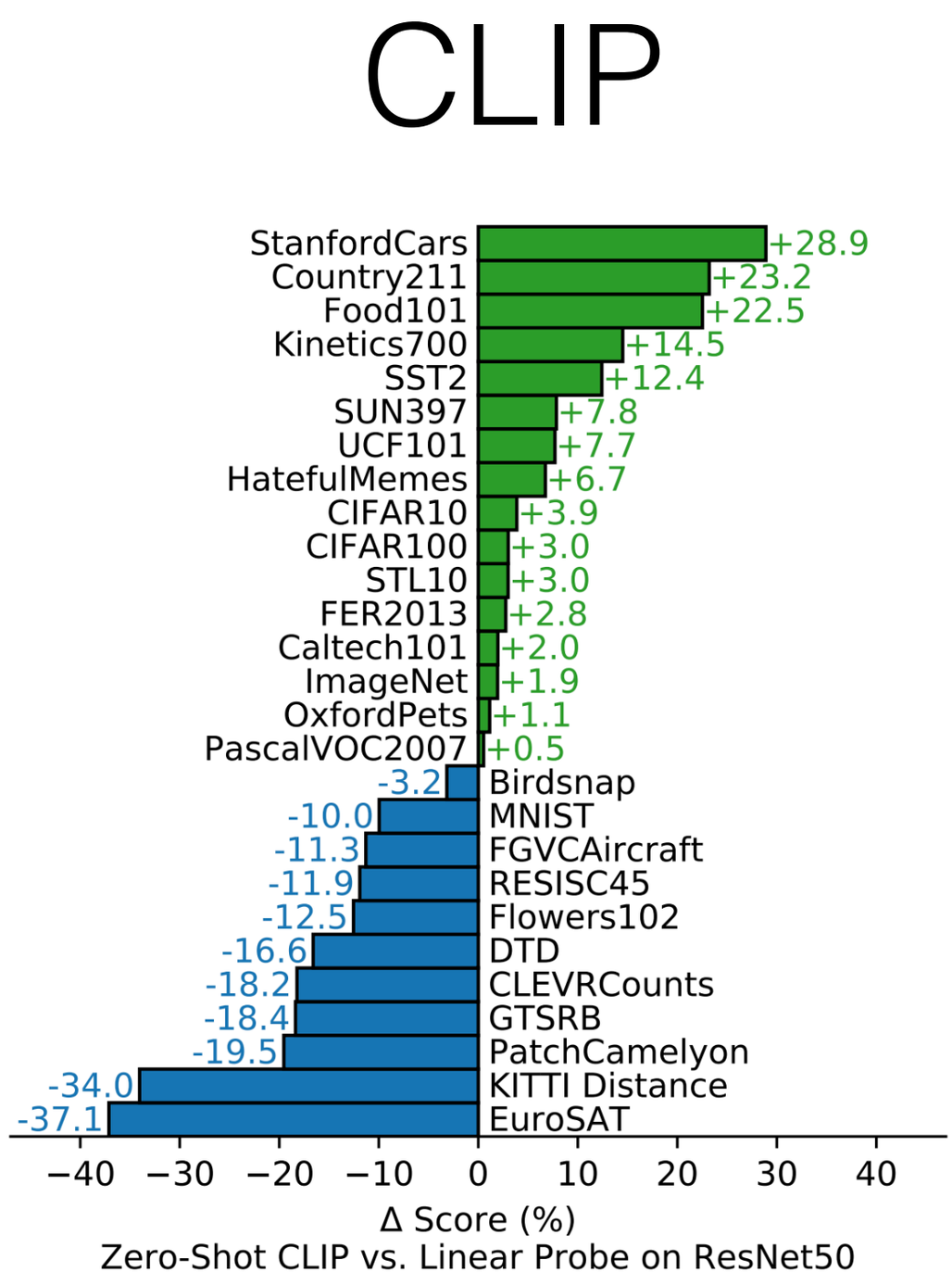
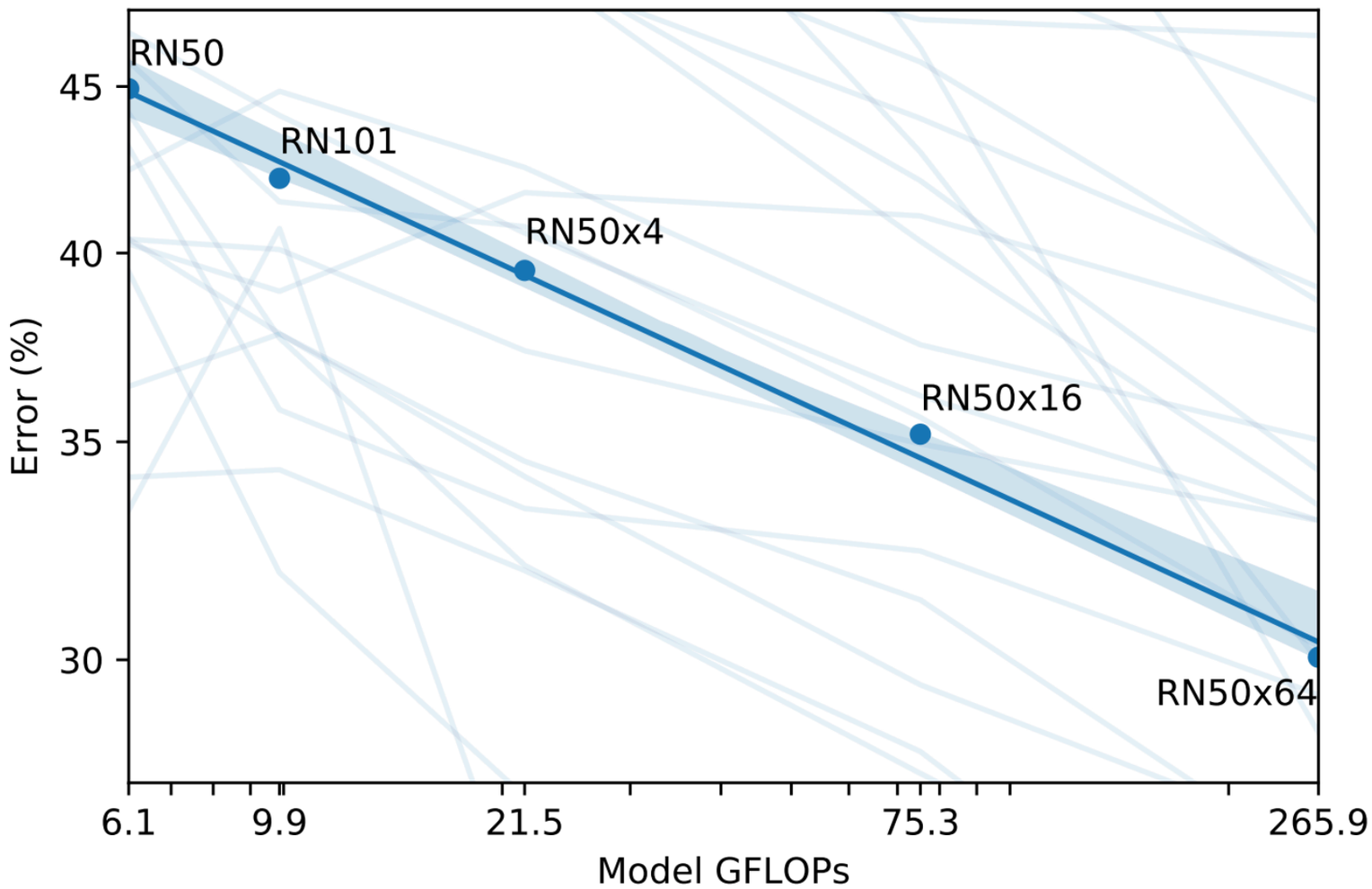


