Deep Learning (BEV033DLE)

Lecture 12: Learning Representations II

Czech Technical University in Prague

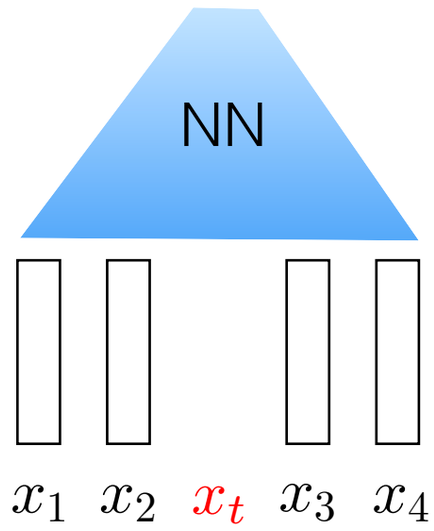
- Representation Learning with Surrogate Labels
  - Instance Classification
  - CLIP /SigLIP
- Autoencoders
  - Plain Autoencoders
  - Masked Autoencoders
  - Variational Autoencoders

Recap

- Representation learning for text: predict word (a discrete token) from context

Word vectors:

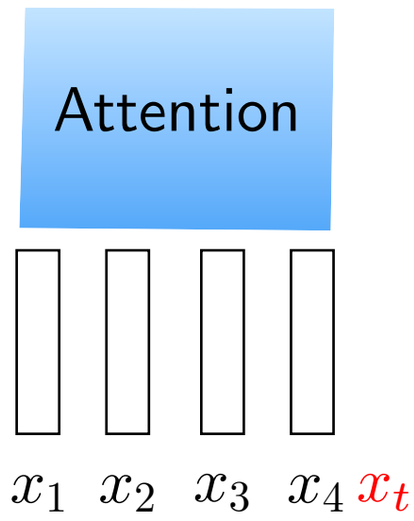
Categorical $p(x_t|x_C)$



- Can generate x_t from x_C
- Cannot generate text
- Useful embeddings

LLM:

Categorical $p(x_t|x_C)$



- Autoregressive generative model:

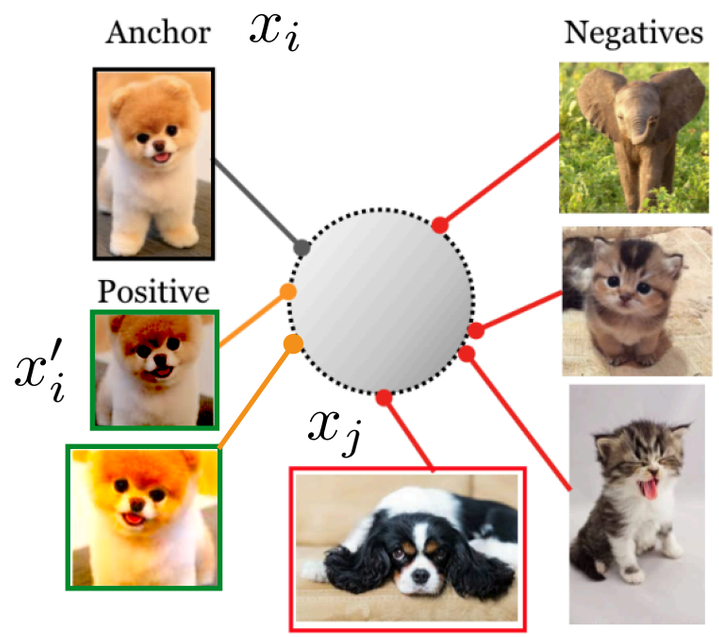
  $p(x) = p(x_0)p(x_1|x_0)p(x_2|x_0, x_1)\ldots p(x_t|x_0, \ldots x_{t-1})$

  - need to handle variable context $x_1, \ldots x_{t-1}$
  - causality for efficiency

- How can we do something like that for images?

Surrogate Categorical Prediction: Instance Classification

- Training data: $x_1 \dots x_N$
  - *Anchor*: x_i
  - *Positive*: $x' = T(x_i)$ – random transform
- Model as classification:
  - As many classes as there are instances (data points)
  - Score of instance i: $s_i = \phi(x_i)^\mathsf{T} \phi(x')$
  - $p(y{=}i|x') = \frac{e^{s_i}}{\sum_j e^{s_j}}$
  - Learning formulation: likelihood of classifying corectly
  - Large sum in the denominator $\rightarrow$ common solution is to restrict to a min-batch
- Properties:
  - Ensures instances can be discriminated
  - Enforces invariance to transformations

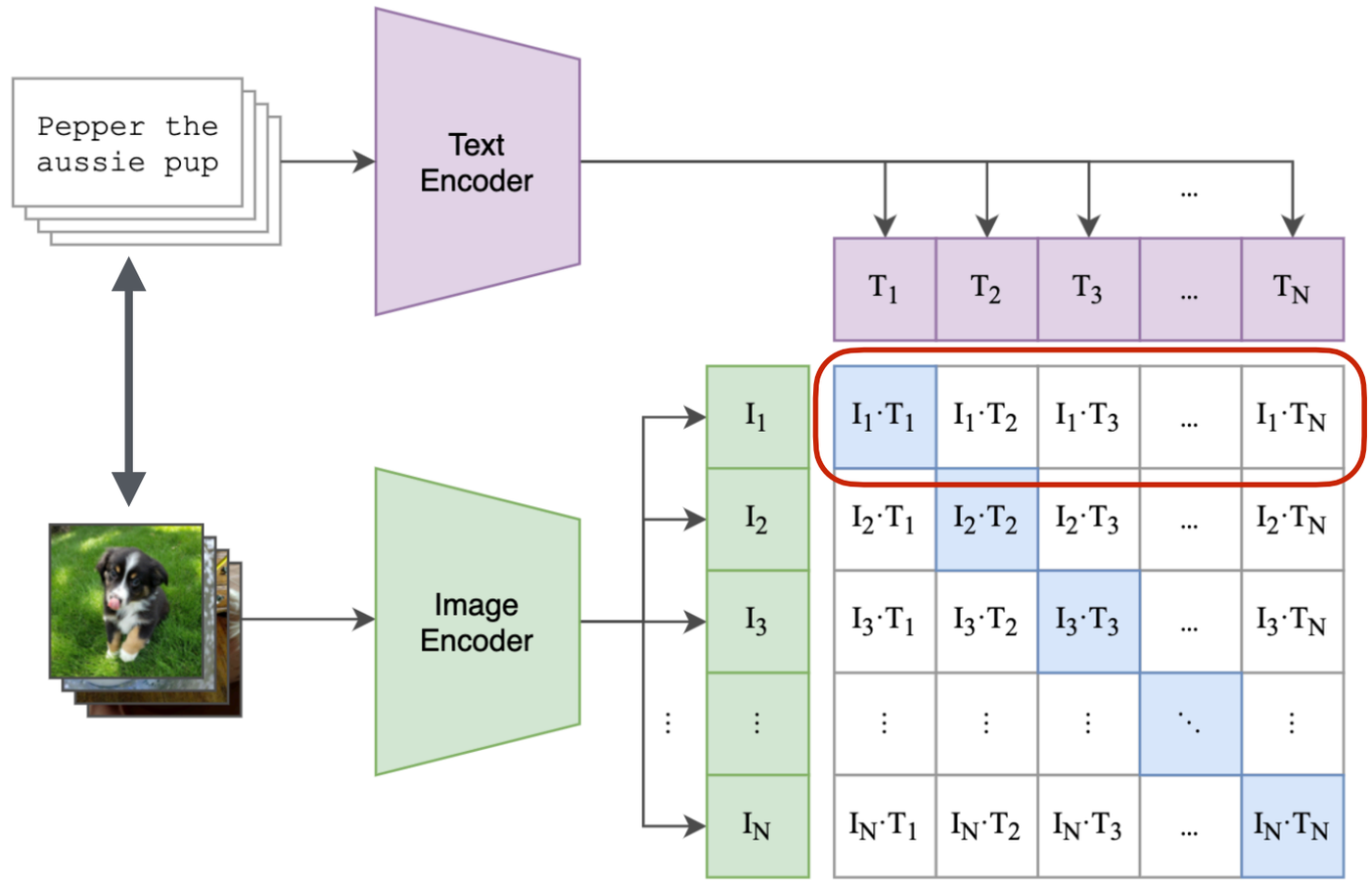


- Applications:
  - Transfer learning / supervised fine-tuning (detection / segmentation)
  - Image retrieval / similarity search / clustering

