Lecture 4

CME 296: Diffusion & Large Vision Models

Afshine Amidi & Shervine Amidi

Stanford University

Recap of last episodes...

Lecture 1: Diffusion, DDPM	$x_t = \sqrt{\overline{\alpha}_t} x_0 + \sqrt{1 - \overline{\alpha}_t} \epsilon$ $x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1 - \alpha_t}{\sqrt{1 - \overline{\alpha}_t}} \epsilon_\theta(x_t, t) \right) + \sigma_t \epsilon$ $\epsilon_\theta(x_t, t)$

Recap of last episodes...

Lecture 1: Diffusion, DDPM	$x_t = \sqrt{\overline{\alpha}_t} x_0 + \sqrt{1 - \overline{\alpha}_t}\epsilon$ $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(x_t, t)\right) + \sigma_t \epsilon$	$\epsilon_\theta(x_t, t)$
Lecture 2: Score matching, SDE	$dx = f(x_t, t)dt + g(t)dW$ $dx = \left[f(x_t, t) - g(t)^2 s_\theta(x_t, t)\right]dt + g(t)d\overline{W}$	$s_\theta(x_t, t)$

Recap of last episodes...

Lecture 1: Diffusion, DDPM	$x_t = \sqrt{\overline{\alpha}_t} x_0 + \sqrt{1 - \overline{\alpha}_t}\epsilon$ $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(x_t, t)\right) + \sigma_t \epsilon$	$\epsilon_\theta(x_t, t)$
Lecture 2: Score matching, SDE	$dx = f(x_t, t)dt + g(t)dW$ $dx = \left[f(x_t, t) - g(t)^2 s_\theta(x_t, t)\right]dt + g(t)d\overline{W}$	$s_\theta(x_t, t)$
Lecture 3: Flow matching	$x_t = (1-t)x_0 + t x_1$ $x_{t+\Delta t} = x_t + u_\theta(x_t, t)\Delta t$	$u_\theta(x_t, t)$

Recap of last episodes...

Lecture 1: Diffusion, DDPM	$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon$ $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)\right) + \sigma_t \epsilon$	$\epsilon_\theta(x_t, t)$
Lecture 2: Score matching, SDE	$dx = f(x_t, t)dt + g(t)dW$ $dx = \left[f(x_t, t) - g(t)^2 s_\theta(x_t, t)\right]dt + g(t)d\overline{W}$	$s_\theta(x_t, t)$
Lecture 3: Flow matching	$x_t = (1-t)x_0 + t x_1$ $x_{t+\Delta t} = x_t + u_\theta(x_t, t)\Delta t$	$u_\theta(x_t, t)$

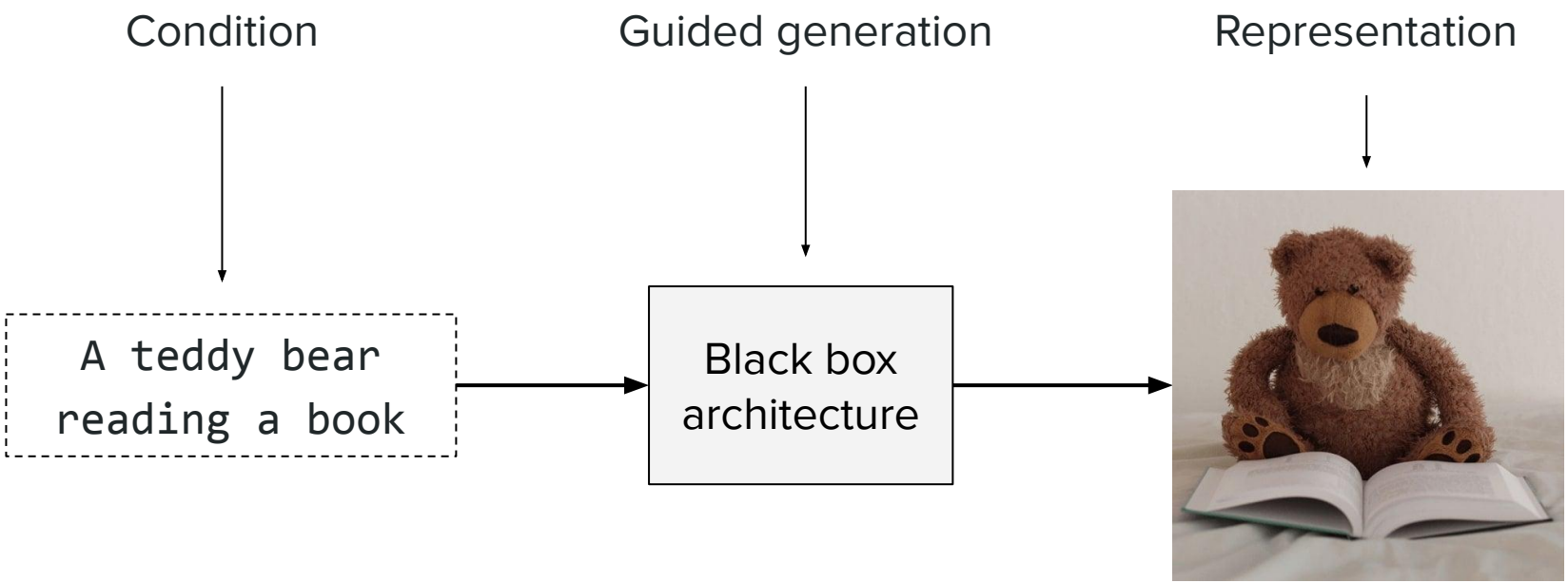
Recap of last episodes...

Lecture 1: Diffusion, DDPM	$x_t = \sqrt{\overline{\alpha}_t} x_0 + \sqrt{1 - \overline{\alpha}_t}\epsilon$ $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(x_t, t)\right) + \sigma_t \epsilon$	$\epsilon_\theta(x_t, t)$
Lecture 2: Score matching, SDE	$dx = f(x_t, t)dt + g(t)dW$ $dx = \left[f(x_t, t) - g(t)^2 s_\theta(x_t, t)\right]dt + g(t)d\overline{W}$	$s_\theta(x_t, t)$
Lecture 3: Flow matching	$x_t = (1-t)x_0 + t x_1$ $x_{t+\Delta t} = x_t + u_\theta(x_t, t)\Delta t$	$u_\theta(x_t, t)$

Recap of last episodes...

Lecture 1: Diffusion, DDPM	$x_t = \sqrt{\overline{\alpha}_t} x_0 + \sqrt{1 - \overline{\alpha}_t}\epsilon$ $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(x_t, t)\right) + \sigma_t \epsilon$	$\epsilon_\theta(x_t, t, \varnothing)$
Lecture 2: Score matching, SDE	$dx = f(x_t, t)dt + g(t)dW$ $dx = \left[f(x_t, t) - g(t)^2 s_\theta(x_t, t)\right]dt + g(t)d\overline{W}$	$s_\theta(x_t, t, \varnothing)$
Lecture 3: Flow matching	$x_t = (1-t)x_0 + t x_1$ $x_{t+\Delta t} = x_t + u_\theta(x_t, t)\Delta t$	$u_\theta(x_t, t, \varnothing)$

Today's lecture



Today's lecture: **Multimodal guided generation**

Today's lecture: Part 1

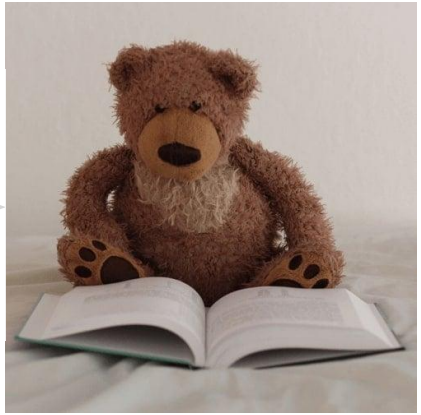
Condition

Guided generation

1 **Representation**

A teddy bear reading a book

Black box architecture



Today's lecture: **Multimodal guided generation**

Diffusion & Large Vision Models

From pixel to latent space

Latent diffusion models

Text representation

Image representation

Contrastive learning

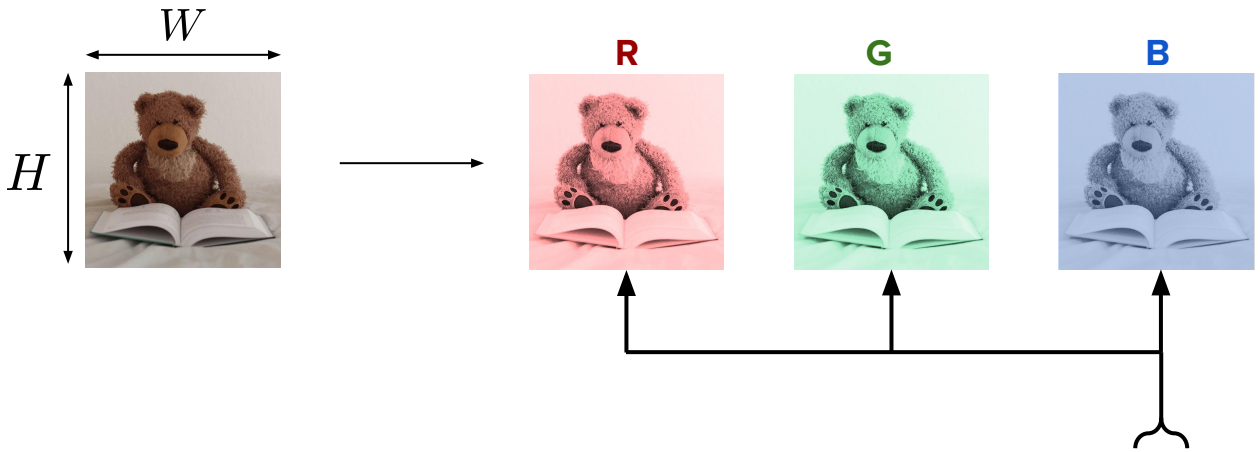
Guidance

Stanford University

ICME

Pixel space

Naive approach. Represent everything in pixel space

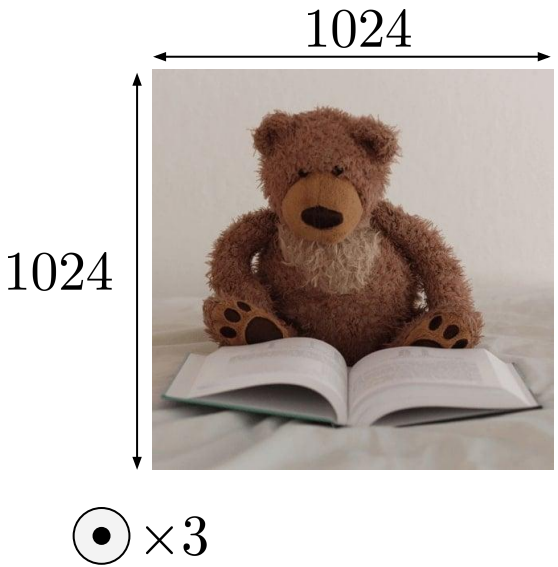


$$x_{\text{image}} \in \mathbb{R}^{H \times W \times 3}$$

$$n = \underbrace{H \times W}_{\text{Number of pixels}} \times 3$$

Limitations of the pixel space

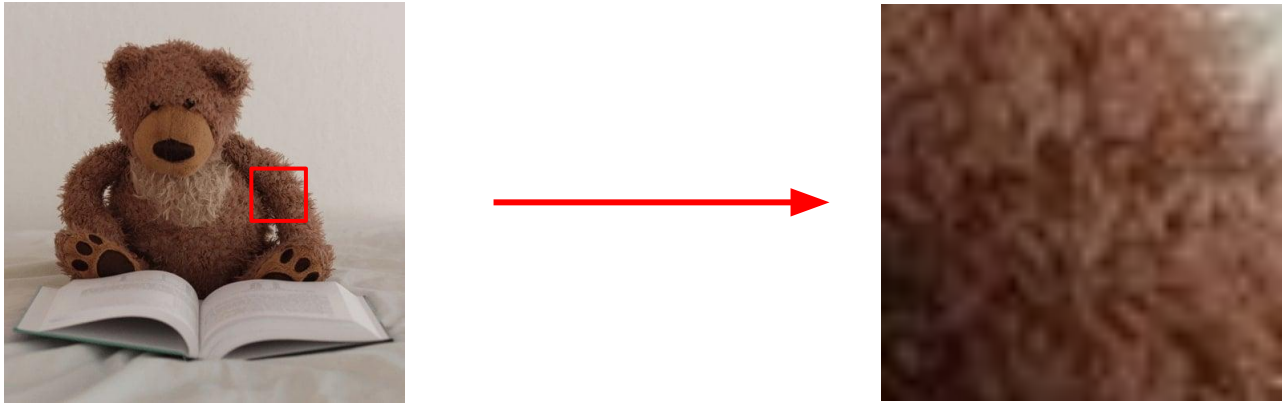
- **High** dimensionality



$$n = 1024^2 \times 3 \approx 10^6$$

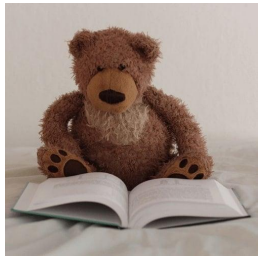
Limitations of the pixel space

- **High** dimensionality
- **Redundant** information (correlated pixels)



Limitations of the pixel space

- **High** dimensionality
- **Redundant** information (correlated pixels)
- Representation **not meaningful** (if we move in space, image becomes gibberish)

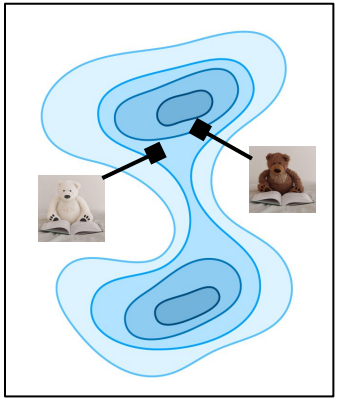


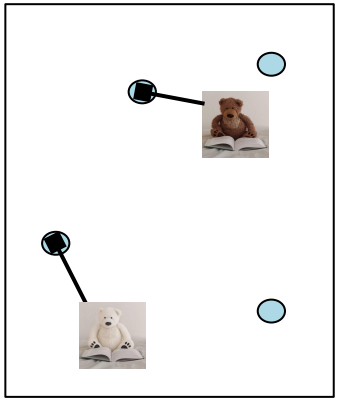
z

$z + \epsilon$

Limitations of the pixel space

- **High** dimensionality
- **Redundant** information (correlated pixels)
- Representation **not meaningful** (if we move in space, image becomes gibberish)





Wish list of an ideal space

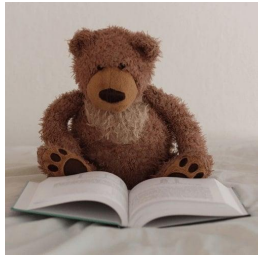
- ~~**High** dimensionality~~ ← **Tractable dimension**
- **Redundant** information (correlated pixels)
- Representation **not meaningful** (if we move in space, image becomes gibberish)

Wish list of an ideal space

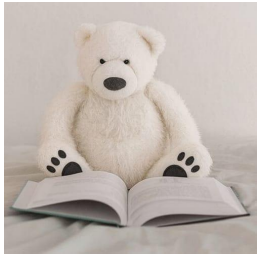
- ~~**High** dimensionality~~ ← **Tractable dimension**
- ~~**Redundant** information (correlated pixels)~~ ← **Compact representation**
- Representation **not meaningful** (if we move in space, image becomes gibberish)