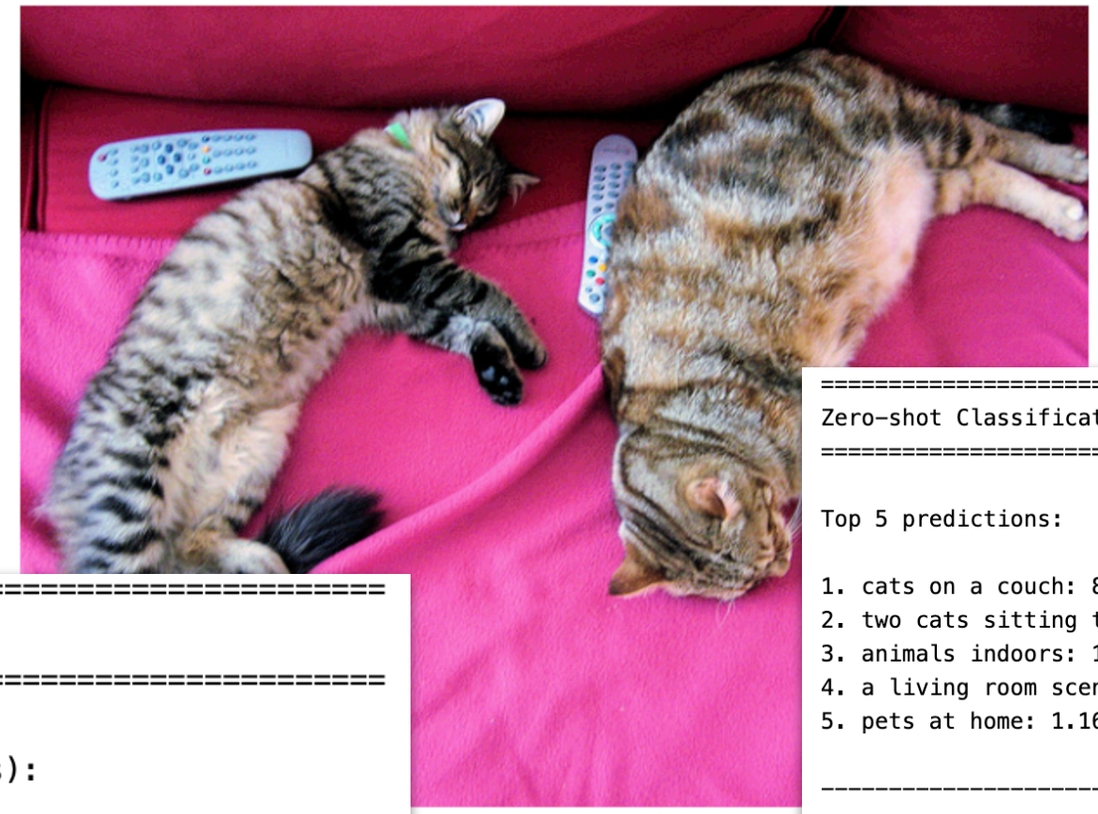

*Figure 5.* **Zero-shot CLIP is competitive with a fully supervised baseline.** Across a 27 dataset eval suite, a zero-shot CLIP classifier outperforms a fully supervised linear classifier fitted on ResNet-50 features on 16 datasets, including ImageNet.

# CLIP




Zero-shot performance scaling as a function of pre-training compute

# Code example



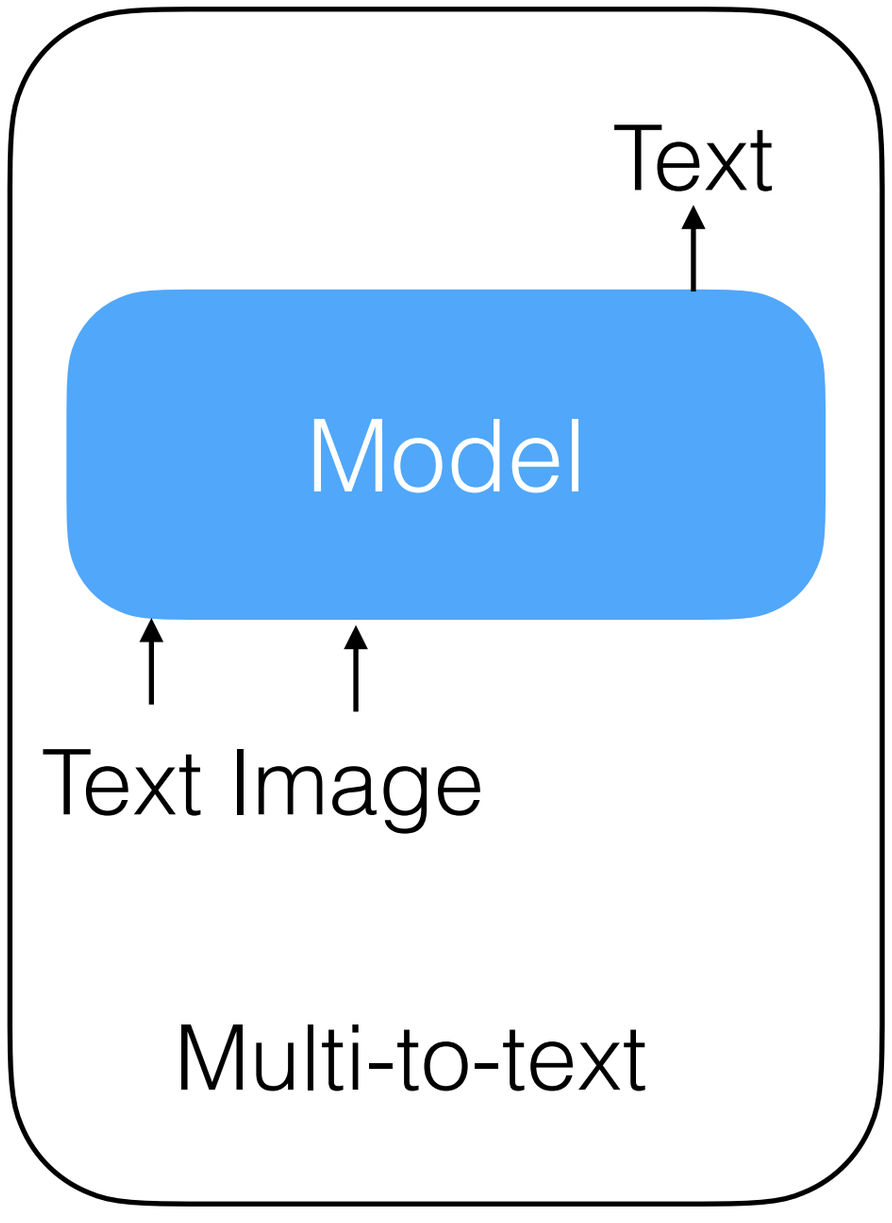

```
==================================================
Cat vs Dog Classification
==================================================

Raw similarity scores (logits):
  a photo of a cat: 18.904
  a photo of a dog: 11.716

Probabilities:
  a photo of a cat: 99.9%
  a photo of a dog: 0.1%

Prediction: a photo of a cat (99.9%)
```

```
============================================================
Zero-shot Classification Results
============================================================

Top 5 predictions:

1. cats on a couch: 89.08%
2. two cats sitting together: 5.39%
3. animals indoors: 1.97%
4. a living room scene: 1.62%
5. pets at home: 1.16%

----------------------------------------

All predictions:

cats on a couch                ████████████████████ 89.08%
two cats sitting together      ▉ 5.39%
animals indoors                 1.97%
a living room scene             1.62%
pets at home                    1.16%
a photo of furniture            0.35%
a photo of a cat                0.33%
a photo of a kitten             0.10%
a photo of a person             0.00%
a photo of a dog                0.00%
```