Surrogate Categorical Prediction: CLIP

Contrastive Language-Image Pre-training

- Training both:
  - text encoder
  - image encoder

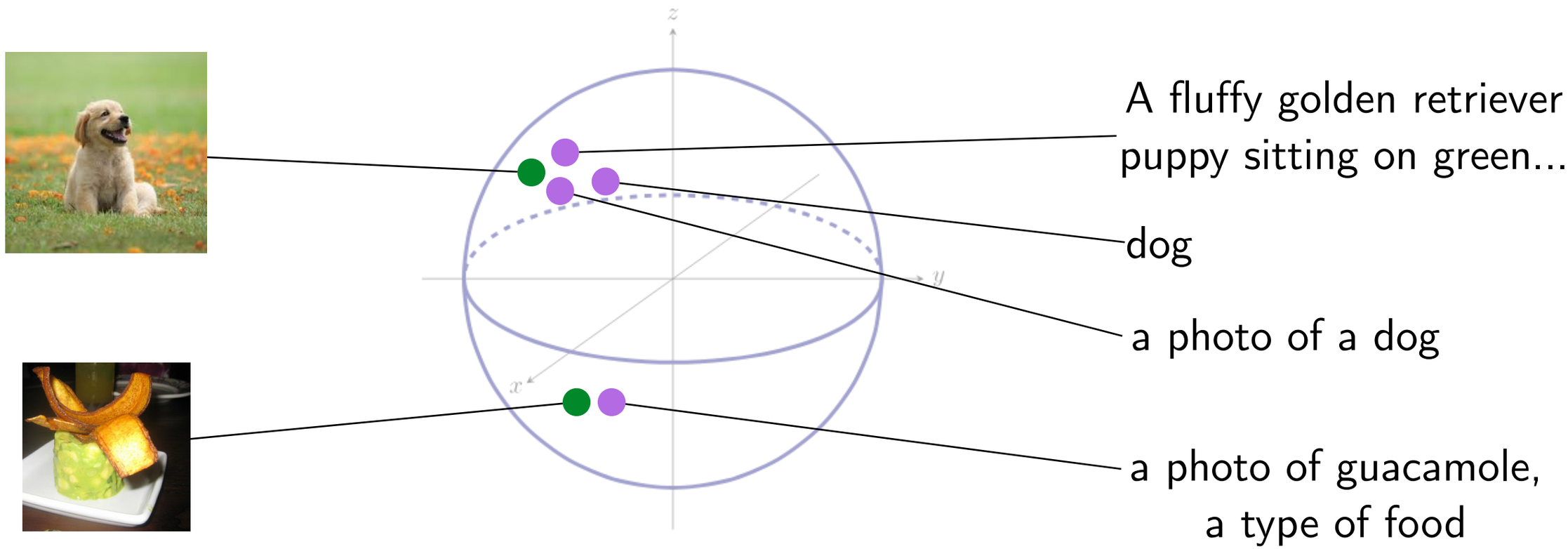


- In each batch of image-text pairs with embeddings I_i, T_j:
  - Predict which text corresponds to image i, model: $p_1(j|i) = \frac{e^{\langle I_i, T_j \rangle / \tau}}{\sum_{j'} e^{\langle I_i, T_{j'} \rangle / \tau}}$
  - Predict which image corresponds to text j, model: $p_2(i|j) = \frac{e^{\langle I_i, T_j \rangle / \tau}}{\sum_{i'} e^{\langle I_{i'}, T_j \rangle / \tau}}$
  - Learning: symmetric cross-entropy loss:

$$-\sum_i \Big(\log p_1(j|i) + \log p_2(i|j) \Big)$$

[Radford et al. 2021]

CLIP Representations



- Can be used to classify "zero-shot" if you have a finite set of text labels
- Great foundational model for fine-tuning (linear probing) to supervised tasks

Surrogate Categorical Prediction: SigLIP

- Using the pairwise loss formulation:

  Binary labels: $z_{ij} = 1$ if image I_i is paired with text T_j $(i = j)$ and $z_{ij} = -1$ otherwise

  Predictive model: $p_{ij}(z_{ij}{=}1|I_i, T_j) = \sigma(\langle I_i, T_j \rangle / \tau + b)$

  Loss: $-\sum_i \sum_j \log p(z_{ij}|I_i, T_j) = \sum_i \sum_j \log(1 + e^{z_{ij}\langle I_i, T_j \rangle / \tau + b})$

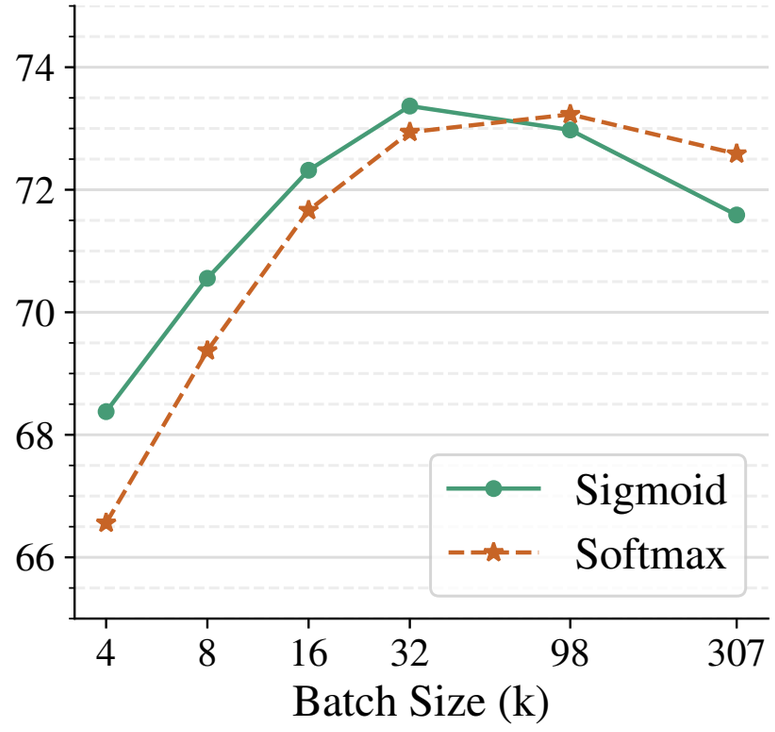
  b is an offset to compensate for imbalanced labels initially

  τ controls the scale of the score

- Properties:
  - Better suited for a distributed implementation
  - Does not require a large batch size, in theory

