10-423/10-623 Generative AI

Machine Learning Department
School of Computer Science
Carnegie Mellon University

Vision-Language Models (VLMs)

Matt Gormley & Pat Virtue
Lecture 14
March 10, 2025

Reminders

- Homework 3: Applying and Adapting LLMs
  - Out: Sun, Feb 23
  - Due: Thu, Mar 13 at 11:59pm
- Quiz 4
  - in-class, Mon, Mar 17
  - lectures 12 – 15
- Homework 4: Multimodal Foundation Models
  - Out: Thu, Mar 13
  - Due: Mon, Mar 24 at 11:59pm

(Slides with blue titles from Henry Chai)

VISION LANGUAGE MODELS (VLMS)

Multimodal Models

- Previously: Text-to-image models – adapt generative models for vision in order to guide their output toward some desired target using natural language
  - Output is still an image
- Today: visual language models (VLMs) – adapt generative models for text in order to allow them to interact with images (as well as text) as input
  - Output is (typically) still text

VLM: Tasks

- Common benchmarks for VLMs include
  - **Visual reasoning**: given an image (or a pair of images) determine if some natural language statement about the image(s) is true or false
  - **Visual grounding**: locate an object in some image given a natural language description
  - **Visual question answering**: given an image (or images), respond to arbitrary, potentially open-ended questions about the content.
  - **Caption generation**: create natural language descriptions of content of some image

VLM: Tasks

- Common benchmarks for VLMs include
  - **Visual reasoning**: given an image (or a pair of images) determine if some natural language statement about the image(s) is true or false

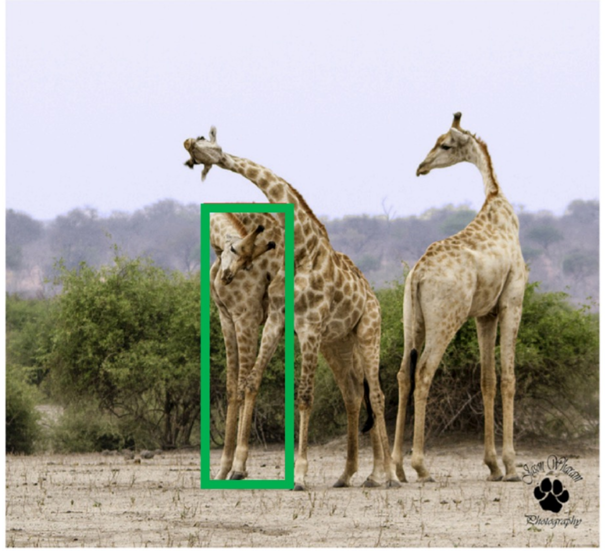
The left image contains twice the number of dogs as the right image, and at least two dogs in total are standing.

One image shows exactly two brown acorns in back-to-back caps on green foliage.

Source: https://aclanthology.org/P19-1644.pdf

VLM: Tasks

- Common benchmarks for VLMs include
  - **Visual reasoning**: given an image (or a pair of images) determine if some natural language statement about the image(s) is true or false
  - **Visual grounding**: locate an object in some image given a natural language description



RefCOCO:
1. giraffe on left
2. first giraffe on left

RefCOCO+:
1. giraffe with lowered head
2. giraffe head down

RefCOCOg:
1. an adult giraffe scratching its back with its horn
2. giraffe hugging another giraffe

Source: https://arxiv.org/pdf/1608.00272

VLM: Tasks

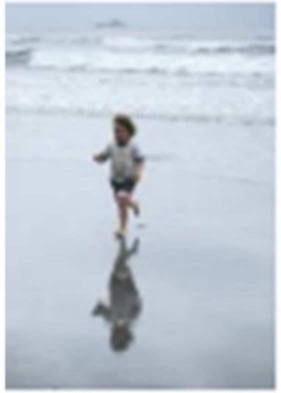
- Common benchmarks for VLMs include

- **Visual question answering**: given an image (or images), respond to arbitrary, potentially open-ended questions about the content.

Source: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8100153

VLM: Tasks

- Common benchmarks for VLMs include



Ground Truth Caption: A little boy runs away from the approaching waves of the ocean.

Generated Caption: A young boy is running on the beach.

Ground Truth Caption: A brunette girl wearing sunglasses and a yellow shirt.

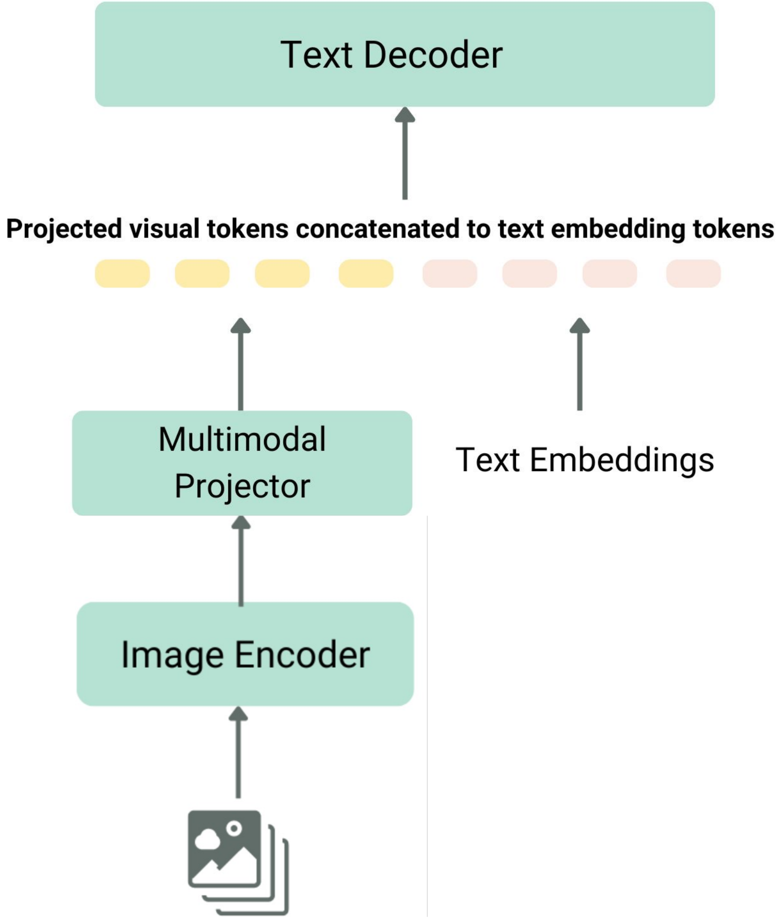
Generated Caption: A woman in a black shirt and sunglasses smiles.

- **Caption generation**: create natural language descriptions of content of some image

Source: https://dl.acm.org/doi/pdf/10.1145/3295748

VLM: Architecture

- High-level idea: convert both the image and the text inputs into embedding vectors, then pass those vectors into a decoder-only transformer and do next (text) token prediction

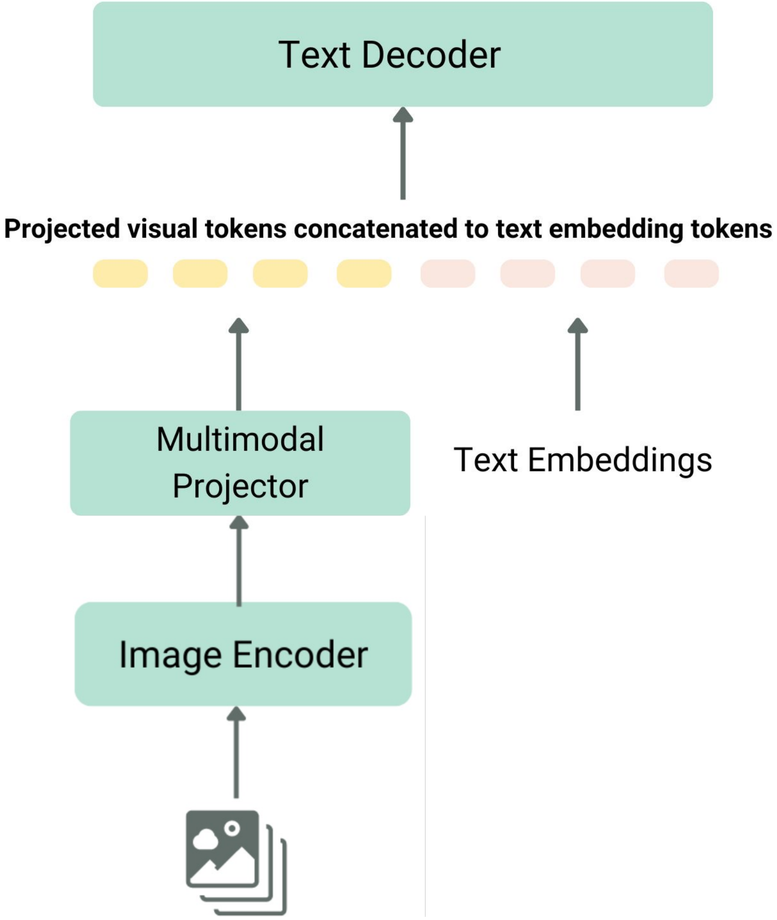


- Two common encoders:
  - VQ-VAE encoder followed by an embedding layer that converts the discrete tokens into dense numerical vectors
  - CLIP encoder, that directly learns an embedding vector using a contrastive pre-training objective

Source: https://huggingface.co/blog/vlms

VLM: Architecture

- High-level idea: convert both the image and the text inputs into embedding vectors, then pass those vectors into a decoder-only transformer and do next (text) token prediction

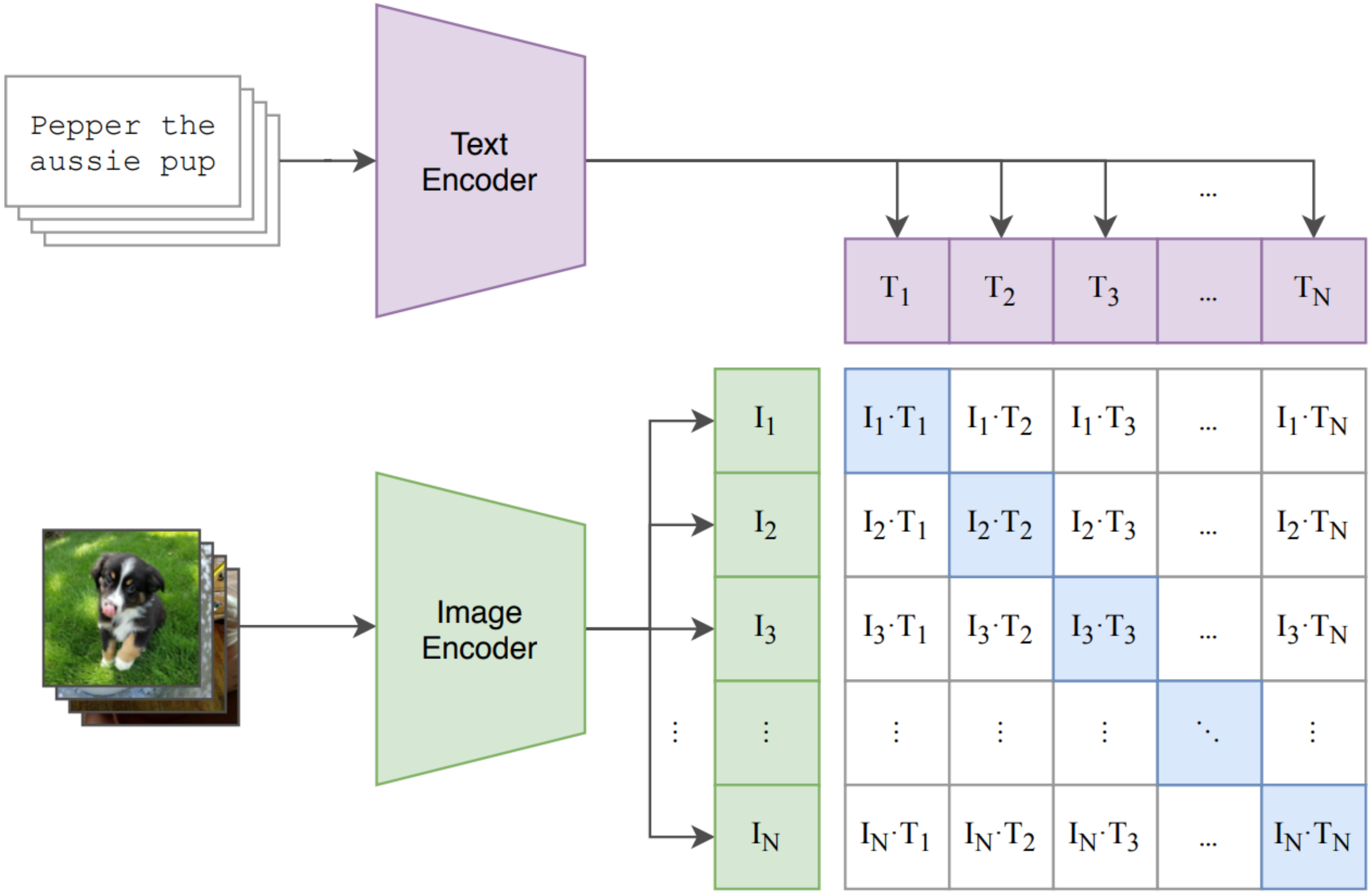


Source: https://huggingface.co/blog/vlms

- Two common encoders:
  - VQ-VAE encoder followed by an embedding layer that converts the discrete tokens into dense numerical vectors
  - CLIP encoder, that directly learns an embedding vector using a contrastive pre-training objective