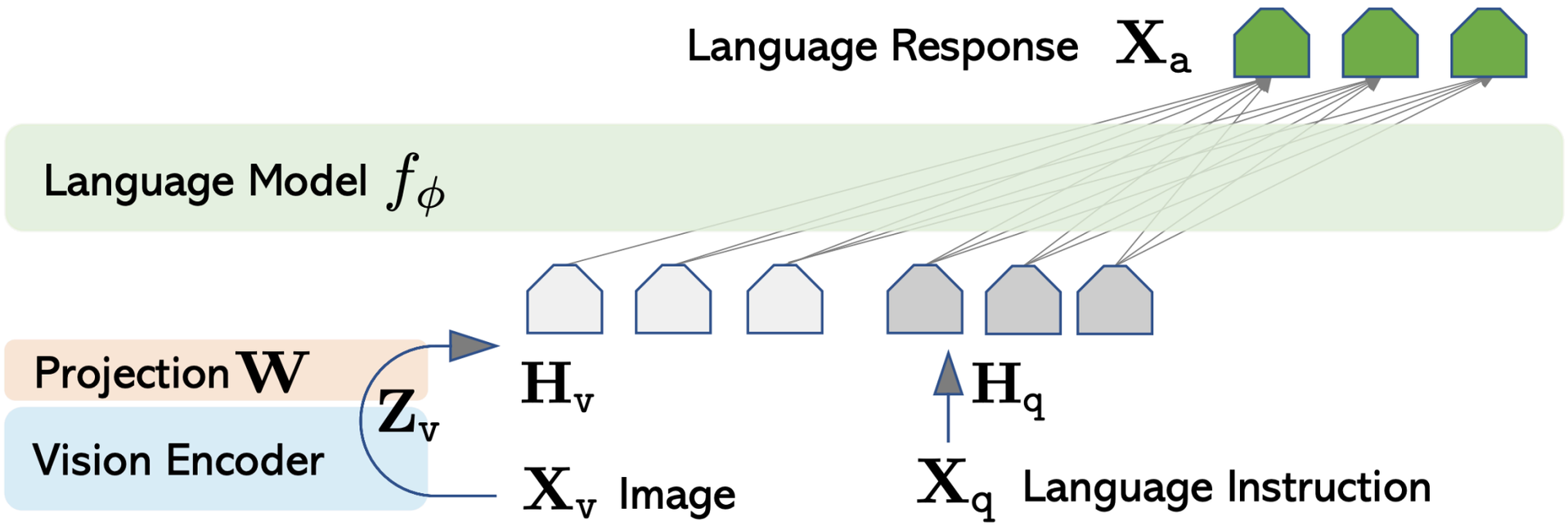
https://github.com/cmu-l3/anlp-fall2025-code

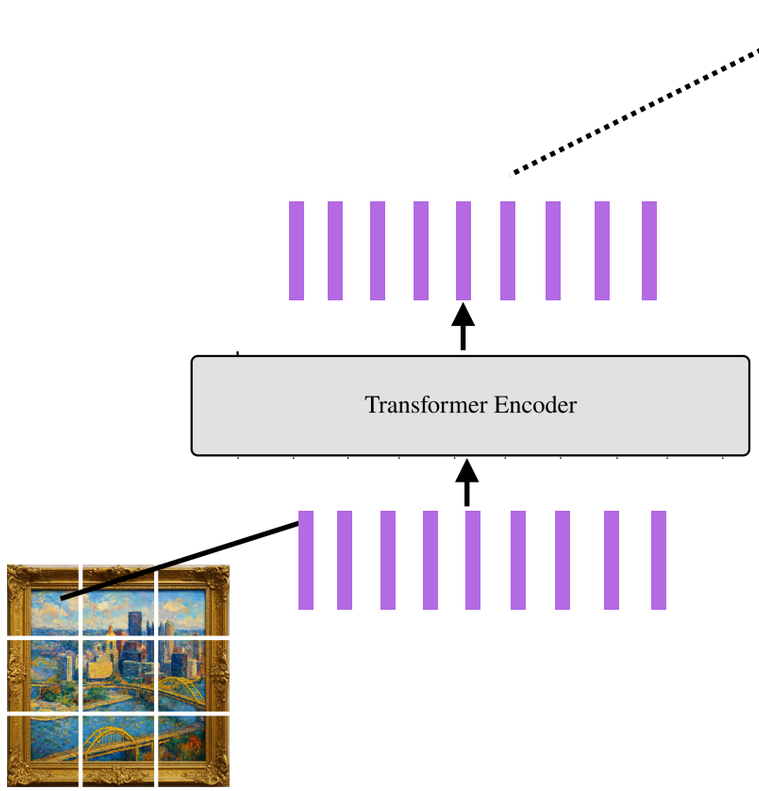
# Today's lecture

- Vision architecture basics
  - ViT
- Learning image representations
  - CLIP
- **Combining with a language model**
  - **Llava**



# Llava




- ‘Image tokens’:
  - Get sequence of vectors from the pretrained CLIP ViT
- Text tokens: same as usual

