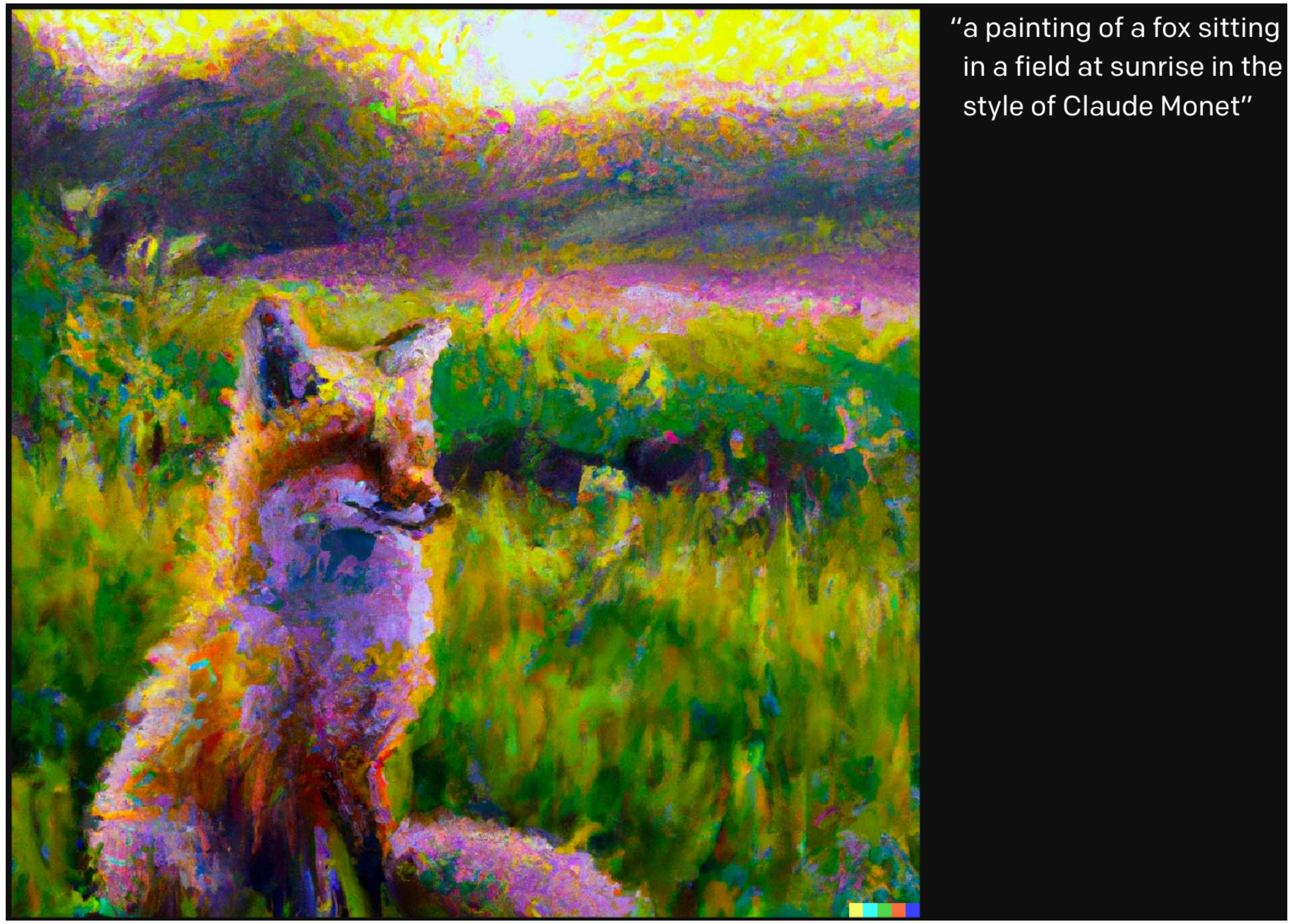
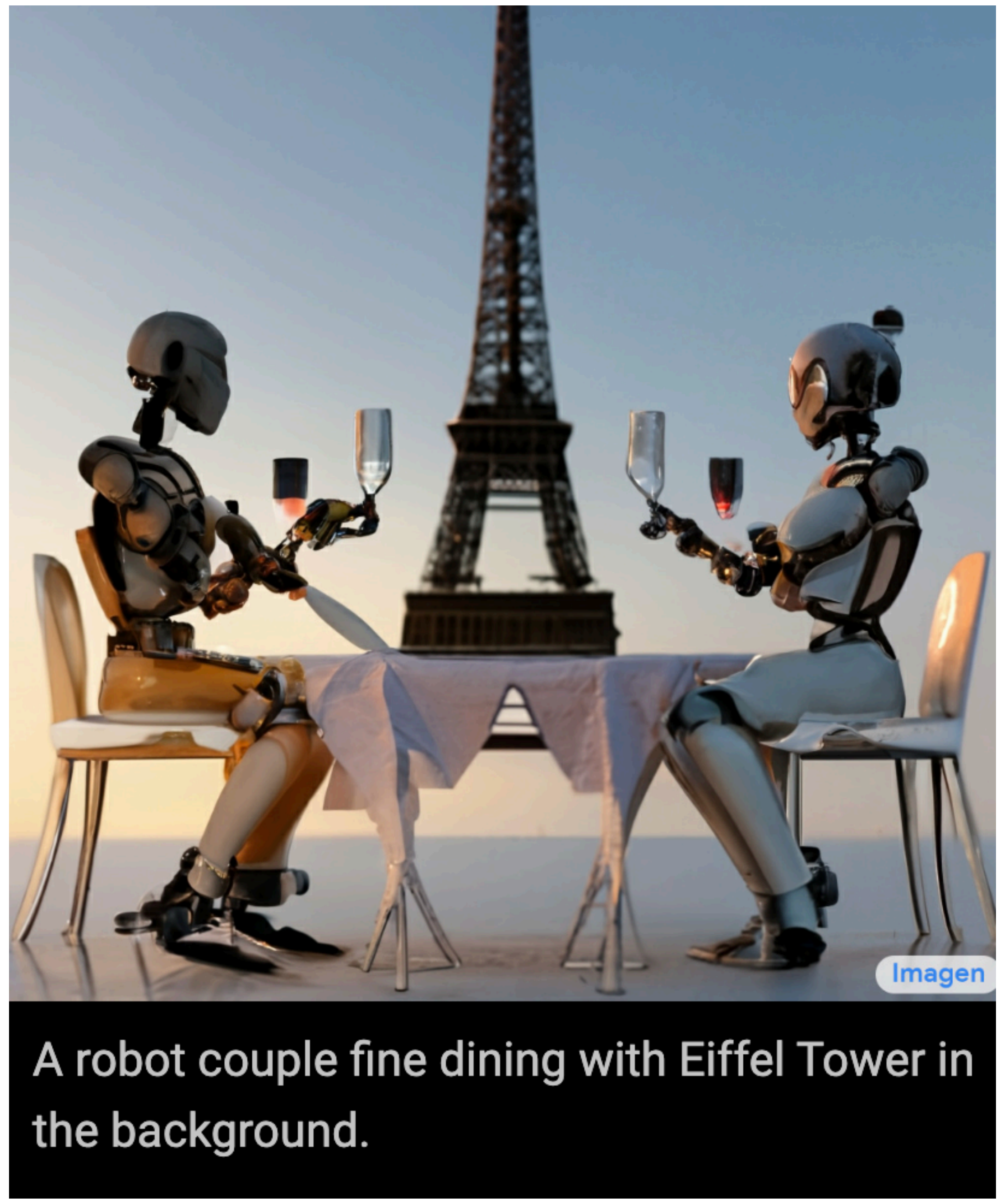
Generative Artificial Intelligence