CLIP

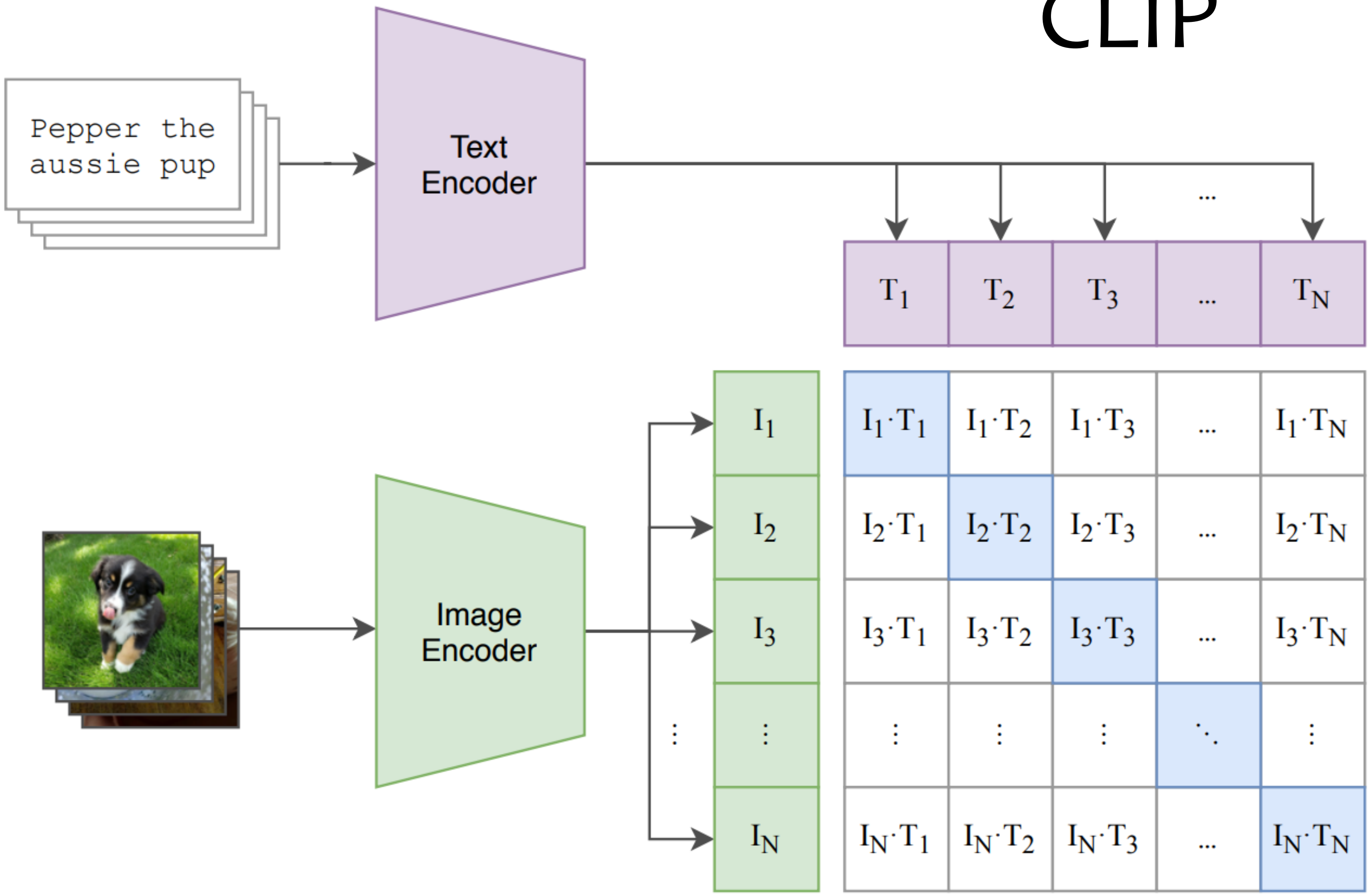
CLIP



- The text encoder is, e.g., an encoder-only transformer
- The image encoder is, e.g., a ResNet-like CNN or ViT
- Both are linearly projected into same-dimensional vectors i.e., the multi-modal embedding space

Figure from Radford et al. (2021) https://arxiv.org/pdf/2103.00020

CLIP



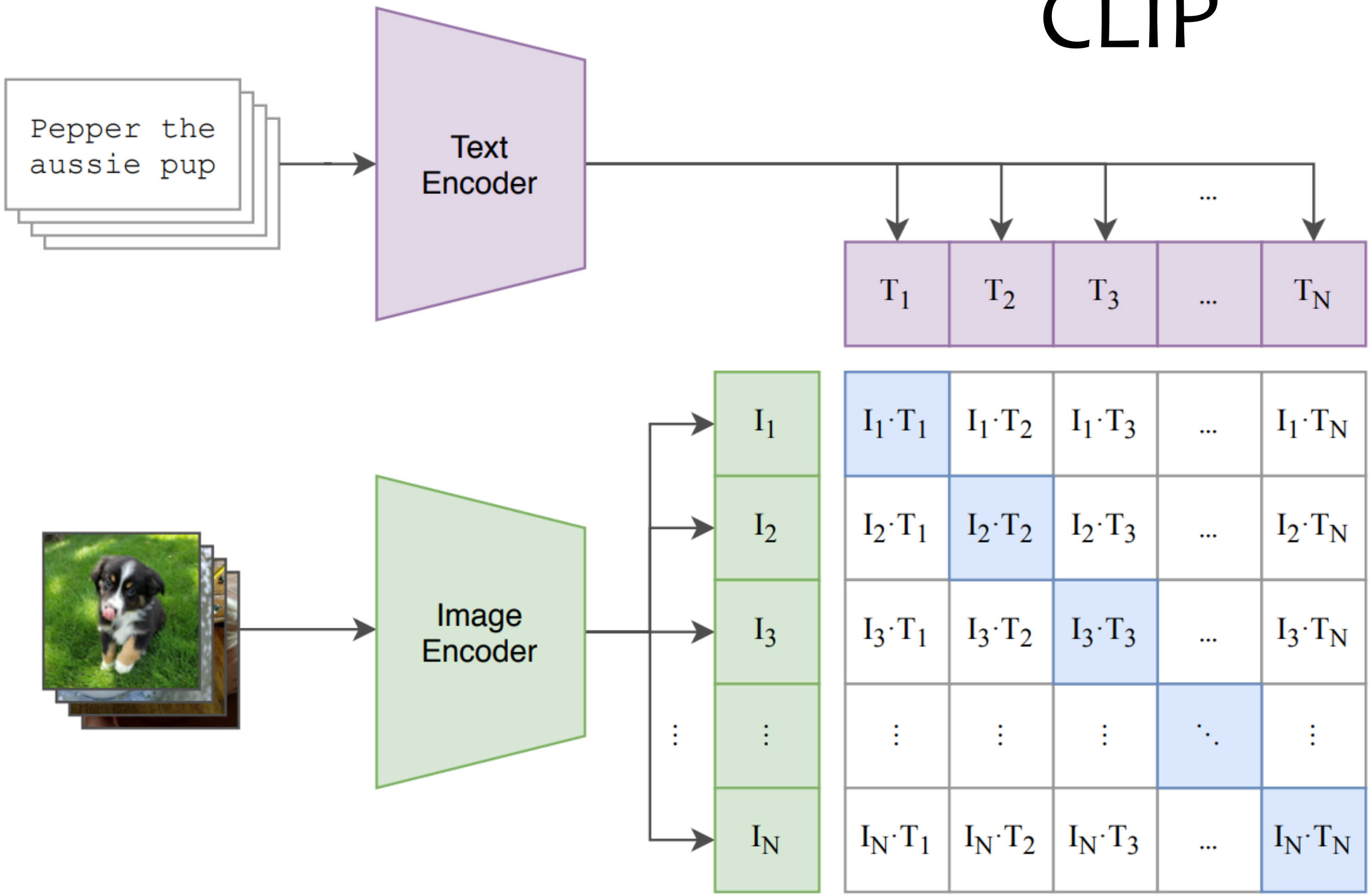
Incorrect (but intuitive) objective function:

$$\max \left[\sum_{i=1}^{N} I_i^\top T_i - \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} I_i^\top T_j \right]$$

Given a mini-batch of N (image, caption) pairs, both encoders are simultaneously pre-trained to maximize the cosine similarity of corresponding image-caption embedding vectors and minimize all other pairwise cosine similarities

Figure from Radford et al. (2021) https://arxiv.org/pdf/2103.00020

CLIP



Correct objective function:

$$\max \sum_{i=1}^{N} \left[\log \frac{\exp\left(\frac{I_i^\top T_i}{\tau}\right)}{\sum_{j=1}^{N} \exp\left(\frac{I_i^\top T_j}{\tau}\right)} + \log \frac{\exp\left(\frac{I_i^\top T_i}{\tau}\right)}{\sum_{j=1}^{N} \exp\left(\frac{I_j^\top T_i}{\tau}\right)} \right]$$

Given a mini-batch of N (image, caption) pairs, both encoders are simultaneously pre-trained to maximize the cosine similarity of corresponding image-caption embedding vectors and minimize all other pairwise cosine similarities