# Llava

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

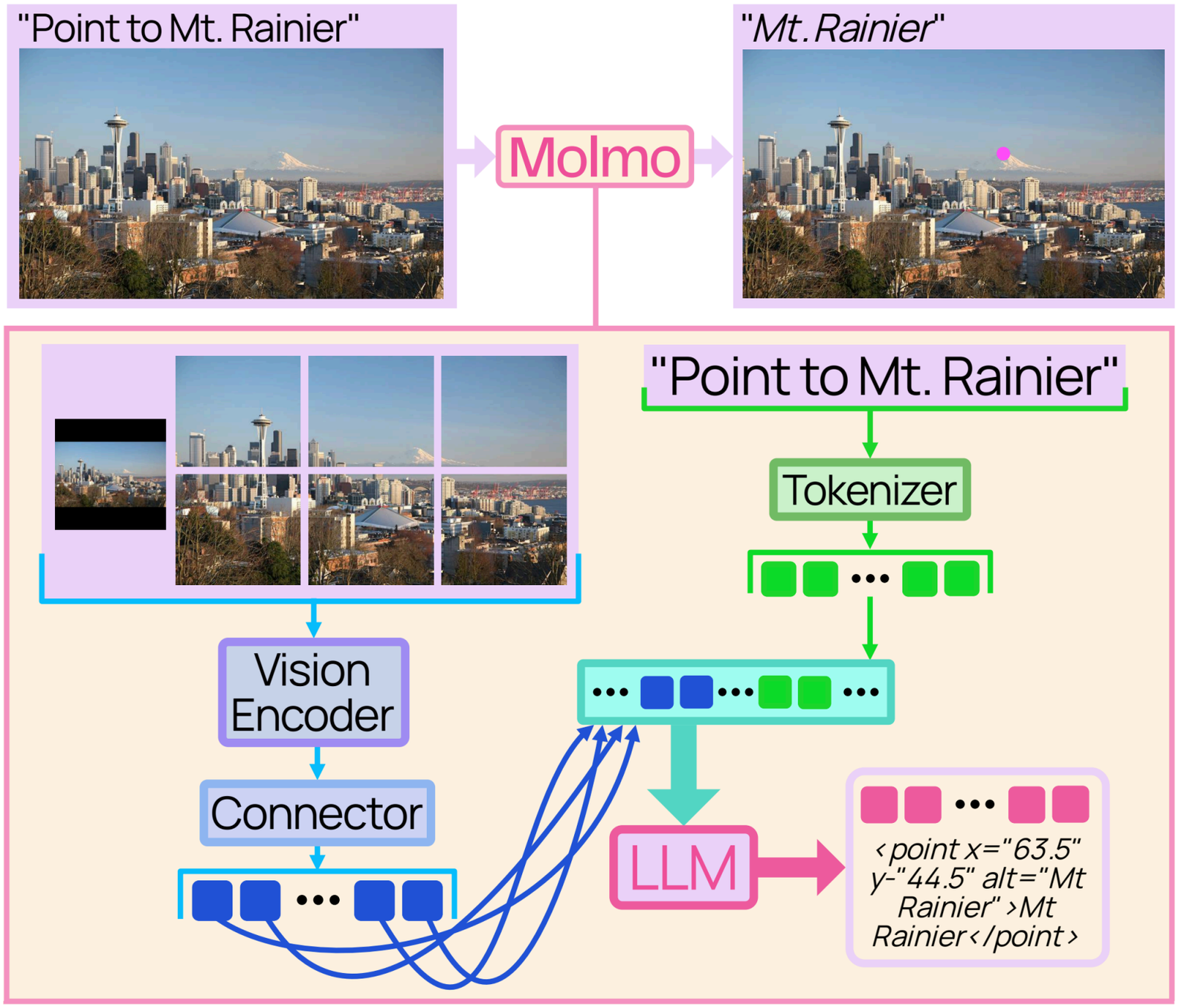
User What is unusual about this image?

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.

# General pipeline

- Image preprocessing
    - E.g. split into patches and vectorize
- Image encoding
    - E.g. use a pre-existing encoding model (e.g., CLIP, get the ViT vectors from the last layer)
- Provide the encodings to a LLM
    - E.g. linearly transform the vectors to be the model's embedding dimension
- Train/fine-tune on data that has text and images
    - For image positions, skip the loss