JAKUB M. TOMCZAK
VRIJE UNIVERSITEIT AMSTERDAM

Recent breakthroughs





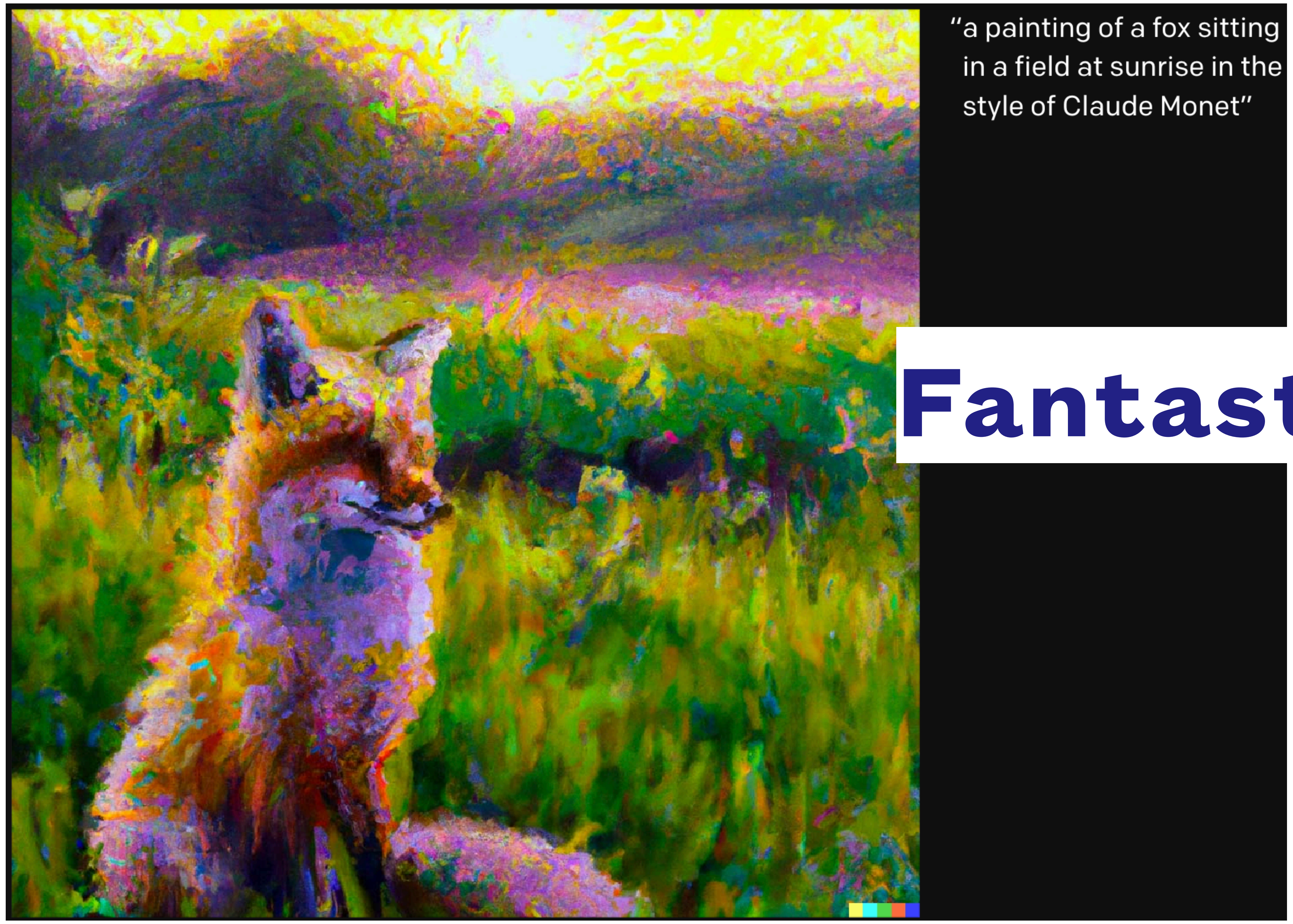
Dalle·2

Imagen

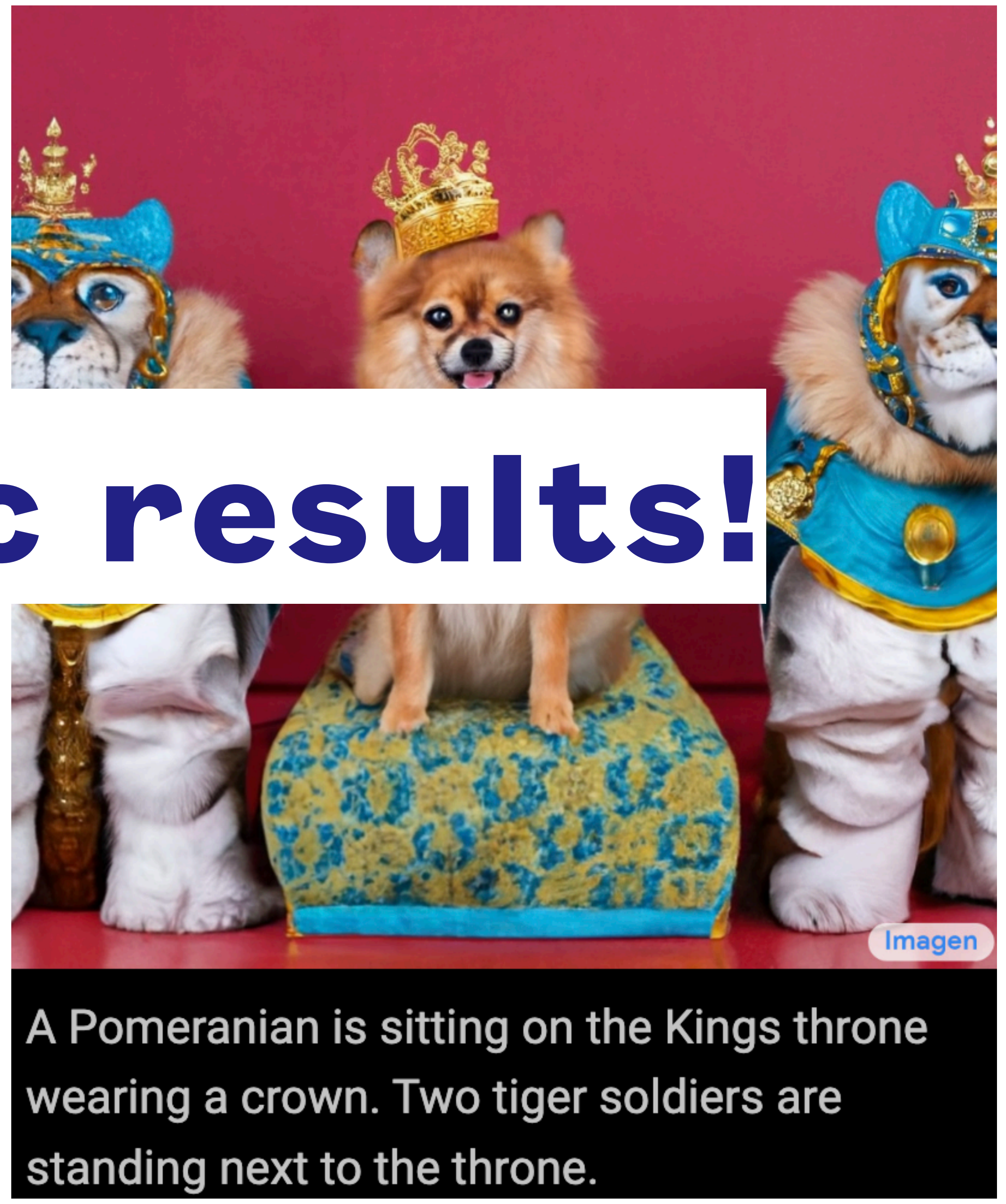
Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", **April 13, 2022**

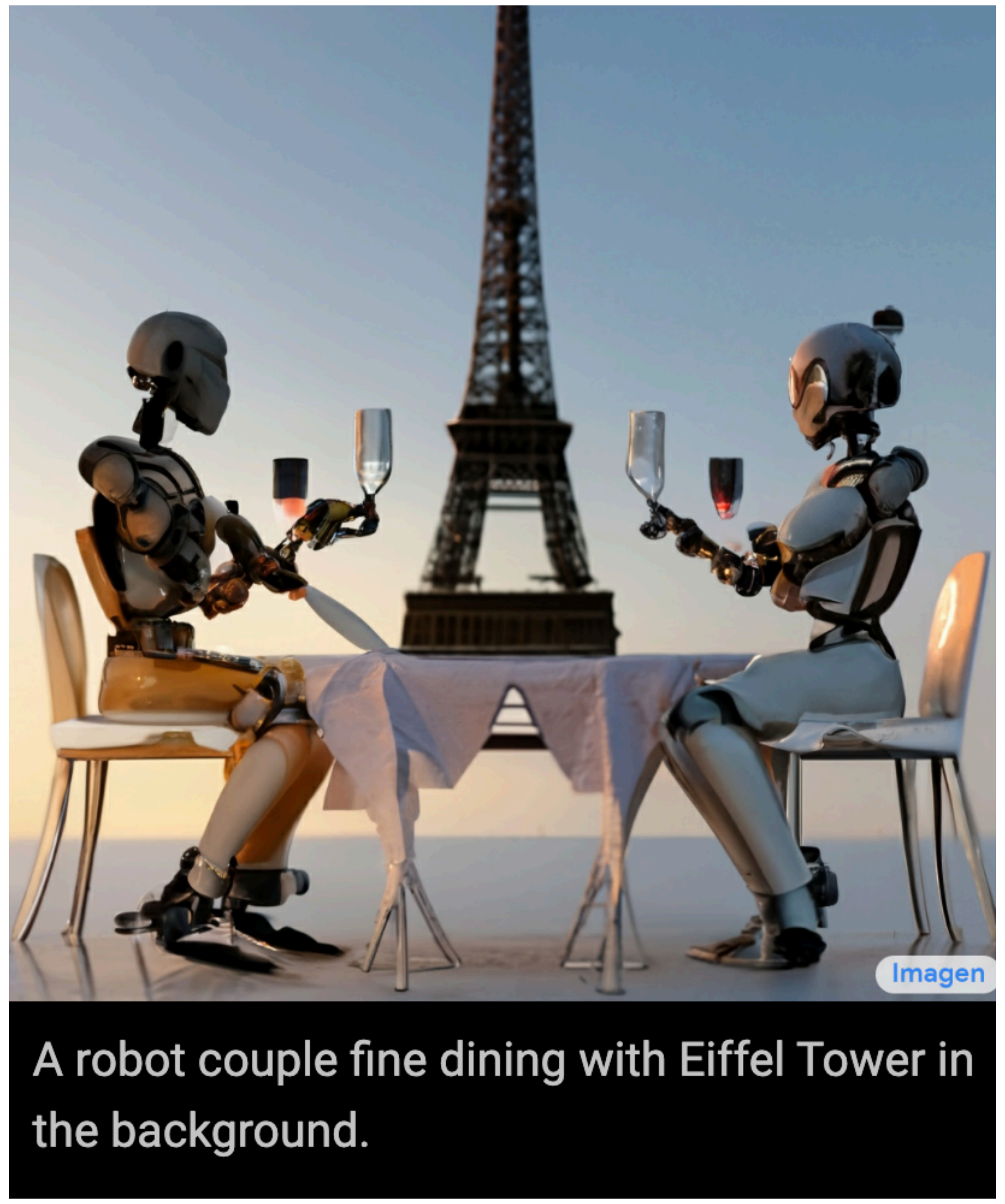
Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", **May 23, 2022**

Recent breakthroughs



Fantastic results!





Dalle·2

Imagen

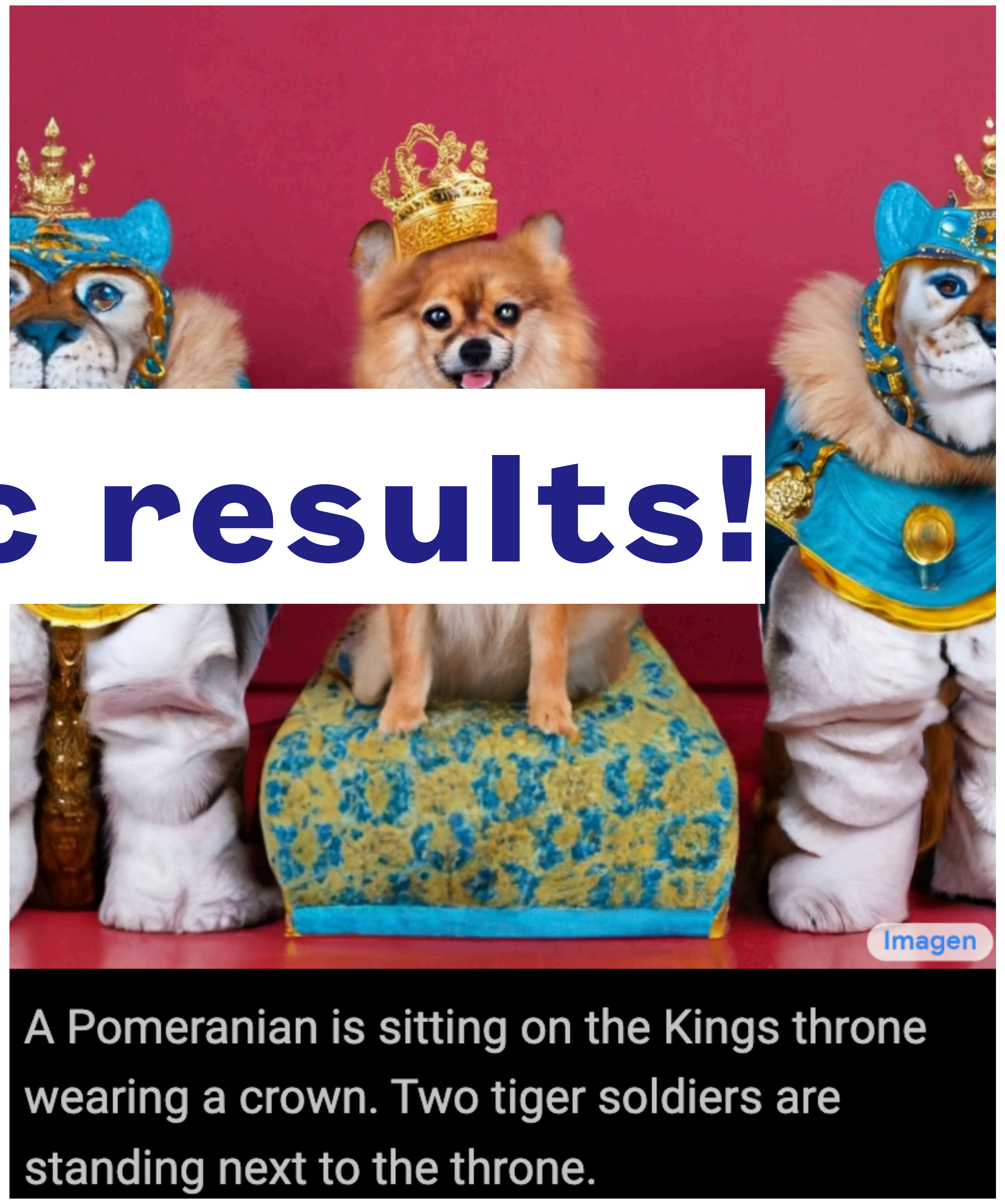
Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", **April 13, 2022**