- Main takeaway: variants of triplet loss perform similarly

Zero-shot ImageNet performance

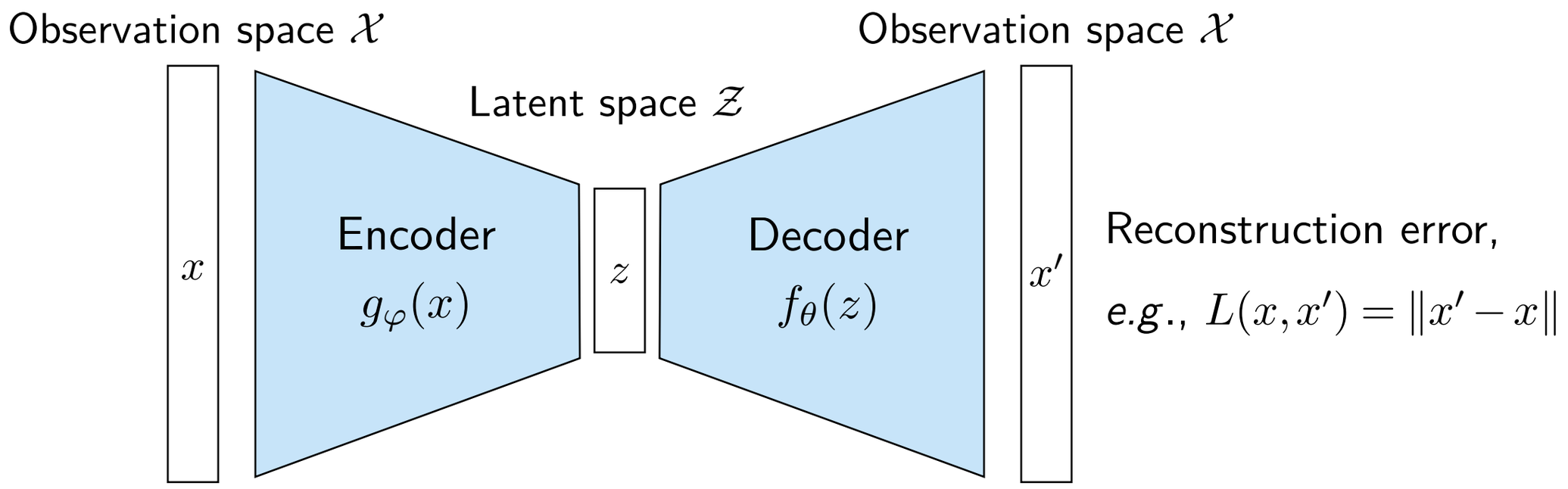


[Zhai et al. 2023: "Sigmoid Loss for Language Image Pre-Training"]

Autoencoders

Autoencoders

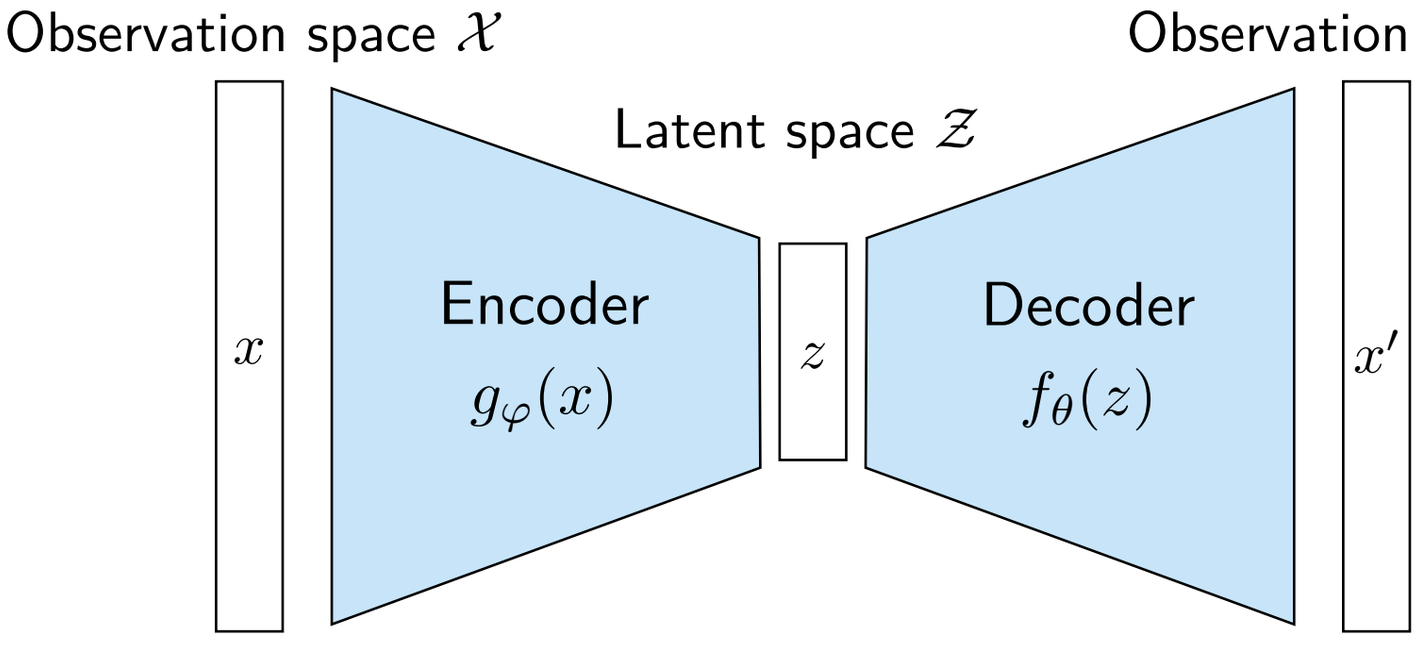
- Motivation / goals:
  - Nonlinear dimensionality reduction (data compression)
  - Learning of features / internal structure in the data in unsupervised way



- Learning: $\mathbb{E}_{x\sim p^*}\Big[L\big(x, x'\big)\Big] = \mathbb{E}_{x\sim p^*}\Big[L\big(x, f_\theta(g_\varphi(x))\big)\Big] \to \min\limits_{\theta,\varphi}$

- Usage:
  - Efficient representations, unsupervised pretraining, anomaly detection

Autoencoders



Reconstruction error,

e.g., $L(x, x') = \|x' - x\|$

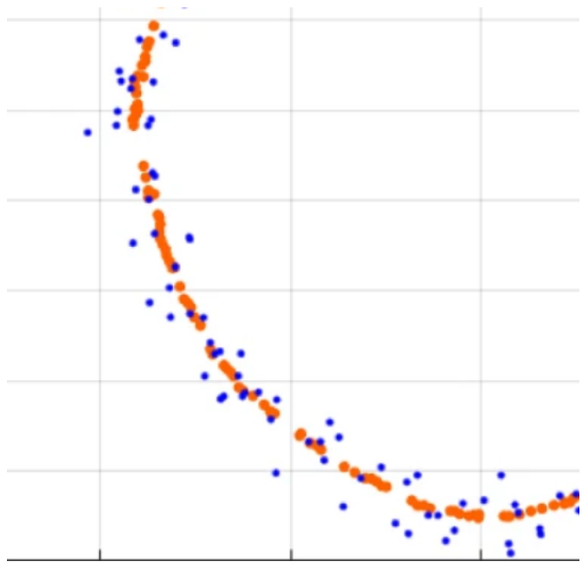
- Variants:

$$\min_{\theta,\varphi} \left(\mathbb{E}_{x\sim p^*} \left[\mathbb{E}_{\delta\sim\mathcal{N}(0,\varepsilon)} \left[L\big(x, f_\theta(g_\varphi(x+\delta))\big) \right] \right] \right. \text{ – denoising}$$

$$\left[\max_{\|\delta\|\le\varepsilon} \left(L\big(x, f_\theta(g_\varphi(x+\delta))\big) \right) \right] \text{ – adversarially robust}$$

$$\left[L\big(x, f_\theta(g_\varphi(x))\big) + \left\| \nabla_\varphi g_\varphi(x) \right\|_F^2 \right] \text{ – contracting}$$

noisy manifold



- Learn representations robust to small changes in the input
- Latent space becomes more smooth (similar codes reconstruct similar images)