Wish list of an ideal space

- ~~**High** dimensionality~~ ← **Tractable dimension**
- ~~**Redundant** information (correlated pixels)~~ ← **Compact representation**
- ~~Representation **not meaningful** (if we move in space, image becomes gibberish)~~ ← **Meaningful representation**



z

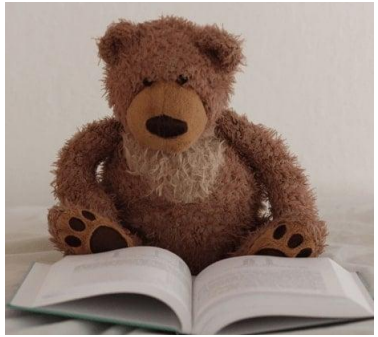


$z + \epsilon$

Terminology

Semantic similarity

- Structural
- Global geometry
- "Low" frequency



Terminology

Semantic similarity

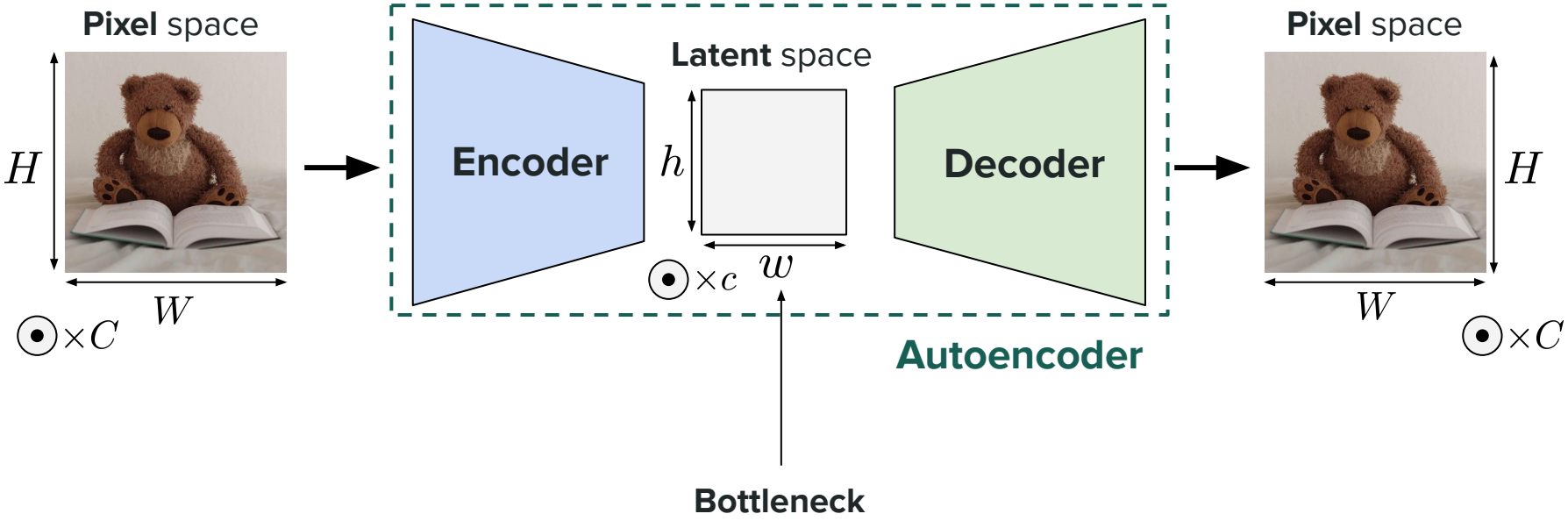
- Structural
- Global geometry
- "Low" frequency

Perceptual similarity

- Local
- Texture
- "High" frequency

Attempt 1: Learn representation via an autoencoder

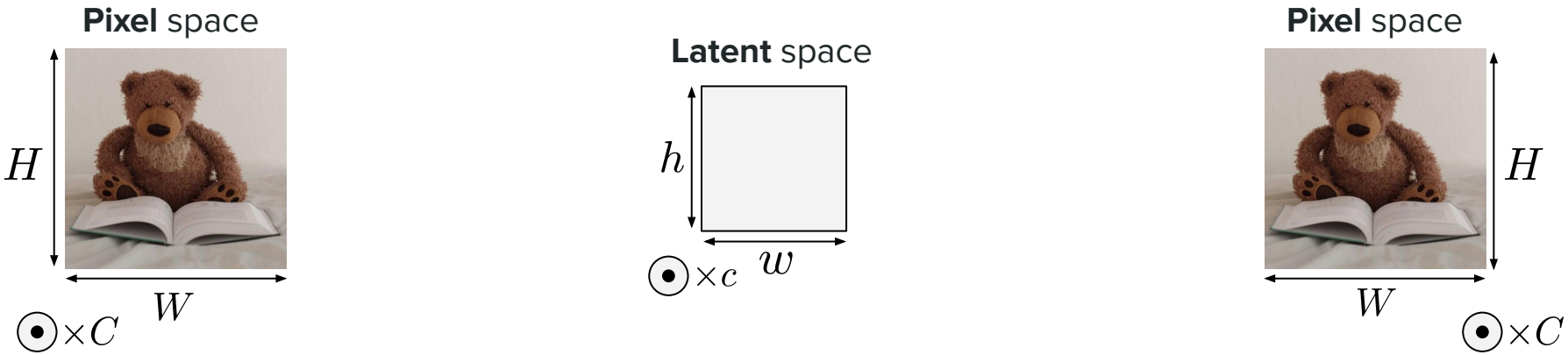
Goal. Learn latent representation via "proxy task" (reconstruction)



Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction, Masci et al., 2011.

Attempt 1: Learn representation via an autoencoder

Goal. Learn latent representation via "proxy task" (reconstruction)



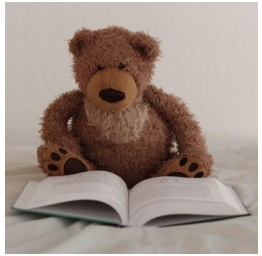
Spatial compression ratio

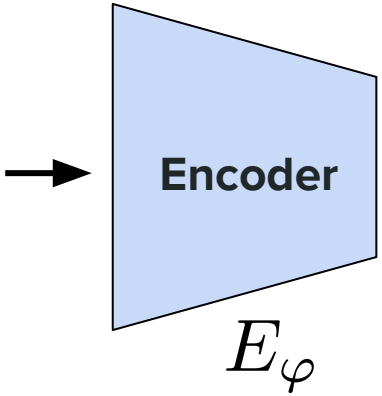
$$f = \frac{H}{h} = \frac{W}{w}$$

Encoder

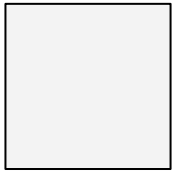
x

Pixel space





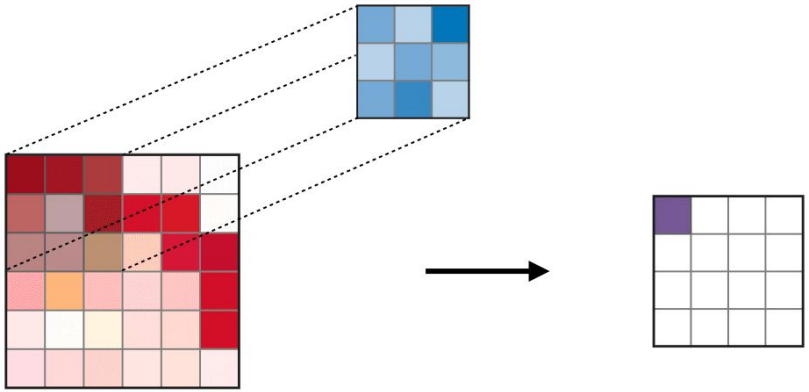
Latent space



$$z = E_\varphi(x)$$

Downsampling operations via convolutions

Quick refresher on convolutions

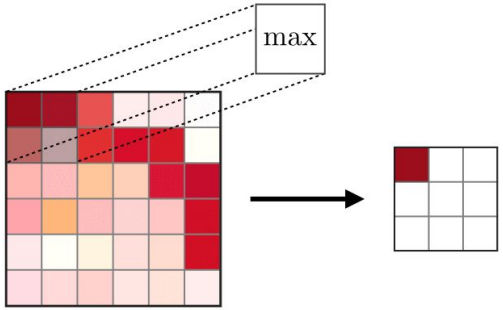


Convolution

VIP cheatsheets on Convolutional Neural Networks, Amidi, 2018.

Quick refresher on convolutions

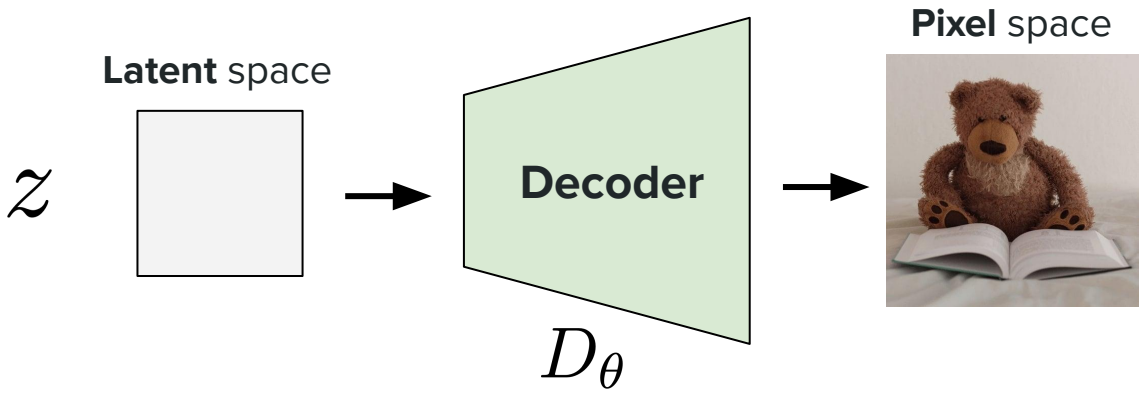
Convolution



Pooling

VIP cheatsheets on Convolutional Neural Networks, Amidi, 2018.

Decoder



$$\widehat{x} = D_\theta(z)$$

Upsampling operations via reverse convolutions

Loss function of autoencoder

AE = **A**uto**E**ncoder

Goal. Learn how to reconstruct input

Loss. Compare input x with reconstructed input $\widehat{x}$

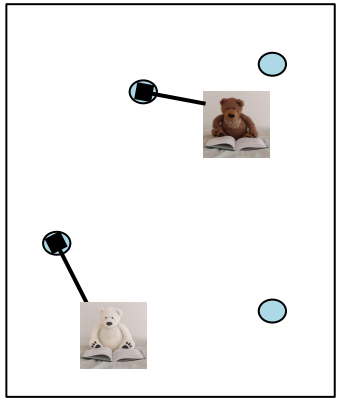
$$\mathcal{L}_{\mathrm{AE}} = \|x - \underbrace{D_\theta(\overbrace{E_\varphi(x)}^{z})}_{\widehat{x}}\|^2$$

Checklist for attempt 1

✅ **Tractable** dimension

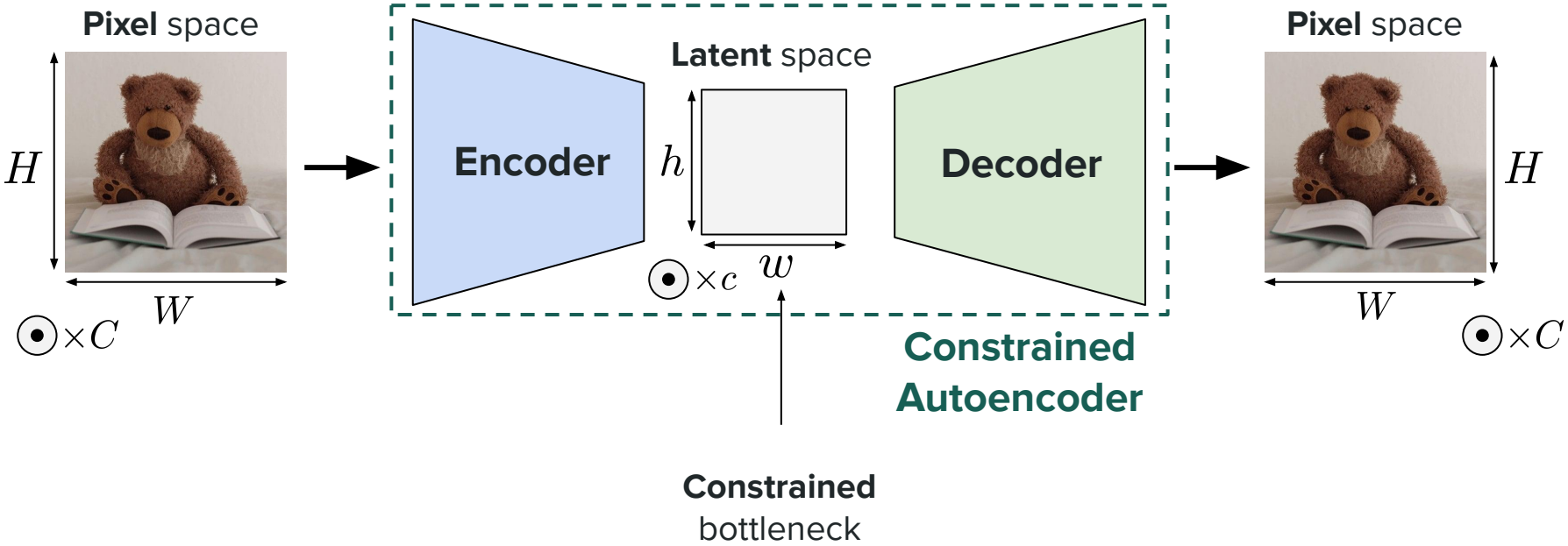
✅ **Compact** representation

❌ **Meaningful** representation



Attempt 2: Learn a constrained latent space

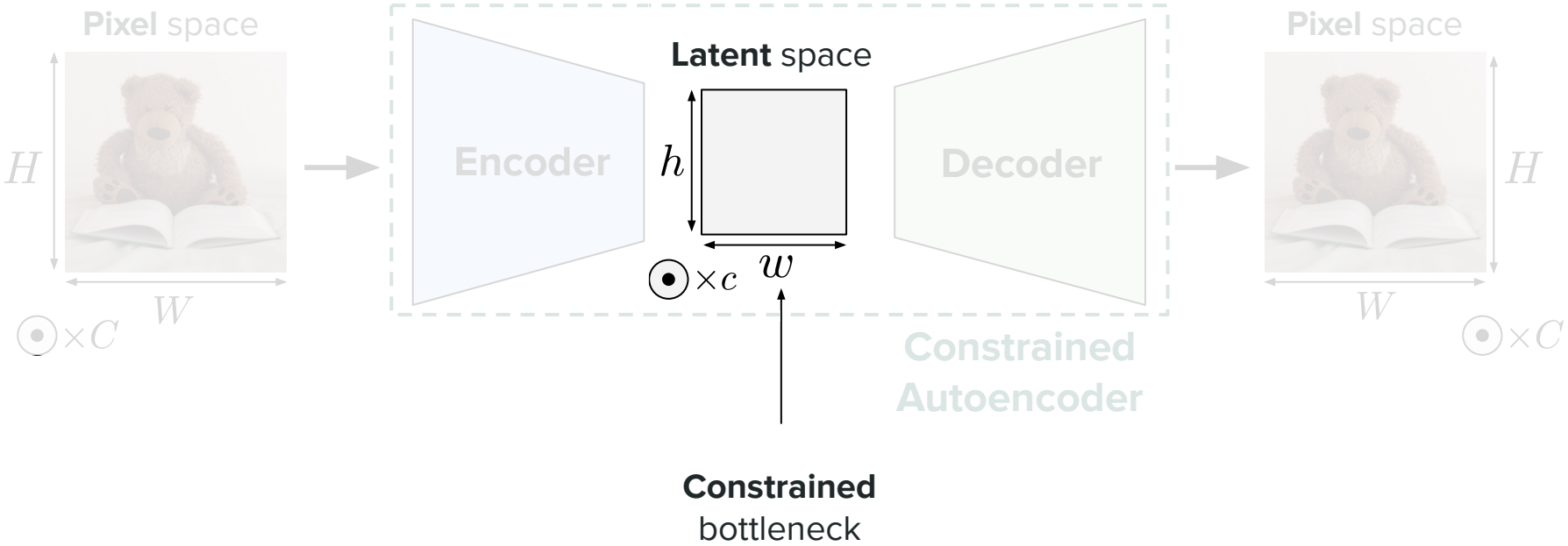
Goal. Enforce some "**structure**" on the latent space



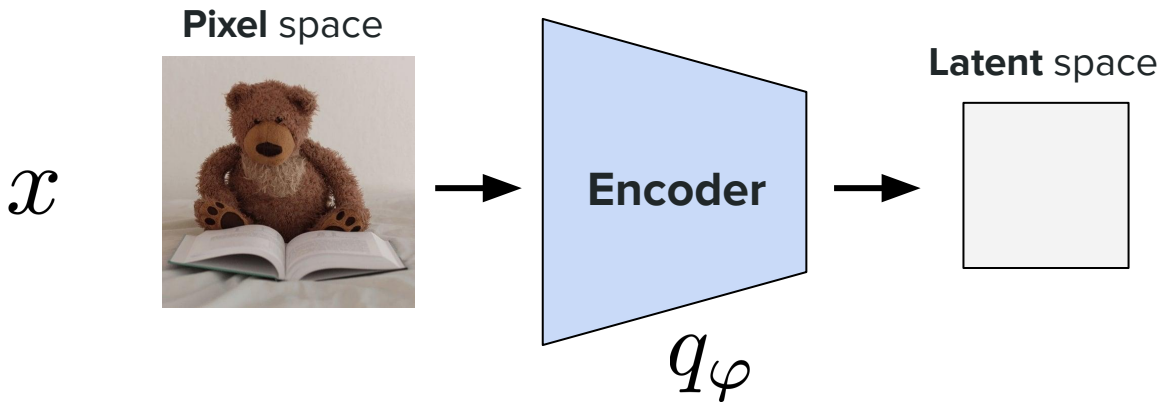
Auto-Encoding Variational Bayes, Kingma et al., 2013.