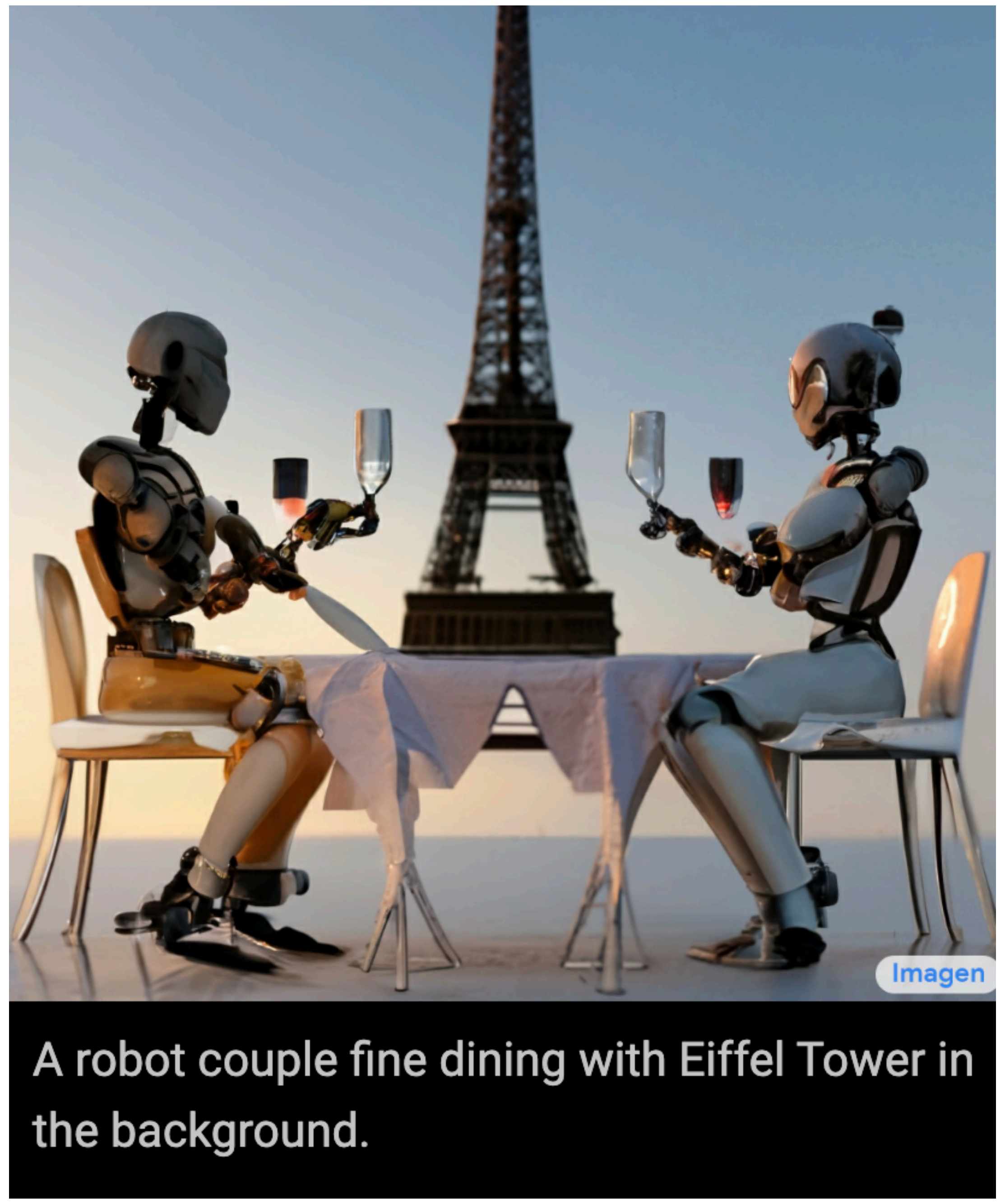
Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", **May 23, 2022**

Recent breakthroughs

Fantastic results!

How big are the models?





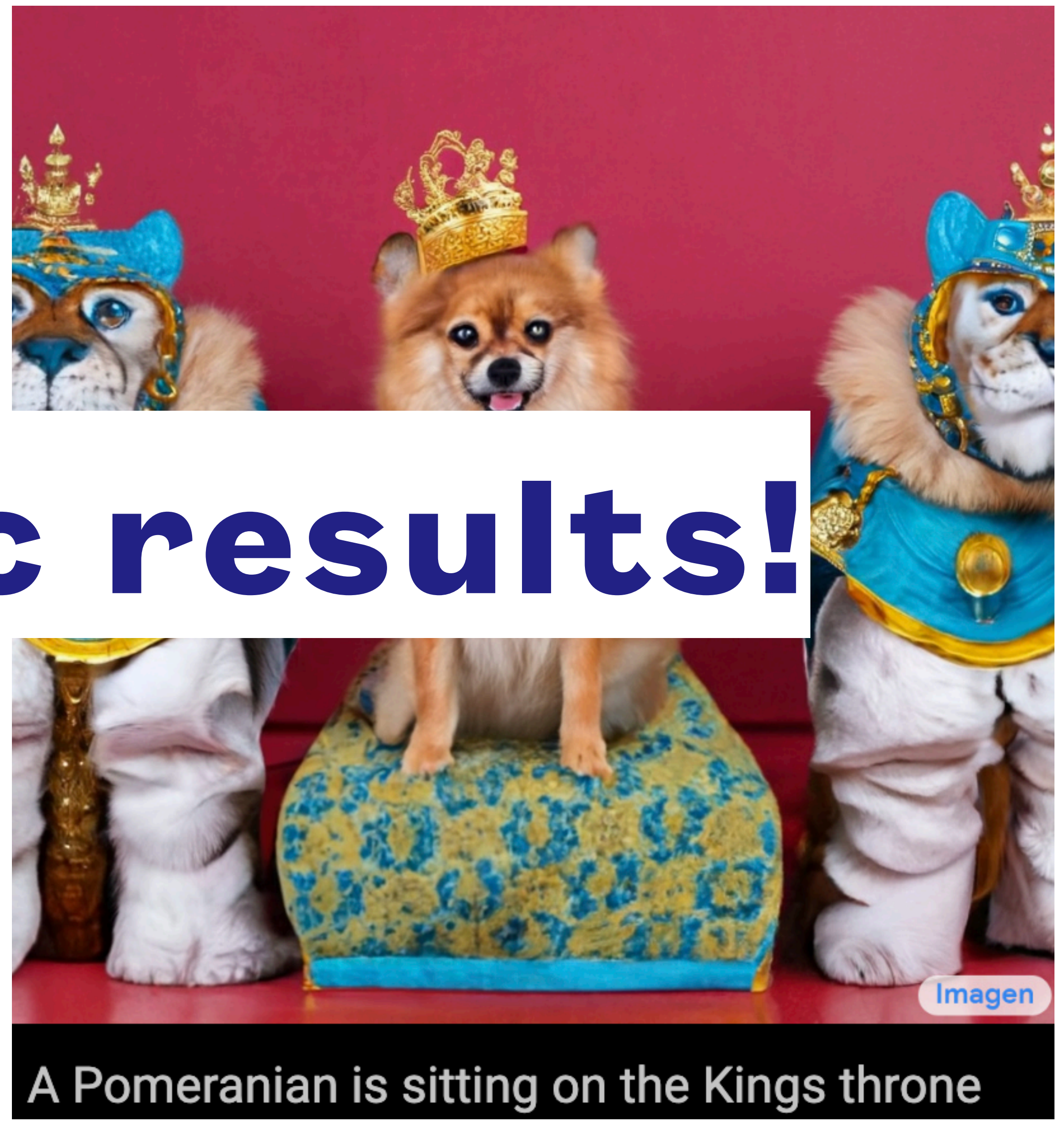
Dalle·2

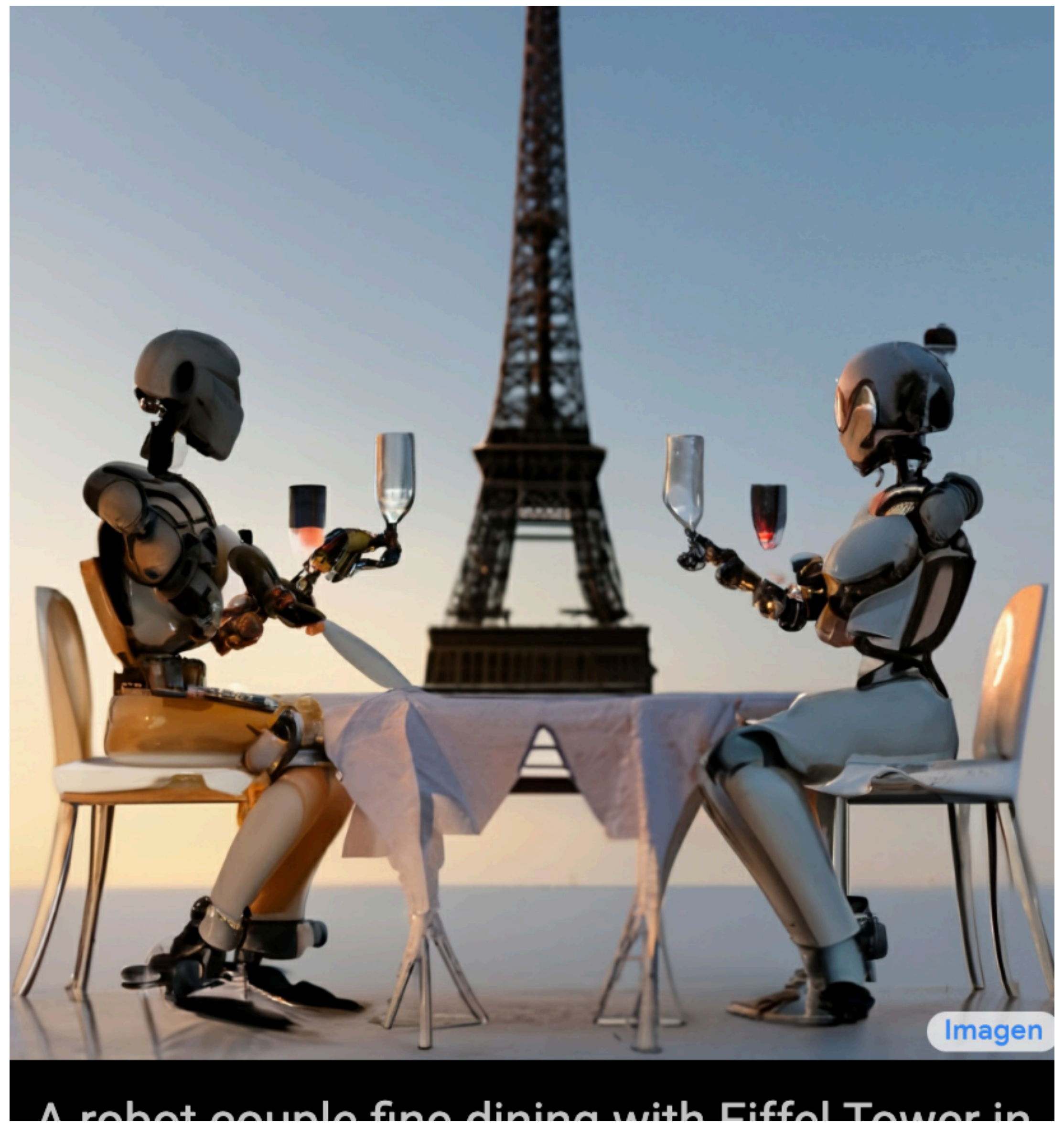
Imagen

Ramesh et al., "Hierarchical Text-Conditional Image Generation with CLIP Latents", **April 13, 2022**

Saharia et al., "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding", **May 23, 2022**

Recent breakthroughs





Fantastic results!

How big are the models?

Are the samples cherry-picked?

Can we use these models beyond data synthesis?

How long did it take to train these models?

Modeling: Discriminative vs. Generative

Modeling: Discriminative vs. Generative

- **Discriminative models**: drawing boundaries in the data space, $p(y \mid \mathbf{x})$.
- **Generative models**: explaining how the data was generated, $p(\mathbf{x}, y)$.

Jebara, T. (2012). Machine learning: discriminative and generative (Vol. 755). Springer Science & Business Media.