Figure from Radford et al. (2021) https://arxiv.org/pdf/2103.00020

CLIP for Zero Shot Classification

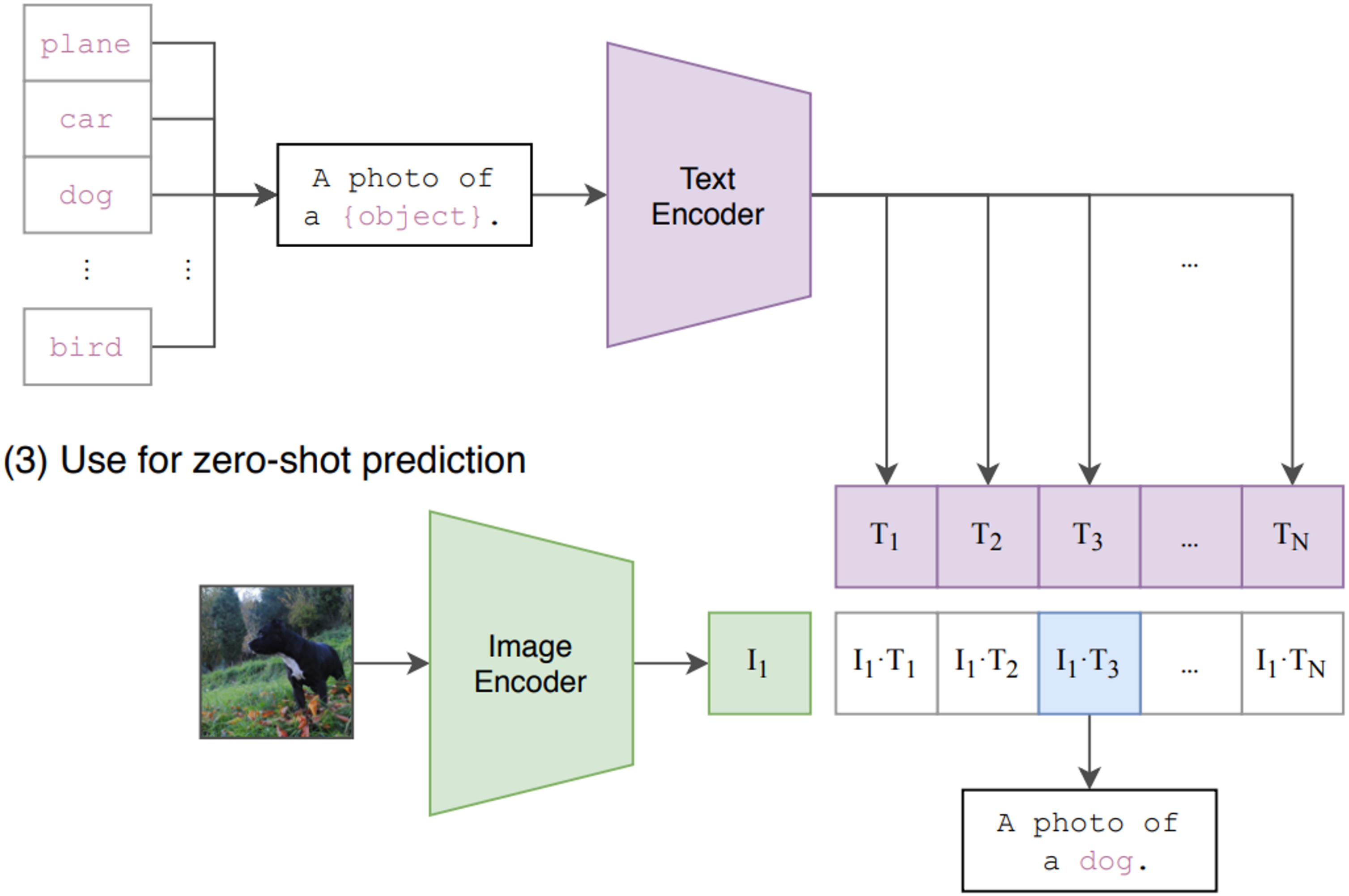
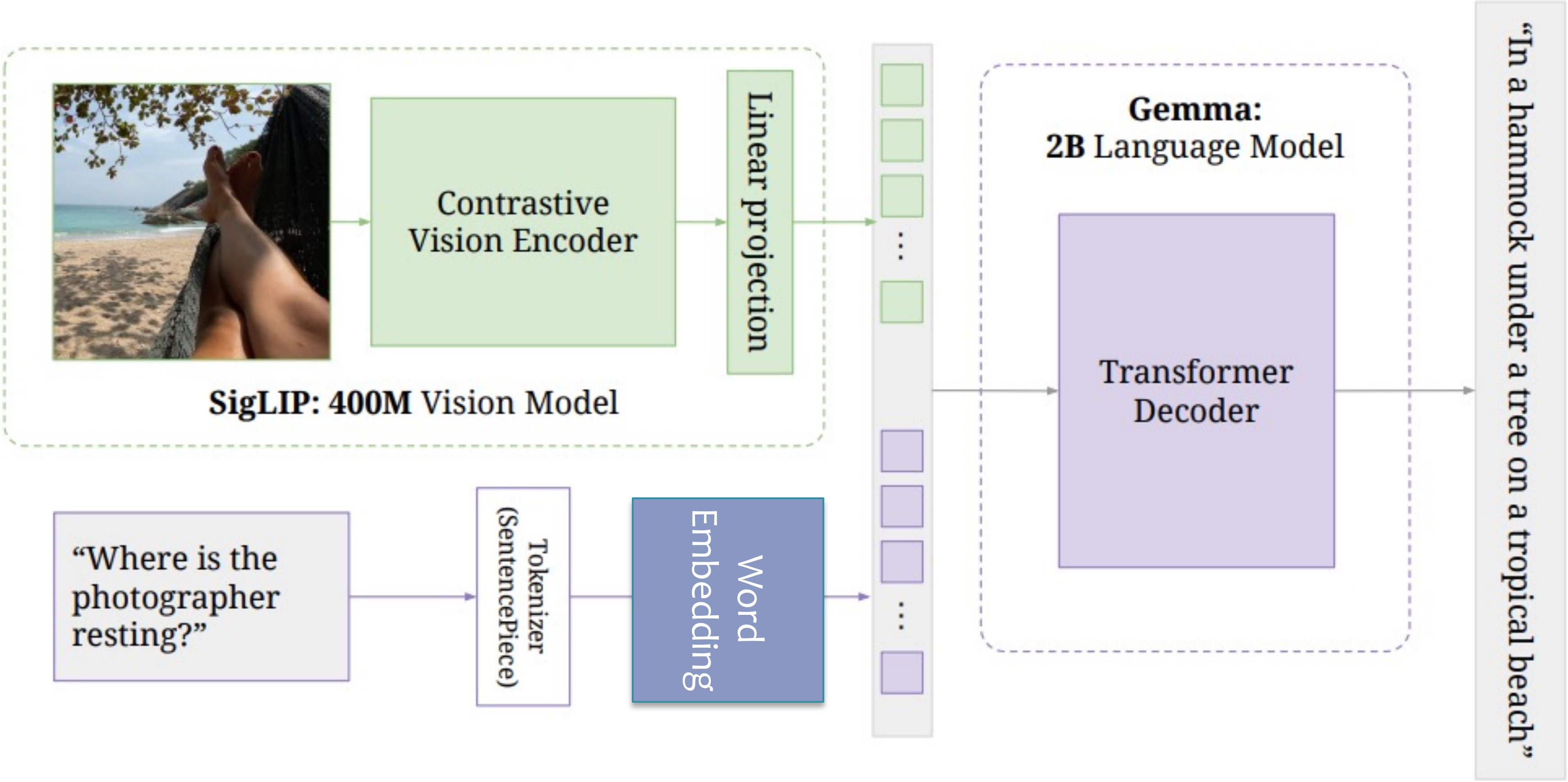


Figure from Radford et al. (2021) https://arxiv.org/pdf/2103.00020

VLMS WITH TEXT-ONLY DECODERS

PaliGemma

- SigLIP is a variant of CLIP
- Gemma is a 2B LLM (open source counterpart to Gemini)



Qwen-VL

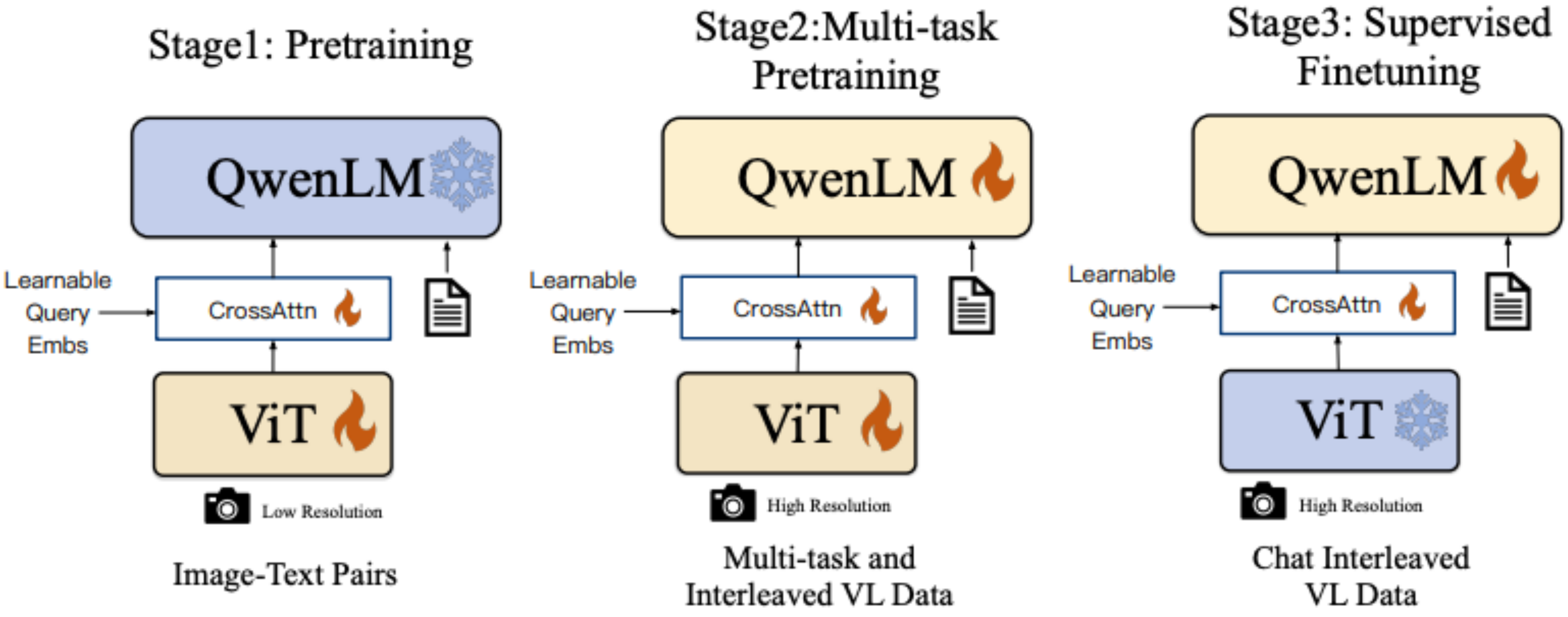
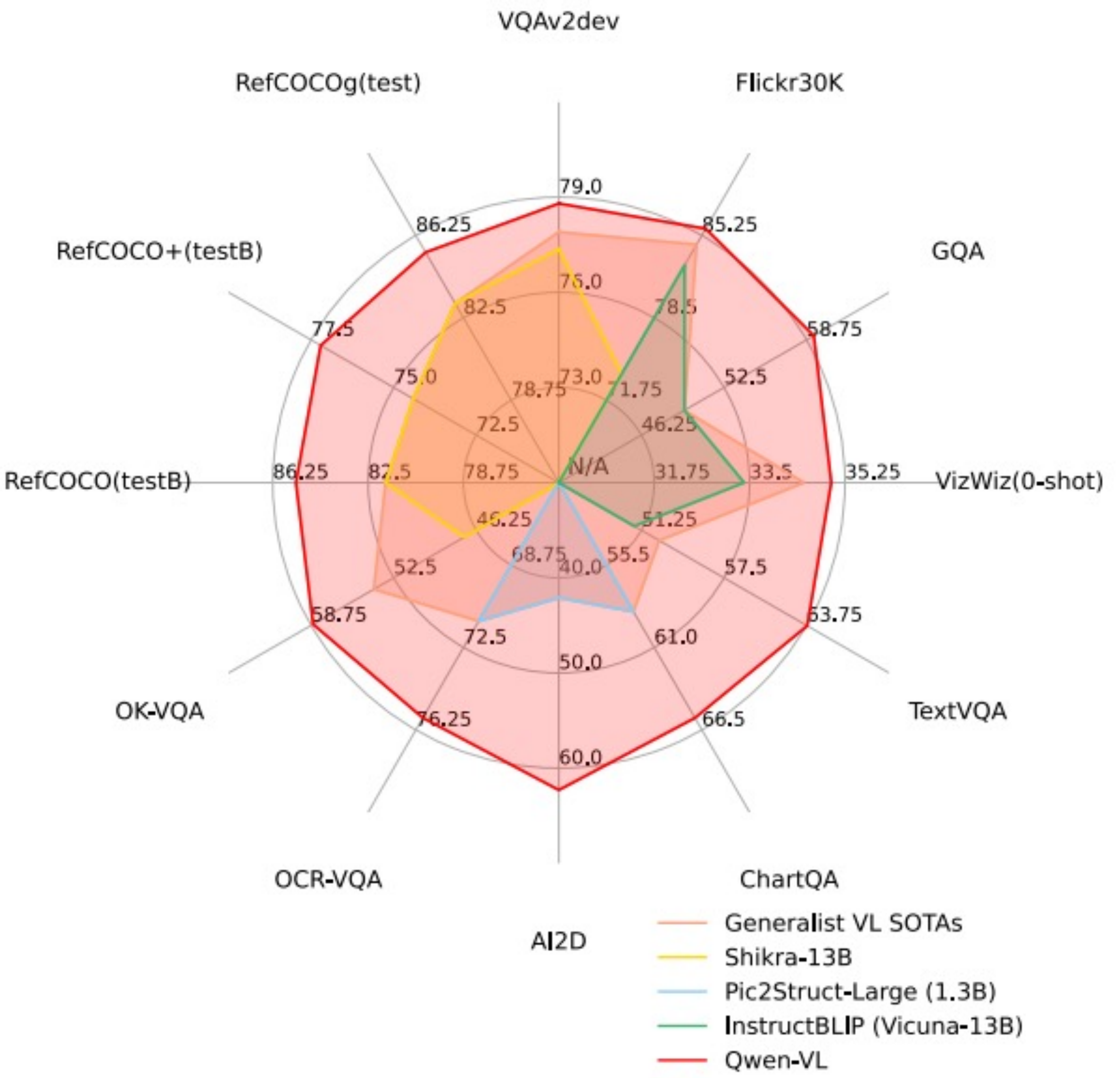


Figure 3: The training pipeline of the Qwen-VL series.