Attempt 2: Learn a constrained latent space

Goal. Enforce some "**structure**" on the latent space



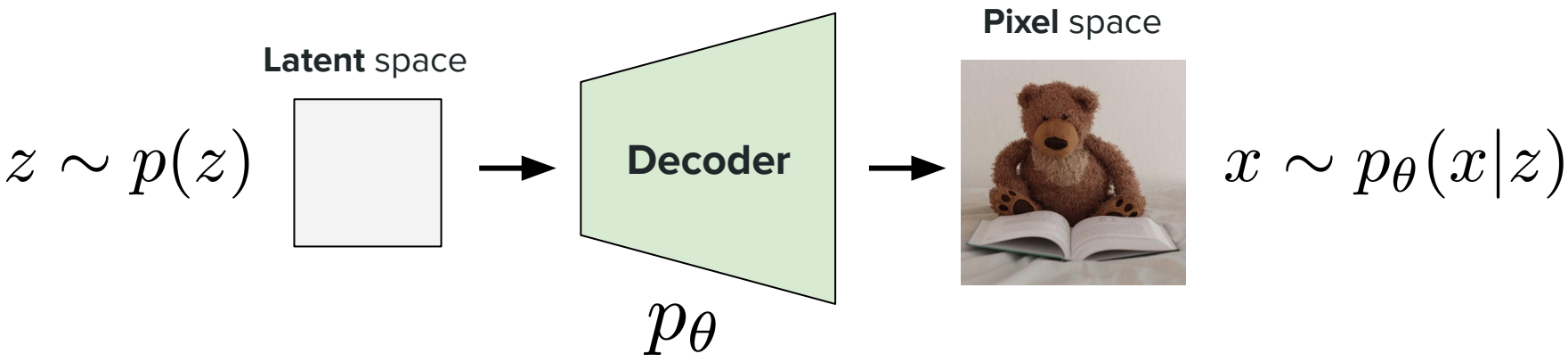
Revised encoder



$$z \sim q_\varphi(z|x)$$

$$q_\varphi(.|x) = \mathcal{N}(\mu_\varphi(x), \sigma^2_\varphi(x))$$

Revised decoder



$$p_\theta(.|z) = \mathcal{N}(\mu_\theta(z), \sigma_\theta^2(z))$$

Revised decoder

$$p_\theta(.|z) = \mathcal{N}(\mu_\theta(z), \sigma_\theta^2(z))$$

$$\uparrow$$

$$\sigma^2 I$$

Assumption: constant variance (for simplicity)

Objective

$$\max_{\theta} \log p_\theta(x)$$

Step 1: Re-using ELBO trick from Lecture 1!

1 Derive **lower bound** for maximum (log-)likelihood estimation

$$\log p_\theta(x) \geqslant \boxed{\mathbb{E}_{z \sim q_\varphi(z|x)} \left[\log \frac{p_\theta(x, z)}{q_\varphi(z|x)} \right]}$$

ELBO = **E**vidence **L**ower **BO**und

Derivation: Use Jensen's inequality on a convenient variational distribution

Step 2: Expand terms of the lower bound

2 **Expand terms** of lower bound

$$\text{ELBO} = \mathbb{E}_z[\log p_\theta(x|z)] - \text{KL}(q_\varphi(z|x)\|p(z))$$

Derivation: Use properties of the log function and rearrange terms

Loss function of variational autoencoder

VAE = **V**ariational **A**uto**E**ncoder

Goal. Learn how to reconstruct input **using a structured latent space**

Loss. Trade-off between reconstruction and latent space structure:

$$\mathcal{L}_{\mathrm{VAE}} = \underbrace{-\mathbb{E}_z[\log p_\theta(x|z)]}_{\mathcal{L}_{\mathrm{rec}}} + \underbrace{\mathrm{KL}(q_\varphi(z|x)\|p(z))}_{\mathcal{L}_{\mathrm{KL}}}$$

Reconstruction

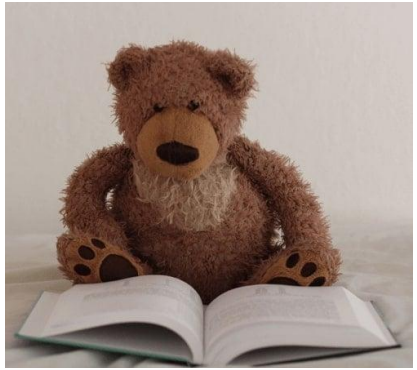
Regularization of latent space

Auto-Encoding Variational Bayes, Kingma et al., 2013.

Checklist for attempt 2

- ✅ **Tractable** dimension
- ✅ **Compact** representation
- ✅ **Meaningful** representation
- ❌ **Truthful** representation

x



$\widehat{x}$

Attempt 3: address limitations of original VAE

$$\mathcal{L}_{\mathrm{VAE}} = \underbrace{\lambda_{\mathrm{rec}}\ \mathcal{L}_{\mathrm{rec}}}_{\textbf{Reconstruction}} + \underbrace{\lambda_{\mathrm{KL}}\ \mathcal{L}_{\mathrm{KL}}}_{\text{Latent space } \textbf{regularization}}$$

Goal. Avoid blurriness!

Refresher on reconstruction loss

$$\mathcal{L}_{\text{rec}} = \|x - \widehat{x}\|^2$$

Idea. Check **reconstruction** of input

Description. L_1 or L_2 **pixel-wise distance** between output and actual

Weight. $\lambda_{\text{rec}} \approx 1$

If λ_{rec} too high, this can produce blurry outputs

Refresher on KL regularization

$$\mathcal{L}_{\mathrm{KL}} = \mathrm{KL}(q_\varphi(z|x) \| p(z))$$

Idea. Make **latent** space more **structured**

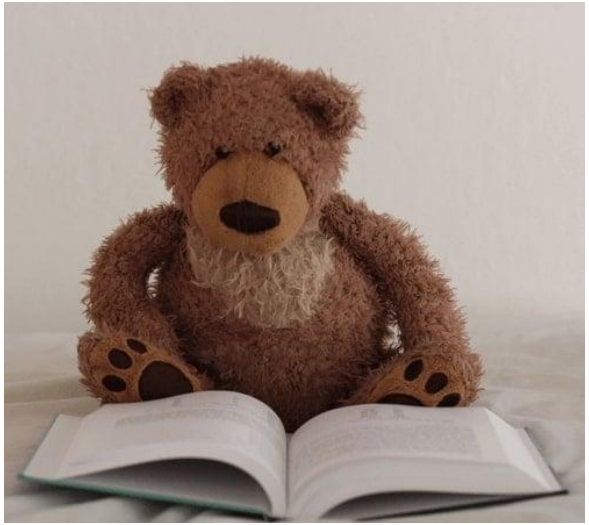
Description. KL divergence between the output of the encoder and the prior

Weight. $\lambda_{\mathrm{KL}} \approx 10^{-6}$

If λ_{KL} too high, this can lead to "**posterior collapse**"

Combat blurriness with perceptual loss

x



Combat blurriness with perceptual loss

$$\phi_l(x)$$

Example of **feature maps** in early layers

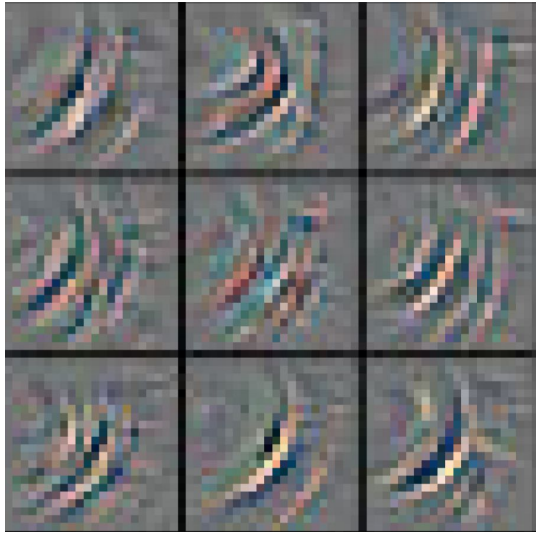


Illustration from Visualizing and Understanding Convolutional Networks, Zeiler et al., 2013.

Combat blurriness with perceptual loss

Idea. Force model to pay attention to **shapes sensitive to human eye** (edges, shapes) and have some **translation invariance**

Combat blurriness with perceptual loss

$$\mathcal{L}_{\text{perc}} = \sum_{l} \frac{1}{H_l W_l} \left\| w^l \odot (\phi_l(x) - \phi_l(\hat{x})) \right\|^2$$

Idea. Force model to pay attention to **shapes sensitive to human eye** (edges, shapes) and have some **translation invariance**

Description. **L**earned **P**erceptual **I**mage **P**atch **S**imilarity (LPIPS)

The Unreasonable Effectiveness of Deep Features as a Perceptual Metric, Zhang et al., 2018.

Combat blurriness with perceptual loss

$$\mathcal{L}_{\text{perc}} = \sum_l \frac{1}{H_l W_l} \left\| w^l \odot (\phi_l(x) - \phi_l(\hat{x})) \right\|^2$$

Idea. Force model to pay attention to **shapes sensitive to human eye** (edges, shapes) and have some **translation invariance**

Description. **L**earned **P**erceptual **I**mage **P**atch **S**imilarity (LPIPS)

The Unreasonable Effectiveness of Deep Features as a Perceptual Metric, Zhang et al., 2018.

Combat blurriness with perceptual loss

$$\mathcal{L}_{\text{perc}} = \sum_l \frac{1}{H_l W_l} \left\| w^l \odot (\phi_l(x) - \phi_l(\hat{x})) \right\|^2$$

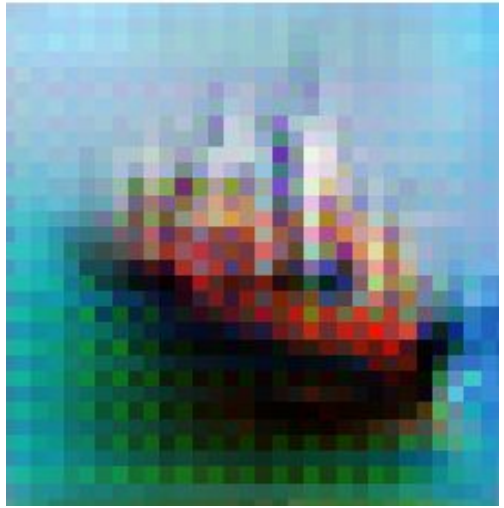
Idea. Force model to pay attention to **shapes sensitive to human eye** (edges, shapes) and have some **translation invariance**

Description. **L**earned **P**erceptual **I**mage **P**atch **S**imilarity (LPIPS)

Weight. $\lambda_{\text{perc}} \approx 1$

Combat blurriness with perceptual loss

No
"checkerboard artifact"



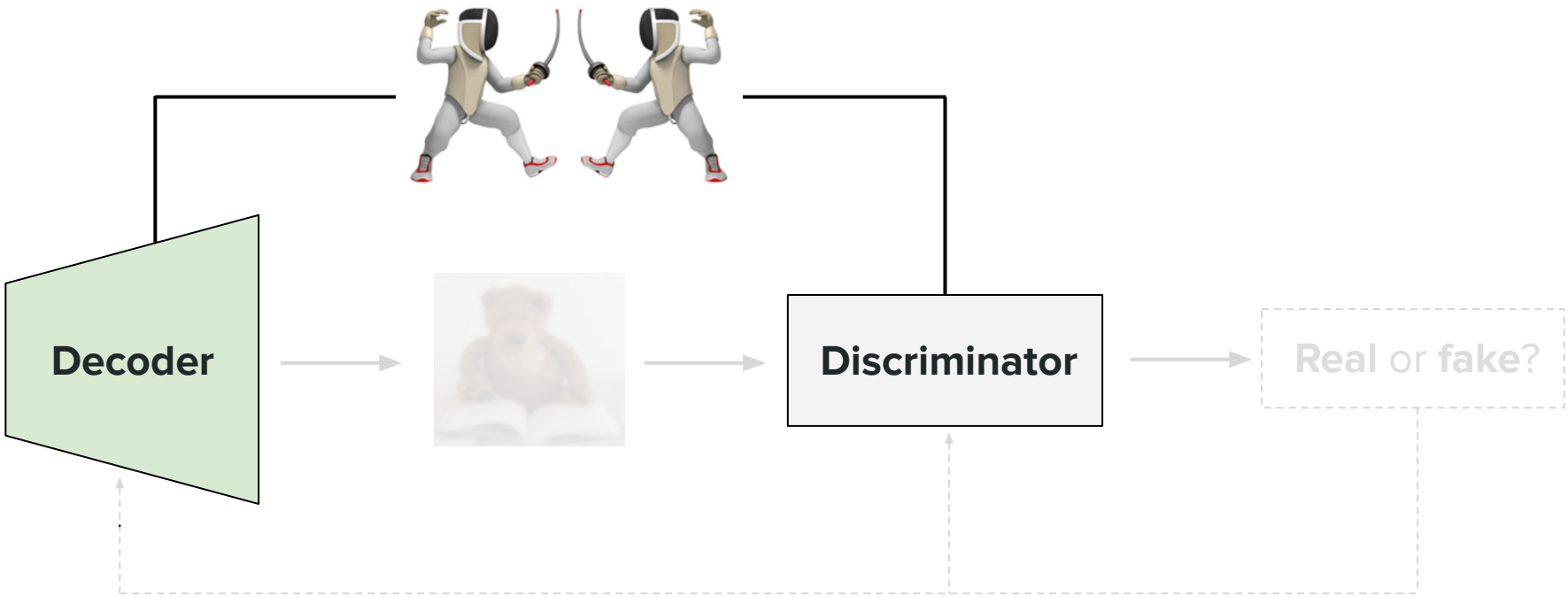
Presence of
"checkerboard artifact"

If λ_{perc} too high, this can produce "checkerboard artifacts"

Illustrations from Deconvolution and Checkerboard Artifacts, Odena et al., 2016.

Make output more realistic with adversarial loss

Make output more realistic with adversarial loss



Make output more realistic with adversarial loss

$$\mathcal{L}_{\text{adv}} = -\mathbb{E}_x[C_\omega(\widehat{x})]$$

Idea. Prevent blurriness by being force to produce a **realistic image**

Description. Incentivize discriminator (critic) to be fooled

Weight. $\lambda_{\text{adv}} \approx 0.1 \text{ to } 1$

If λ_{adv} too high, this may lead to "**mode collapse**" and ignore the latent

Summary of refined VAE

$$\mathcal{L}_{\text{VAE}} = \begin{cases} \lambda_{\text{rec}}\ \mathcal{L}_{\text{rec}} & \textbf{Reconstruction} \\ + \lambda_{\text{KL}}\ \mathcal{L}_{\text{KL}} & \text{Latent space } \textbf{regularization} \\ + \lambda_{\text{perc}} \mathcal{L}_{\text{perc}} & \textbf{Perception} \\ + \lambda_{\text{adv}}\ \mathcal{L}_{\text{adv}} & \textbf{Adversarial} \end{cases}$$

Mitigate blurriness

Diffusion & Large Vision Models

From pixel to latent space

Latent diffusion models

Text representation

Image representation

Contrastive learning

Guidance

Diffusion in latent space

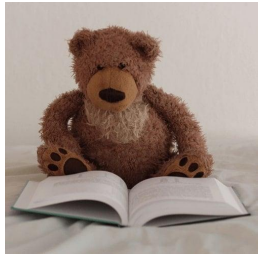
Training.

1. Train **VAE**
2. Train **image generation model** in **VAE latent space** using frozen **VAE encoder**

Training leverages encoder

Pixel space

$x_1 \sim p_{\text{data}}$



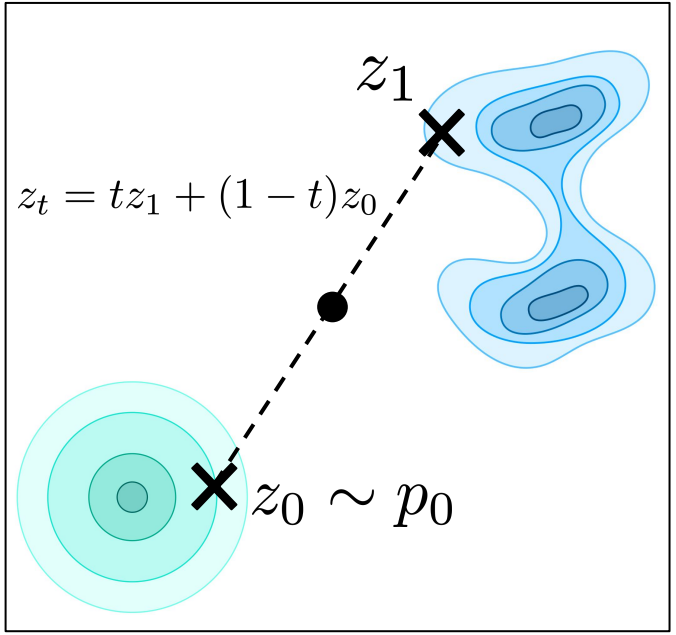
Encoder

q_φ

Latent space

$z_1 \sim q_\varphi(z_1 | x_1)$

Latent space



Training leverages encoder

$$\mathcal{L} = \|u_t^\theta(z_t) - (z_1 - z_0)\|^2$$

Pixel space

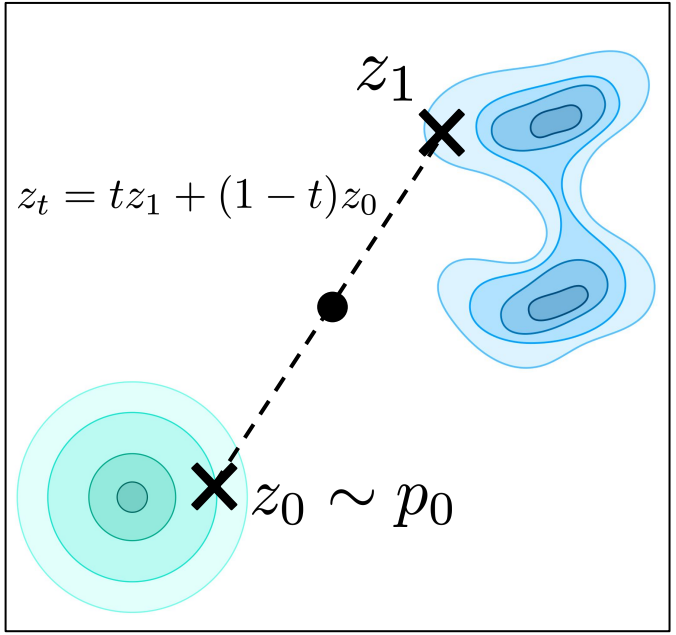
$x_1 \sim p_{\text{data}}$ →

Encoder

q_φ

→ $z_1 \sim q_\varphi(z_1|x_1)$

Latent space



Diffusion in latent space

Training.