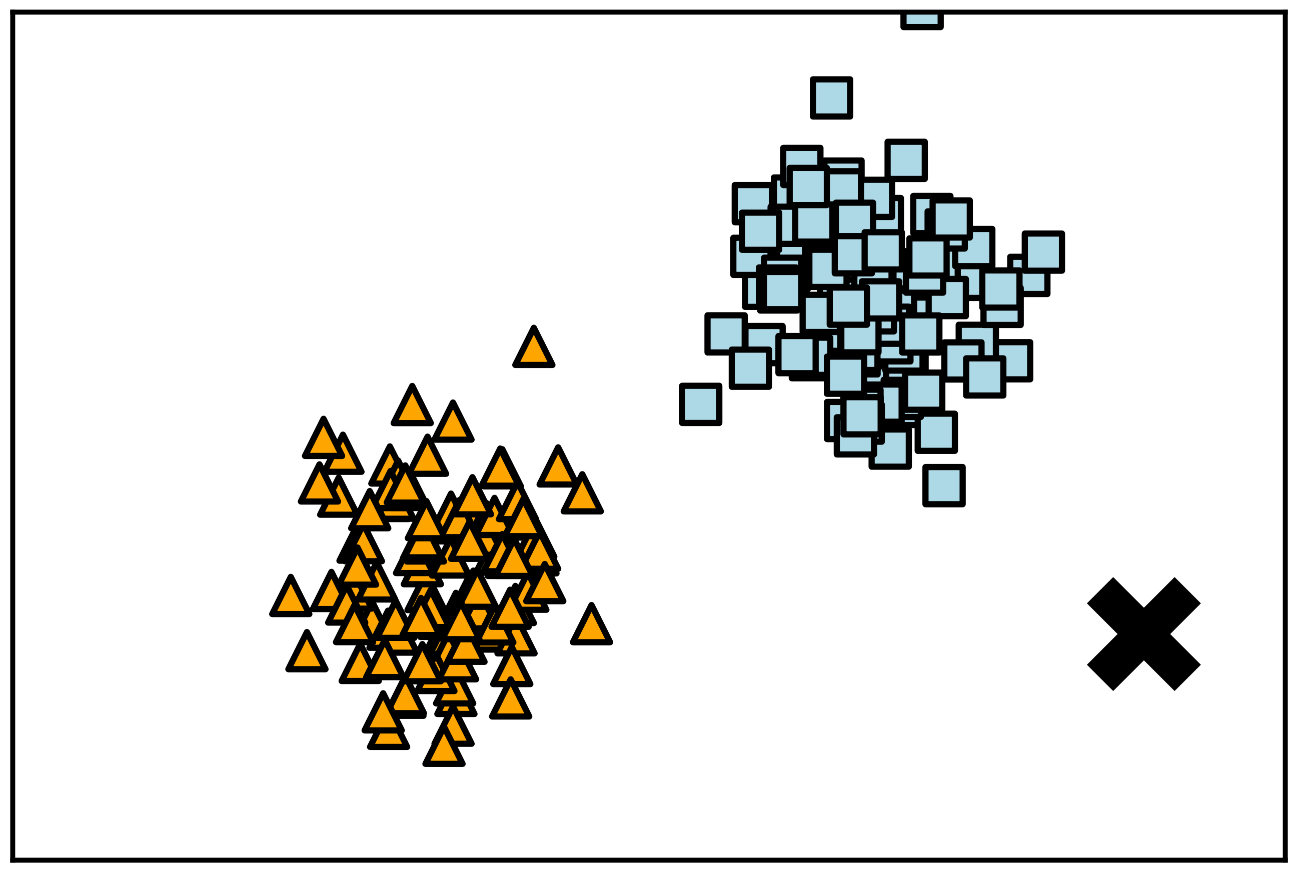
Modeling: Discriminative vs. Generative

- **Discriminative models**: drawing boundaries in the data space, $p(y \mid \mathbf{x})$.
- **Generative models**: explaining how the data was generated, $p(\mathbf{x}, y)$.
- In ML, many models are **generative**:
  - Naive Bayes, Linear Discriminant Analysis
  - Bayesian networks & Markov random fields
  - (Gaussian) Mixture Models, Latent Dirichlet Allocation, Factor Analysis, PCA
  - Chinese restaurant process, Indian buffet process

Jebara, T. (2012). Machine learning: discriminative and generative (Vol. 755). Springer Science & Business Media.