Qwen-VL



VQAv2dev
Flickr30K
GQA
VizWiz(0-shot)
TextVQA
ChartQA
AI2D
OCR-VQA
OK-VQA
RefCOCO(testB)
RefCOCO+(testB)
RefCOCOg(test)
79.0
76.0
73.0
85.25
78.5
71.75
58.75
52.5
46.25
35.25
33.5
31.75
N/A
63.75
57.5
51.25
66.5
61.0
55.5
60.0
50.0
40.0
76.25
72.5
68.75
58.75
52.5
46.25
86.25
82.5
78.75
77.5
75.0
72.5
86.25
82.5
78.75
Generalist VL SOTAs
Shikra-13B
Pic2Struct-Large (1.3B)
InstructBLIP (Vicuna-13B)
Qwen-VL

Llama 3.2 Vision

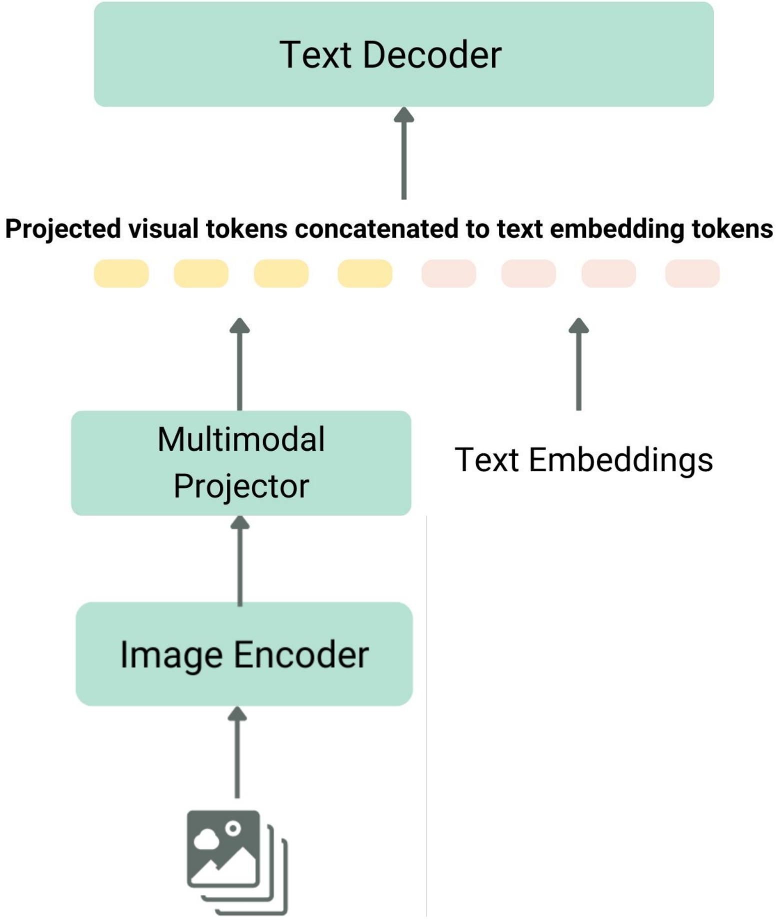
Vision instruction-tuned benchmarks

Modality	Category / Benchmark	Llama 3.2 11B	Llama 3.2 90B	Claude 3 – Haiku	GPT-4o-mini
Image	College-level Problems and Mathematical Reasoning MMMU (val, 0-shot CoT, micro avg accuracy)	50.7	**60.3**	50.2	59.4
	MMMU-Pro, Standard (10 opts, test)	33.0	**45.2**	27.3	42.3
	MMMU-Pro, Vision (test)	23.7	33.8	20.1	**36.5**
	MathVista (testmini)	51.5	**57.3**	46.4	56.7
	Charts and Diagram Understanding ChartQA (test, 0-shot CoT relaxed accuracy)*	83.4	85.5	81.7	—
	AI2 Diagram (test)*	91.1	92.3	86.7	—
	DocVQA (test, ANLS)*	88.4	90.1	88.8	—
	General Visual Question Answering VQAv2 (test)	75.2	78.1	—	—
Text	General MMLU (0-shot, CoT)	73.0	**86.0**	75.2 (5-shot)	82.0
	Math MATH (0-shot, CoT)	51.9	68.0	38.9	**70.2**
	Reasoning GPQA (0-shot, CoT)	32.8	**46.7**	33.3	40.2
	Multilingual MGSM (0-shot, CoT)	68.9	86.9	75.1	**87.0**

VISION LANGUAGE MODELS (VLMS)

VLM: Architecture

- High-level idea: convert both the image and the text inputs into embedding vectors, then pass those vectors into a decoder-only transformer and do next (text) token prediction

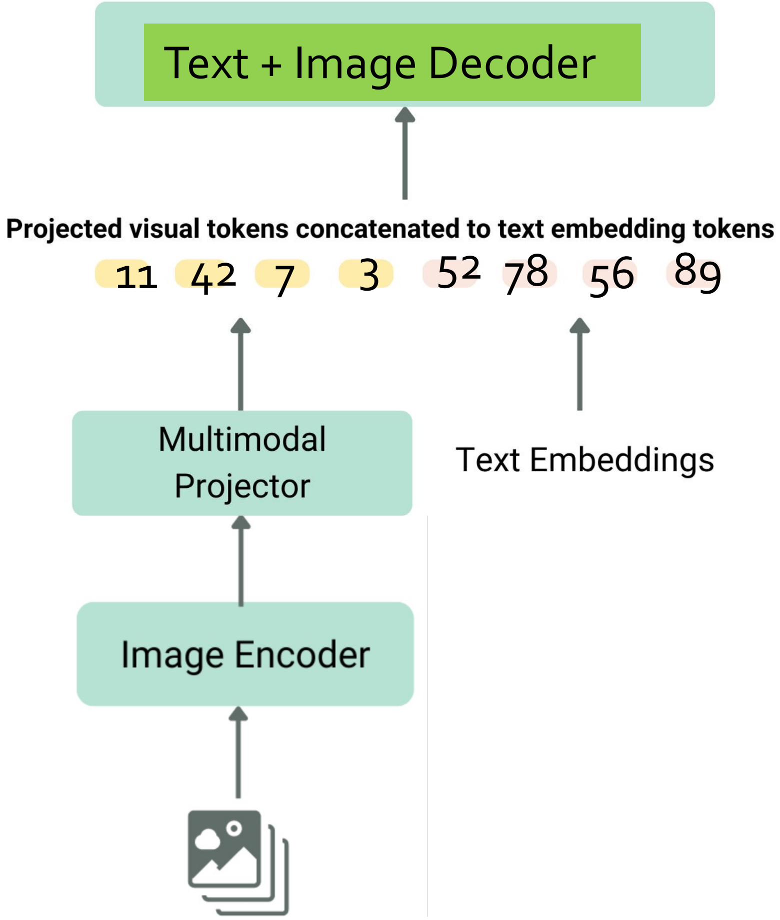


- Two common encoders:
  - VQ-VAE encoder followed by an embedding layer that converts the discrete tokens into dense numerical vectors
  - CLIP encoder, that directly learns an embedding vector using a contrastive pre-training objective

Source: https://huggingface.co/blog/vlms

VLM: Architecture

- High-level idea: convert both the image and the text inputs into **integers**, then pass those **integers** into a decoder-only transformer and do next (**text or image**) token prediction



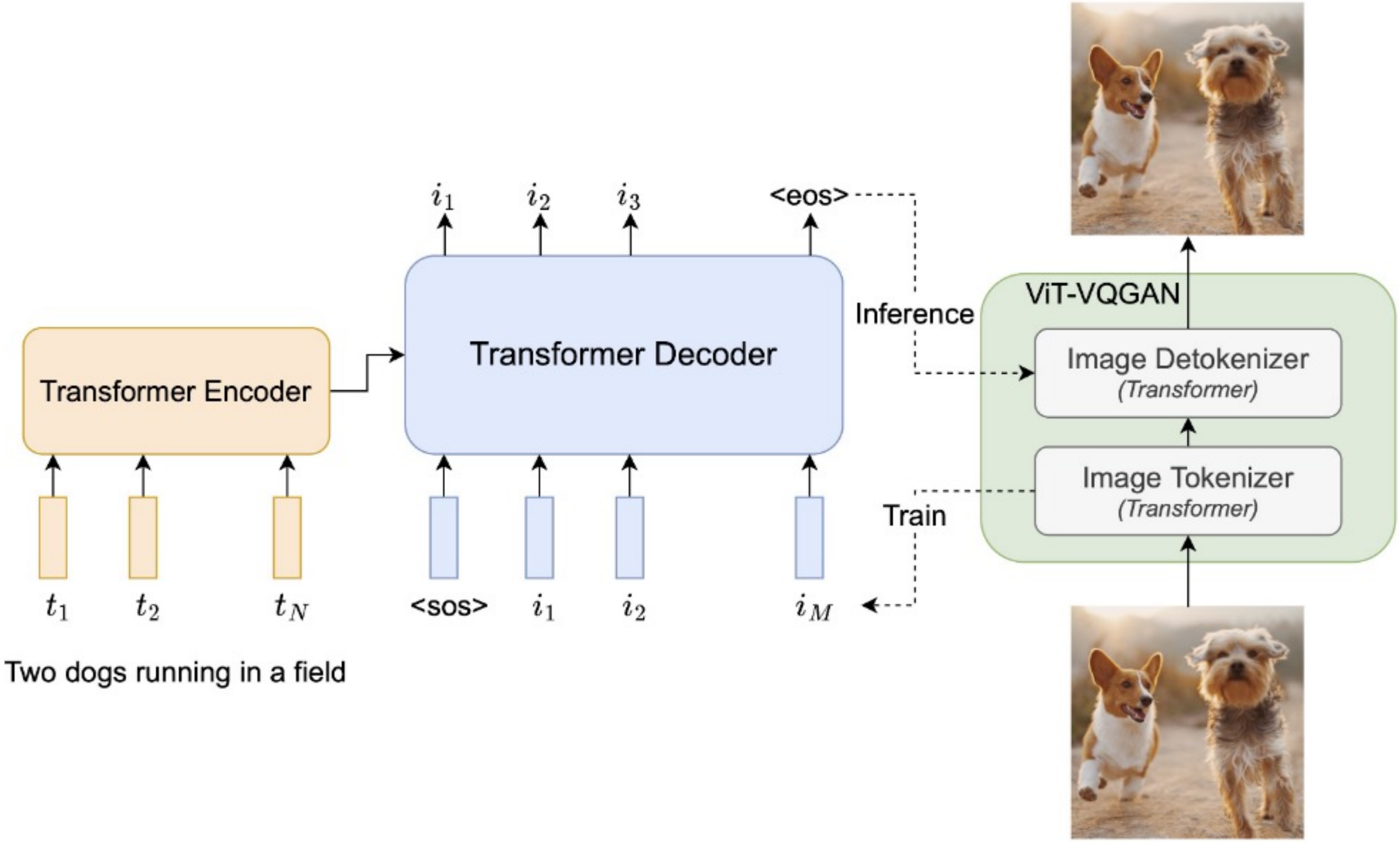
- Two common encoders:
  - VQ-VAE encoder followed by an embedding layer that converts the discrete tokens into dense numerical vectors
  - CLIP encoder, that directly learns an embedding vector using a contrastive pre-training objective

Source: https://huggingface.co/blog/vlms

Why VLMs with Integer Tokens?