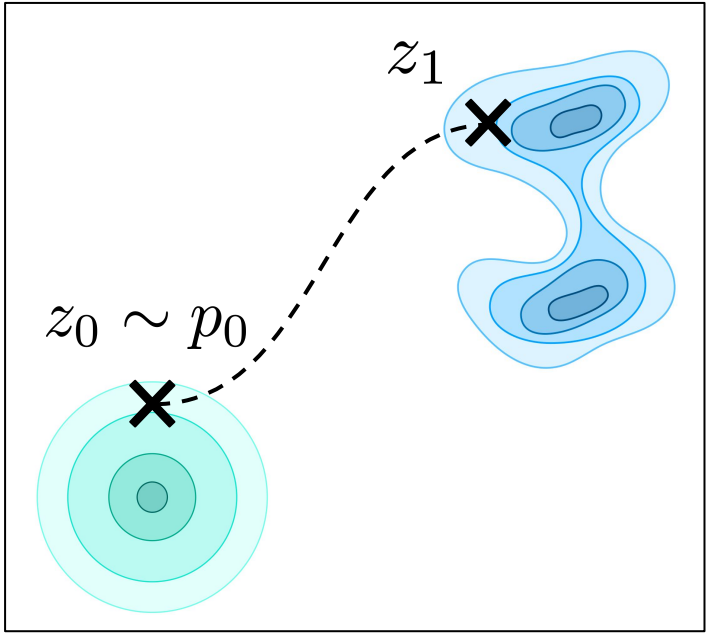
1. Train **VAE**
2. Train **image generation model** in **VAE latent space** using frozen **VAE encoder**

Inference.

1. Start from a noisy latent and perform diffusion / score matching / flow matching in **latent space**
2. Use **VAE decoder** to obtain final image in **pixel space**

Inference leverages decoder

Latent space



$z_1 \longrightarrow$ **Decoder** $\longrightarrow$

p_θ

Pixel space

$x_1 \sim p_\theta(x_1|z_1)$

VAE used for image generation models

- **Encoder**: acts as a "low-pass filter"

Interesting experiment results in FLUX.2: Analyzing and Enhancing the Latent Space of FLUX – Representation Comparison, 2025.
High-Resolution Image Synthesis with Latent Diffusion Models, Rombach et al., 2021.

VAE used for image generation models

- **Encoder**: acts as a "low-pass filter"

- **Decoder**:
  - Responsible to convey low-level details via "texture hallucination"
  - Takes in a highly compressed low-dimensional vector and needs to convert it to a high-dimensional one

Interesting experiment results in FLUX.2: Analyzing and Enhancing the Latent Space of FLUX – Representation Comparison, 2025.
High-Resolution Image Synthesis with Latent Diffusion Models, Rombach et al., 2021.

VAE used for image generation models

- **Encoder**: acts as a "low-pass filter"

- **Decoder**:
  - Responsible to convey low-level details via "texture hallucination"
  - Takes in a highly compressed low-dimensional vector and needs to convert it to a high-dimensional one

- Decoder is **2x bigger** than Encoder

Interesting experiment results in FLUX.2: Analyzing and Enhancing the Latent Space of FLUX – Representation Comparison, 2025.
High-Resolution Image Synthesis with Latent Diffusion Models, Rombach et al., 2021.

Diffusion & Large Vision Models

From pixel to latent space

Latent diffusion models

Text representation

Image representation

Contrastive learning

Guidance

Today's lecture: Part 2

2 **Condition**

Guided generation

1 Representation

A teddy bear reading a book

Black box architecture

Today's lecture: **Multimodal guided generation**

Class of models

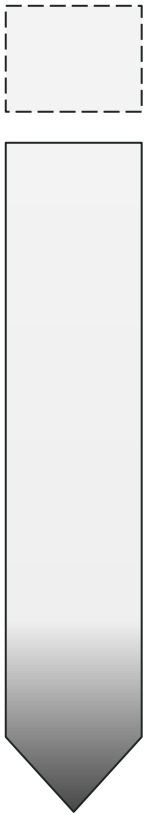
1980s Recurrent neural networks (RNNs)

1997 Long short-term memory (LSTM)

2013 Word2vec

2017 Transformers

2020s Large Language Models



Tokenization

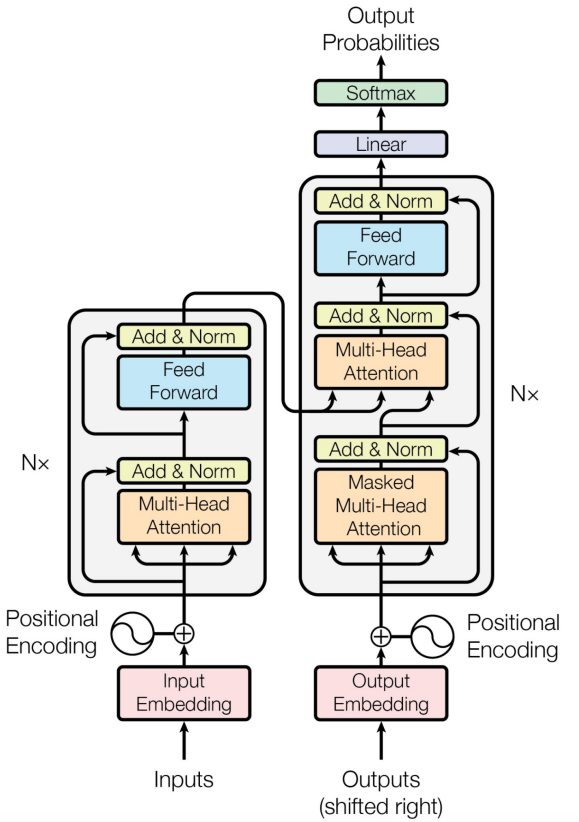
A cute teddy bear is reading.

arbitrary	A	cute	teddy bear			is	reading	.	
word	A	cute	teddy	bear		is	reading	.	
sub-word	A	cute	ted	##dy	bear	is	read	##ing	.

A	_	c	u	t	e	_	t	e	d	d	y	_	b	e	a	r	_	i	s	_	r	e	a	d	i	n	g	.

character

Transformer architecture



Context

- Introduced in 2017 for machine translation
- Relies on the concept of **attention**
- Concepts of query, key, value
- Proxy task: next token prediction

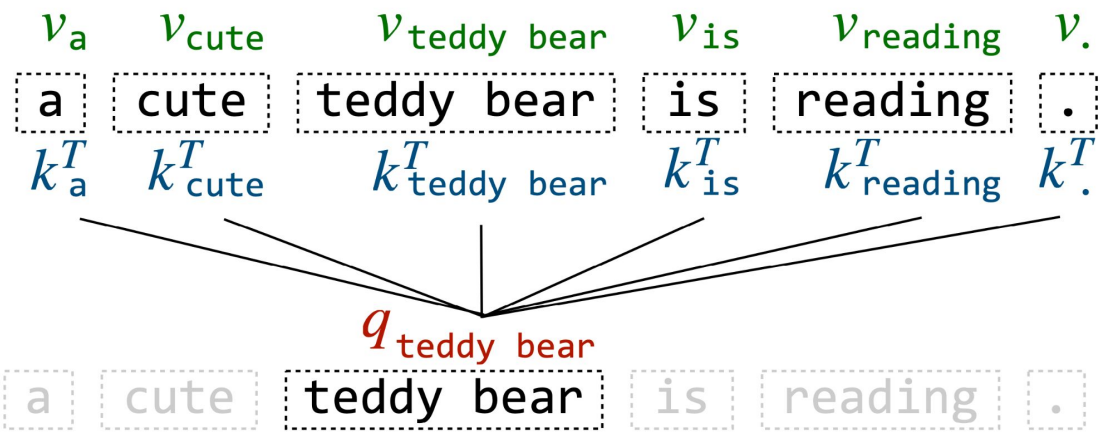
Benefits

- Weaker inductive biases
- More generalizability

Attention Is All You Need, Vaswani et al., 2017.

Attention mechanism

Concept of **Q**uery, **K**ey, **V**alue



"Super Study Guide: Transformers & Large Language Models", by Amidi, 2024.

Attention mechanism

Efficient computations with matrices:

$$\boxed{\mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V}$$

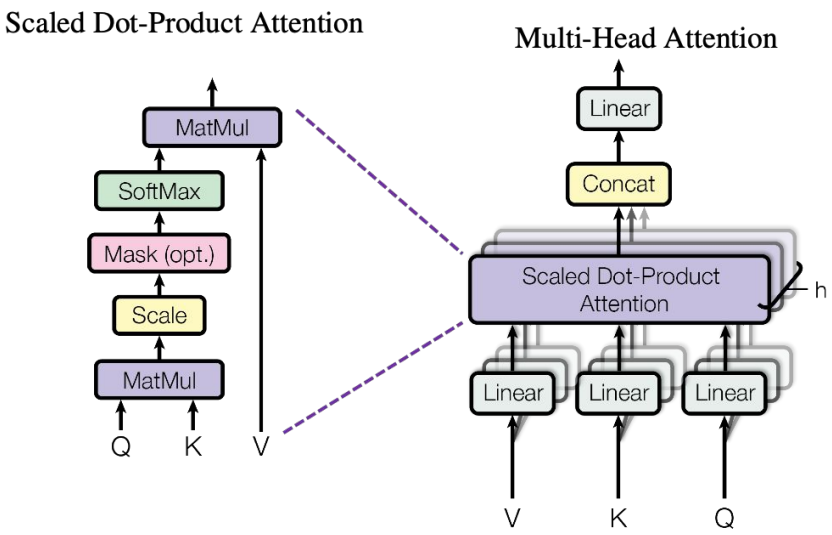
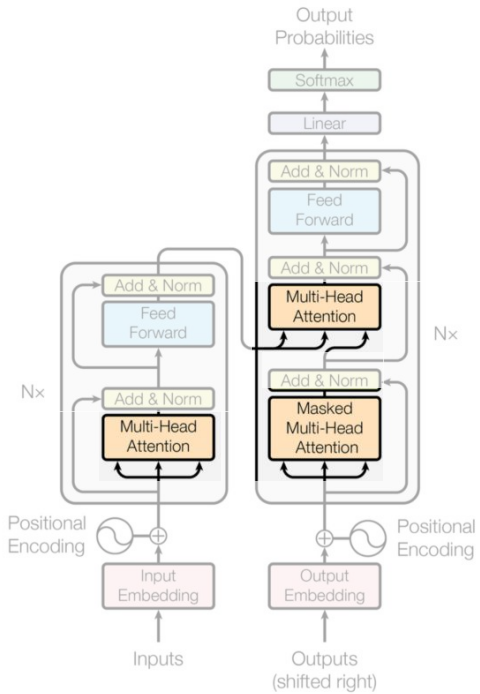


Figure adapted from "Attention Is All You Need", Vaswani et al., 2017.

Attention locations



Self-attention

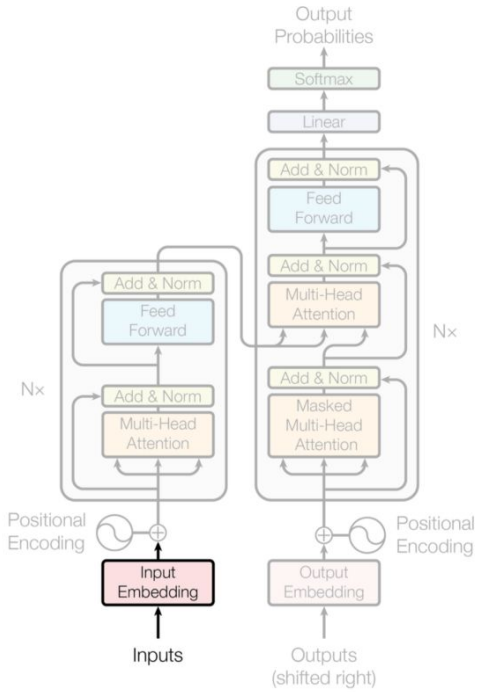
- Encoder-Encoder
- Decoder-Decoder

Cross-attention

- Encoder-Decoder

Figure adapted from "Attention Is All You Need", Vaswani et al., 2017.

Input



Overview

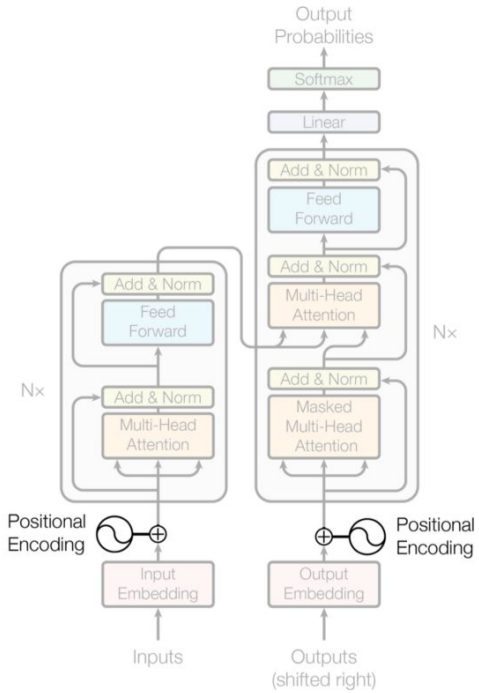
- Text is "tokenized"
- Learned embeddings for tokens

Parameters

- V: vocabulary size
- `d_model`: embedding dimensions

Figure adapted from "Attention Is All You Need", Vaswani et al., 2017.

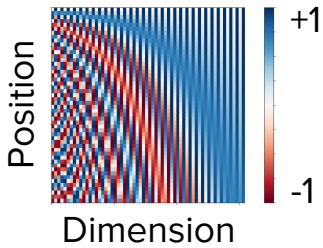
… with a trick!

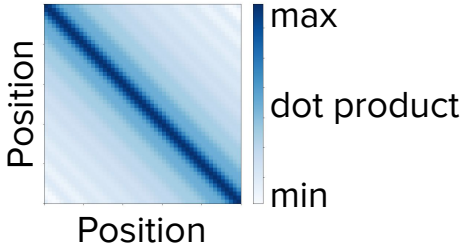


Positional encoding

Idea:

- Add **position information** to inputs
- Can be either learned or hardcoded

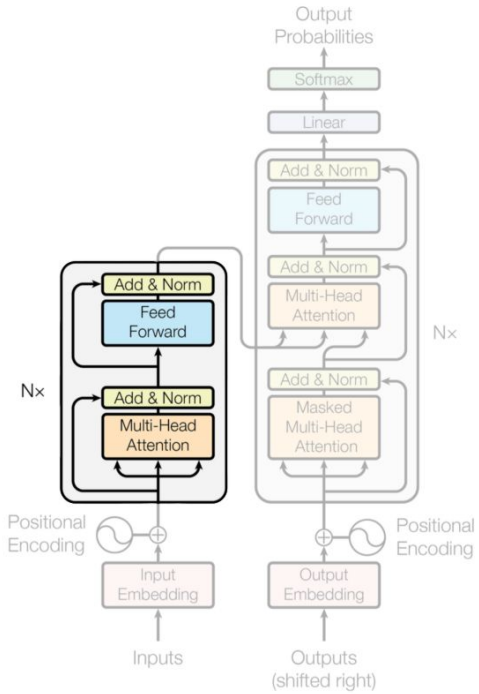




Goal: let model understand relative input position

Left figure adapted from "Attention Is All You Need", Vaswani et al., 2017.
Right figures adapted from "Transformer Architecture: The Positional Encoding", Kazemnejad, 2019.