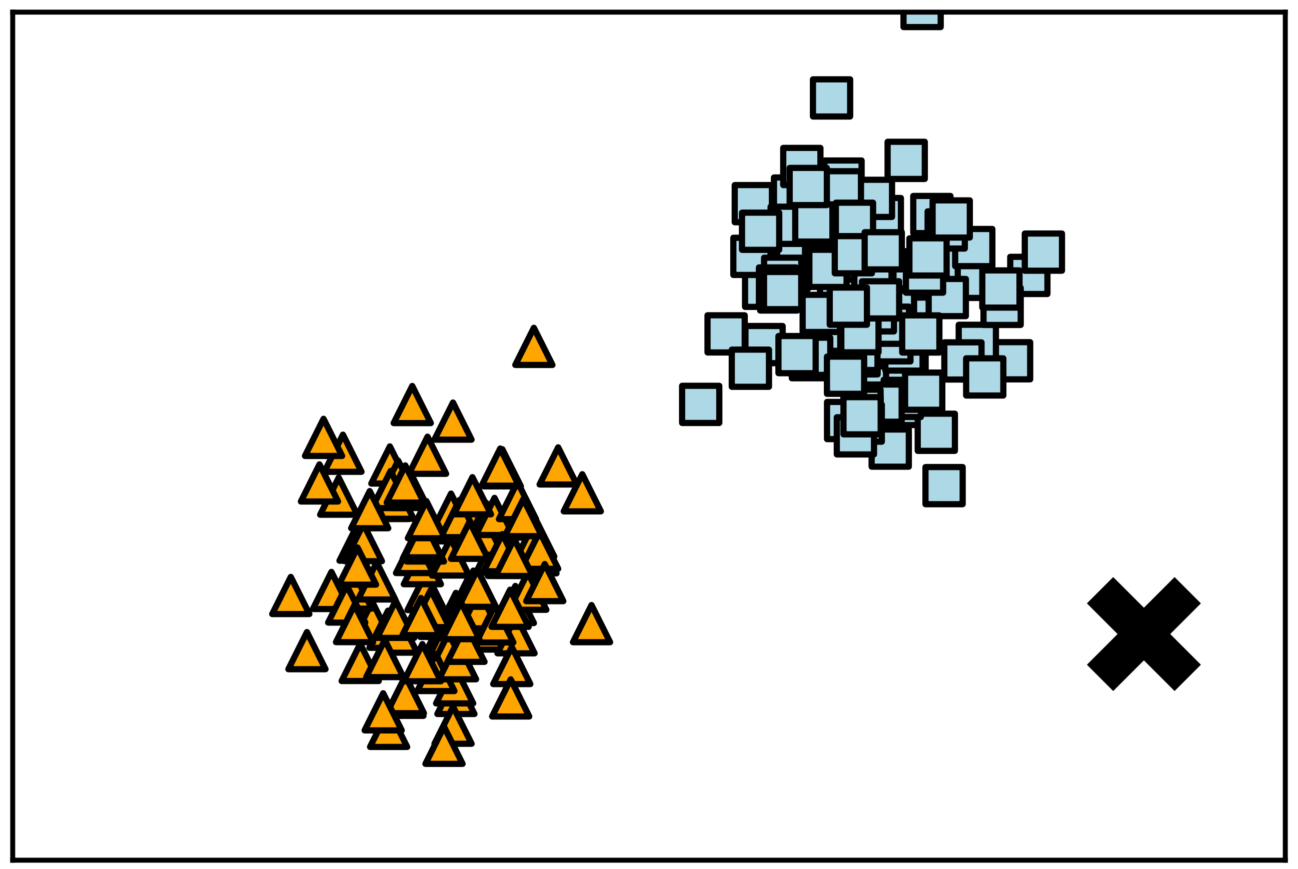
Generative = How data is generated

Generative = How data is generated

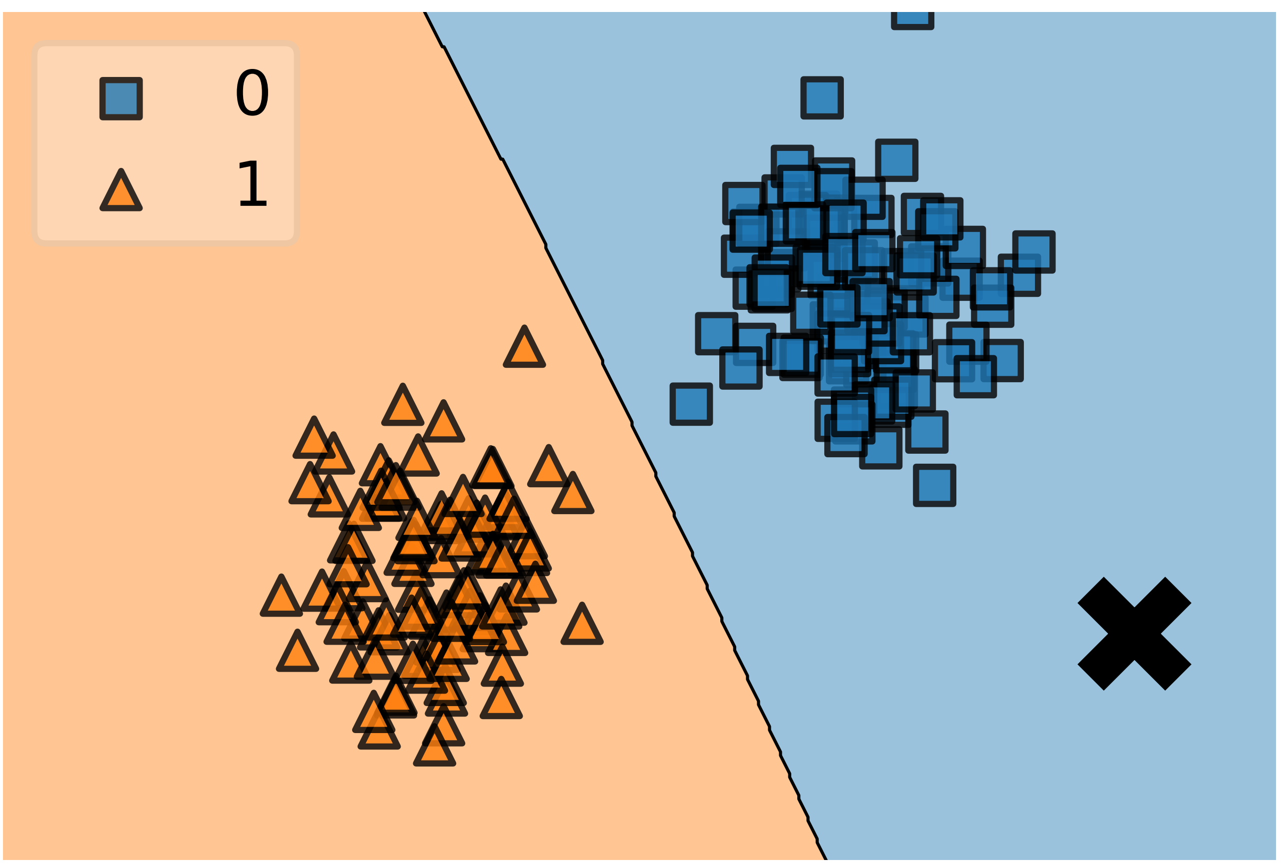


Data

Generative = How data is generated



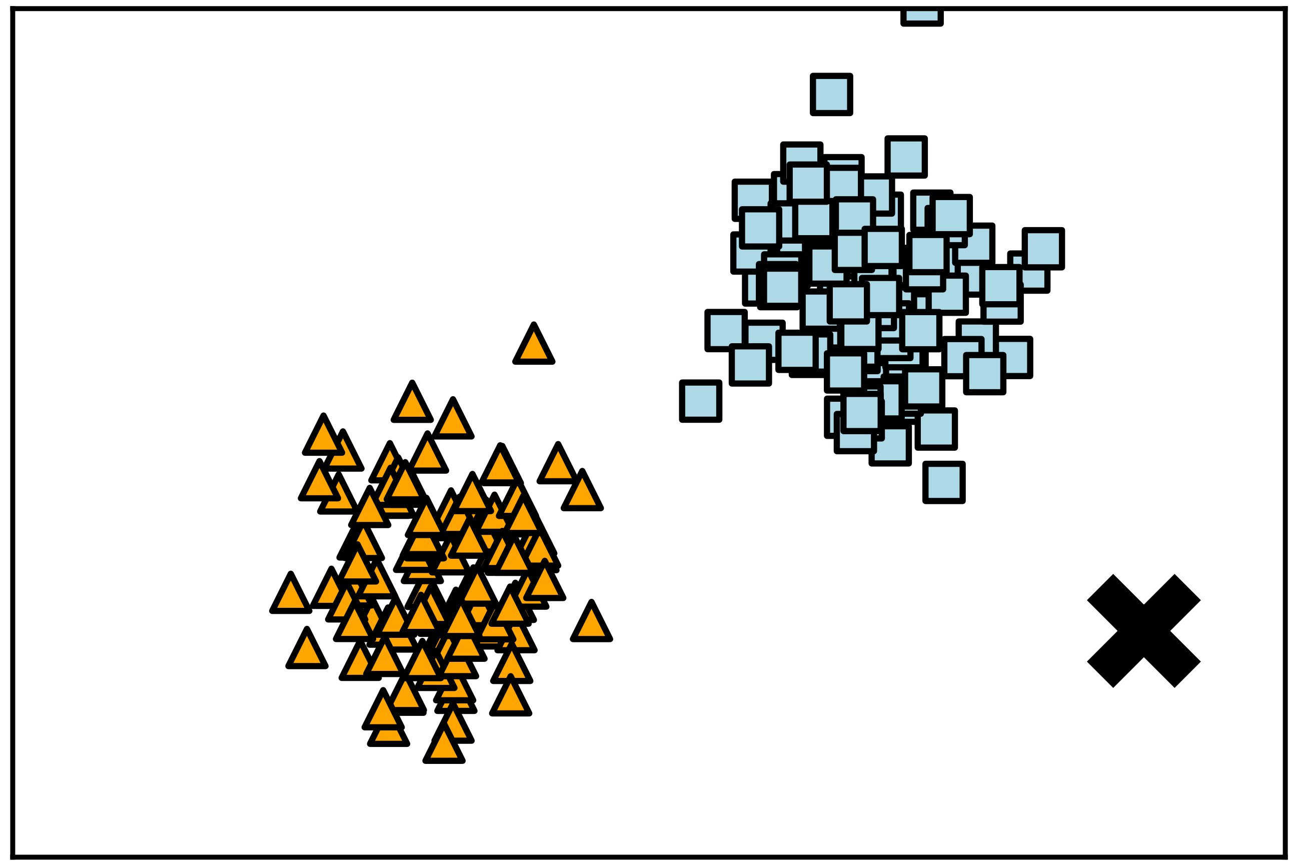
Data



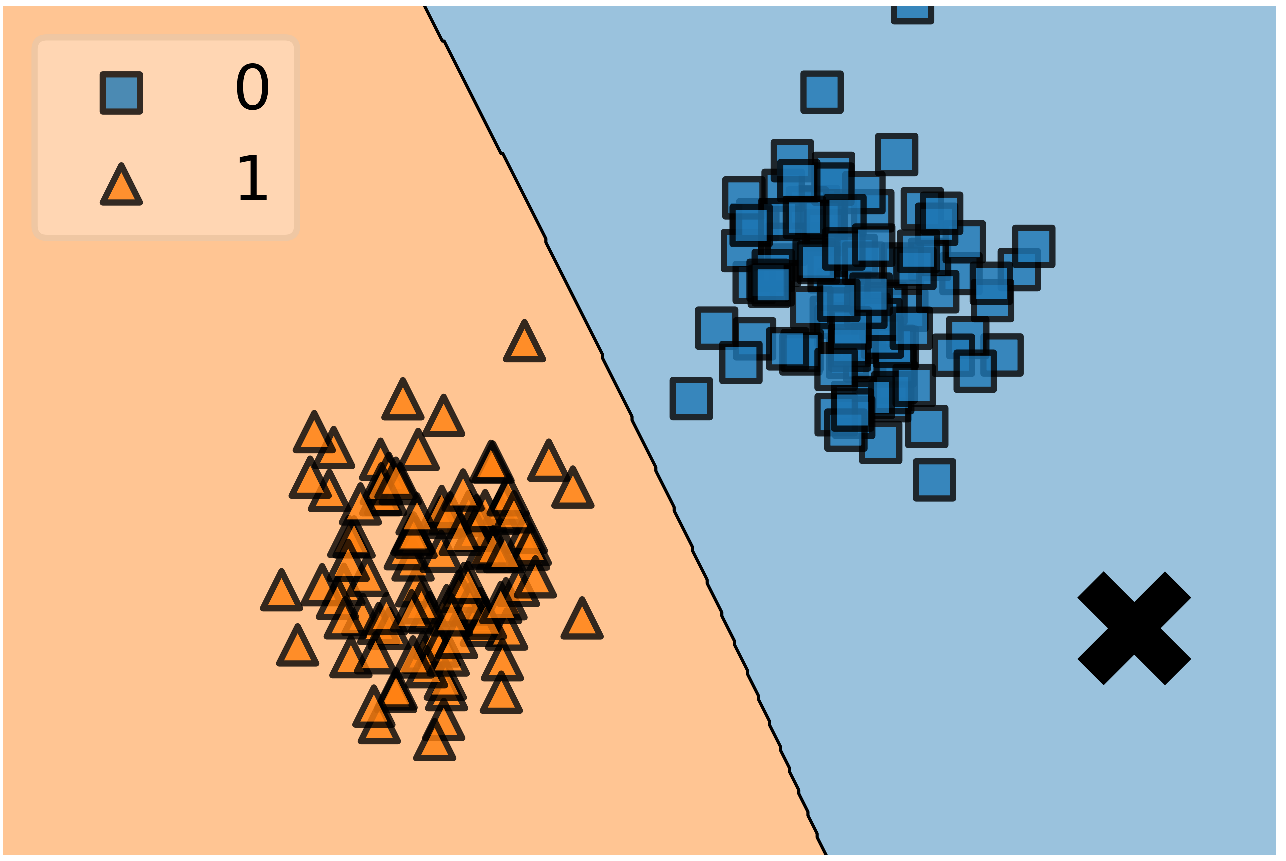
$p(y|\mathbf{x})$

$p(blue|\mathbf{x})$ is high
= certain decision!

Generative = How data is generated

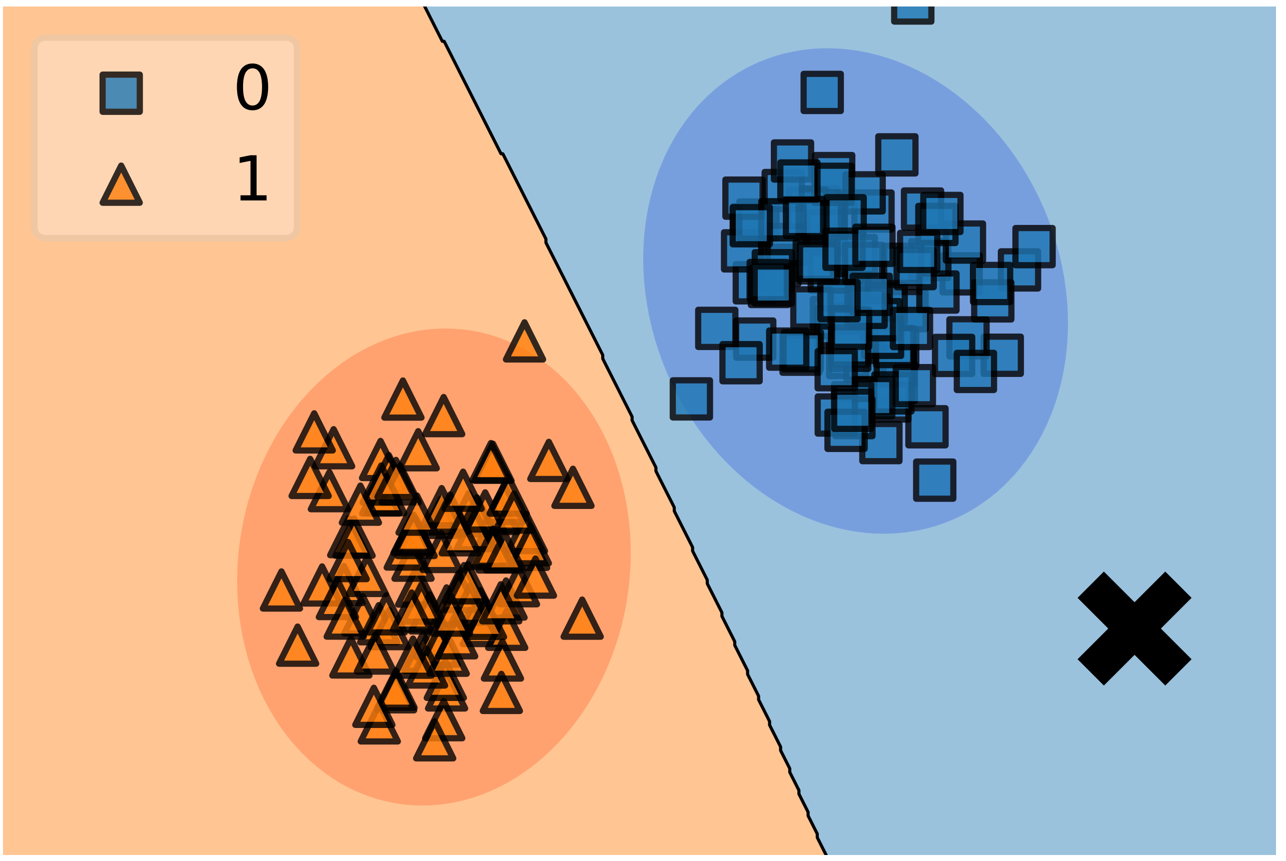


Data



$p(y|\mathbf{x})$

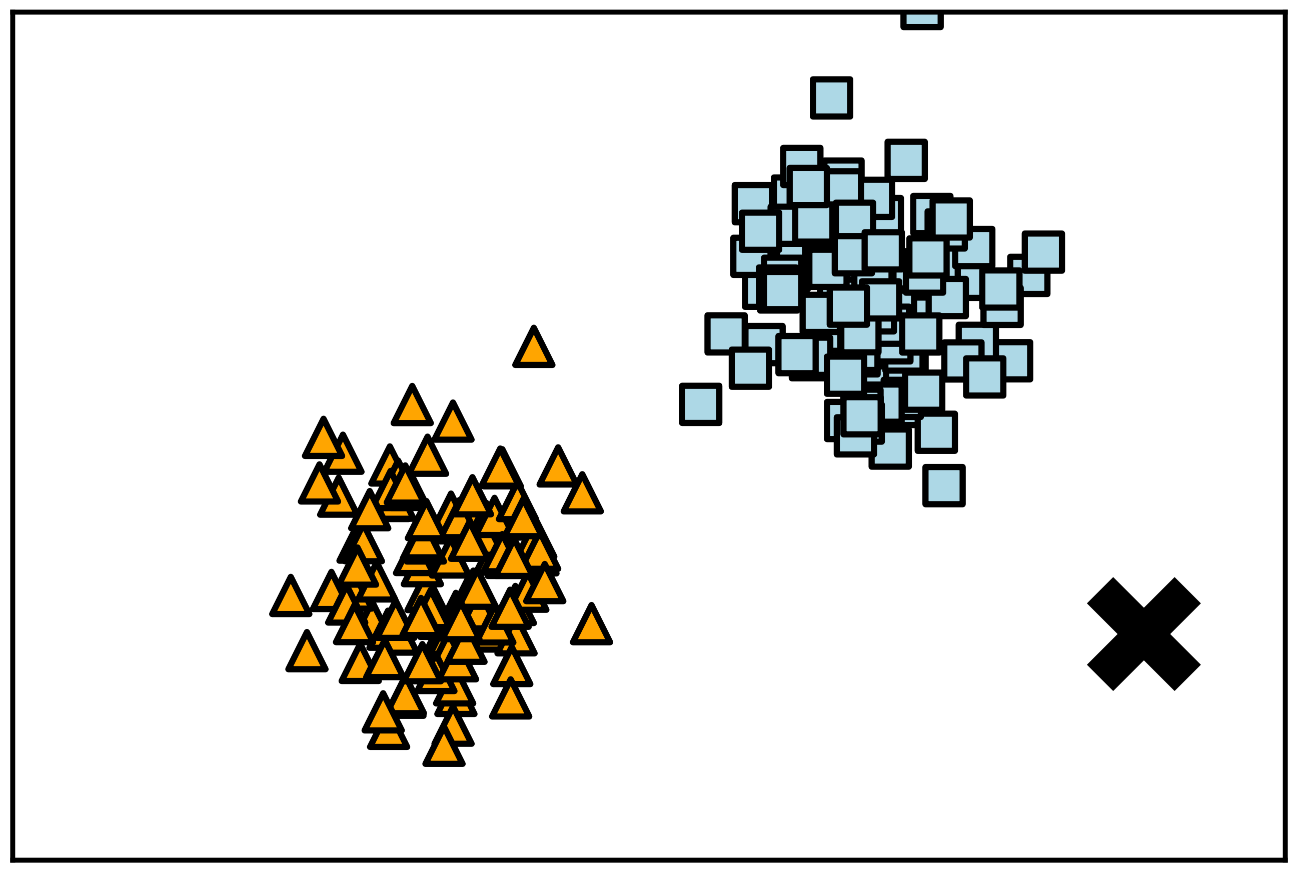
$p(blue|\mathbf{x})$ is high
= certain decision!