# Example: MOLMO (AI2)

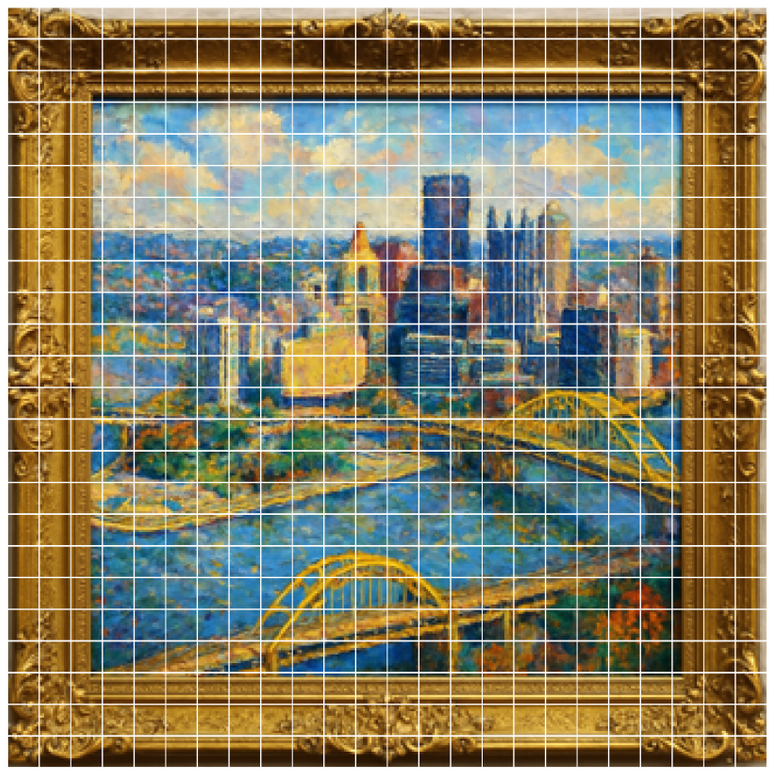


# Example: MOLMO (AI2)

- Image preprocessing

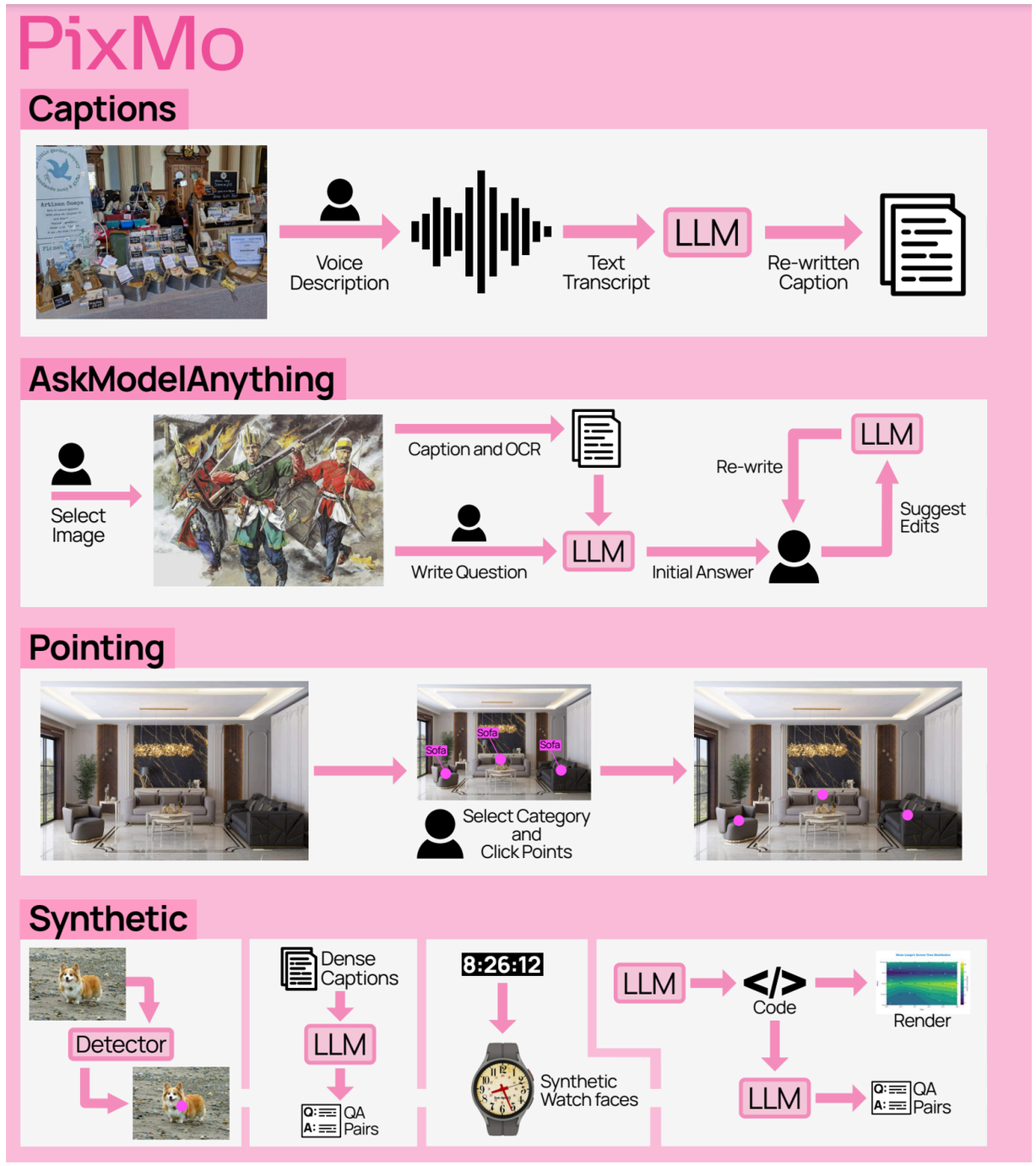
# Example: MOLMO (AI2)

- Image encoding: CLIP ViT-L/14 336px
  - 336 x 336 image
  - 14 x 14 patches
    - => 24 x 24 grid
- Pool together each 2x2 patch subset then transform to the LLM's embedding dimension
  - => 12 x 12 vectors
- Do the above for 1 full image and 12 crops

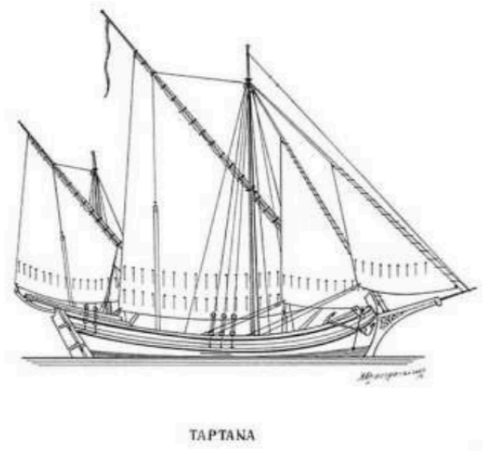


# Example: MOLMO (AI2)

- Data



# Example: MOLMO (AI2)




**Generate a long caption about this image.**

This black and white sketch depicts a boat, elegantly stationed at a pier. The boat, seemingly wooden, showcases a classic design with four prominent sails catching an imagined breeze. Below the boat, text inscribed reads "TAPTANA" in clear, uppercase letters. In the lower right-hand corner, another text, written in cursive and appearing less legible, adds an air of authenticity to the artwork. The meticulous detail in the sails and the overall structure of the boat highlights the craftsmanship and artistry involved in this sketch.

# Example: MOLMO (AI2)

**how many musicians.**

Counting the <points x1="8.2" y1="37.6" x2="10.9" y2="45.3" x3="13.9" y3="19.4" x4="16.3" y4="11.4" x5="17.4" y5="28.8" x6="23.3" y6="35.5" x7="39.1" y7="27.3" x8="42.4" y8="23.5" x9="56.0" y9="4.8" x10="63.1" y10="28.3" x11="69.3" y11="19.6" x12="71.5" y12="29.7" x13="86.2" y13="17.1" x14="94.6" y14="28.…

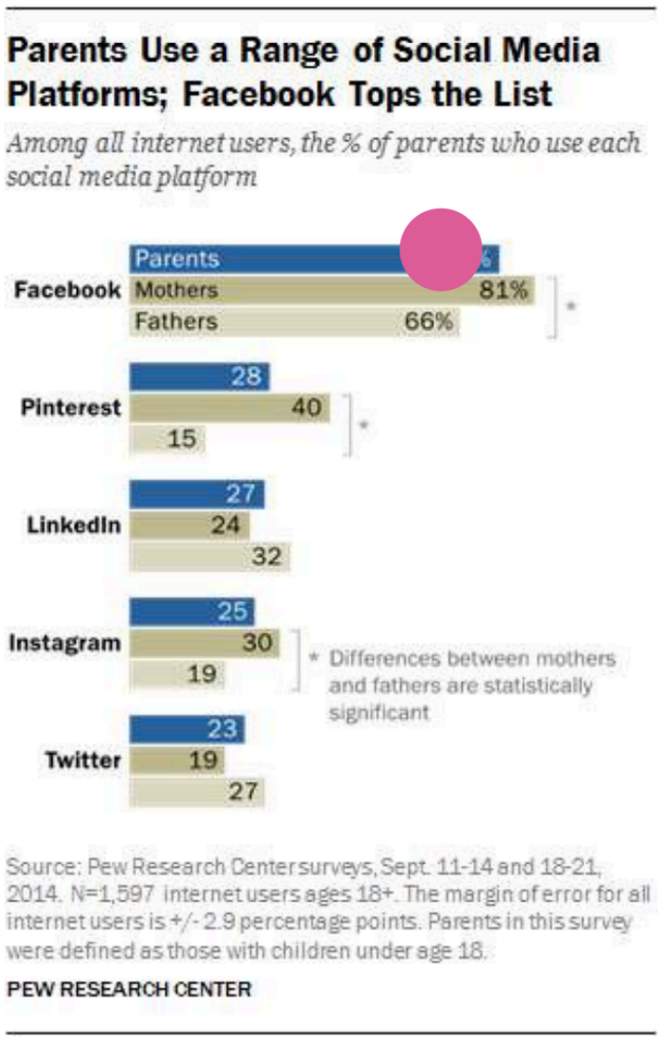
# Example: MOLMO (AI2)

**What kind of donut is this?**

This appears to be a cinnamon sugar donut. The donut itself has a light tan color typical of a traditional fried dough. It's coated with a generous layer of cinnamon sugar, giving it that classic sweet and slightly spicy flavor profile.