Encoder



Overview

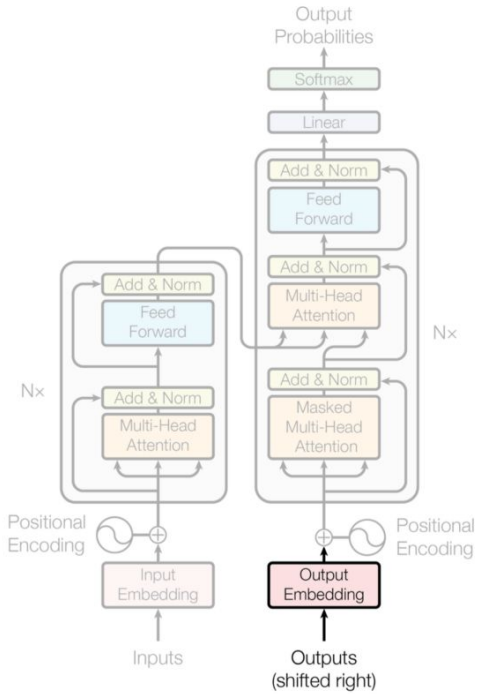
- Encoder-Encoder self-attention
- Feed Forward Neural Network
- Normalization layer

Parameters

- N: layers stacked
- h: number of attention heads
- `d_FF`, `d_key`, `d_value`: sub-layer dimension
- `d_model`: embedding dimensions

Figure adapted from "Attention Is All You Need", Vaswani et al., 2017.

Output "shifted right"



Overview

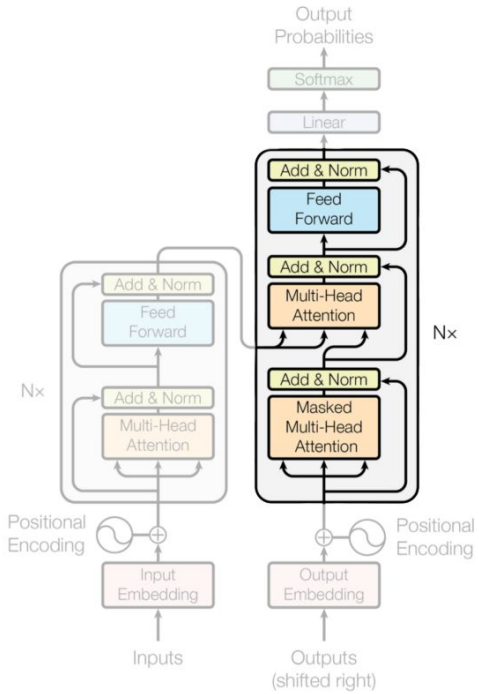
- Learned embeddings for output tokens
- In practice, will start with `[BOS]`

Parameters

- V: vocabulary size
- `d_model`: embedding dimensions

Figure adapted from "Attention Is All You Need", Vaswani et al., 2017.

Decoder



Overview

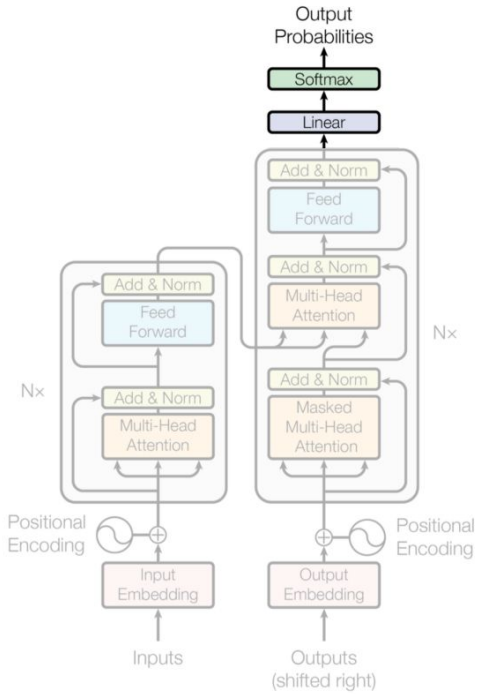
- Decoder-Decoder self-attention
- Encoder-Decoder cross-attention
- Feed Forward Neural Network
- Normalization layer

Parameters

- `N`: layers stacked
- `h`: number of attention heads
- `d_FF`, `d_key`, `d_value`: sub-layer dimension
- `d_model`: embedding dimensions

Figure adapted from "Attention Is All You Need", Vaswani et al., 2017.

Output



Overview

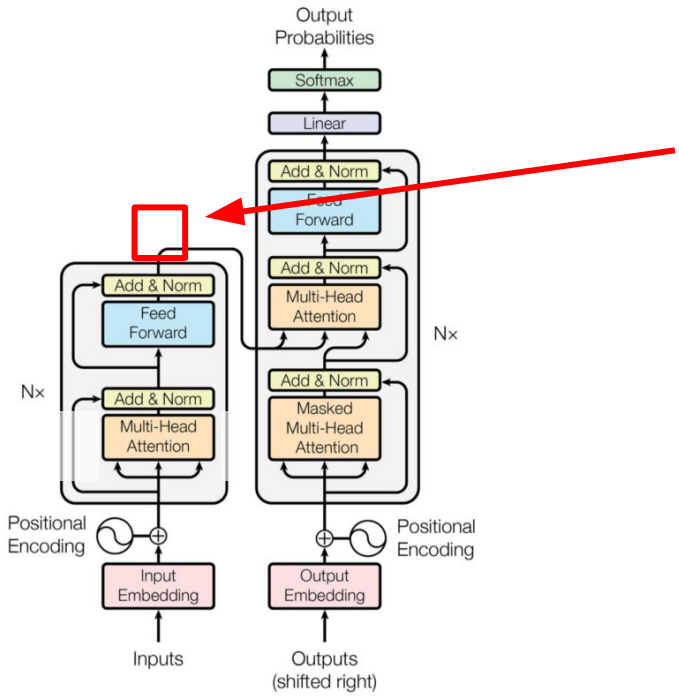
- Linear projection
- Classification problem that outputs probability of belonging to a class, where class = word

Parameters

- V: vocabulary size
- `d_model`: embedding dimensions

Figure adapted from "Attention Is All You Need", Vaswani et al., 2017.

Typical embeddings location



last hidden state of the encoder

Figure adapted from "Attention Is All You Need", Vaswani et al., 2017.

Diffusion & Large Vision Models

From pixel to latent space

Latent diffusion models

Text representation

Image representation

Contrastive learning

Guidance

Stanford University

ICME

Today's lecture: Part 2

2 **Condition**

Guided generation

1 Representation

Black box architecture

Today's lecture: **Multimodal guided generation**

Adapting attention mechanism to images

It's just **numbers**...

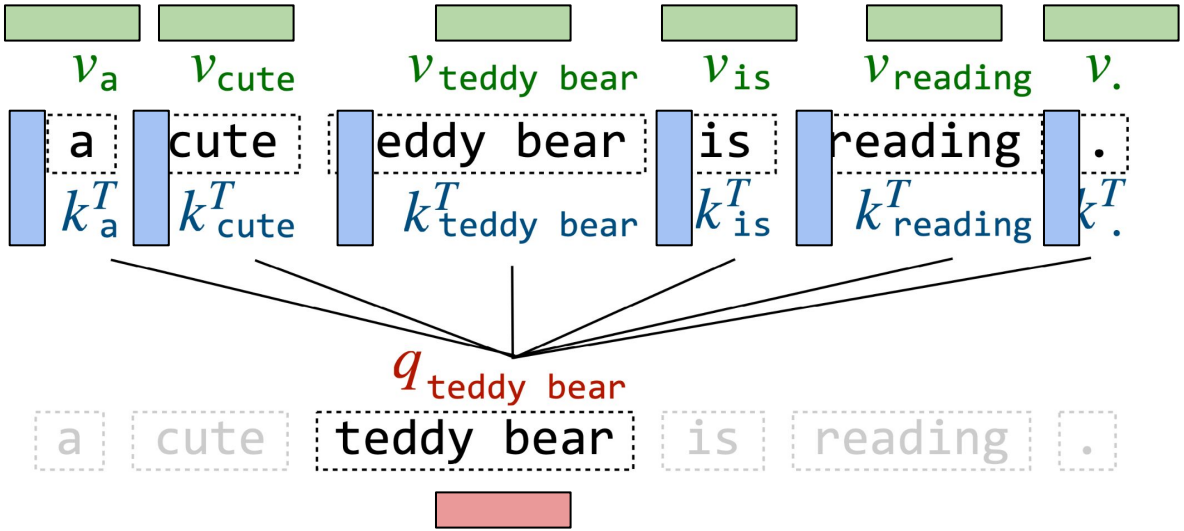
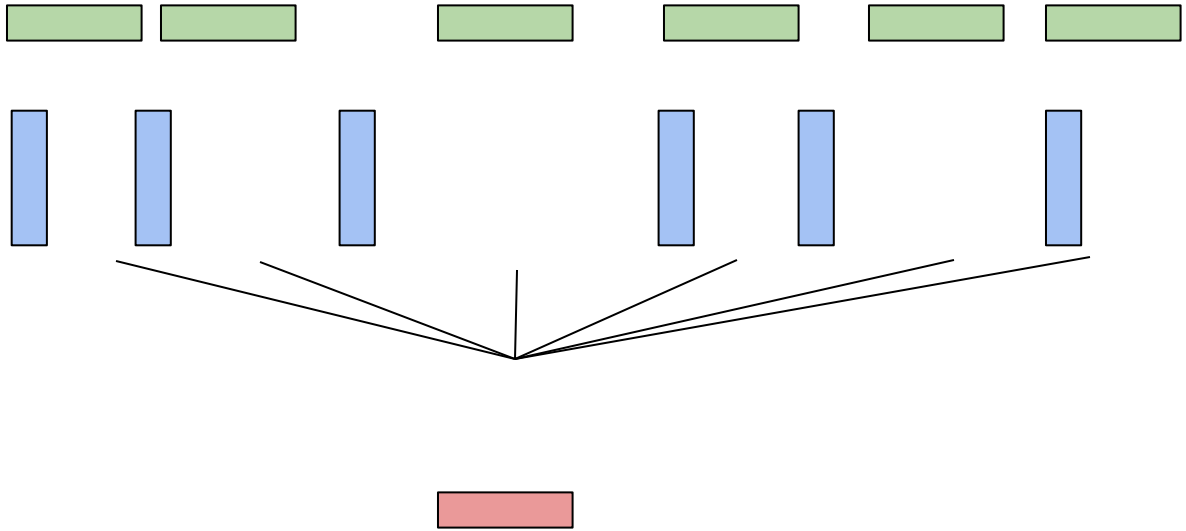


Figure adapted from "Super Study Guide: Transformers & Large Language Models", by Amidi, 2024.

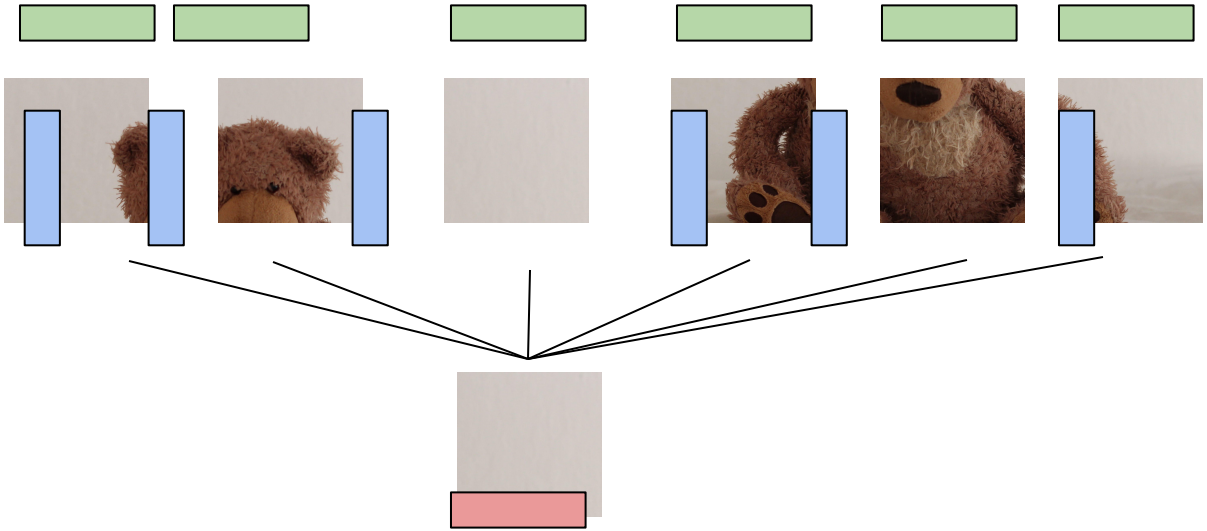
Adapting attention mechanism to images

It's just **numbers**...



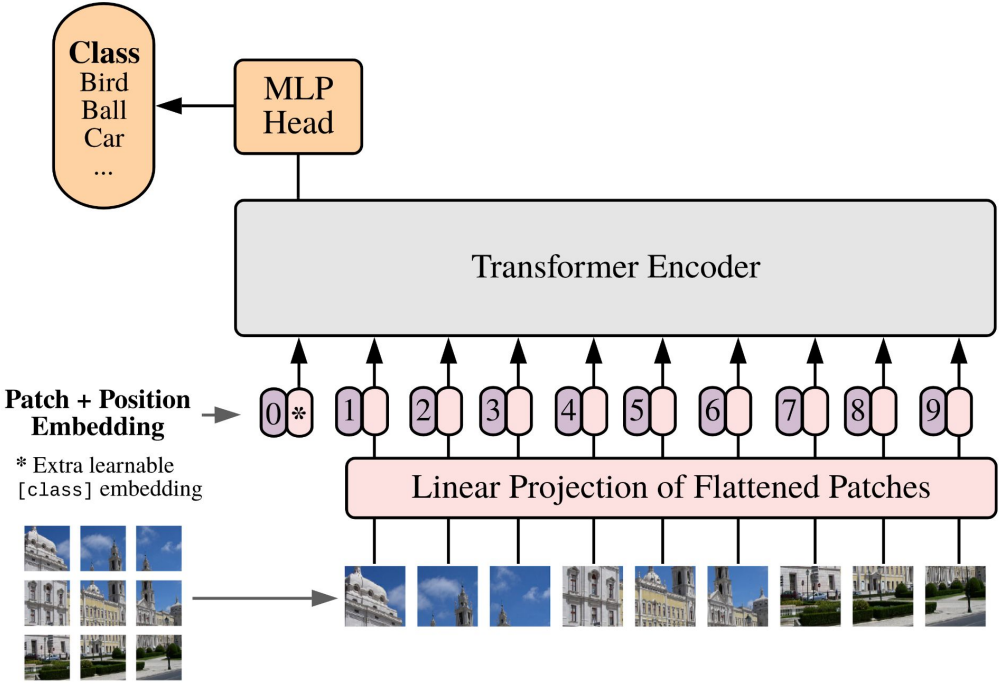
Adapting attention mechanism to images

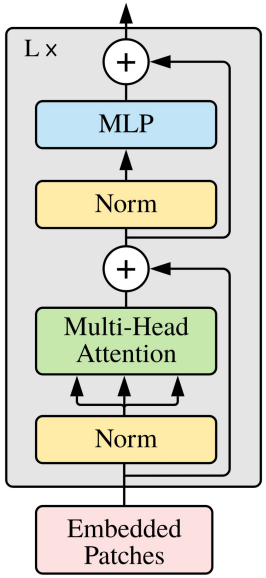
...that could well represent **images**!



Vision Transformer

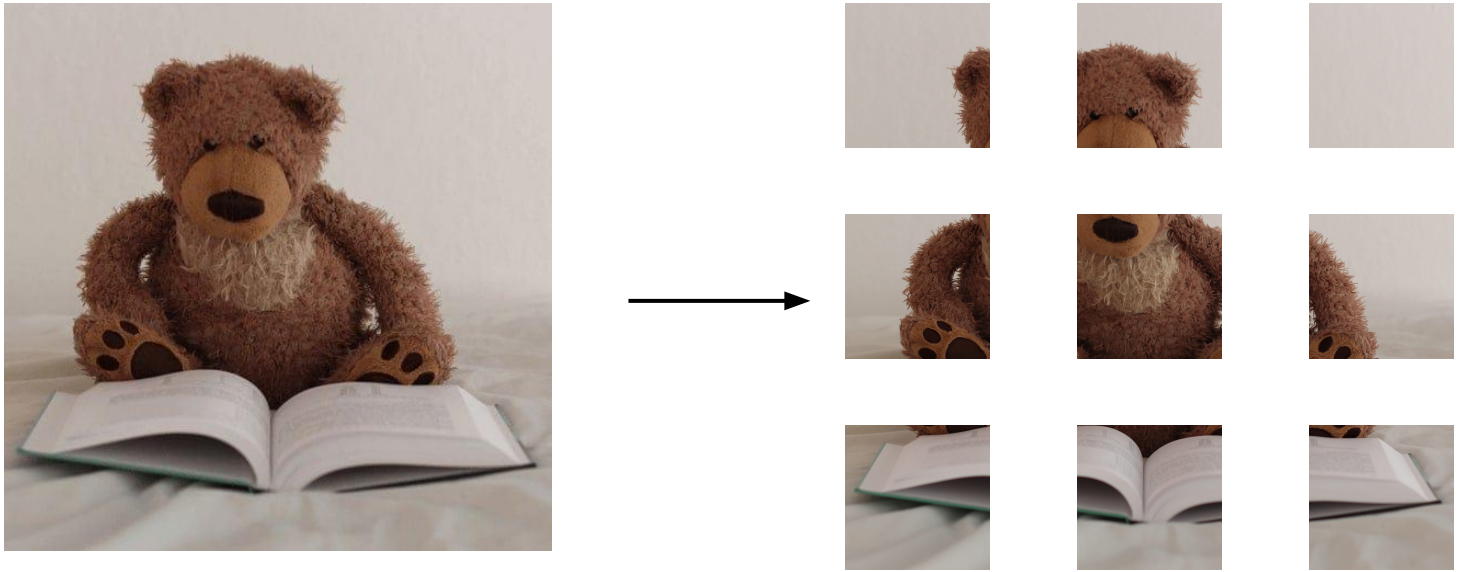
ViT = **Vi**sion **T**ransformer



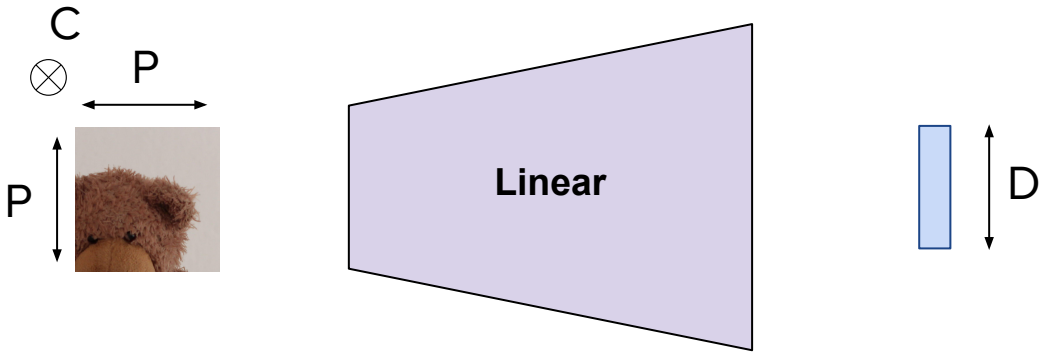


An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, Dosovitskiy et al., 2020.