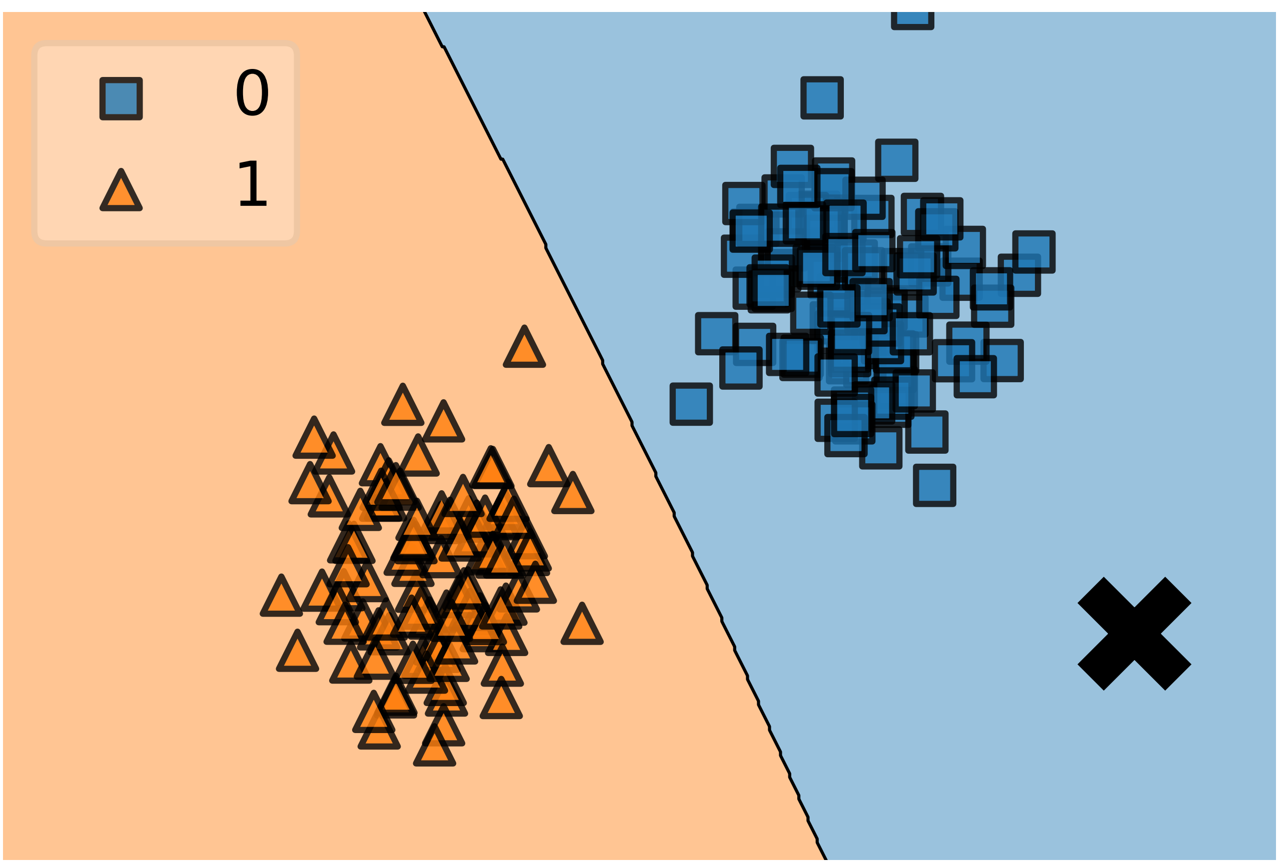


$p(\mathbf{x}, y) = p(y|\mathbf{x})\ p(\mathbf{x})$

$p(blue|\mathbf{x})$ is high
and $p(\mathbf{x})$ is low
= uncertain decision!

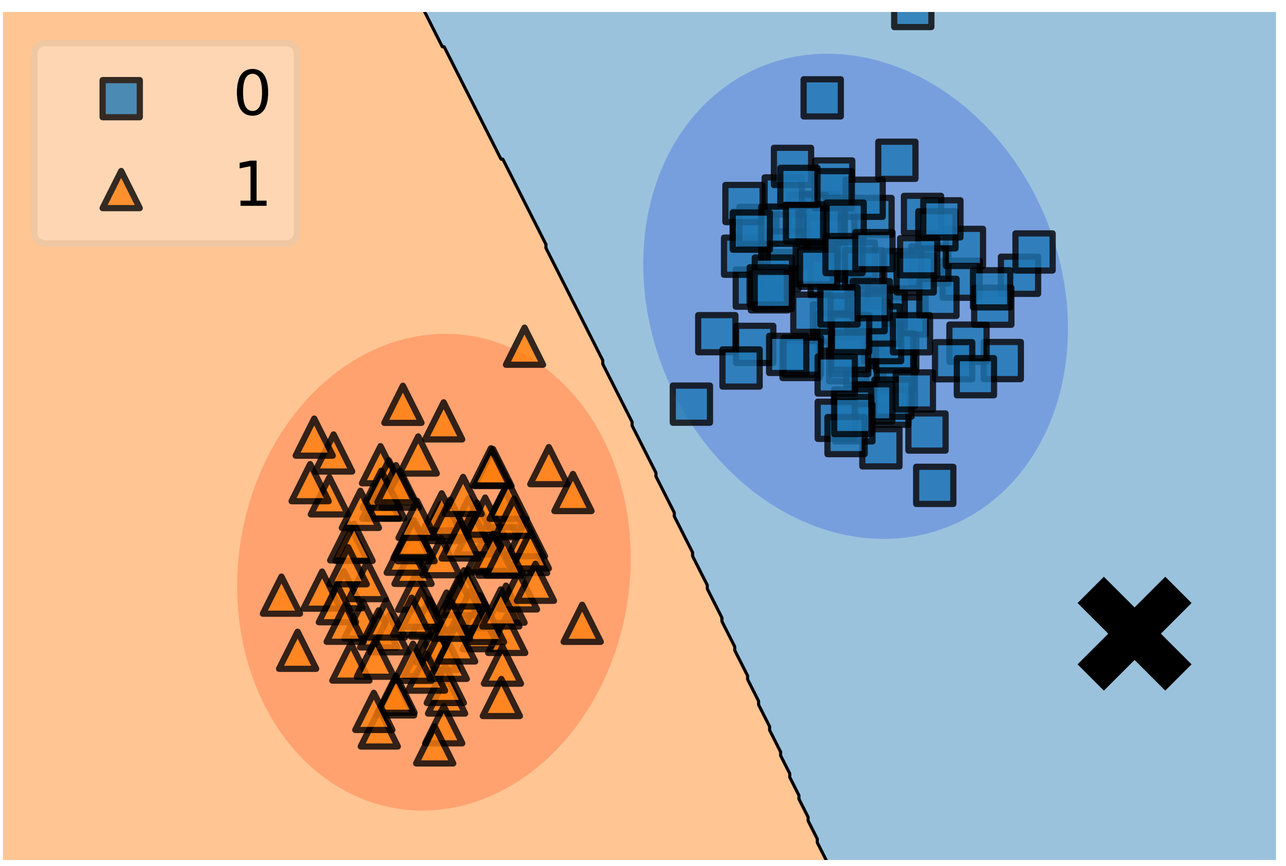
Generative = How data is generated


Data



$p(y|\mathbf{x})$

$p(blue|\mathbf{x})$ is high
= certain decision!



$p(\mathbf{x}, y) = p(y|\mathbf{x})\ p(\mathbf{x})$

$p(blue|\mathbf{x})$ is high
and $p(\mathbf{x})$ is low
= uncertain decision!

Knowing the joint distribution tells us a lot about the phenomenon!

Why Generative AI?

Human intelligence is intrinsically generative

Why Generative AI?

Human intelligence is intrinsically generative

GAI = AGI?