Autoencoders

- Example of latent space continuity / smoothness:
  - Interpolation in the latent space between 3 and 8:
    (MNIST data, 2-layer MLPs, 16-dim latent space, training: 10 epochs)

DAE:

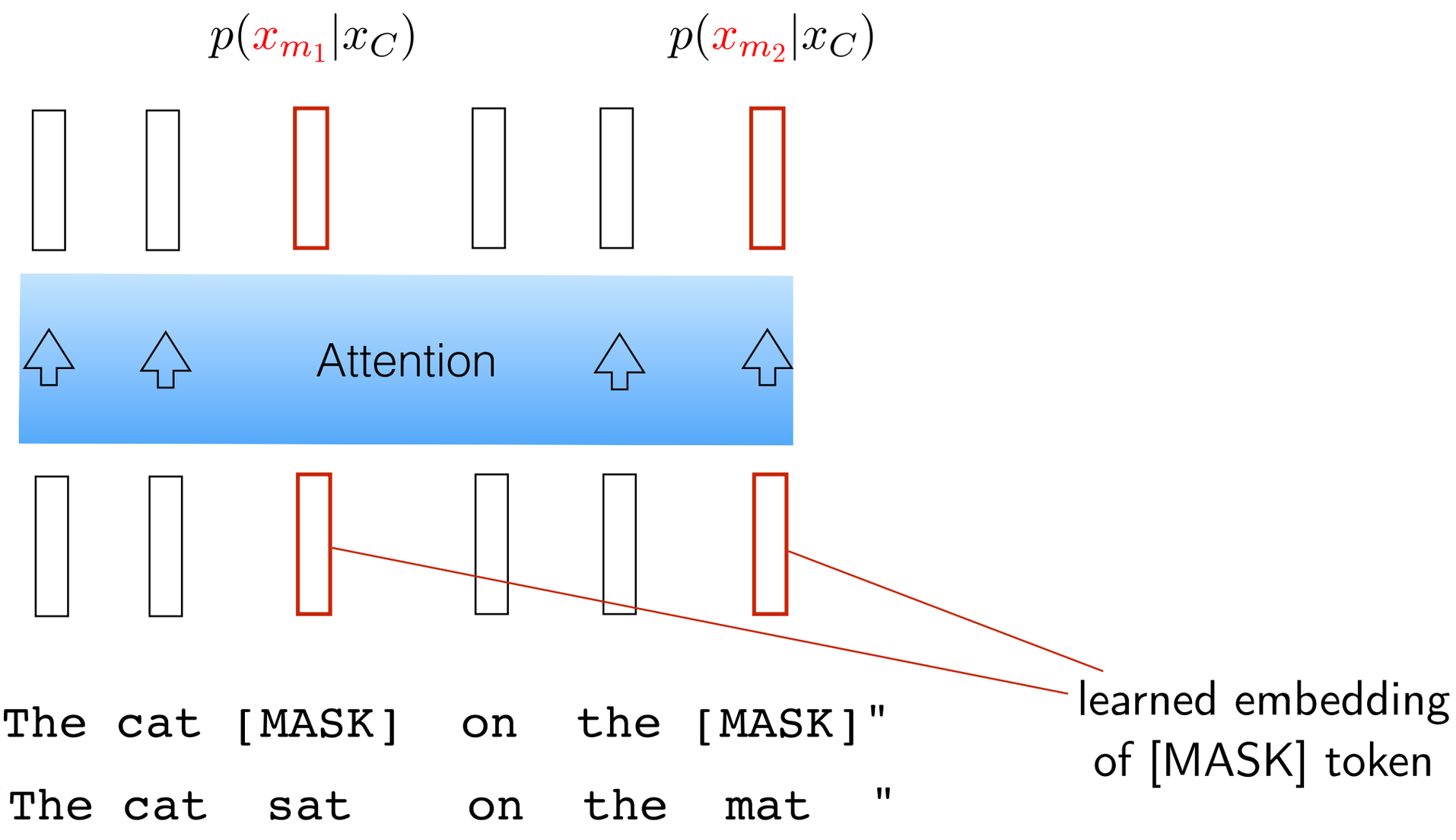
CAE:

- Limitations:
  - No principled way to sample new data
  - There may be clusters and gaps in the latent space
  - No probabilistic interpretation

Masked Autoencoders

Masked Language Modeling

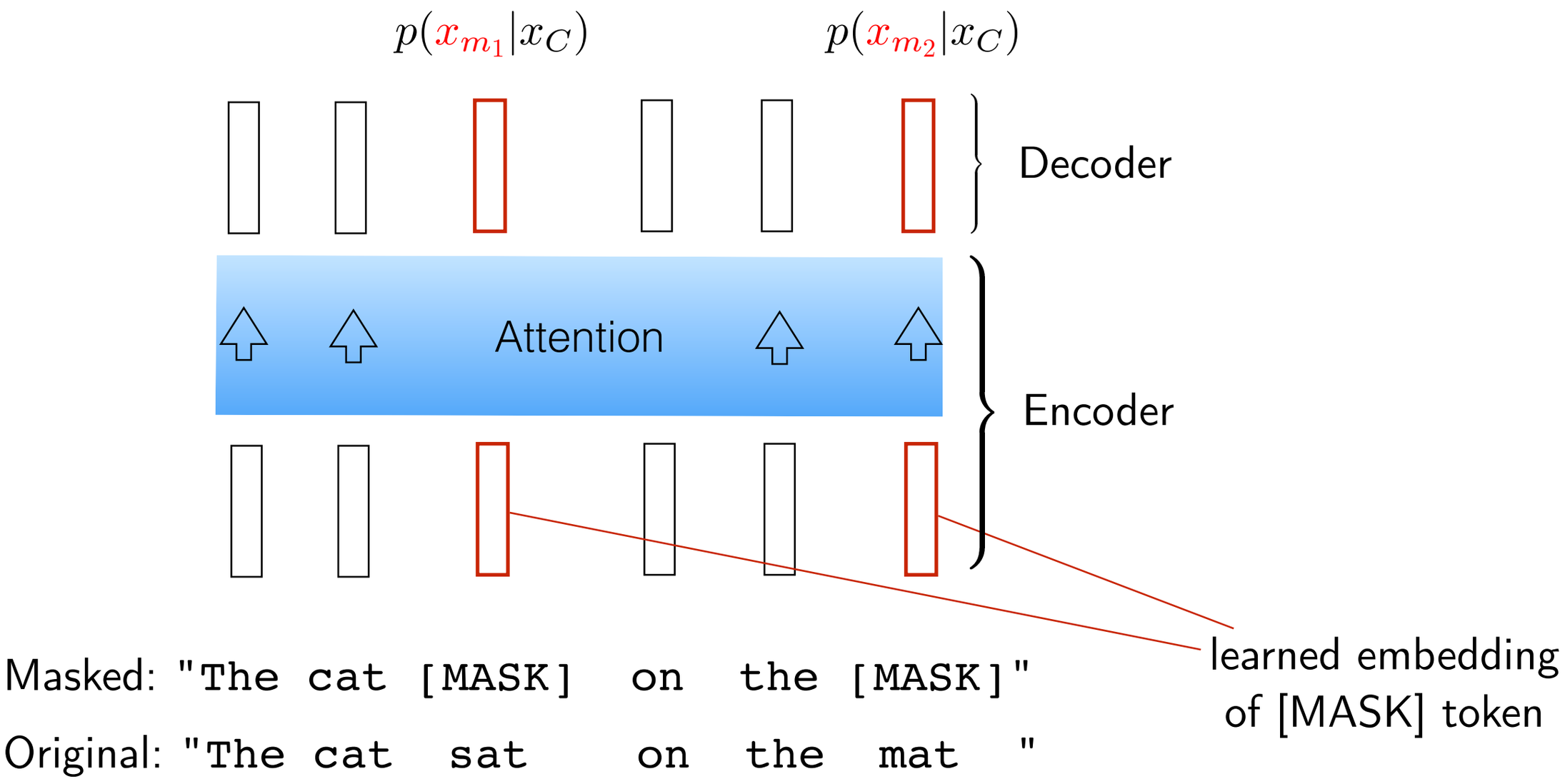
- BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding