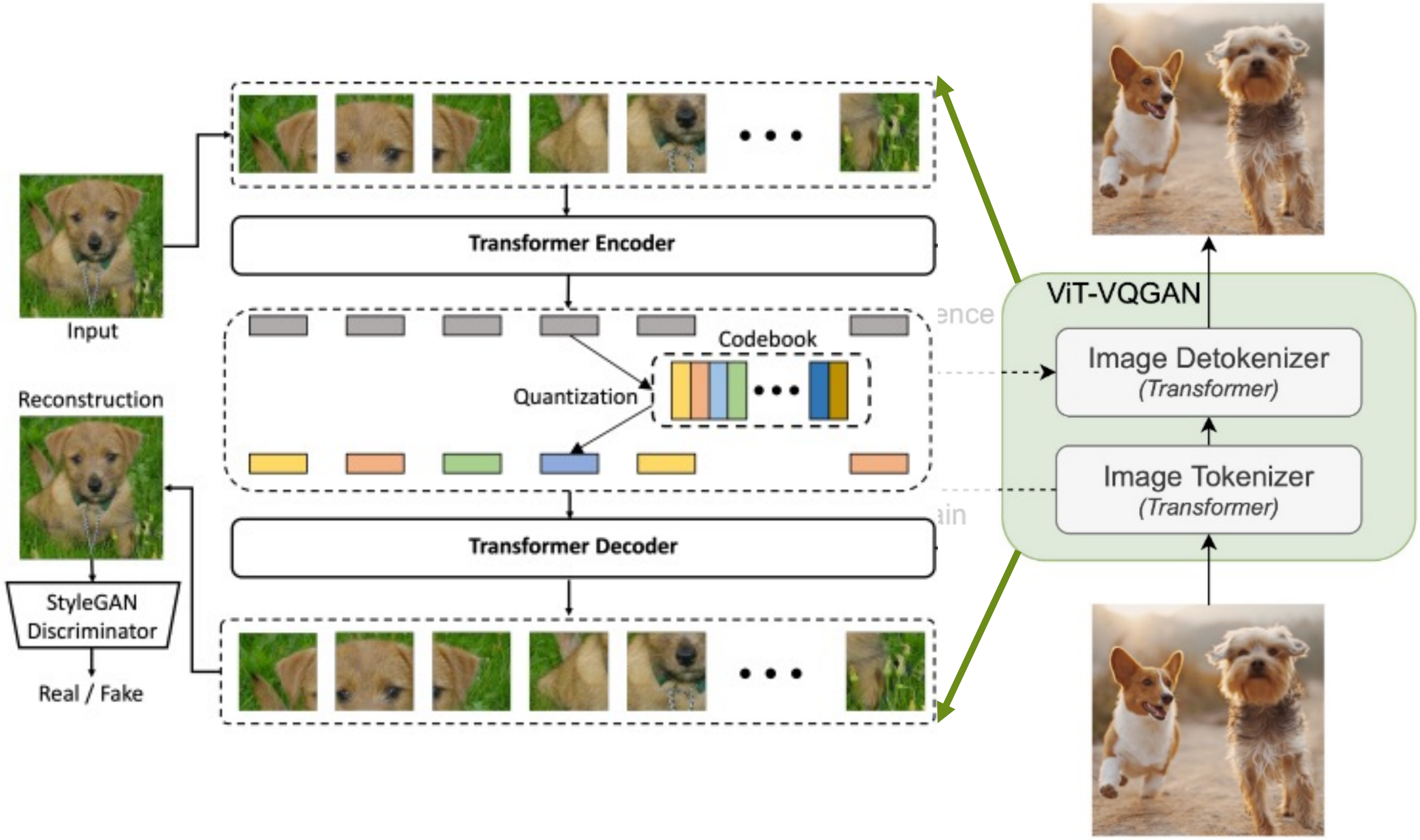
VQ-VAES

Recall: Parti



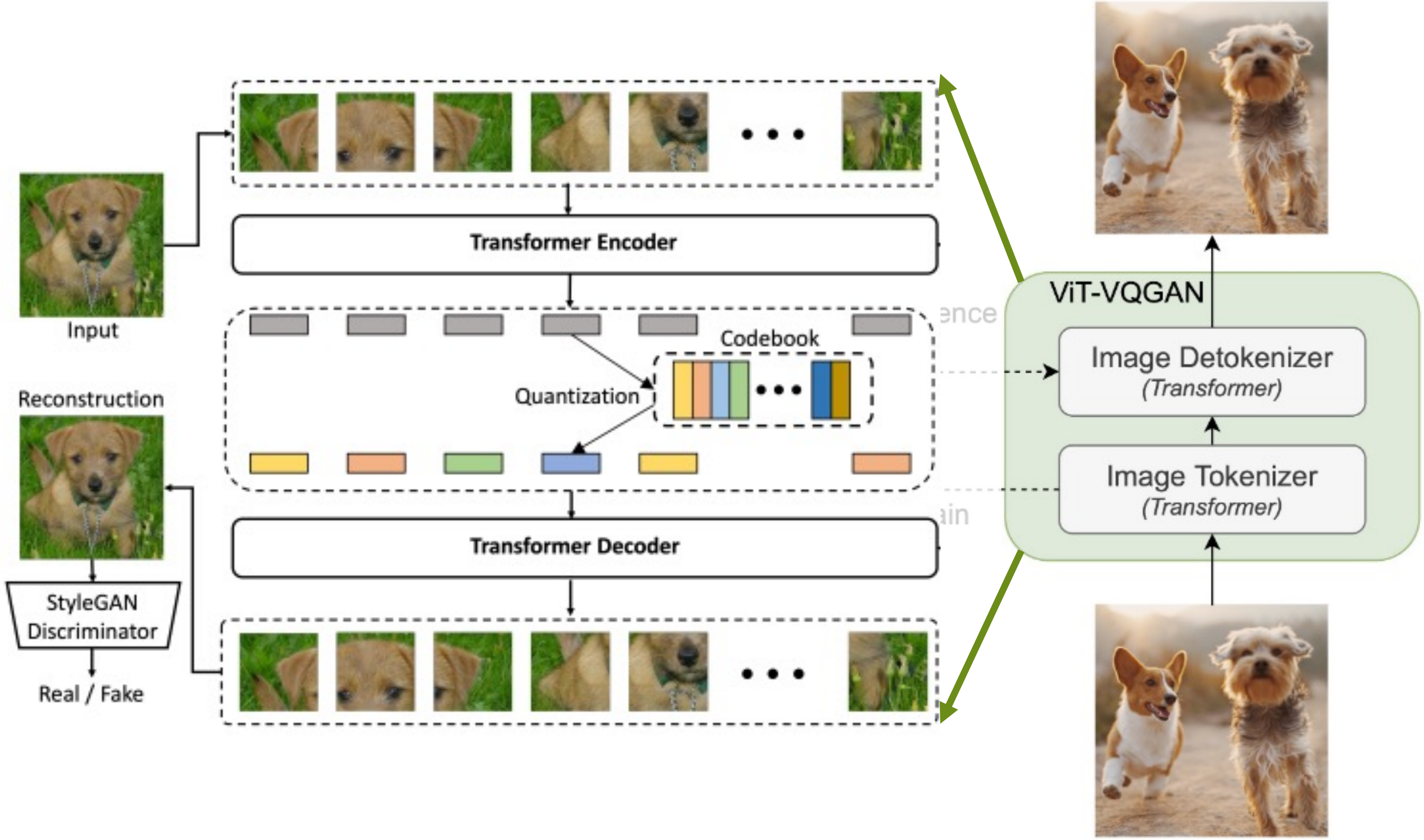
Source: https://arxiv.org/pdf/2206.10789

Recall: Image Tokenization

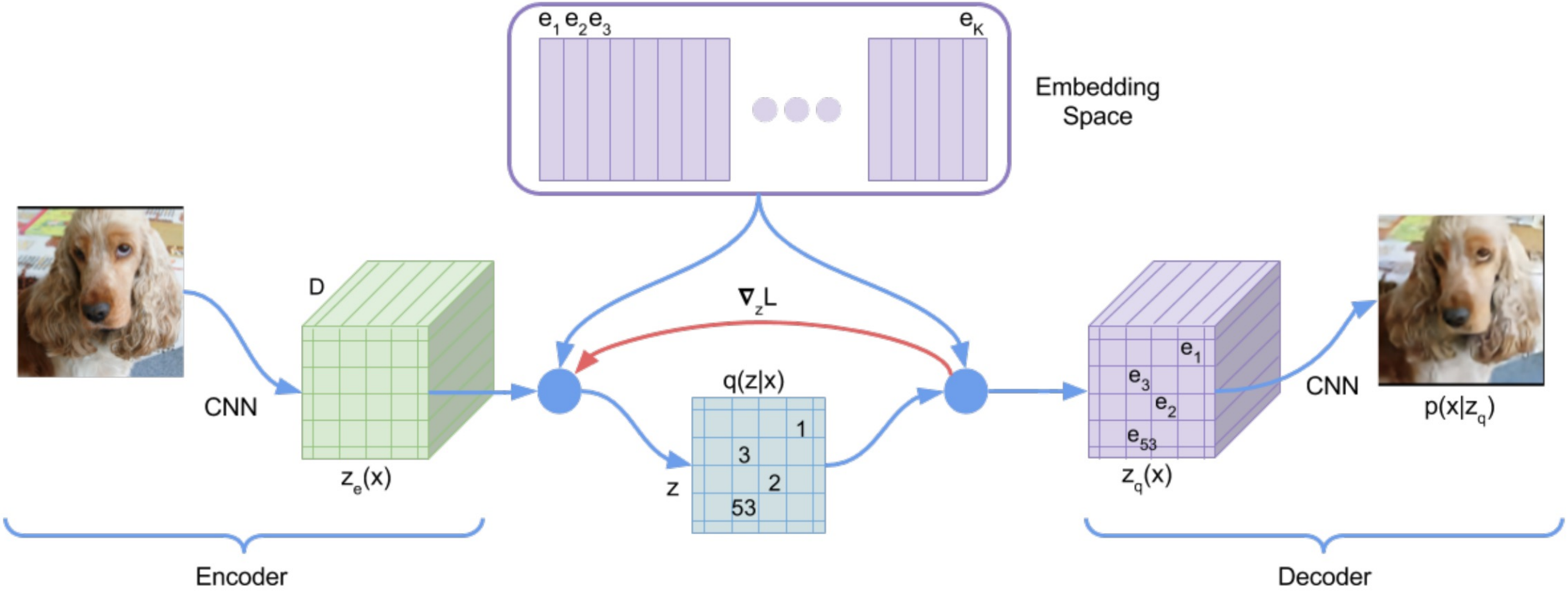


Source: https://arxiv.org/pdf/2110.04627

How can we (pre-)train these models given the non-differentiable quantization operation?

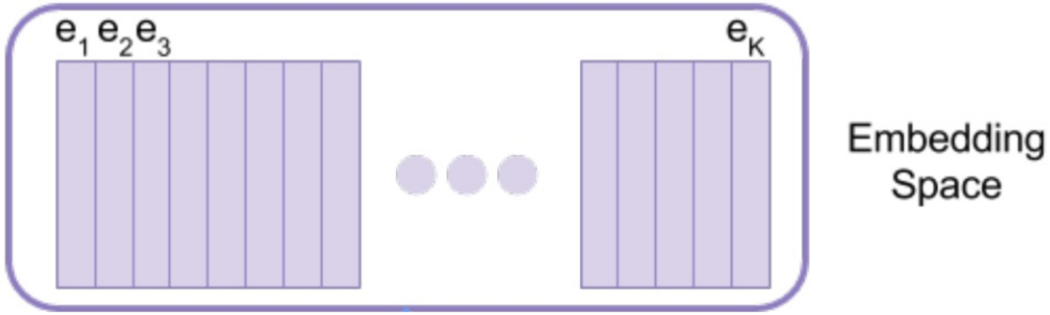


Source: https://arxiv.org/pdf/2110.04627



Vector-Quantized VAEs

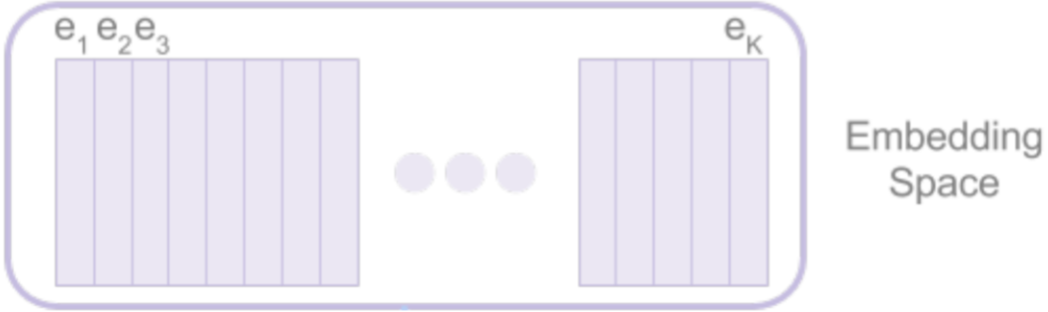
Source: https://arxiv.org/pdf/1711.00937

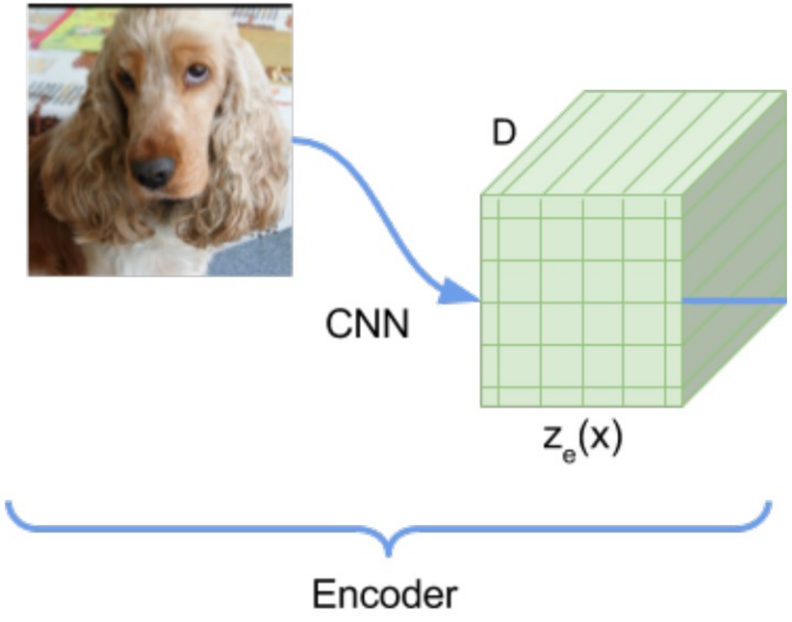


- Embedding space consists of K D-dimensional latent vectors $\{e_1, \ldots, e_K\}$ which are learned during training
- The indices $[1, \ldots, K]$ of each latent vector correspond to the "image tokens" in some fixed-length codebook