The topping isn't cheese as…

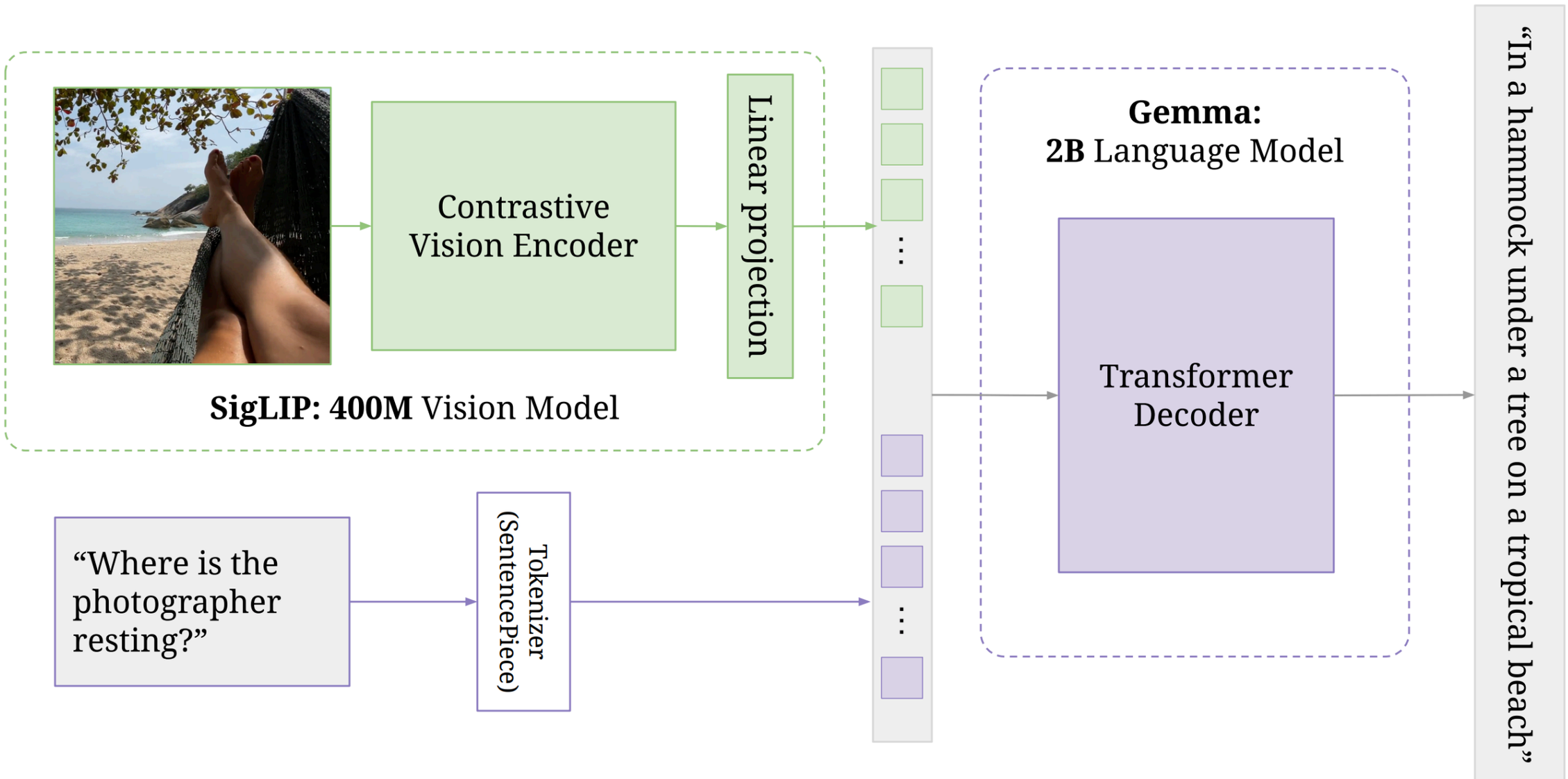
# Example: MOLMO (AI2)



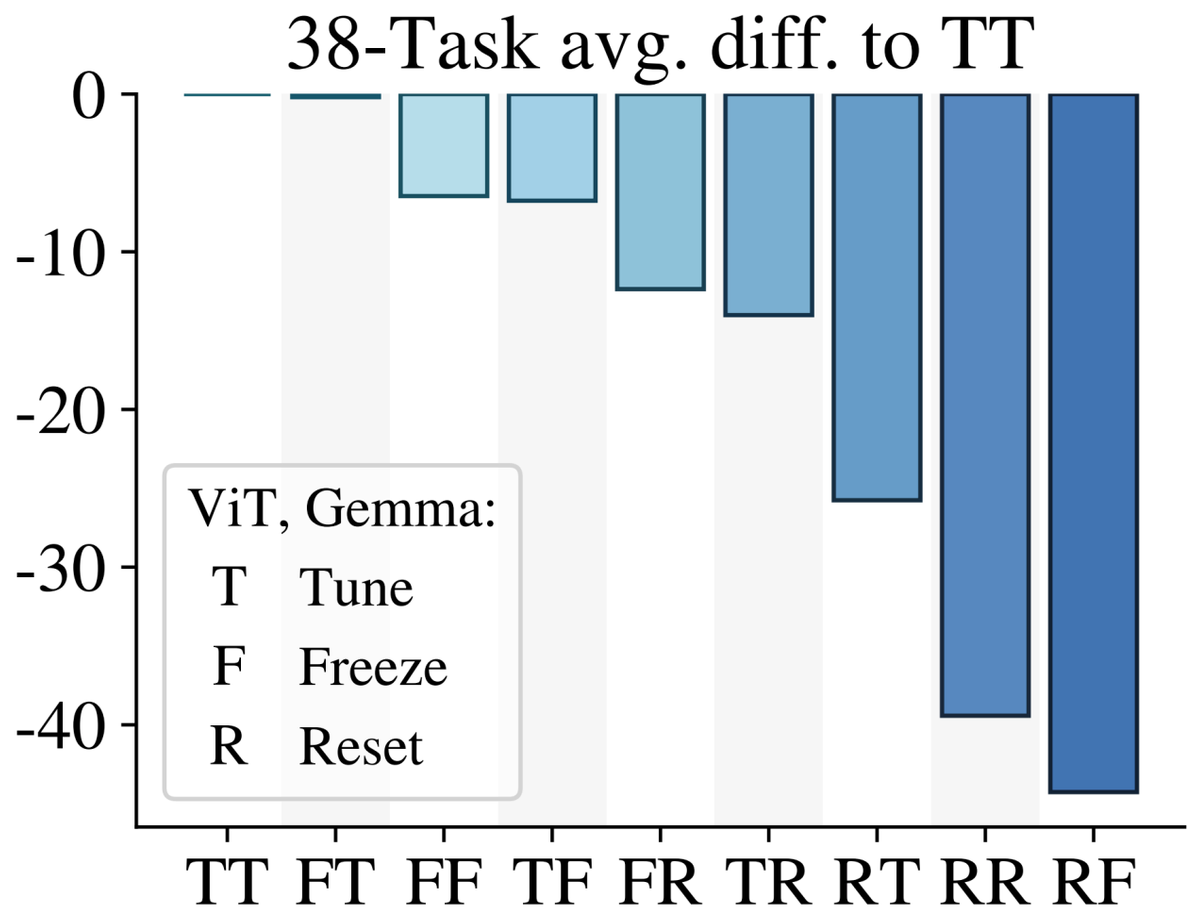

**What percentage of parents use Facebook?**

The image shows a breakdown of social media usage among parents. <point x="66.6" y="24.2" alt="Percentage of parents using Facebook">74 percent</point> of parents use Facebook, making it the most popular social media platform among this demographic. This statistic combines both mothers an…