- Note:
  - Predictions are modeled as conditionally independent (problem for ambiguous but strongly correlated cases, e.g. "Manchester United" vs "Real Madrid")
  - Cannot be used directly for complete text generation
  - Strong for learning representations

Masked Language Modeling

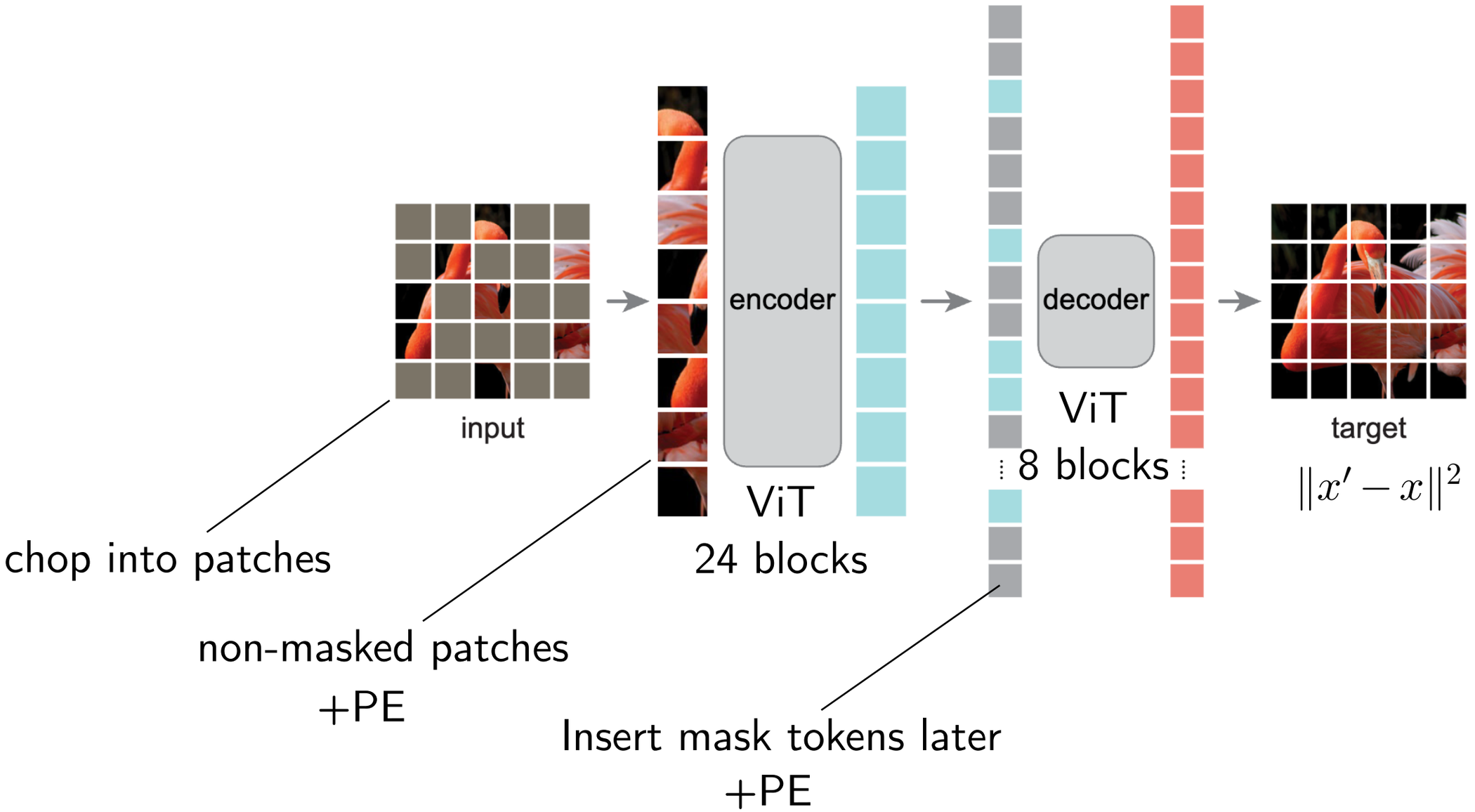
- BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding



- Note:
  - Predictions are modeled as conditionally independent (problem for ambiguous but strongly correlated cases, e.g. "Manchester United" vs "Real Madrid")
  - Cannot be used directly for complete text generation
  - Strong for learning representations

Masked (Image) Autoencoders

- Masked modeling idea extends to the image domain (and video, etc.)



[He et al. 2021: "Masked Autoencoders Are Scalable Vision Learners"]

Masked (Image) Autoencoders

- Reconstructions are blurry, but that's Ok
  - The main goal is to learn strong representations, the decoder gets thrown away
  - more complex decoders (diffusion, perceptual losses) might not necessarily improve representations

- Decoder needs to be small, or it could do all the job

[He et al. 2021: "Masked Autoencoders Are Scalable Vision Learners"]