Vector-Quantized VAEs

Source: https://arxiv.org/pdf/1711.00937

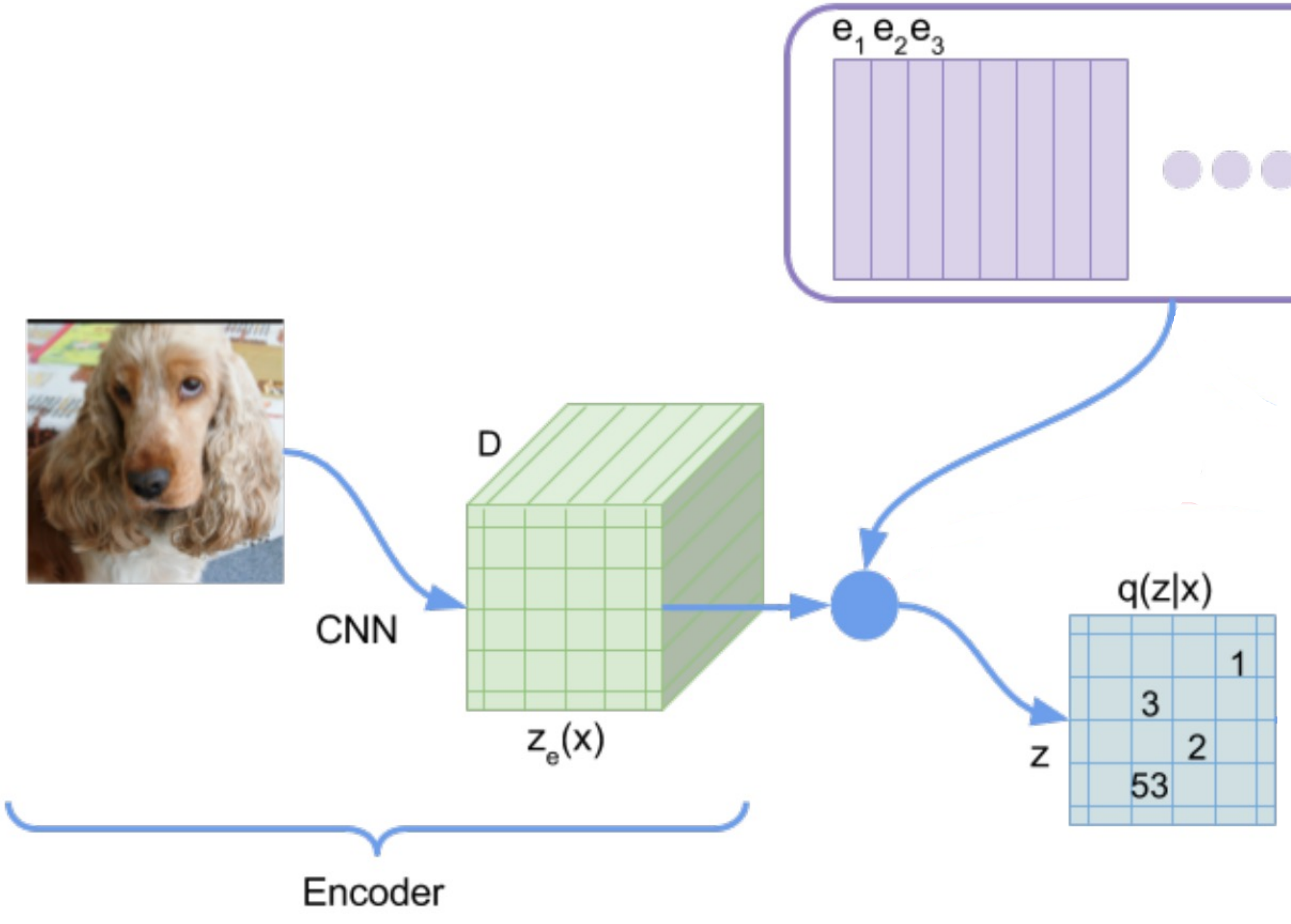




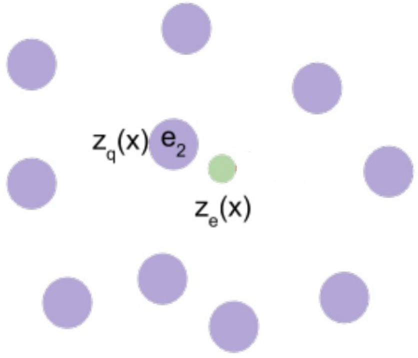
- The encoder (e.g., a ResNet-like CNN) maps images to N D-dimensional vectors

Vector-Quantized VAEs

Source: https://arxiv.org/pdf/1711.00937



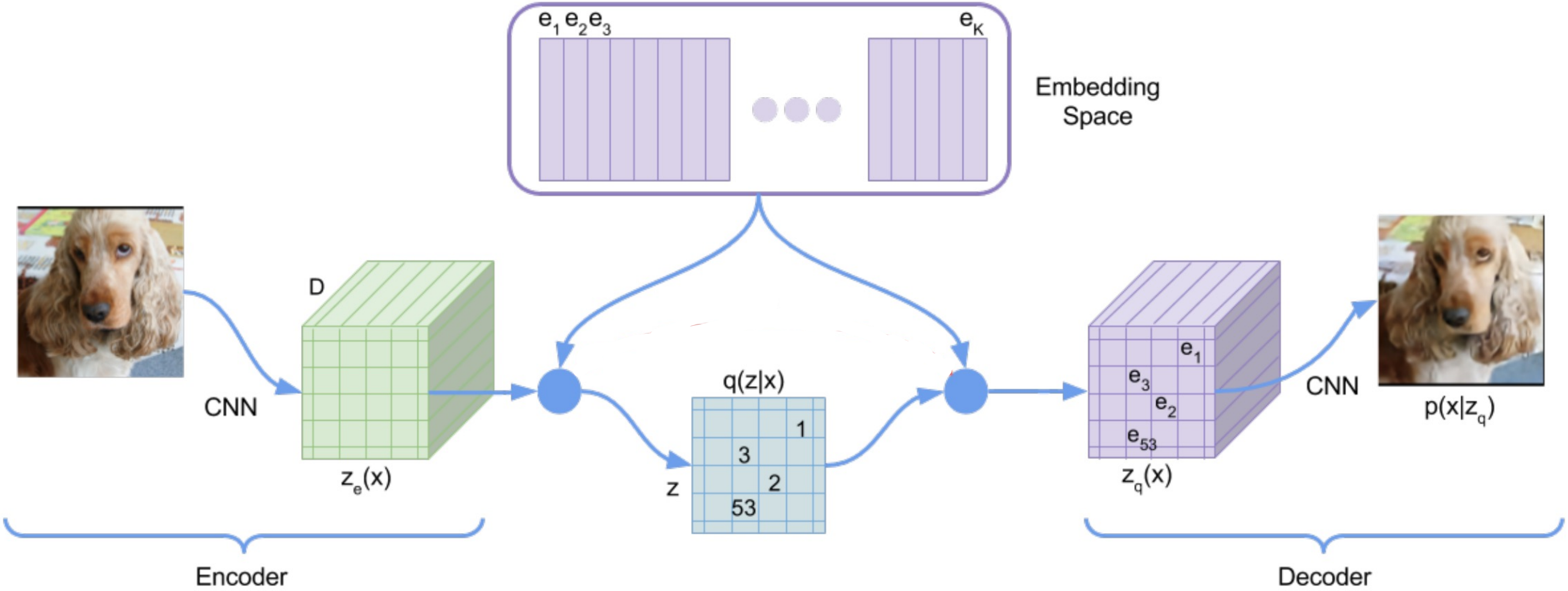
- Each output vector from the encoder is mapped to the nearest latent vector to get the discretized encoding



$$z_q(x) = \underset{e \in \{e_1,\dots,e_K\}}{\operatorname{argmin}} \|z_e(x) - e\|_2^2$$

Vector-Quantized VAEs

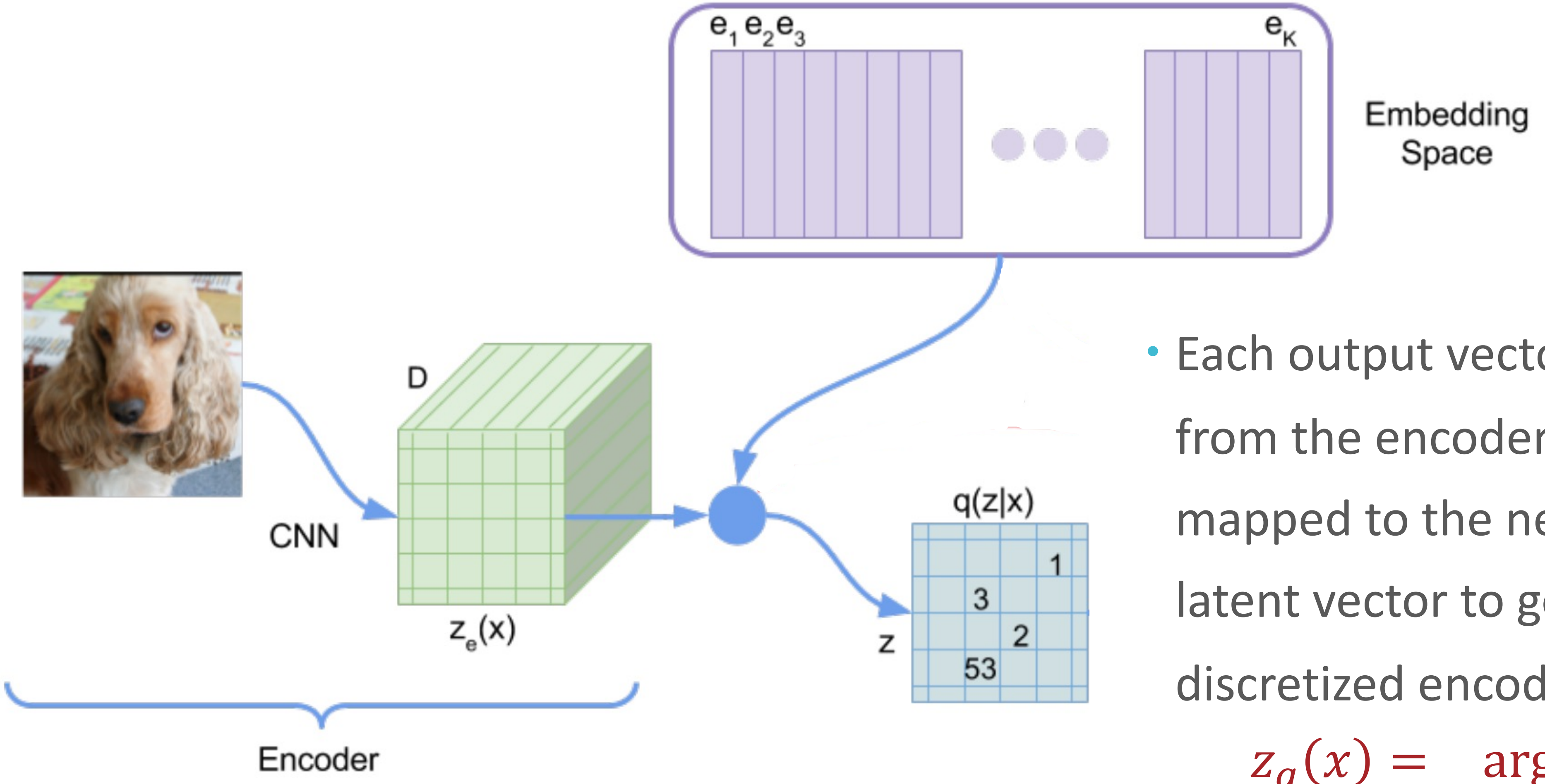
Source: https://arxiv.org/pdf/1711.00937



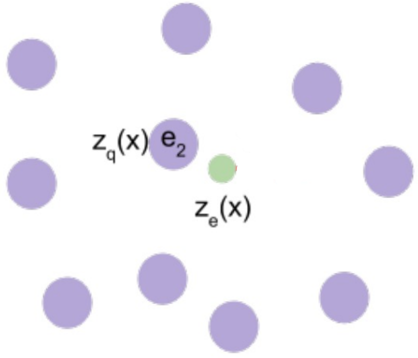
- The decoder takes the discretized representation and recreates the original image