ViT end-to-end example



ViT end-to-end example

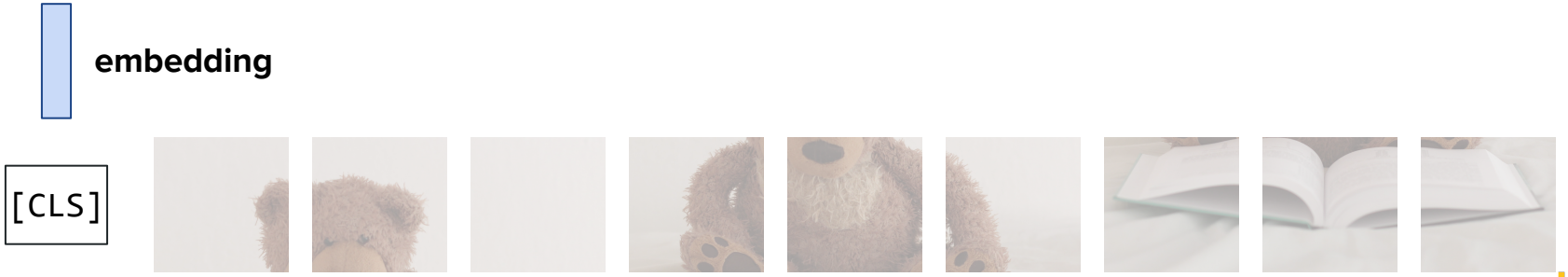


ViT end-to-end example

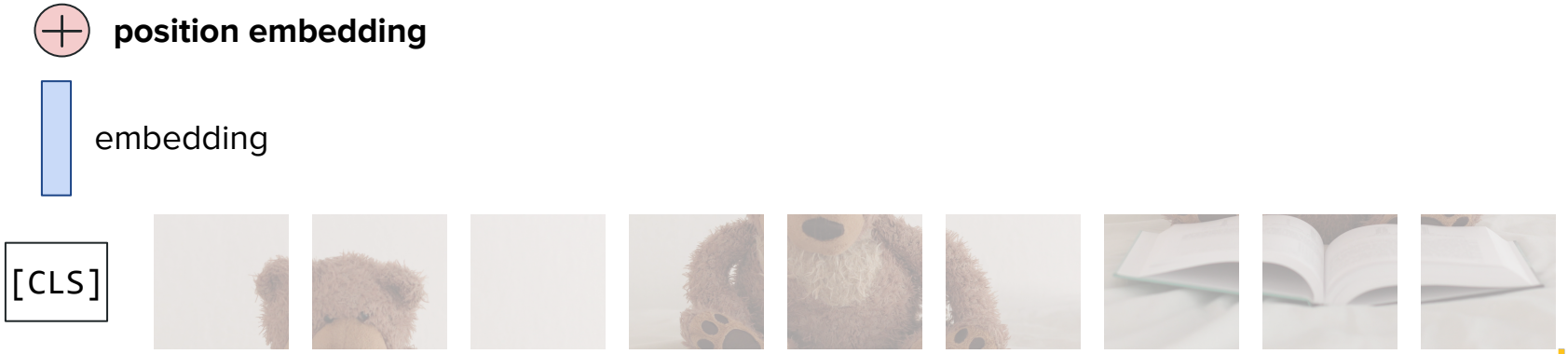
ViT end-to-end example

[CLS]

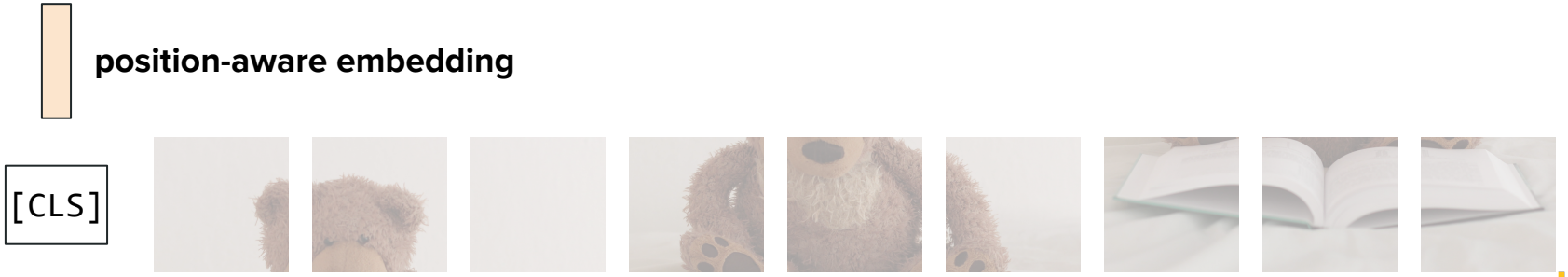
ViT end-to-end example



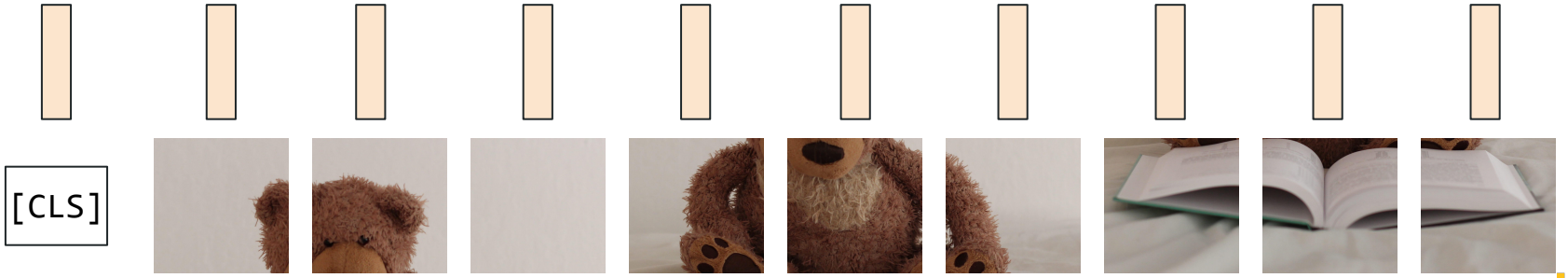
ViT end-to-end example



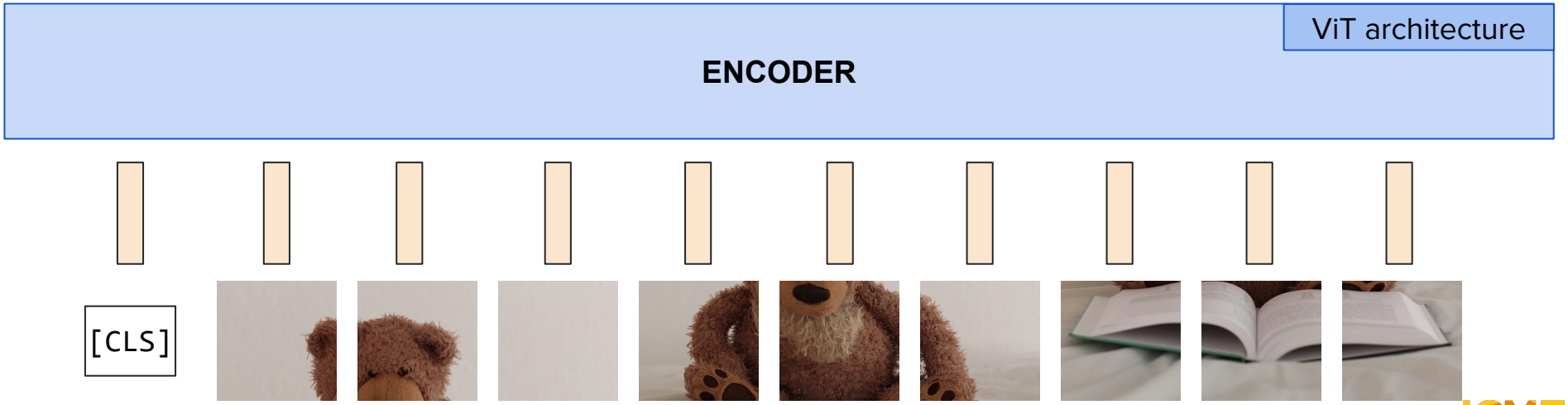
ViT end-to-end example



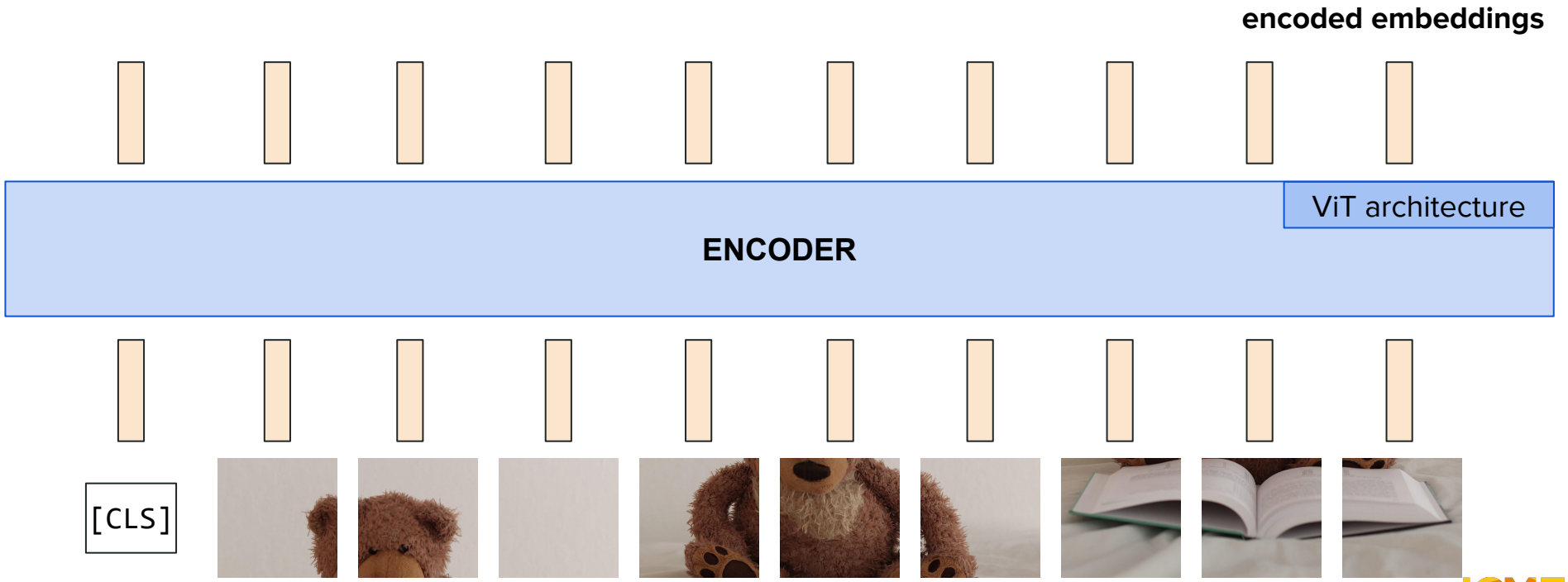
ViT end-to-end example



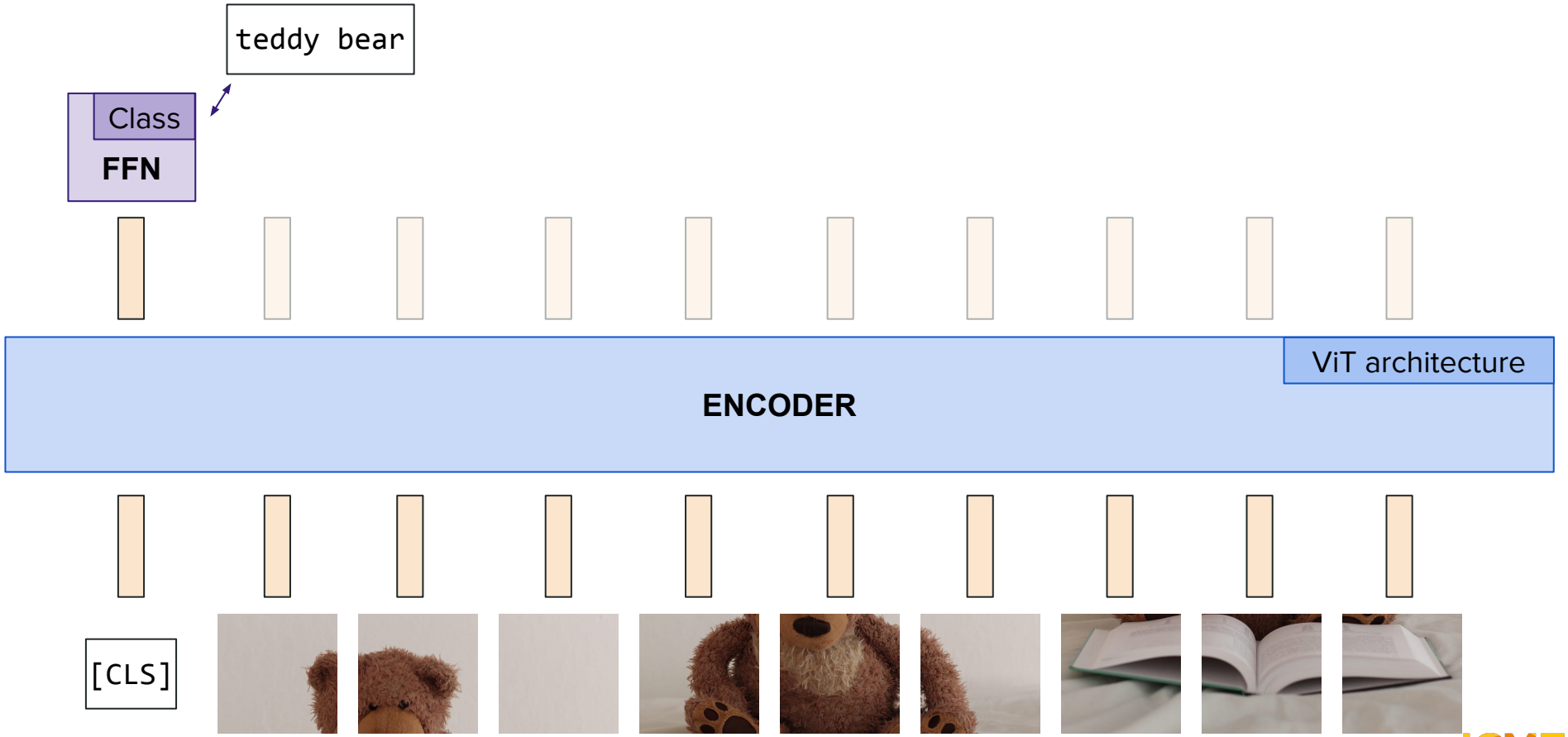
ViT end-to-end example



ViT end-to-end example



ViT end-to-end example



Discussion

Limitations.

- Supervised ViT good but needs labels
- Fits to narrow category-level representation that has to be specified in advance

Discussion

Limitations.

- Supervised ViT good but needs labels
- Fits to narrow category-level representation that has to be specified in advance

Other approaches.

- We didn't have that problem for text because tasks that work are already self-supervised
- Popular work: self-**DI**stillation with **NO** labels

Emerging Properties in Self-Supervised Vision Transformers, Caron et al., 2021.

Discussion

How to make image and text embeddings comparable?