or

AGI through GAI

Generative AI

$$p(\mathbf{x}, y) = p(y \mid \mathbf{x})\; p(\mathbf{x})$$

Generative AI

$$p(\mathbf{x}, y) = p(y \mid \mathbf{x})\ p(\mathbf{x})$$

Any deep learning predictor

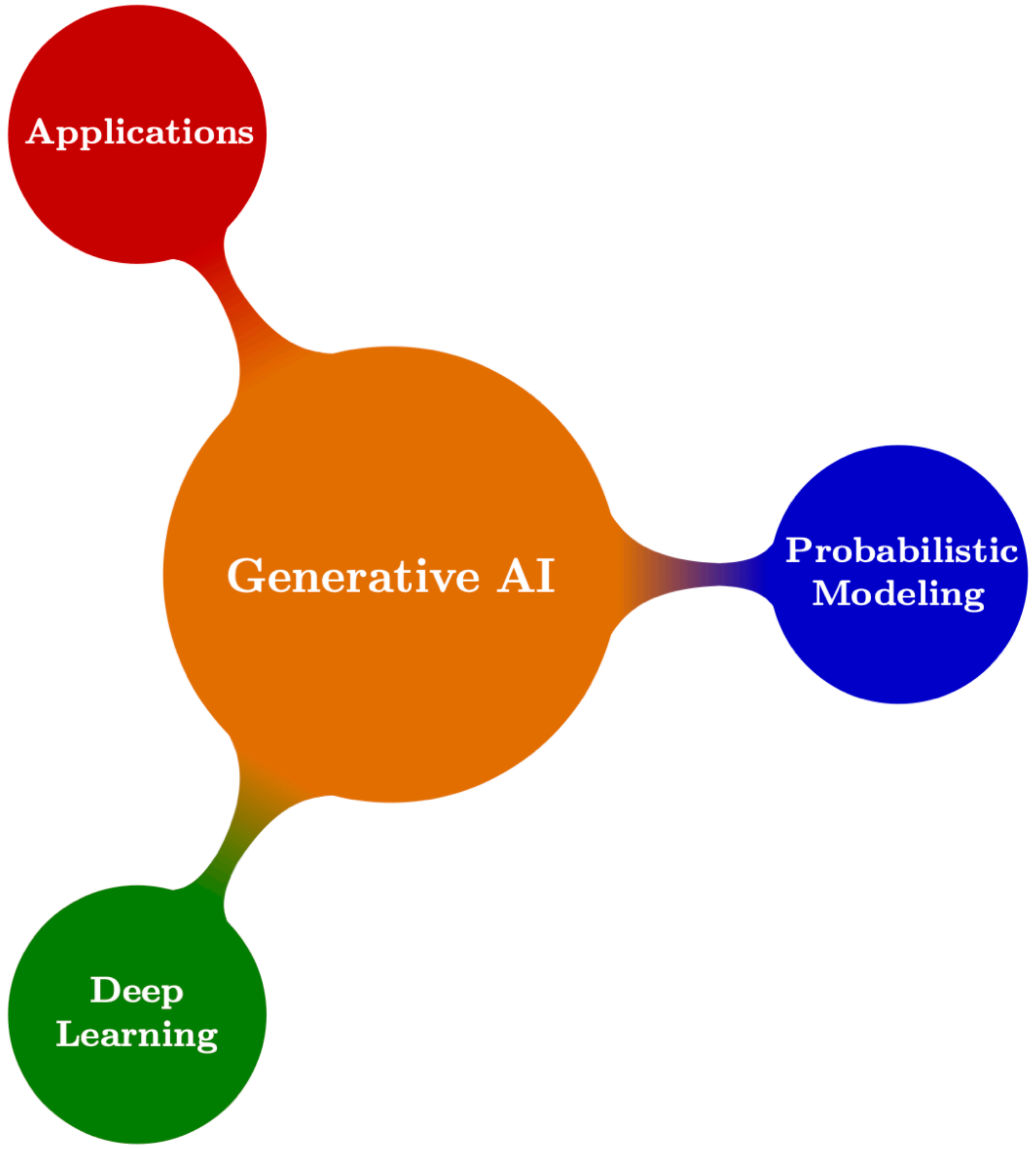
Relatively easy

A density estimator

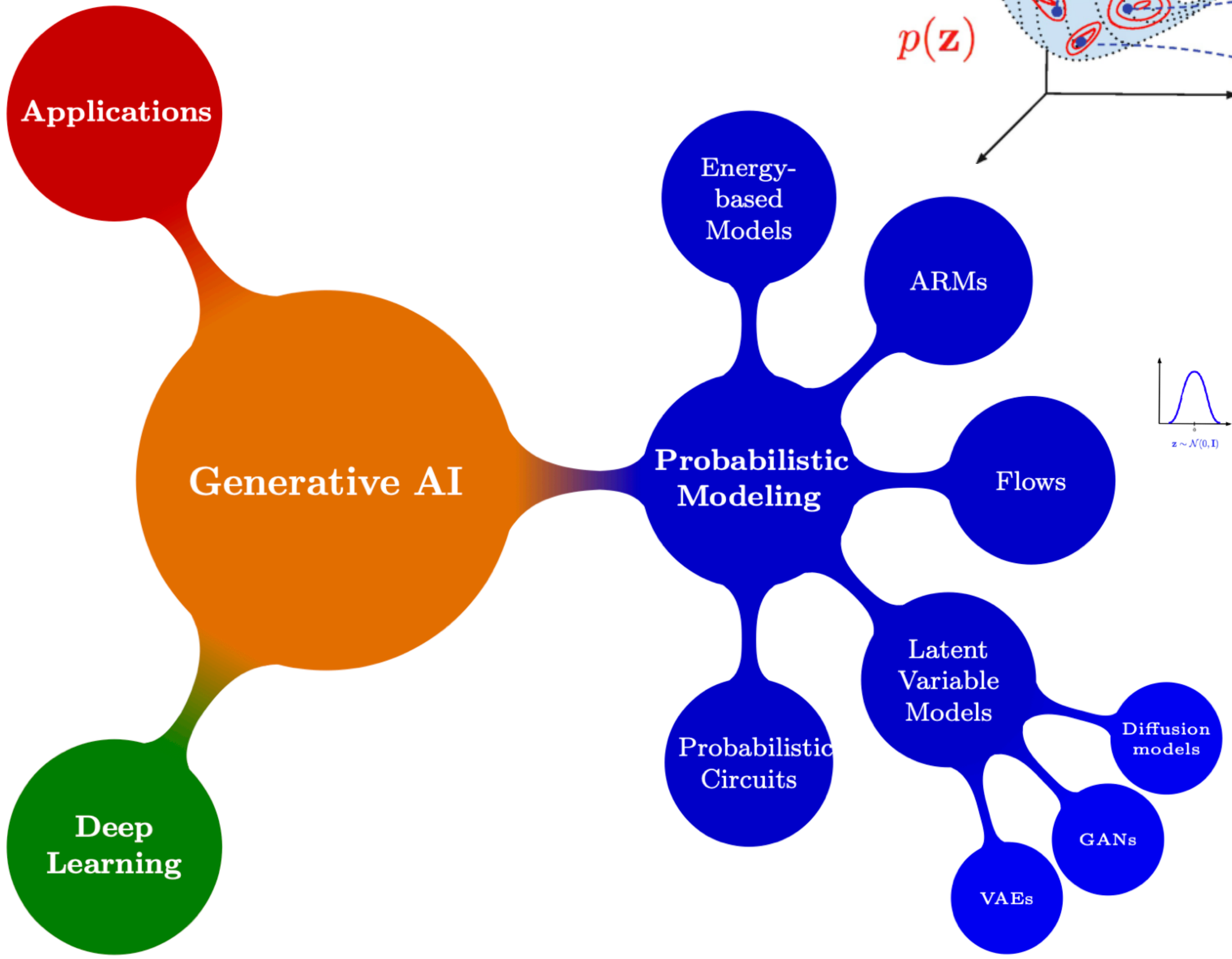
A probabilistic model

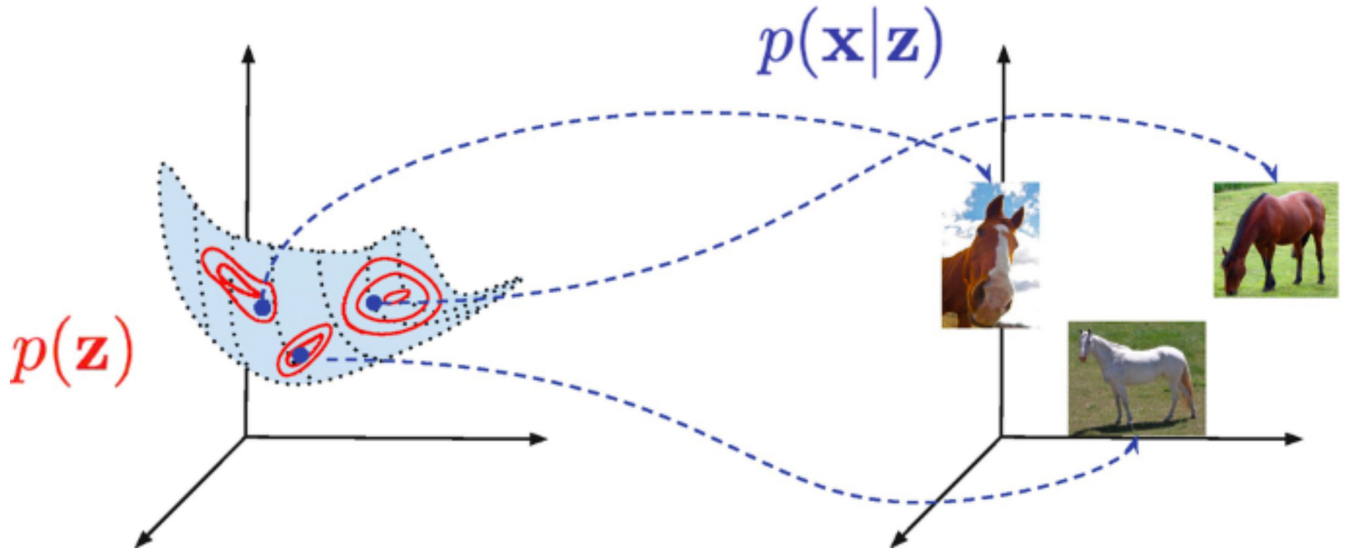
Challenging!