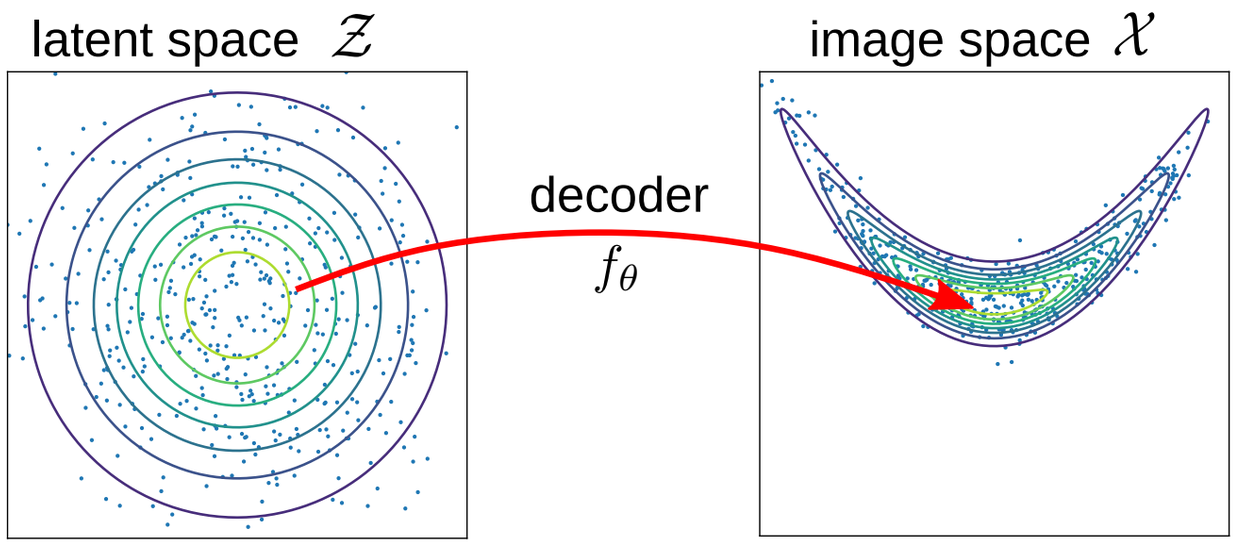
Variational Autoencoders

Generative Models

Generative models: Given training data $\mathcal{T} = \{x_i \,|\, i = 1, \ldots, N\}$ drawn i.i.d. from an unknown distribution $p^*(x)$, the goal is to learn a model that allows to generate random instances of x similar to $x \sim p^*(x)$.

Approach this task by using *latent variable models:*

- fix a latent noise space $\mathcal{Z}$ and a distribution $p(z)$ on it,
- design a neural network f_θ that maps $\mathcal{Z}$ to the feature space $\mathcal{X}$,
- learn its parameters θ so that the resulting distribution $p_\theta(x)$ "reproduces" the data distribution.

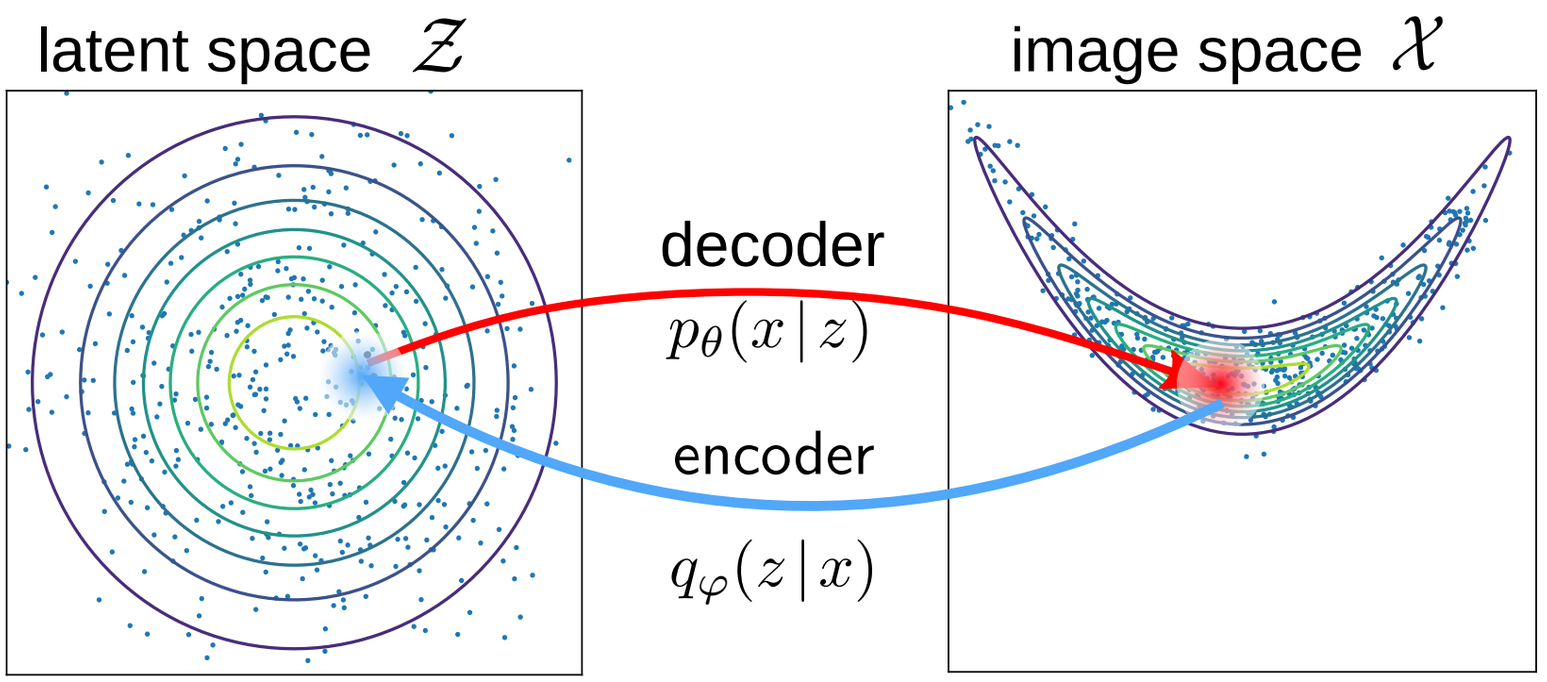


- E.g. maximum likelihood learning: $\log p_\theta(x) = \log p_{\mathcal{Z}}(f_\theta^{-1}(x)) + \log \det \frac{\mathrm{df}_\theta^{-1}(\mathrm{x})}{\mathrm{dx}}$

  — normalizing flow model, need invertible f_θ with tractable determinant Jacobian

(Gaussian) Variational Autoencoders

- Prior distribution $p(z)\colon \mathcal{N}(0, \mathbb{I})$
- **Decoder:** conditional distribution $p_\theta(x \,|\, z)\colon \mathcal{N}(f_\theta(z), \sigma^2 \mathbb{I})$
  - $f_\theta(z)$ is a (deep, convolutional) *decoder network* $\mathcal{Z} \to \mathcal{X}$.
- **Sampling:** $z \sim p(z)$, $x \sim p(x \,|\, z)$, induced marginal distribution $p_\theta(x) = \mathbb{E}_{z \sim p(z)}\big[p(x|z)\big]$



- **Encoder (inference model):** $q_\varphi(z|x)$ will be learned to approximate the decoder posterior $p_\theta(z|x)$ (i.e. probabilistic inverse)
- **Likelihood:** $\log p_\theta(x) = \mathbb{E}_{z \sim q_\varphi(z|x)}\Big[\log p_\theta(x|z)p(z) - \log q_\varphi(z|x)\Big]$ if $q_\varphi(z|x) = p_\theta(z|x)$

Variational Evidence Lower Bound (ELBO)

- Evidence

Likelihood: $\log p_\theta(x) = \int_z p(z) p_\theta(x|z) \to \max_\theta$ — intractable

- NELBO (Negative Evidence Lower Bound):

$$\underbrace{-\log p_\theta(x)}_{\text{NLL}} \leq \underbrace{-\log p_\theta(x)}_{\text{NLL}} + \underbrace{D_{KL}(q_\varphi(z|x) \| p_\theta(z|x))}_{\text{Encoder-Decoder Consistency}} \qquad (D_{KL} \geq 0)$$

$$= -\log p_\theta(x) + \sum_z q_\varphi(z|x) \log \frac{q_\varphi(z|x)}{p_\theta(z|x)} \qquad \text{(expand)}$$