# Example: PaliGemma (Google)



# Example: PaliGemma (Google)




Updating **both** the image encoder and the LM was best

# Example: PaliGemma (Google)

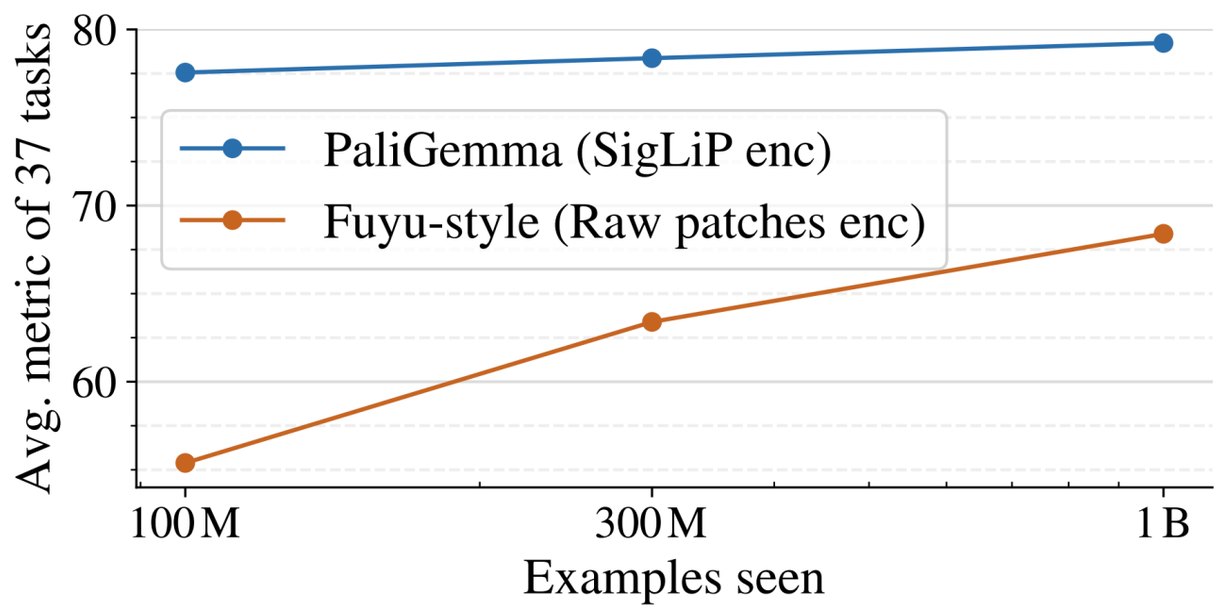
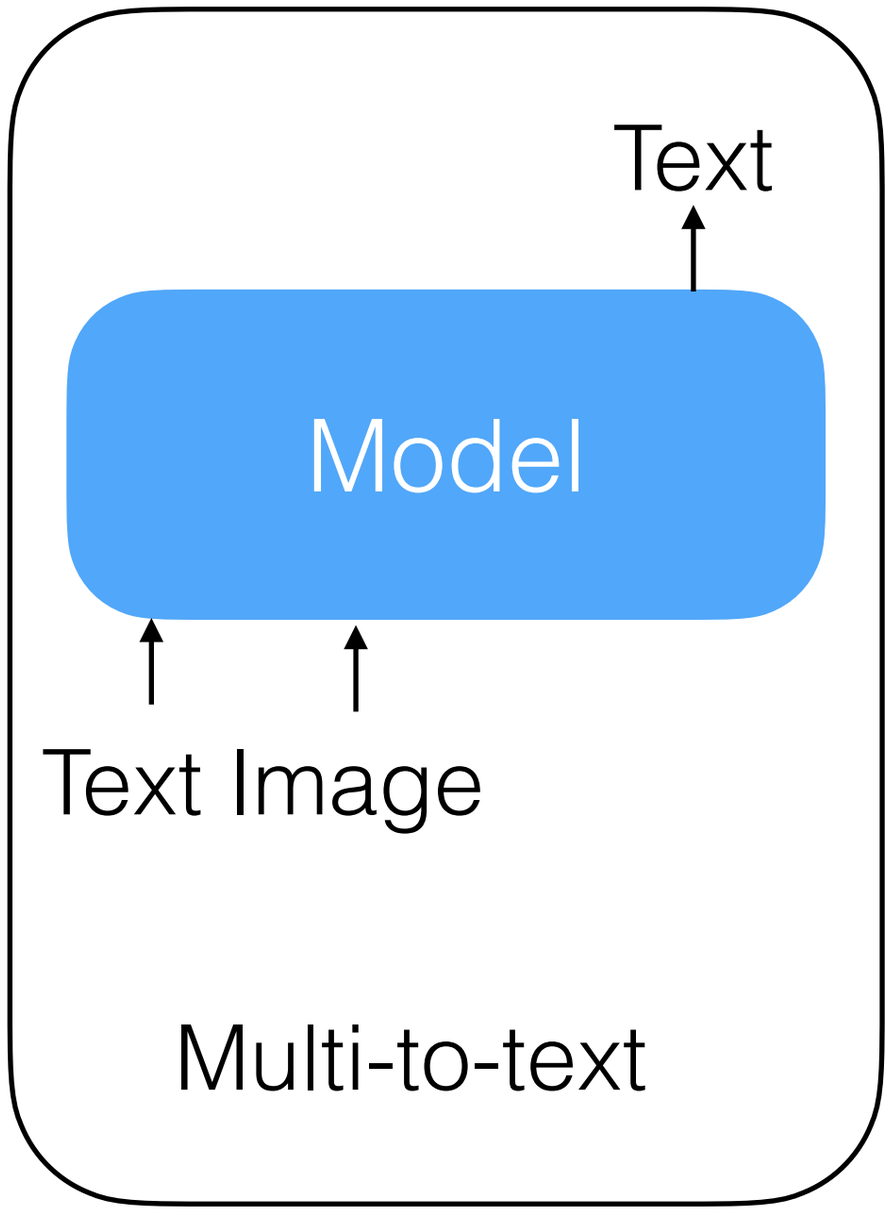