Diffusion & Large Vision Models

From pixel to latent space

Latent diffusion models

Text representation

Image representation

Contrastive learning

Guidance

Contrastive learning

Motivation. How to learn general image/text relationships?

Contrastive learning

Motivation. How to learn general image/text relationships?

Idea. Contrastive learning:

- Group similar items together
- Push away dissimilar items

Contrastive learning

teddy bear

water polo ball

close

Contrastive learning

teddy bear

water polo ball

far

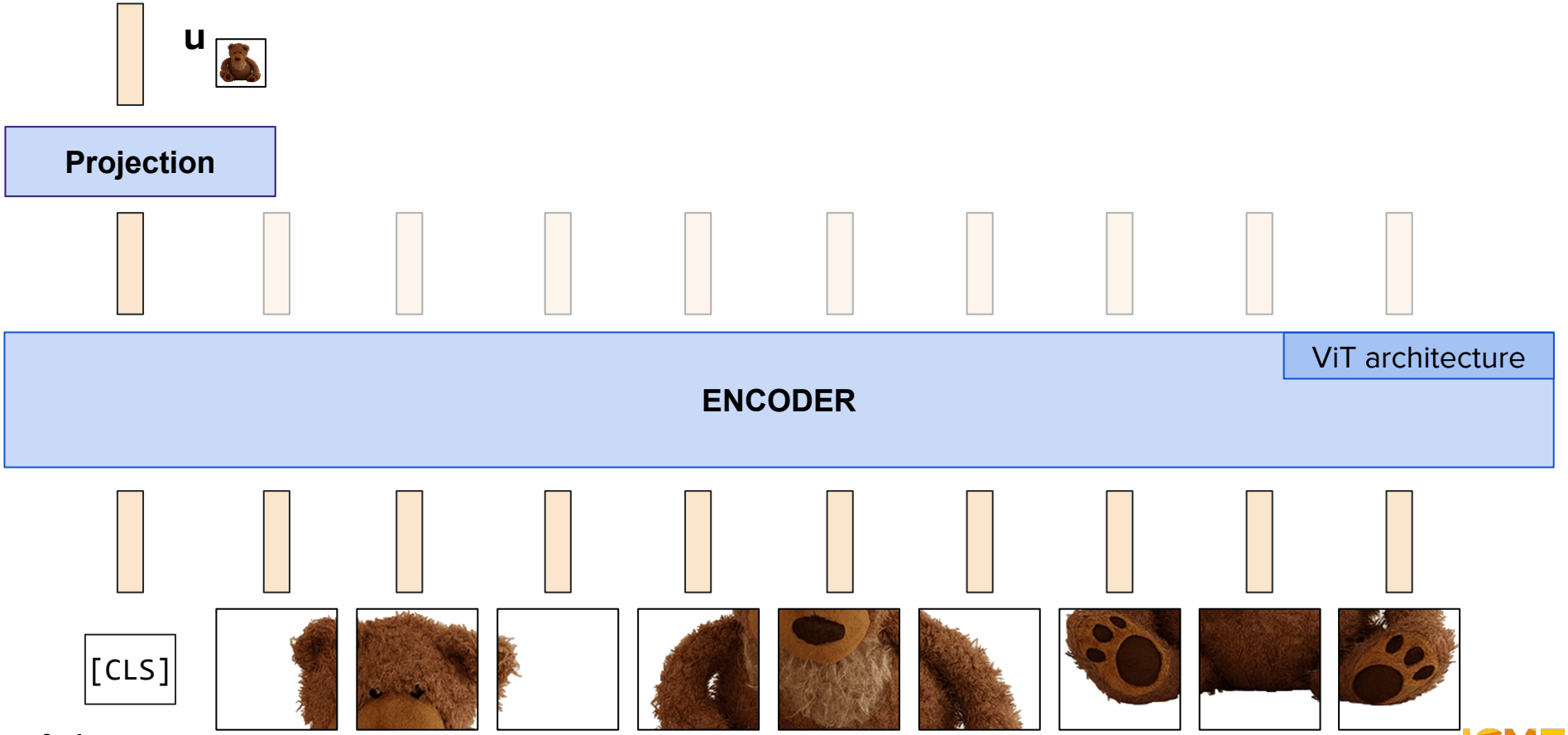
Contrastive learning

	teddy bear	water polo ball
	close	far
	far	close

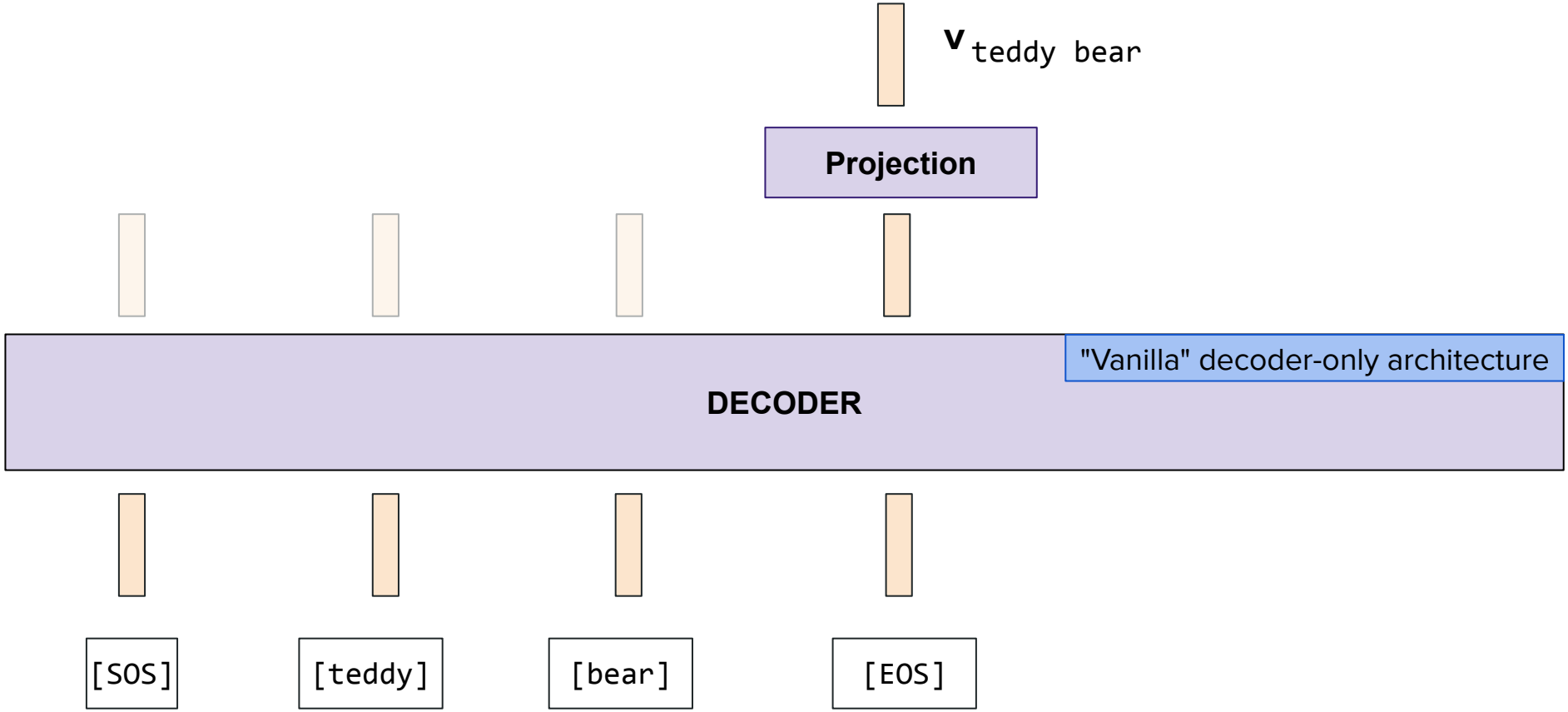
Contrastive learning

	teddy bear	water polo ball
	high similarity	low similarity
	low similarity	high similarity

How to find a same space?



How to find a same space?



Loss formulation?

Notation.

- Similarity: $\mathbf{s}_{,\mathtt{text}} = \dfrac{\mathbf{u}^T \bullet \mathbf{v}_{\mathtt{text}}}{||\mathbf{u}|| \bullet ||\mathbf{v}_{\mathtt{text}}||}$

Loss formulation?

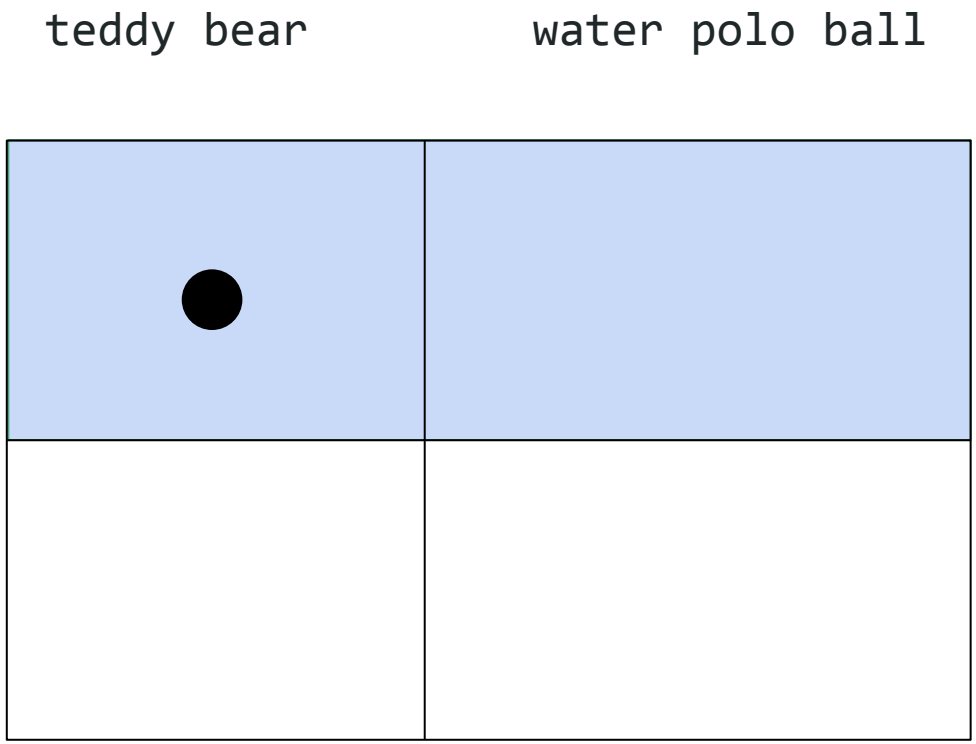
Notation.

- Similarity: $s_{\text{,text}} = \dfrac{\mathbf{u}^T \cdot \mathbf{v}_{\text{text}}}{\|\mathbf{u}\| \cdot \|\mathbf{v}_{\text{text}}\|}$

- Probability that image " " matches text "teddy bear":

$$p_{\boxed{\text{🧸}}} \longrightarrow \texttt{teddy bear} = \frac{\exp(s_{\boxed{\text{🧸}},\texttt{teddy bear}})}{\exp(s_{\boxed{\text{🧸}},\texttt{teddy bear}}) + \exp(s_{\boxed{\text{🧸}},\texttt{water polo ball}})}$$

Contrastive learning



Loss formulation?

Notation.

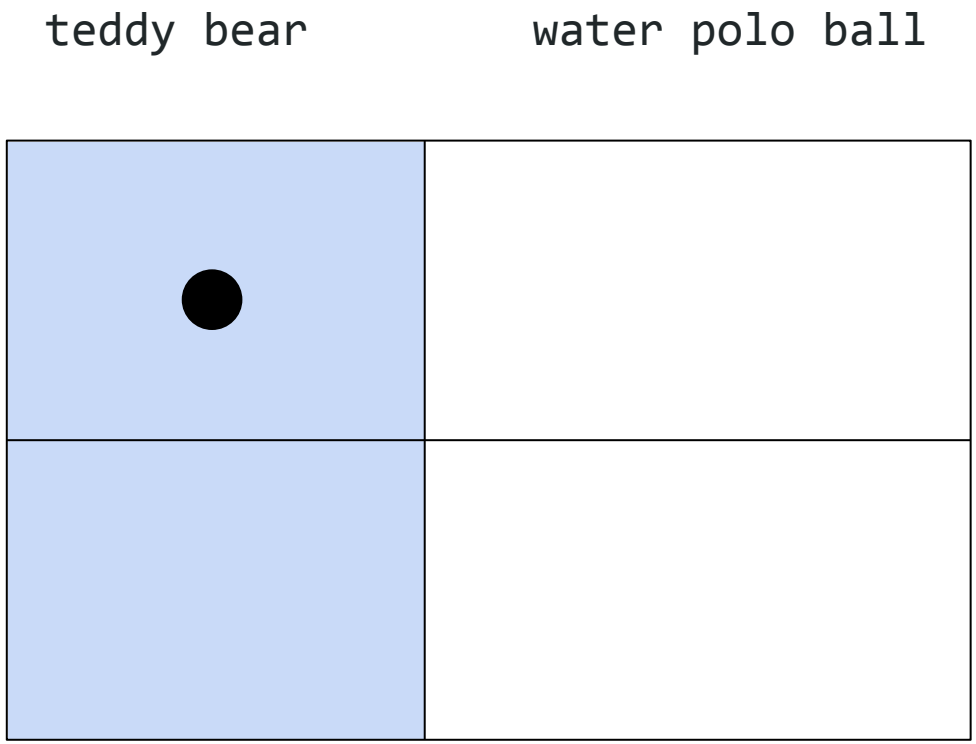
- Similarity: $s_{,\text{text}} = \dfrac{\mathbf{u}^T \cdot \mathbf{v}_{\text{text}}}{\|\mathbf{u}\| \cdot \|\mathbf{v}_{\text{text}}\|}$

- Probability that image "" matches text "teddy bear":

- Probability that text "teddy bear" matches image " ":

$$p_{\text{teddy bear} \rightarrow \text{ }} = \frac{\exp(s_{\text{ },\text{teddy bear}})}{\exp(s_{\text{ },\text{teddy bear}}) + \exp(s_{\text{ },\text{teddy bear}})}$$

Contrastive learning



Loss formulation

Each term.

$$\mathbf{L}_{\longrightarrow \texttt{text}} = -\frac{1}{2}\left(\log \mathbf{p}_{\boxed{}\,\longrightarrow \texttt{teddy bear}} + \log \mathbf{p}_{\boxed{}\,\longrightarrow \texttt{water polo ball}}\right)$$

$$\mathbf{L}_{\texttt{text}\longrightarrow} = -\frac{1}{2}\left(\log \mathbf{p}_{\texttt{teddy bear}\,\longrightarrow\boxed{}} + \log \mathbf{p}_{\texttt{water polo ball}\,\longrightarrow\boxed{}}\right)$$

Loss formulation

Each term.

$$\mathbf{L}_{\longrightarrow \texttt{text}} = -\frac{1}{2}\left(\log \mathbf{p}_{\longrightarrow \texttt{teddy bear}} + \log \mathbf{p}_{\longrightarrow \texttt{water polo ball}}\right)$$

$$\mathbf{L}_{\texttt{text}\longrightarrow} = -\frac{1}{2}\left(\log \mathbf{p}_{\texttt{teddy bear}\longrightarrow} + \log \mathbf{p}_{\texttt{water polo ball}\longrightarrow}\right)$$

Total loss.