Vector-Quantized VAEs

Source: https://arxiv.org/pdf/1711.00937



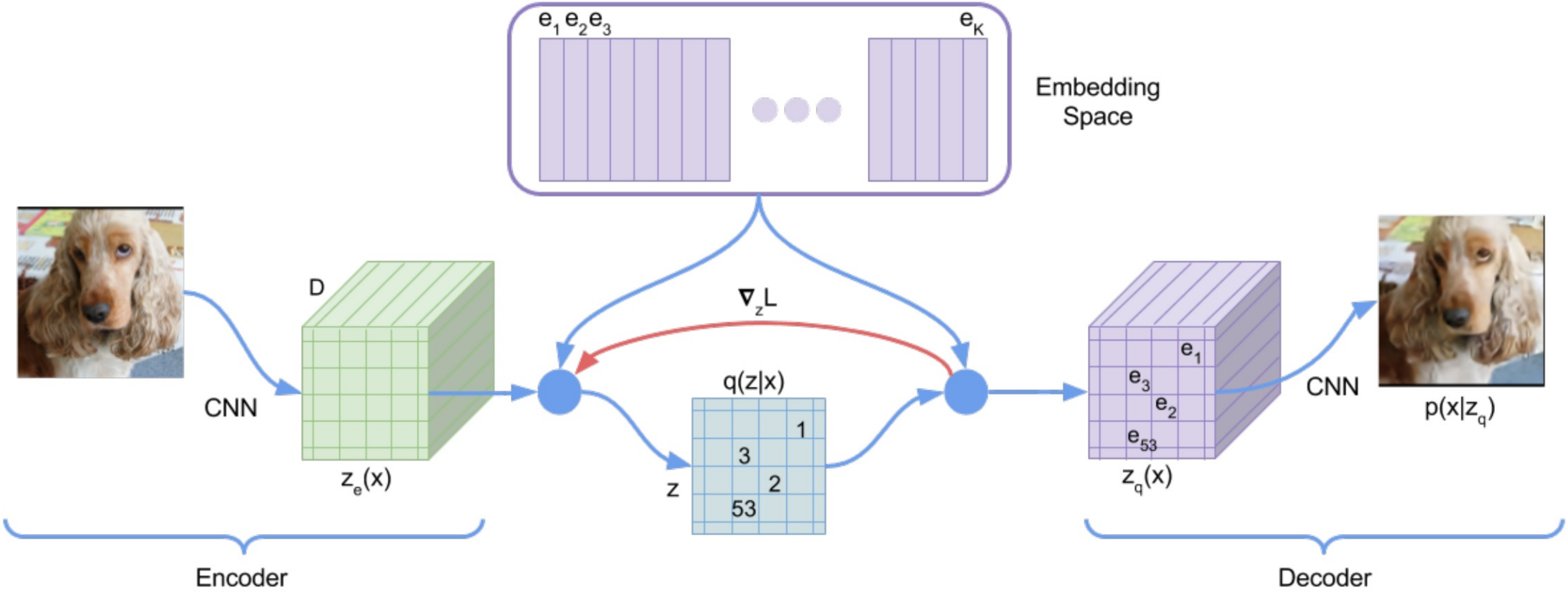
- Each output vector from the encoder is mapped to the nearest latent vector to get the discretized encoding



$$z_q(x) = \underset{e\,\in\{e_1,\ldots,e_K\}}{\operatorname{argmin}} \|z_e(x) - e\|_2^2$$

Wait, how would we take the gradient through the argmin?

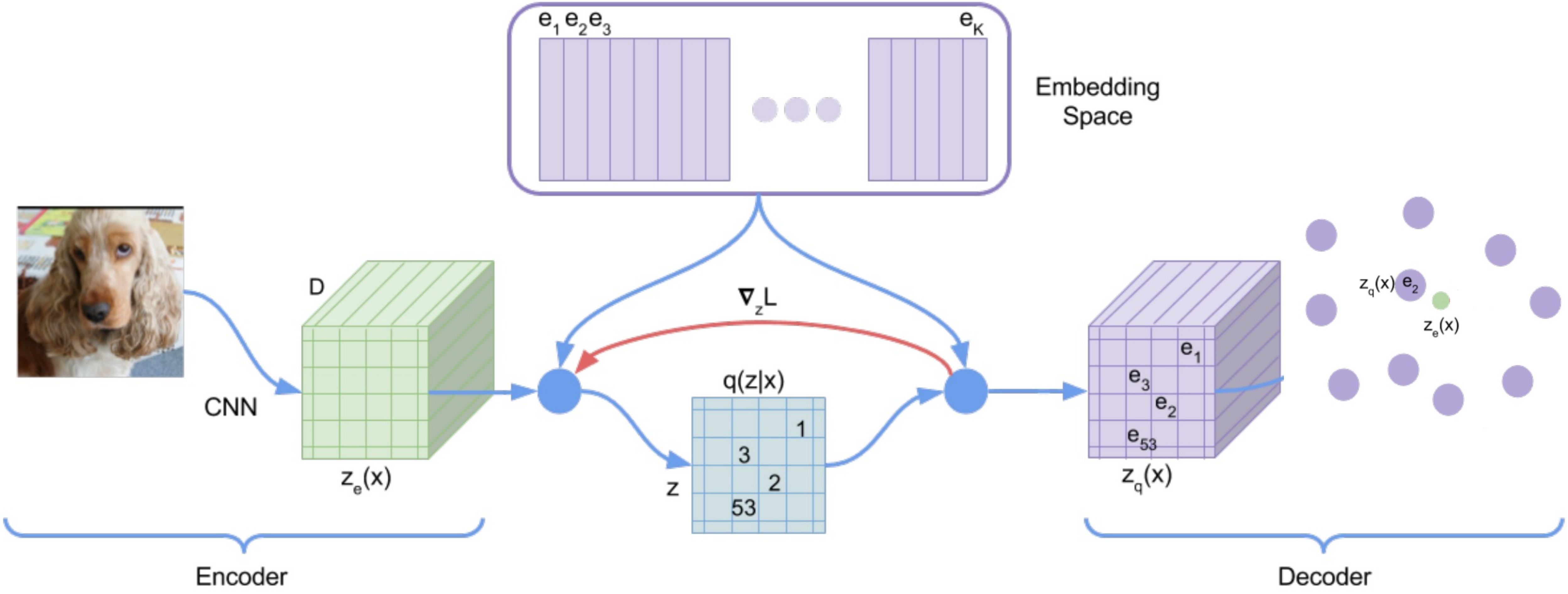
Source: https://arxiv.org/pdf/1711.00937



- Treat the gradient w.r.t. $z_q(x)$ as an estimate of the gradient w.r.t. $z_e(x)$

Straight-through Estimator

Source: https://arxiv.org/pdf/1711.00937



- Intuition: the closer $z_q(x)$ and $z_e(x)$, the better the estimate (under certain assumptions)

Straight-through Estimator

Source: https://arxiv.org/pdf/1711.00937

VQ-VAE Objective Function

- Intuition: we want the latent vectors to correspond to relevant points in the embedding space i.e., ones that are near the outputs of the encoder
- However, we also want the encoder to respect the latent vectors and not overfit to the training dataset
- Idea: augment the standard VAE objective with some regularizing terms that drive the two closer to each other

$$-\log p_\theta\left(x \middle| z_q(x)\right) + \quad \left\|\mathrm{sg}[z_e(x)] - z_q(x)\right\|_2^2$$

$$+ \beta \left\|z_e(x) - \mathrm{sg}\left[z_q(x)\right]\right\|_2^2$$

where sg is the stop-gradient operator which fixes the argument to be non-updated constant

VQ-VAE Objective Function

- Intuition: we want the latent vectors to correspond to relevant points in the embedding space i.e., ones that are near the outputs of the encoder
- However, we also want the encoder to respect the latent vectors and not overfit to the training dataset
- Idea: augment the standard VAE objective with some regularizing terms that drive the two closer to each other

$$-\log p_\theta(x|z_q(x)) + \left\|\mathrm{sg}[z_e(x)] - z_q(x)\right\|_2^2$$
$$+ \beta\left\|z_e(x) - \mathrm{sg}[z_q(x)]\right\|_2^2$$

- The first term is the typical reconstruction error objective

VQ-VAE Objective Function

- Intuition: we want the latent vectors to correspond to relevant points in the embedding space i.e., ones that are near the outputs of the encoder
- However, we also want the encoder to respect the latent vectors and not overfit to the training dataset
- Idea: augment the standard VAE objective with some regularizing terms that drive the two closer to each other

$$-\log p_\theta\left(x \middle| z_q(x)\right) + \left\| \mathrm{sg}[z_e(x)] - z_q(x) \right\|_2^2 + \beta \left\| z_e(x) - \mathrm{sg}\left[z_q(x)\right] \right\|_2^2$$

- The second term drives the latent vector to be closer to the encoder output vector that was mapped to it

VQ-VAE Objective Function

- Intuition: we want the latent vectors to correspond to relevant points in the embedding space i.e., ones that are near the outputs of the encoder
- However, we also want the encoder to respect the latent vectors and not overfit to the training dataset
- Idea: augment the standard VAE objective with some regularizing terms that drive the two closer to each other

$$-\log p_\theta\left(x \middle| z_q(x)\right) + \quad \left\| \mathrm{sg}[z_e(x)] - z_q(x) \right\|_2^2$$
$$+ \beta \left\| z_e(x) - \mathrm{sg}\left[z_q(x)\right] \right\|_2^2$$

- The third term drives the encoder to output vectors closer to the latent vectors

CLIP vs. VQ-VAEs

- VLMs with VQ-VAE encoders (or any vector quantized image model) can also generate images in addition to text by defining a loss over the image codebook tokens
- CLIP does not discretize its image embedding so VLMs with CLIP-based encoders cannot (naturally) define a loss over images and thus, can only output text
- However, CLIP embeddings are more expressive than the discrete VQ-VAE encodings so can lead to improved performance in some settings