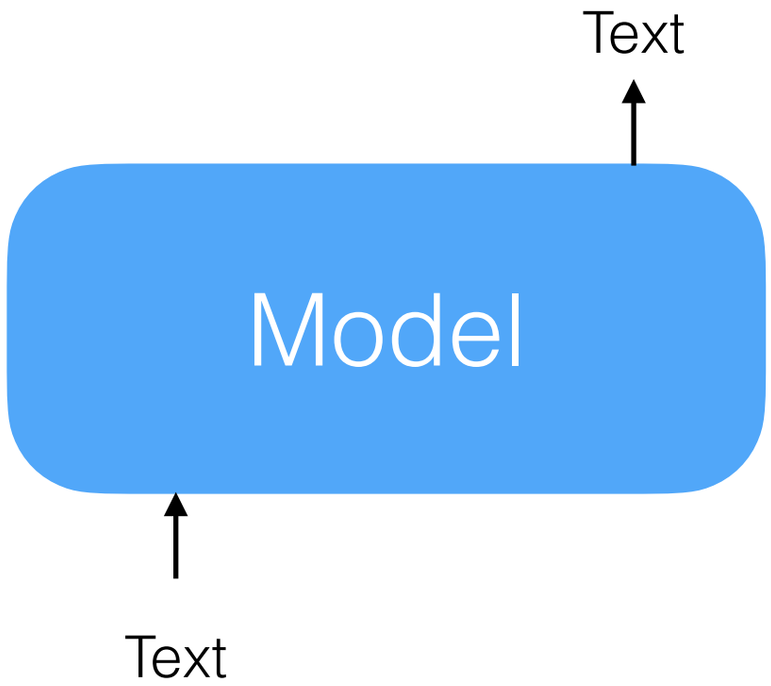

Figure 8 | Not using a SigLiP image encoder at all and instead passing a linear projection of raw RGB patches to Gemma works, but is significantly less sample-efficient.

What if we got rid of the image encoder?

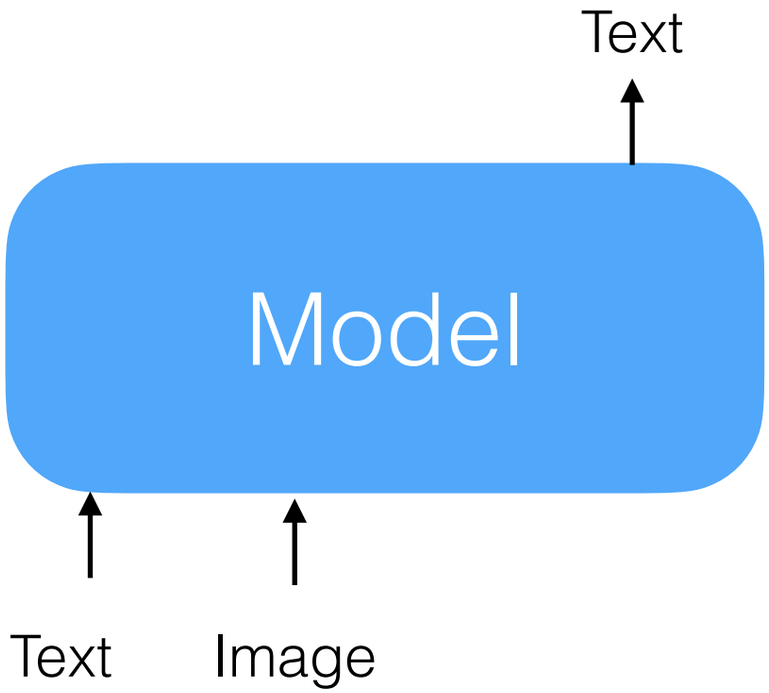
# Today's lecture

- Vision architecture basics
  - ViT
- Learning image representations
  - CLIP
- Combining with a language model
  - Llava



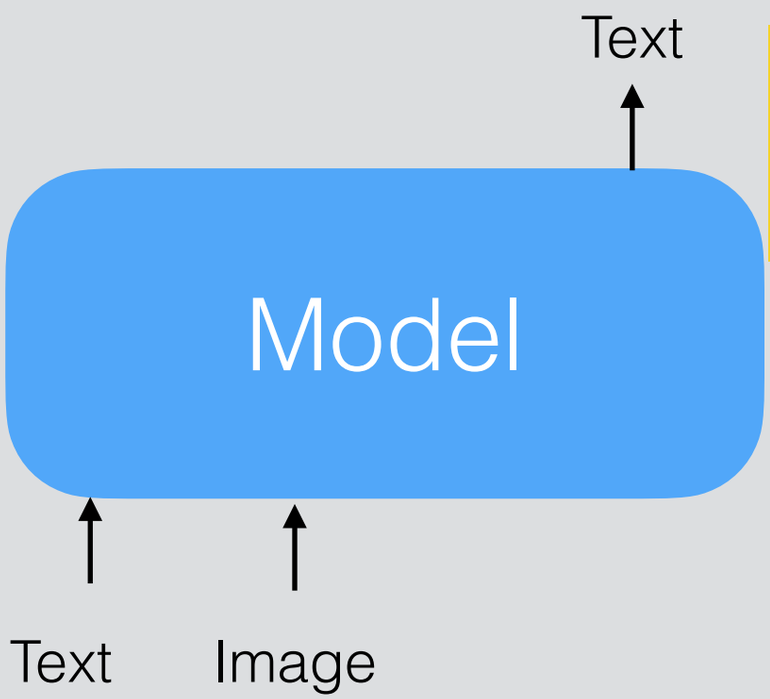



Text-to-text

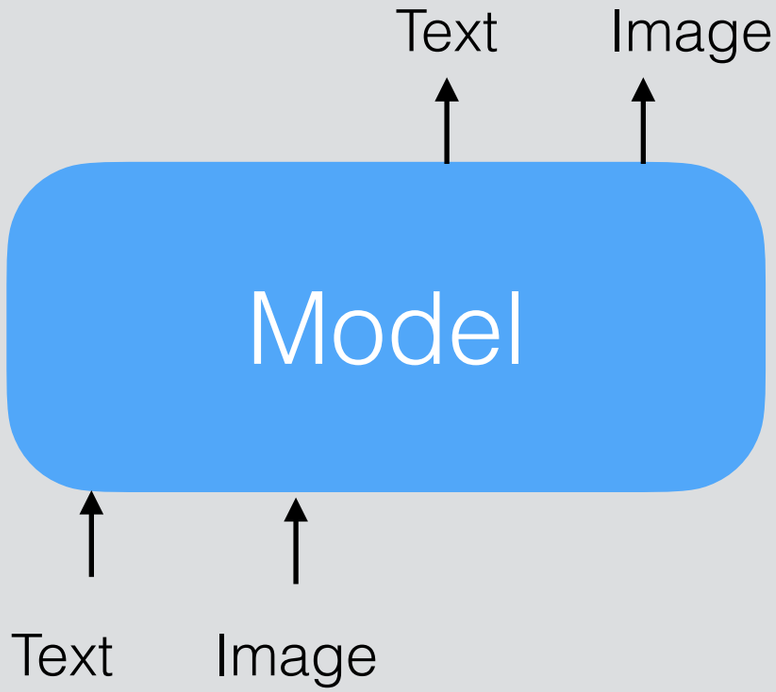



Multi-to-text

Next 2 Lectures




Multi-to-image




Multi-to-multi

Thank you