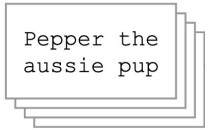
$$\mathbf{L} = \frac{1}{2}\left(\mathbf{L}_{\longrightarrow \texttt{text}} + \mathbf{L}_{\texttt{text}\longrightarrow}\right)$$

CLIP

CLIP = **C**ontrastive **L**anguage-**I**mage **P**retraining

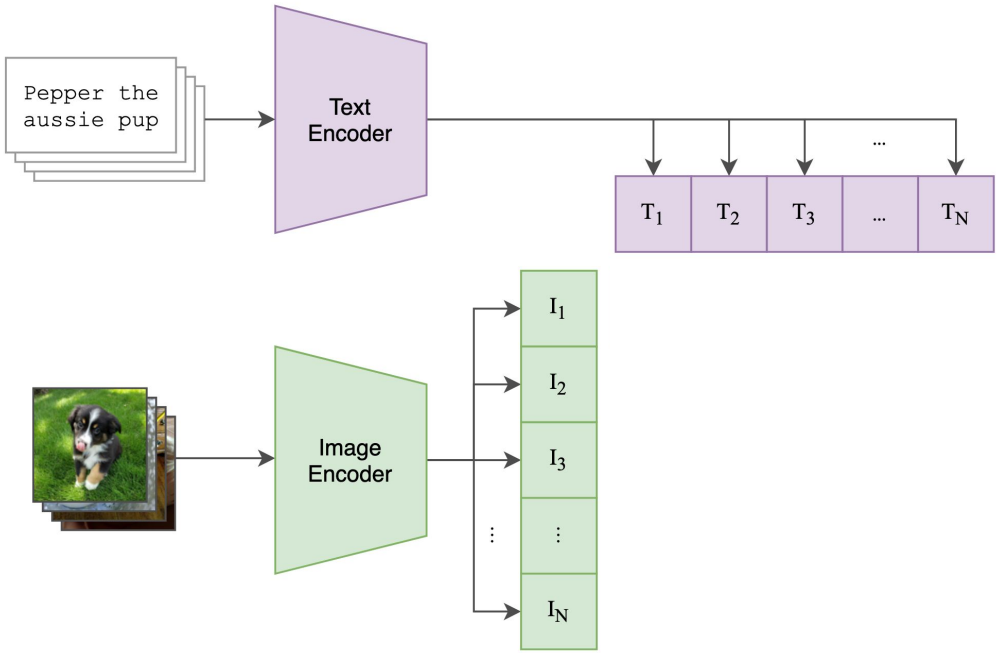
CLIP

CLIP = **C**ontrastive **L**anguage-**I**mage **P**retraining



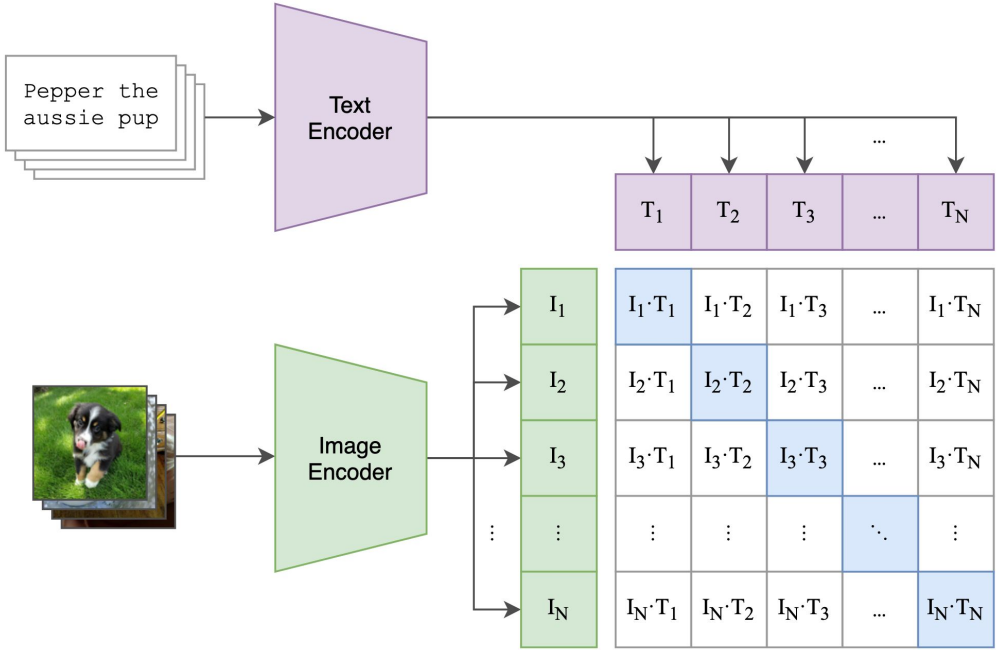
CLIP

CLIP = **C**ontrastive **L**anguage-**I**mage **P**retraining



CLIP

CLIP = **C**ontrastive **L**anguage-**I**mage **P**retraining



Learning Transferable Visual Models From Natural Language Supervision, Radford et al., 2021

CLIP loss

Each term.

$$\mathbf{L}_{\longrightarrow \texttt{text}} = -\frac{1}{N}\left(\log \mathbf{p}_{\boxed{}\,\longrightarrow \texttt{teddy bear}} + \cdots\right)$$

$$\mathbf{L}_{\texttt{text}\longrightarrow} = -\frac{1}{N}\left(\log \mathbf{p}_{\texttt{teddy bear}\,\longrightarrow\boxed{}} + \cdots\right)$$

CLIP loss

Each term.

$$\mathbf{L}_{\longrightarrow \texttt{text}} = -\frac{1}{N}\left(\log \mathbf{p}_{\longrightarrow \texttt{teddy bear}} + \cdots\right)$$

$$\mathbf{L}_{\texttt{text}\longrightarrow} = -\frac{1}{N}\left(\log \mathbf{p}_{\texttt{teddy bear}\longrightarrow} + \cdots\right)$$

Total loss.

$$\mathbf{L} = \frac{1}{2}\left(\mathbf{L}_{\longrightarrow \texttt{text}} + \mathbf{L}_{\texttt{text}\longrightarrow}\right)$$

CLIP characteristics

Training.

- Dataset: 400M of (image, caption) pairs
- Visual encoder: ViT or CNN
- Text encoder: Transformer-based
- Loss works because of the assumption of "in-batch negatives"

Learning Transferable Visual Models From Natural Language Supervision, Radford et al., 2021

CLIP characteristics

Training.

- Dataset: 400M of (image, caption) pairs
- Visual encoder: ViT or CNN
- Text encoder: Transformer-based
- Loss works because of the assumption of "in-batch negatives"

Results.

- Reaches ~76% on ImageNet with no supervised training
- Arguable possible leak from supervised sets, but still remarkable

Learning Transferable Visual Models From Natural Language Supervision, Radford et al., 2021

Discussion of CLIP

Computational challenges.

- Large similarity matrix required for a batch
- Global normalization
- Memory intensive

Discussion of CLIP

Computational challenges.

- Large similarity matrix required for a batch
- Global normalization
- Memory intensive

Other approach.

- Reframe 1 ↔ N softmax problem into a 1 ↔ 1 sigmoid one
- Popular work: **Sig**moid Loiss for **L**anguage **I**mage **P**re-Training

Sigmoid Loss for Language Image Pre-Training, Zhai et al., 2023.

Diffusion & Large Vision Models

From pixel to latent space

Latent diffusion models

Text representation

Image representation

Contrastive learning

Guidance

Today's lecture: Part 3

2 Condition

3 **Guided generation**

1 Representation

A teddy bear reading a book

Black box architecture

Today's lecture: **Multimodal guided generation**

Motivation

$$p(x_t|x_{t+1}) \longrightarrow p(x_t|x_{t+1}, \boxed{y})$$

condition

First idea: guide the generation with a classifier

Idea. Use a classifier's knowledge to feed image generation with information on the target class.

$$p_{\boxed{\theta},\boxed{\phi}}(x_t | x_{t+1}, y)$$

generation weights

classifier weights

Diffusion Models Beat GANs on Image Synthesis, Dhariwal et al., 2021.

Main derivation

We want to compute $p_{\theta,\phi}(x_t|x_{t+1}, y)$

Justification: Apply conditional Bayes' rule on $P(A|B, C)$ with

$A = x_t \quad B = y \quad C = x_{t+1}$

Same distribution as $p_\theta(x_t|x_{t+1})p_\phi(y|x_t)$

Diffusion Models Beat GANs on Image Synthesis, Dhariwal et al., 2021.

Main derivation

We want to compute $p_{\theta,\phi}(x_t|x_{t+1}, y)$

Justification: Apply conditional Bayes' rule on $P(A|B,C)$ with

$A = x_t \quad B = y \quad C = x_{t+1}$

Same distribution as $\boxed{p_\theta(x_t|x_{t+1})}p_\phi(y|x_t)$

Diffusion Models Beat GANs on Image Synthesis, Dhariwal et al., 2021.

Detour 1

Let's focus on $p_\theta(x_t|x_{t+1})$

Justification: Lecture 1's derivation of the DDPM generation process.

Same distribution as $\mathcal{N}(\mu_\theta, \sigma_{t+1} I)$

Justification: $\mathcal{N}(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$

$$\log p_\theta(x_t|x_{t+1}) = -\frac{||x_t - \mu_\theta||^2}{2\sigma_{t+1}^2} + \text{constant}$$

Diffusion Models Beat GANs on Image Synthesis, Dhariwal et al., 2021.

Main derivation

We want to compute $p_{\theta,\phi}(x_t|x_{t+1}, y)$

Justification: Apply conditional Bayes' rule on $P(A|B,C)$ with

$A = x_t \quad B = y \quad C = x_{t+1}$

Same distribution as $p_\theta(x_t|x_{t+1})\boxed{p_\phi(y|x_t)}$

Diffusion Models Beat GANs on Image Synthesis, Dhariwal et al., 2021.

Detour 2

Let's focus on $p_\phi(y|x_t)$

Justification: First-order Taylor approximation

$$f(\mu + (x_t - \mu)) \approx f(\mu) + (x_t - \mu)^T \nabla_{x_t} f(\mu)$$

$$\log p_\phi(y|x_t) \approx (x_t - \mu)^T \nabla_{x_t} \log p_\phi(y|\mu)$$ + constant

Diffusion Models Beat GANs on Image Synthesis, Dhariwal et al., 2021.

Main derivation

We want to compute $p_{\theta,\phi}(x_t|x_{t+1}, y)$

Justification: Apply conditional Bayes' rule on $P(A|B, C)$ with

$A = x_t \quad B = y \quad C = x_{t+1}$

Same distribution as $p_\theta(x_t|x_{t+1})p_\phi(y|x_t)$

Justification: Plug in expressions into $\log\left(p_\theta(x_t|x_{t+1})p_\phi(y|x_t)\right)$

and recognize probability density function of a normal distribution

$$x_t \sim \mathcal{N}(\mu_\theta + \boxed{\sigma_{t+1}^2 \nabla_{x_t} \log p_\phi(y|\mu_\theta)}, \sigma_{t+1}^2 I)$$

shifted mean

Diffusion Models Beat GANs on Image Synthesis, Dhariwal et al., 2021.

And what about the training loss?

$$x_t \sim \mathcal{N}(\boxed{\mu_\theta} + \boxed{\sigma_{t+1}^2} \nabla_{x_t} \log p_\phi(y|\boxed{\mu_\theta}), \boxed{\sigma_{t+1}^2} I)$$

Observe that the per-step generation rule only needs:

- The unconditioned generation model's predictions and parameters

Diffusion Models Beat GANs on Image Synthesis, Dhariwal et al., 2021.

And what about the training loss?

$$x_t \sim \mathcal{N}(\mu_\theta + \sigma_{t+1}^2 \boxed{\nabla_{x_t} \log p_\phi(y|\mu_\theta)}, \sigma_{t+1}^2 I)$$

Observe that the per-step generation rule only needs:

- The unconditioned generation model's predictions and parameters
- The classifier's score

Diffusion Models Beat GANs on Image Synthesis, Dhariwal et al., 2021.

Training

Generation model.

- No retraining needed of an unconditioned generation model.
- Purely a sampling technique.

Classifier.

- Needs to be trained with cross-entropy loss on noisy images
- Sample x_t and train $p_\phi(y|x_t)$ to predict the correct label

Giving more weight to the condition signal

$$x_t \sim \mathcal{N}(\mu_\theta + w\sigma_{t+1}^2 \nabla_{x_t} \log p_\phi(y|\mu_\theta), \sigma_{t+1}^2 I)$$

Empirically, we observe that we need $w > 1$ for good conditioned samples.

Limitations

New prerequisites.

- Classifier been trained on labeled data
- Need to operate on noised input: cannot use out-of-the-box classifier
- Distribution of data between two models to be aligned

Limitations

New prerequisites.

- Classifier been trained on labeled data
- Need to operate on noised input: cannot use out-of-the-box classifier
- Distribution of data between two models to be aligned

Computational considerations.

- Extra classifier pass needed at each step
- New gradient signal needs scaling tuning
- Taylor expansion introduces approximations

Refined goal

$$p_{\theta,\cancel{\phi}}(x_t | x_{t+1}, y)$$

generation weights **only**

Any way to get rid of the classifier part?

Motivation.

- Classifier on noisy data not found in the wild
- We want to avoid extra model calls at inference time

Classifier-Free Diffusion Guidance, Ho et al., 2022.

Any way to get rid of the classifier part?

Motivation.

- Classifier on noisy data not found in the wild
- We want to avoid extra model calls at inference time

Observation. Conditional and unconditional signal defines an implicit classifier!

$$p^{i}(y|x_t) \quad \text{proportional to} \quad \frac{p(x_t|y)}{p(x_t)}$$

Classifier-Free Diffusion Guidance, Ho et al., 2022.

Change of mindset

Idea. Compute both conditioned and unconditioned situations and infer classifier signal.

classifier-based

$$\epsilon_\theta(x_t) - w\sigma_t \nabla_{x_t} \log \boxed{p_\phi}(y|x_t)$$

$$\downarrow$$

classifier-free

$$\epsilon_\theta(x_t, \varnothing) + w \cdot (\boxed{\epsilon_\theta}(x_t, y) - \boxed{\epsilon_\theta}(x_t, \varnothing))$$

Classifier-Free Diffusion Guidance, Ho et al., 2022.

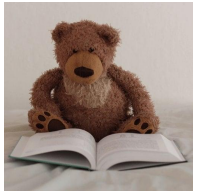
Training for classifier-free guidance

p_{uncond}

1. Sample:

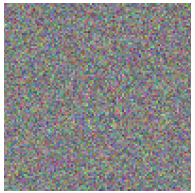
clean image

$x_0 \sim p_{\text{data}}$



noise

$\epsilon \sim \mathcal{N}(0, I)$



time step

$t \sim \mathcal{U}(0, T)$

noised image

$$x_t = \alpha_t x_0 + \sigma_t \epsilon$$

2. Use x_t and t to **predict** ϵ via $\epsilon_\theta(x_t, \varnothing)$

3. Compute **loss** $\mathcal{L} = ||\epsilon_\theta(x_t, \varnothing) - \epsilon||^2$ and **backpropagate** through ϵ_θ

Classifier-Free Diffusion Guidance, Ho et al., 2022.

Training for classifier-free guidance

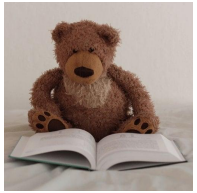
$1 - p_{\text{uncond}}$

1. Sample:

clean image

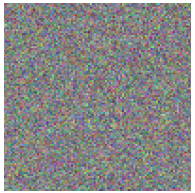
$x_0 \sim p_{\text{data}}$

y



noise

$\epsilon \sim \mathcal{N}(0, I)$



time step

$t \sim \mathcal{U}(0, T)$

noised image

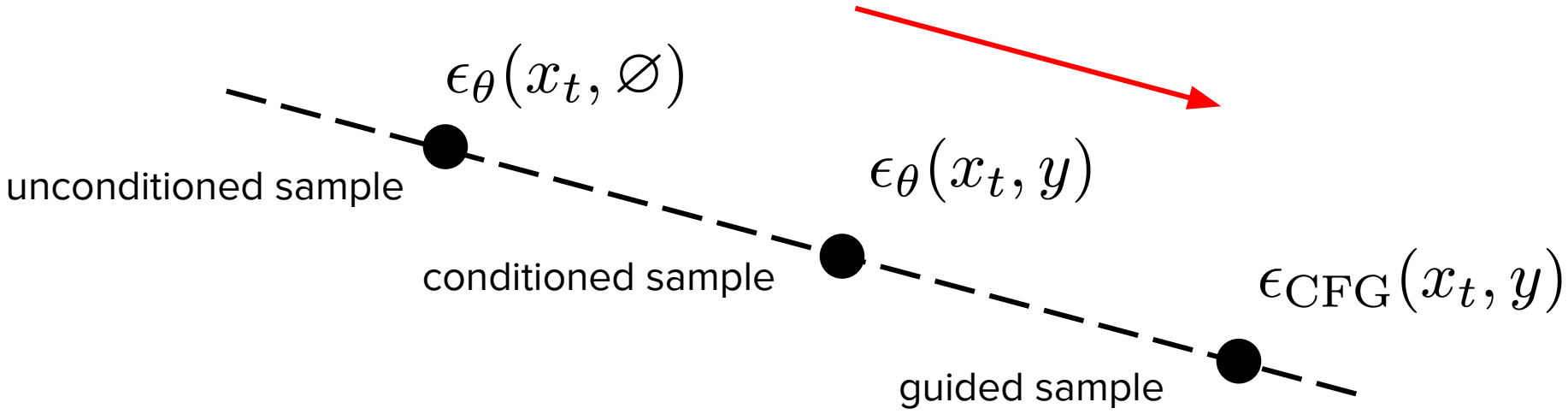
$$x_t = \alpha_t x_0 + \sigma_t \epsilon$$

2. Use x_t and t to **predict** ϵ via $\epsilon_\theta(x_t, y)$

3. Compute **loss** $\mathcal{L} = ||\epsilon_\theta(x_t, y) - \epsilon||^2$ and **backpropagate** through ϵ_θ

Classifier-Free Diffusion Guidance, Ho et al., 2022.

Sampling for classifier-free guidance



Classifier-Free Diffusion Guidance, Ho et al., 2022.

Practical considerations

Modeling.

- Flexibility of conditioning
- CLIP embeddings handy to represent conditioning signal
- Best results with $w > 1$ and $p_{\text{uncond}} = 10 - 20\%$

Classifier-Free Diffusion Guidance, Ho et al., 2022.

Practical considerations

Modeling.

- Flexibility of conditioning
- CLIP embeddings handy to represent conditioning signal
- Best results with $w > 1$ and $p_{\text{uncond}} = 10 - 20\%$

Computational.

- No more bespoke classifier needed...
- ...but still two calls per step!

Classifier-Free Diffusion Guidance, Ho et al., 2022.

Thank you for your attention!