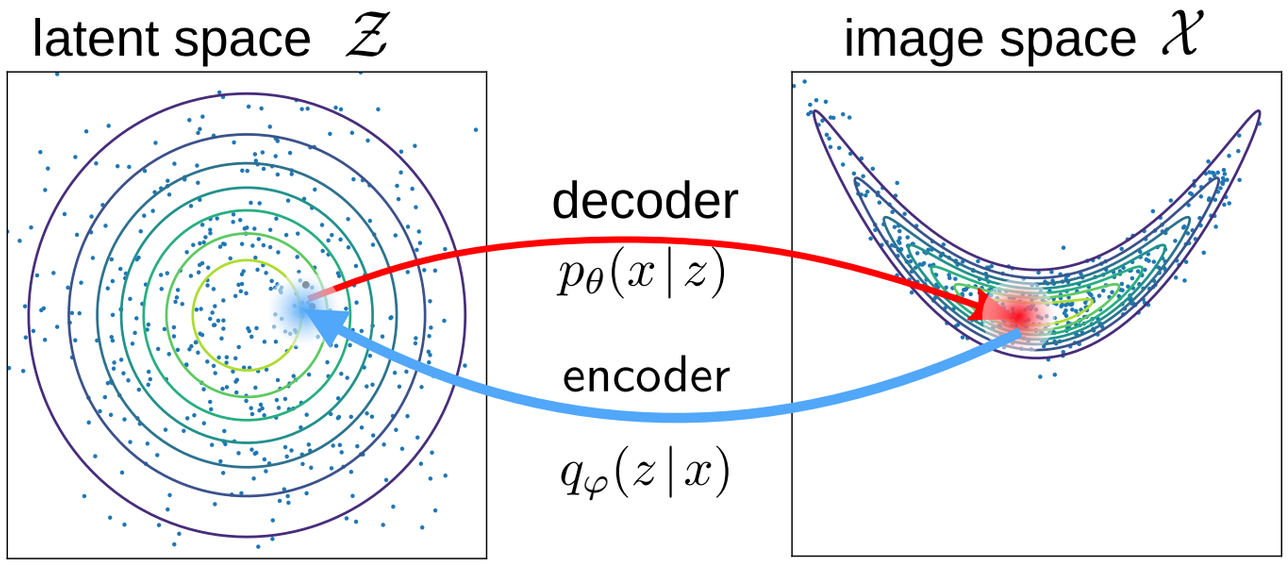
$$= \sum_z q_\varphi(z|x) \left(-\log p_\theta(x) + \log \frac{q_\varphi(z|x)}{p_\theta(z|x)} \right) \qquad (\textstyle\sum_z q_\varphi(z\,|\,x) = 1)$$

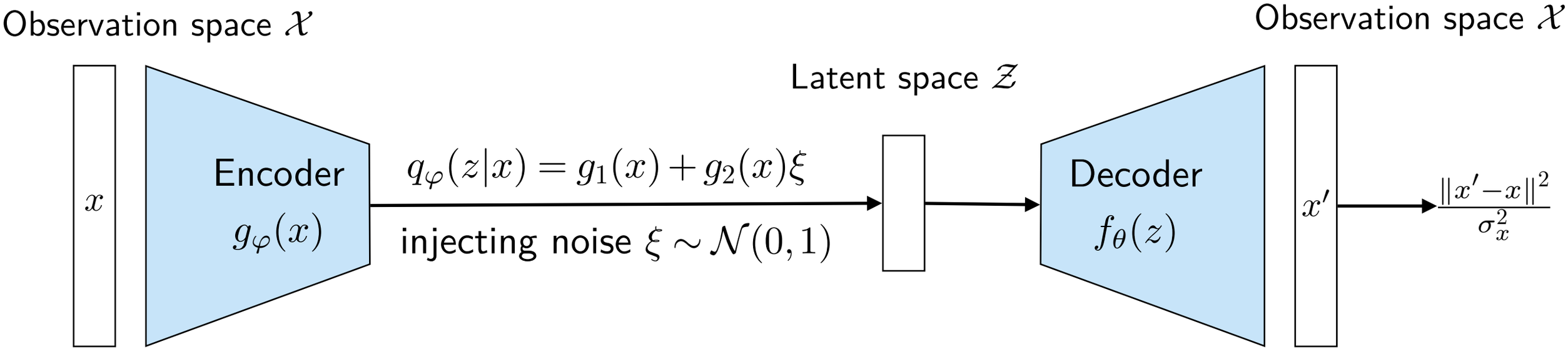
$$= \sum_z q_\varphi(z|x) \left(-\log \frac{p_\theta(x,z)}{q_\varphi(z|x)} \right) \qquad (\ p(z|x)p(x) = p(x,z)\ )$$

$$= -\underbrace{\mathbb{E}_{q_\varphi(z|x)}[\log p_\theta(x|z)]}_{\text{Reconstruction Loss}} + \underbrace{D_{KL}(q_\varphi(z|x) \| p(z))}_{\text{Encoder ``Balooning''}} \qquad (\ p(x,z) = p(x|z)p(z)\ )$$

- Tractable to optimize in θ and φ!