VLMS WITH TEXT AND IMAGE DECODERS

Large World Model

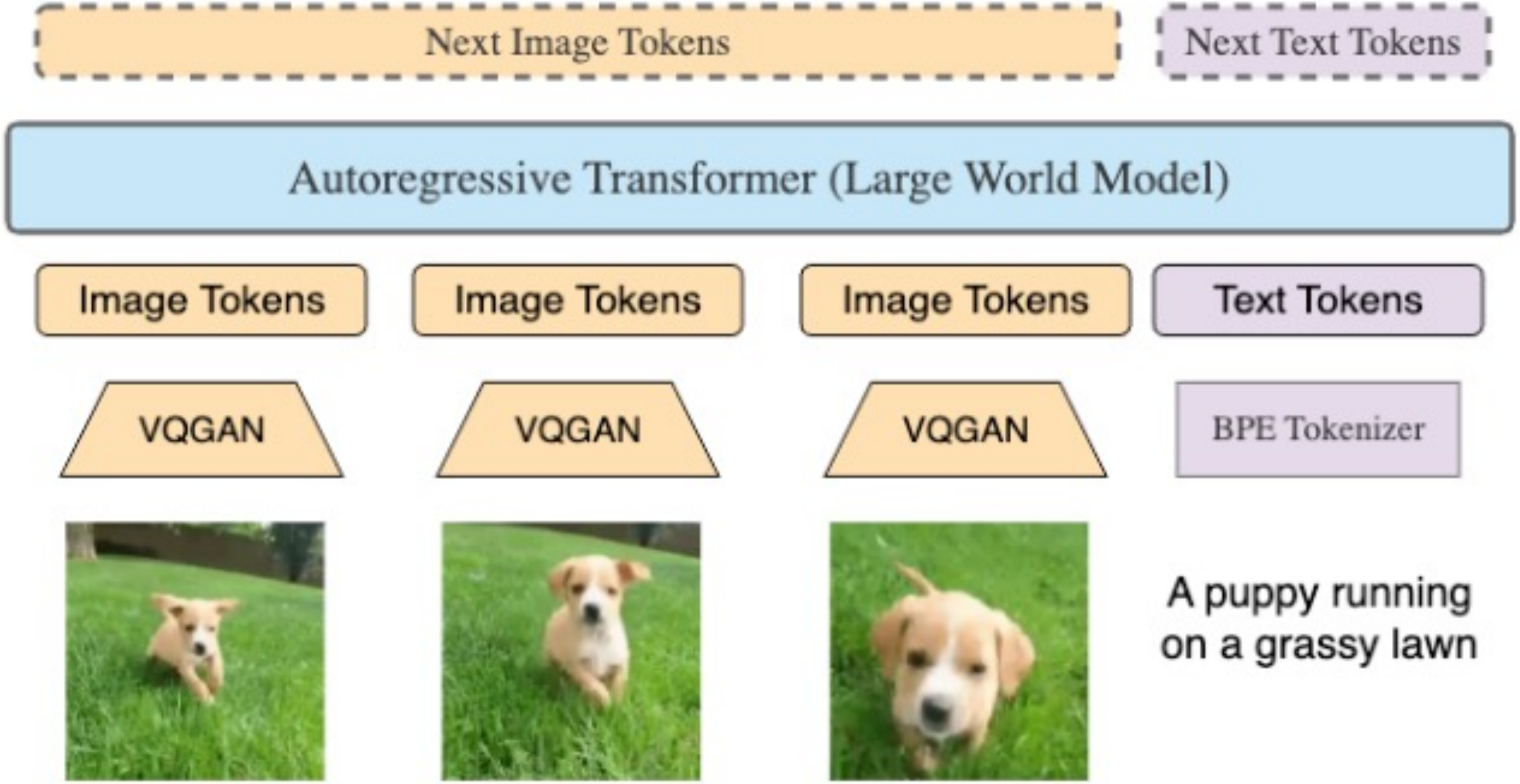


Figure from http://arxiv.org/abs/2402.08268

Large World Model

H More Image Generation Examples

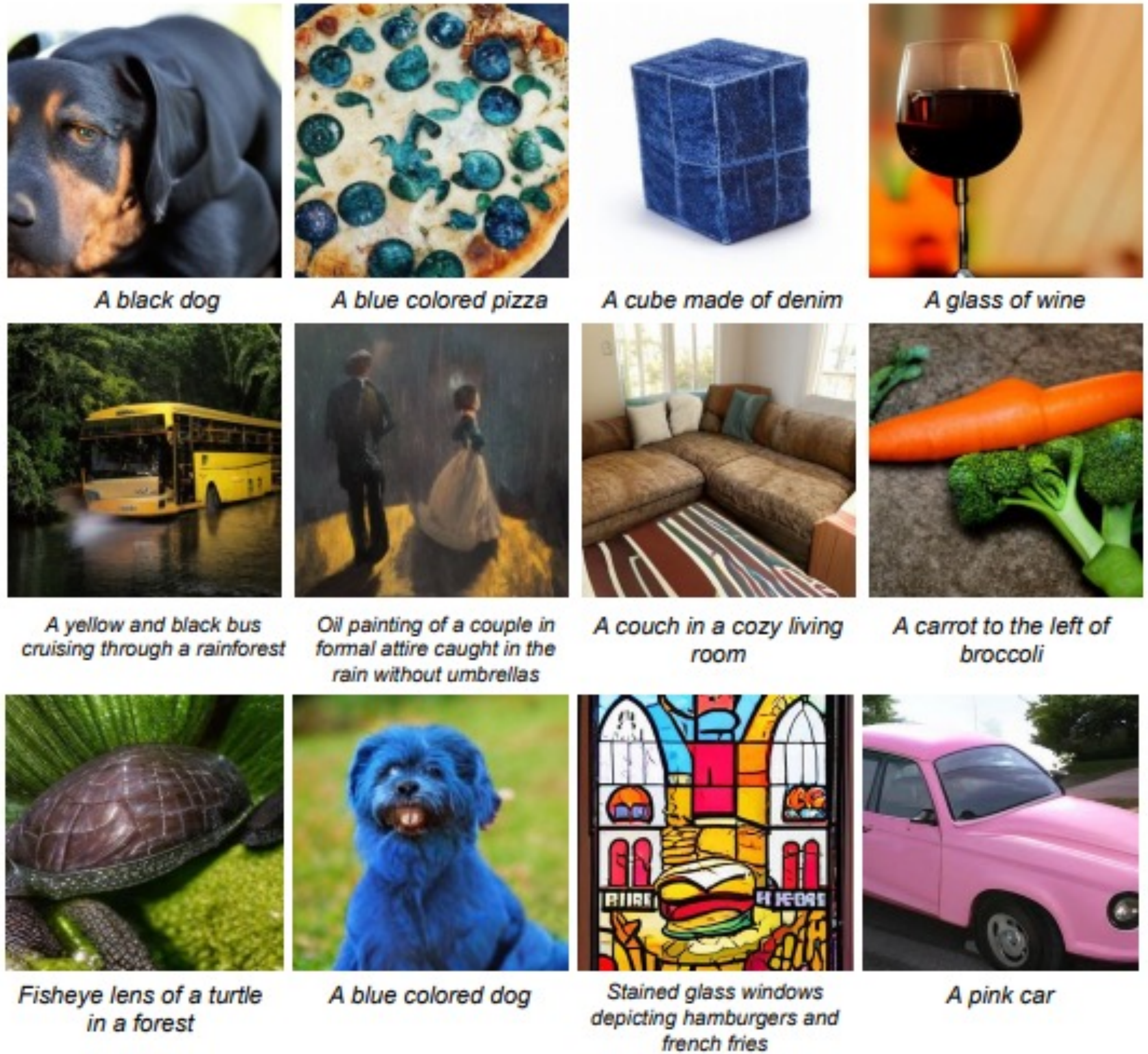
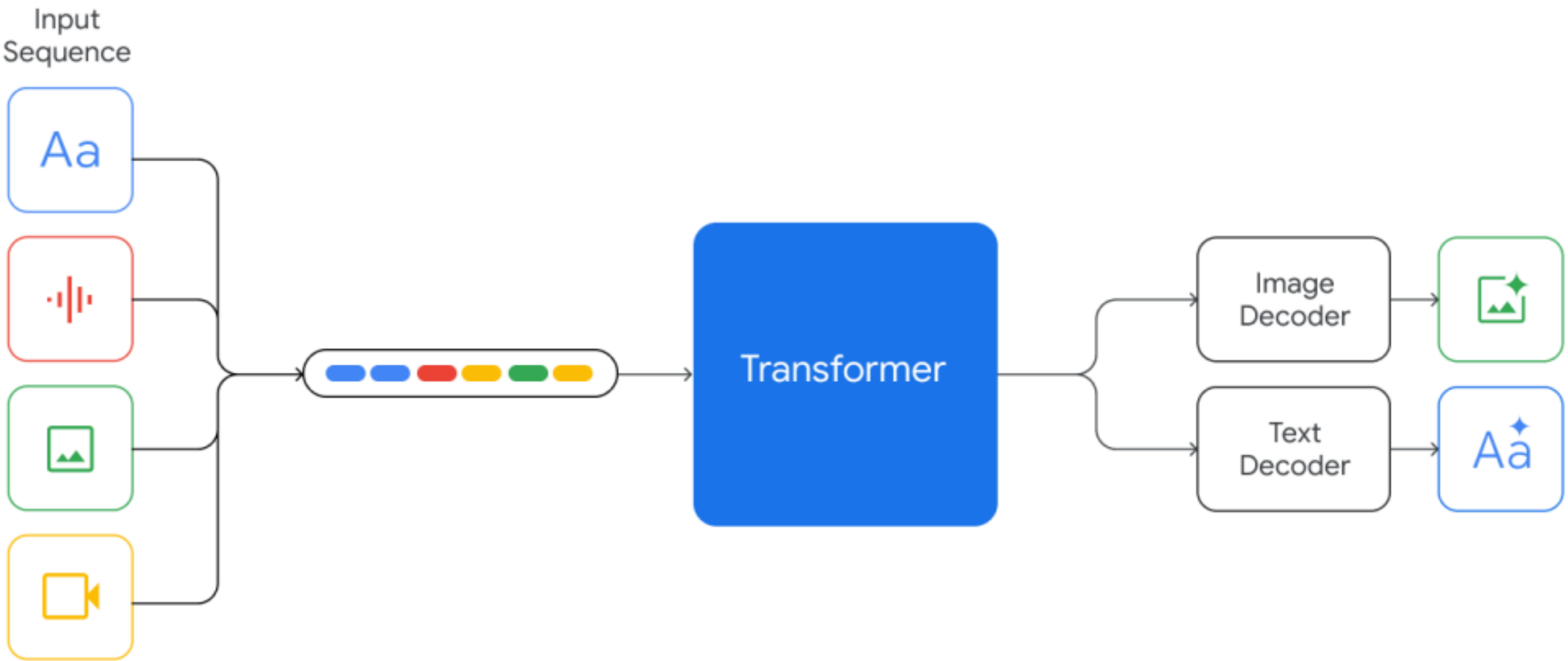


Figure from http://arxiv.org/abs/2402.08268

Gemini


Input
Sequence
Aa
Transformer
Image
Decoder
Text
Decoder
Aa

Gemini

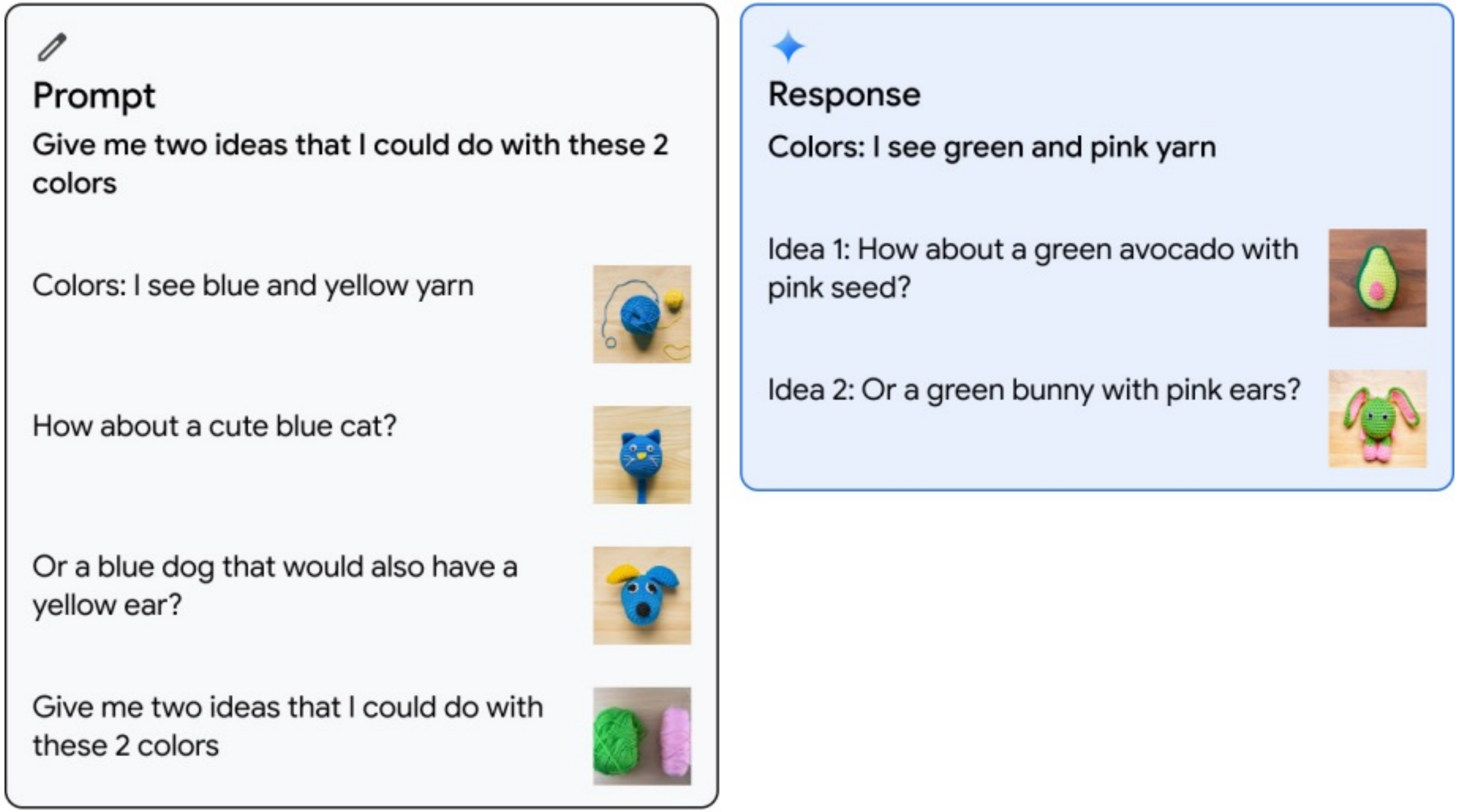


Figure 6 | **Image Generation.** Gemini models can output multiple images interleaved with text given a prompt composed of image and text. In the left figure, Gemini Ultra is prompted in a 1-shot setting with a user example of generating suggestions of creating cat and dog from yarn when given two colors, blue and yellow. Then, the model is prompted to generate creative suggestions with two new colors, pink and green, and it generates images of creative suggestions to make a cute green avocado with pink seed or a green bunny with pink ears from yarn as shown in the right figure.