(Gaussian) Variational Autoencoder

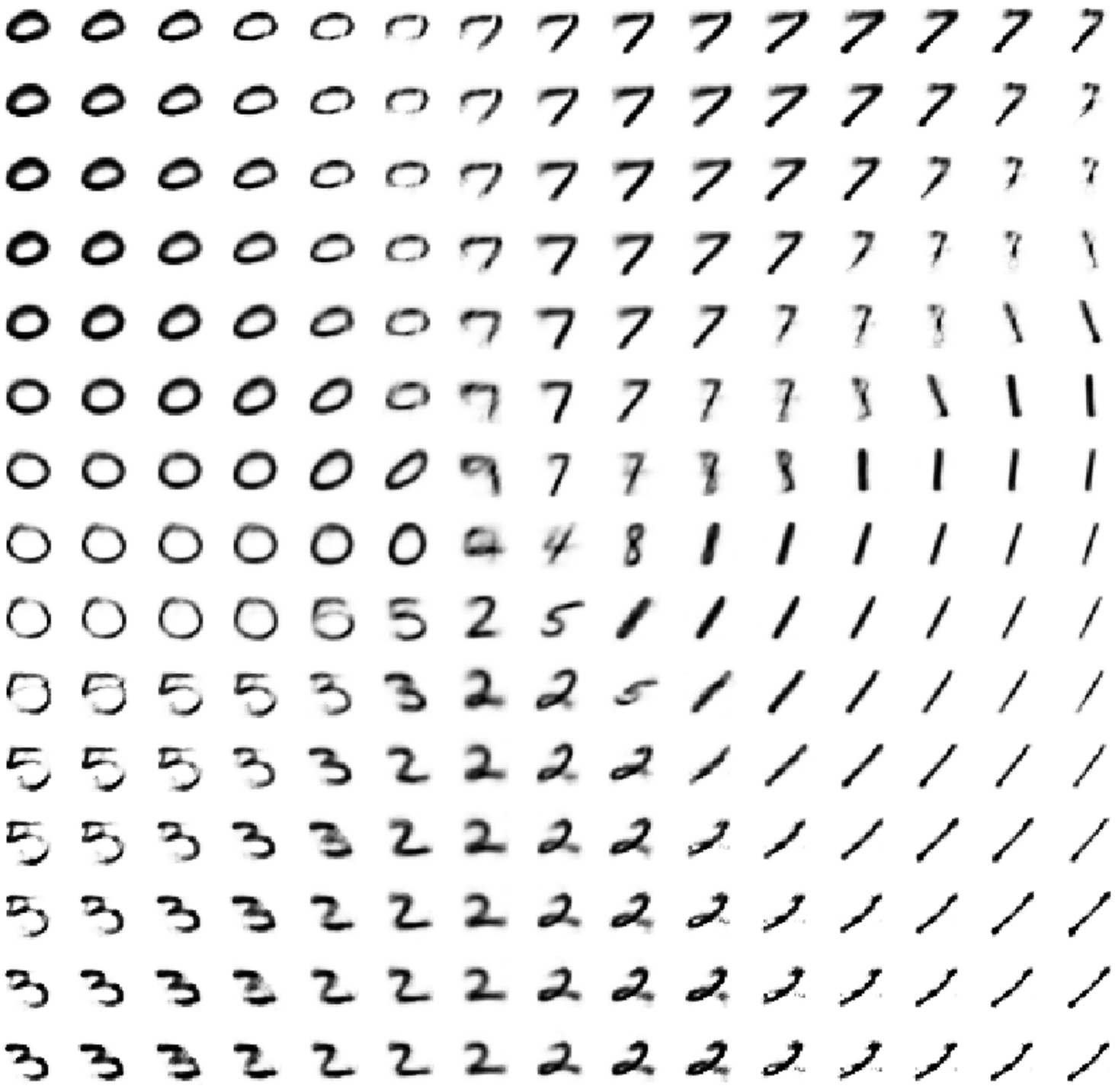




- Takeaway:
  - VAE objective in practice: reconstruction loss
    + regularization that the noise scale g_2 does not diminish. (c.f. denoising AE)
  - A solid learning principle: minimizing upper bound on NLL pushes NLL down
  - Can generate $z \sim p(z) \to x \sim p_\theta(x|z)$
  - Encoder $q_\phi(z|x)$ and posterior $p_\theta(z|x)$ learn to be close

Example: Latent Space Smoothness

- MNIST latent space bilinear interpolation



(*) Hierarchical Variational Autoencoders

Closing the gap between $L(\theta)$ and $L_B(\theta, \varphi)$:

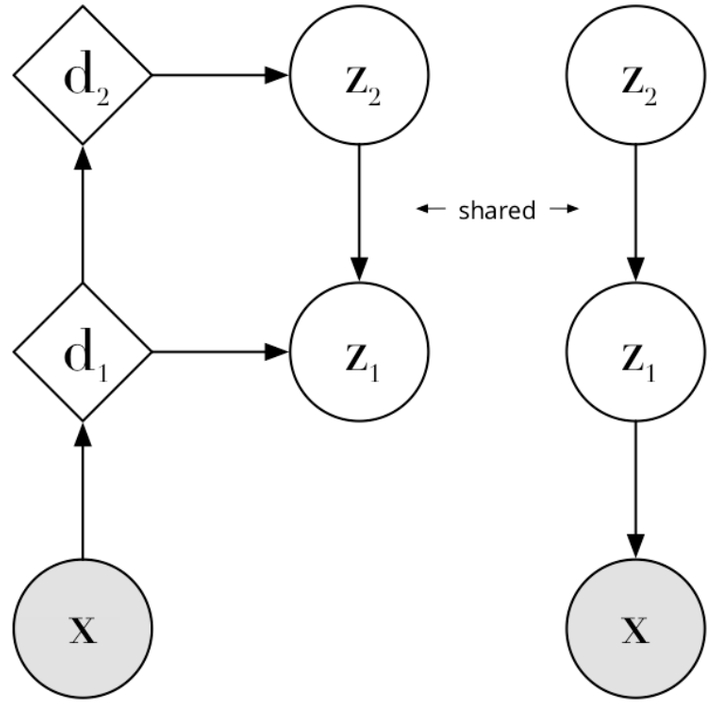
The latent state z consists of variable groups $z_1, \ldots, z_m$.

$$p_\theta(x, z) = p(z_m) \prod_{i=1}^{m-1} p_\theta(z_i \,|\, z_{>i})\, p_\theta(x \,|\, z); \quad q_\varphi(z \,|\, x) = q_\varphi(z_m \,|\, x) \prod_{i=1}^{m-1} q_\varphi(z_i \,|\, z_{>i}, x).$$

The encoder shares parameters with the decoder, by assuming

$$q_{\theta,\varphi}(z_i \,|\, z_{>i}, x) \propto p_\theta(z_i \,|\, z_{>i}) d_i(z_i, x, \varphi),$$

where the functions d_i are hidden layer outputs of a deterministic encoder network whose forward direction is reverse to the factorisation order of the model.



(*) Hierarchical Variational Autoencoders

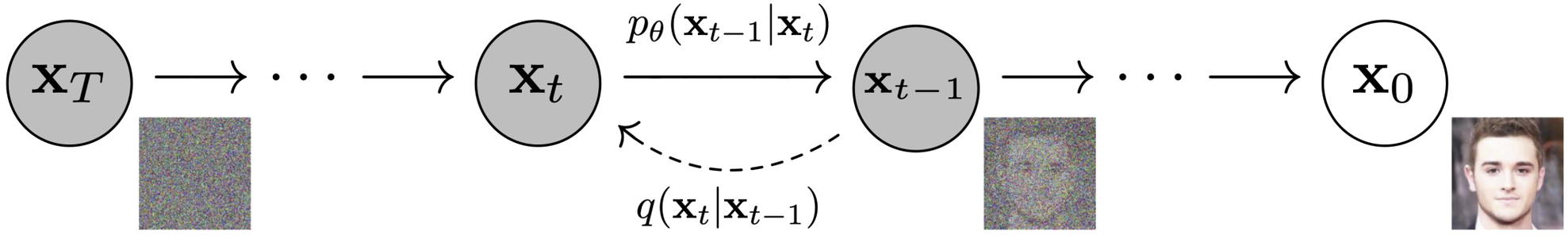
Hierarchical VAEs can be learned by maximising ELBO.

For instance

$$D_{KL}(q_\varphi(z\,|\,x)\,\|\,p(z)) = D_{KL}(q_\varphi(z_m\,|\,x)\,\|\,p(z_m)) + \int dz_m\, q_\varphi(z_m\,|\,x) D_{KL}(q_\varphi(z_{m-1}\,|\,z_m, x)\,\|\,p_\theta(z_{m-1}\,|\,z_m)) + \dots$$

A. Vahdat et al., NeurIPS 2020: A Deep Hierarchical VAE trained on CelebA data.

VAE Elements in Diffusion Models



Diffusion models are homogeneous hierarchical VAEs with the latent spaces same as image: $\mathcal{Z}_i = \mathcal{X}$.

- The encoder $q(x_t, |x_{t-1}) = \mathcal{N}(x_t, \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$ is fixed and gradually adds Gaussian noise to the data. (diffusion forward process)
- The decoder is given by $p_\theta(x_{t-1}|x_t, \beta_t) \sim \mathcal{N}(x_{t-1}, \mu_\theta(x_t, t), \sigma_t^2 I)$ and is implemented by a deep network (typically a UNet). Its parameters θ are shared for all t.

- Same learning principle: Apply a chain of ELBOs
  - Conditionally independent Gaussian predictions are not restrictive any more (many small steps)
  - Great generative capabilities
  - The encoder learns nothing

Diffusion Models

J. Ho et al., NeurIPS 2020, Denoising diffusion probabilistic models