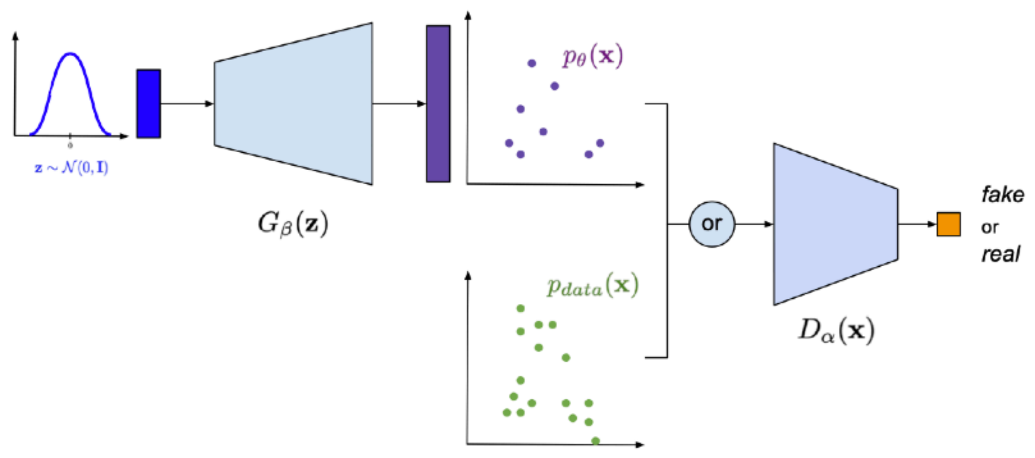
Overview of Generative AI

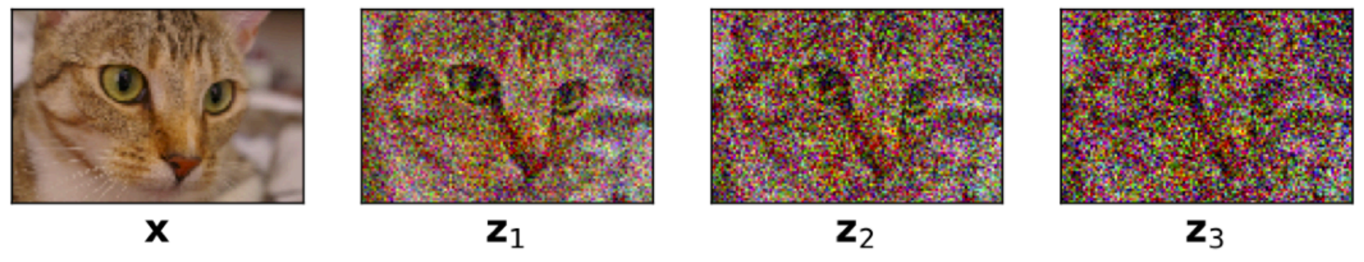


Overview of Generative AI

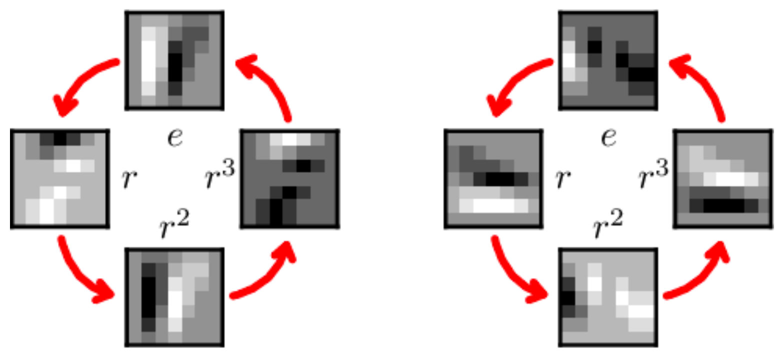


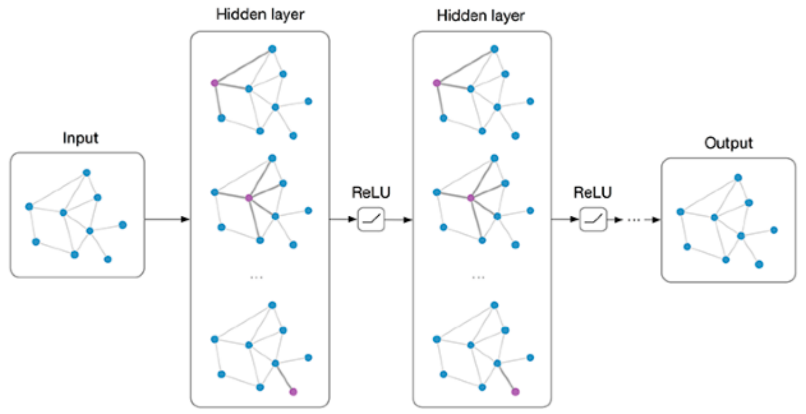


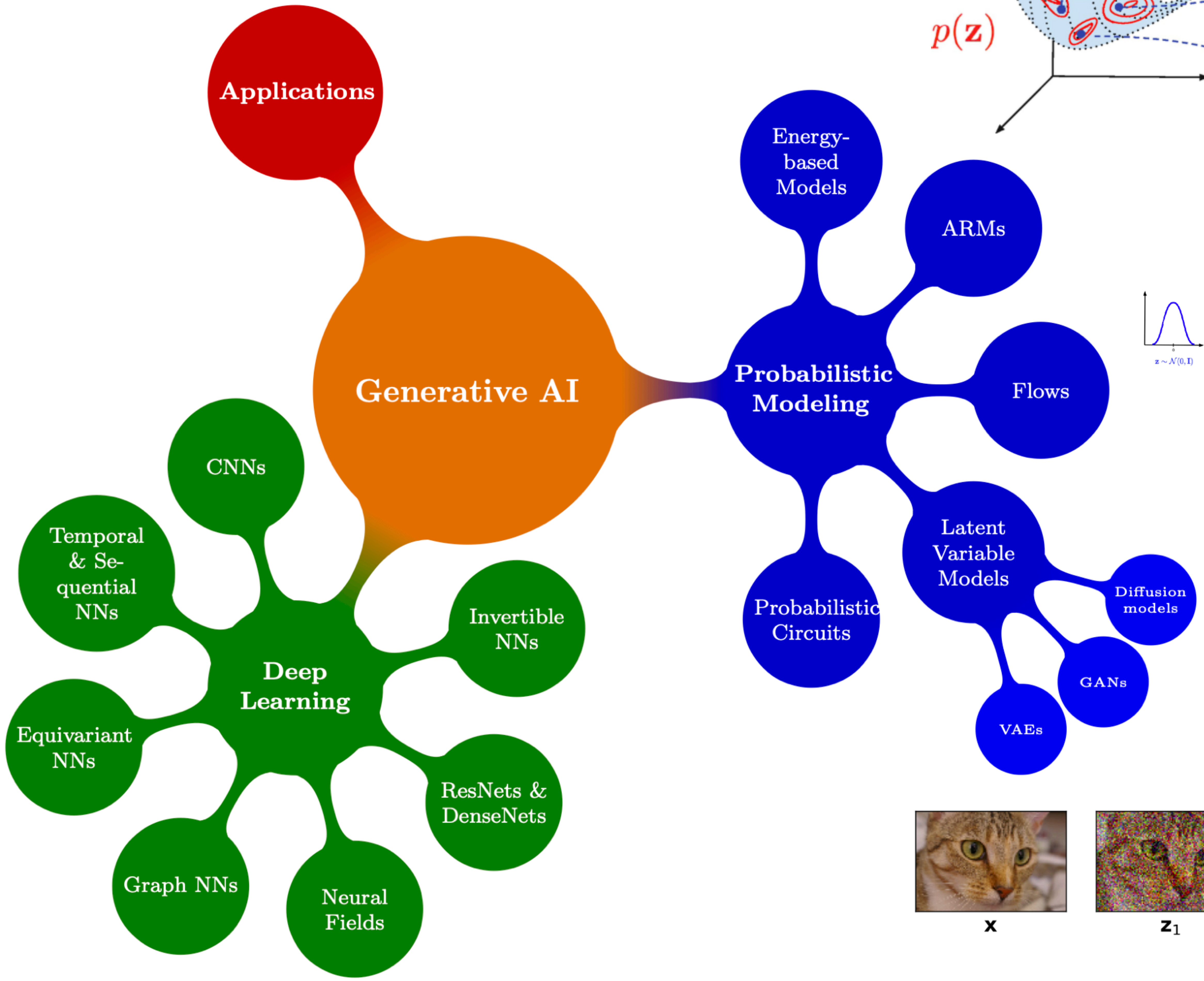


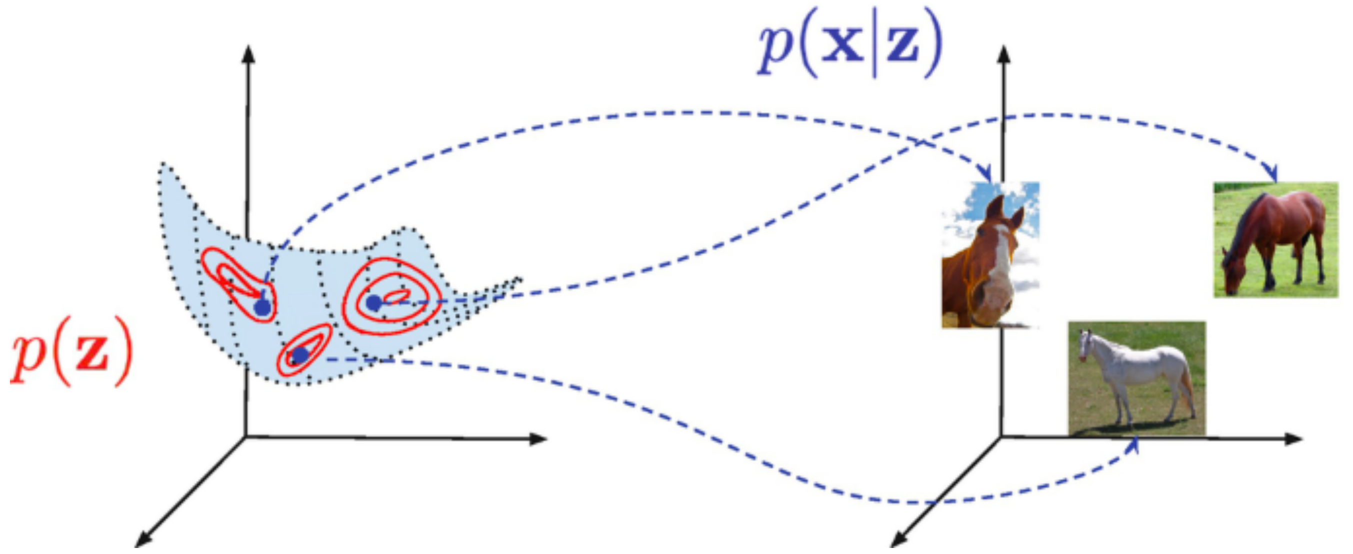


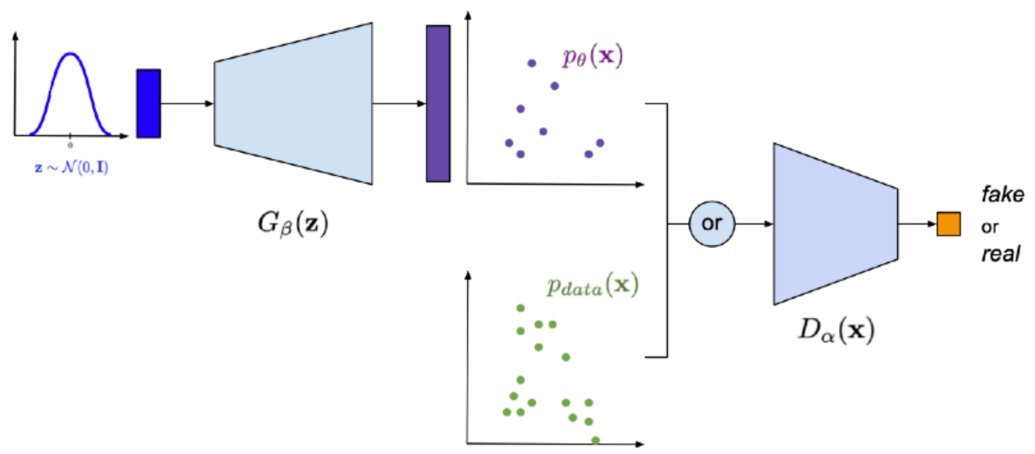
Overview of Generative AI

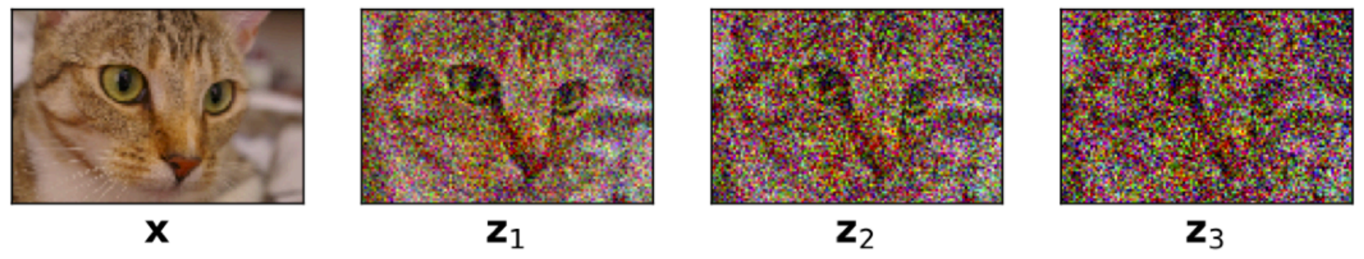




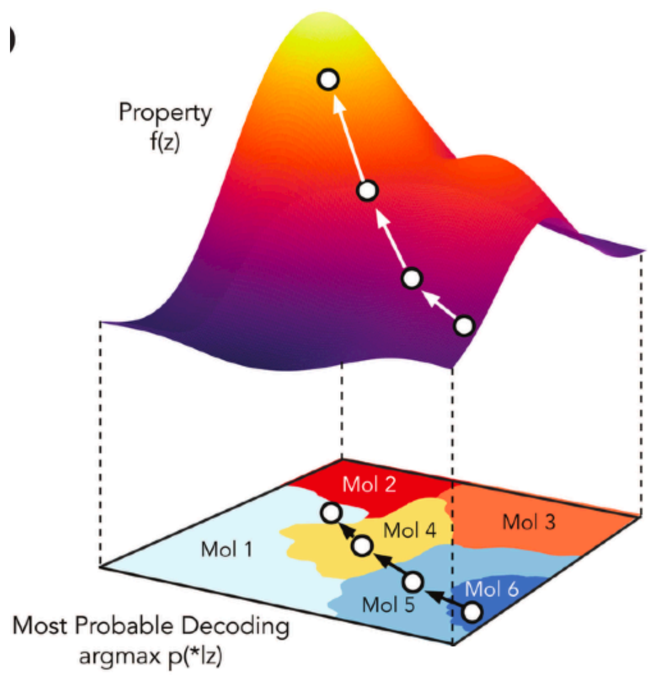


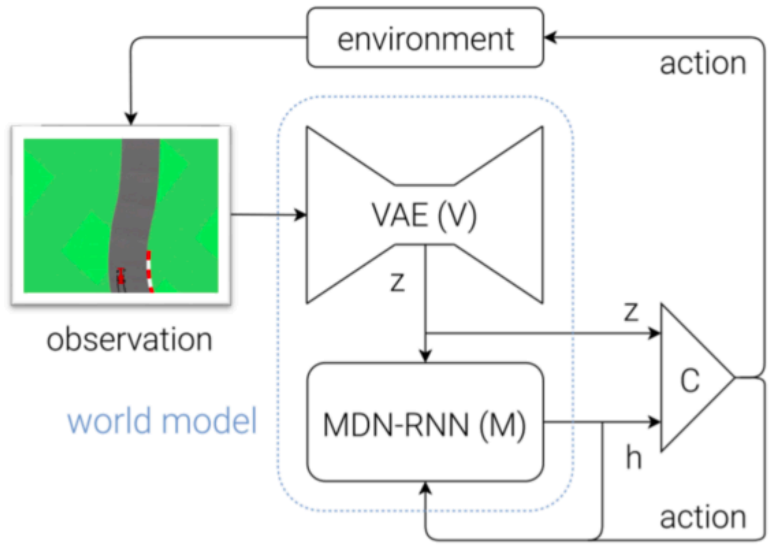


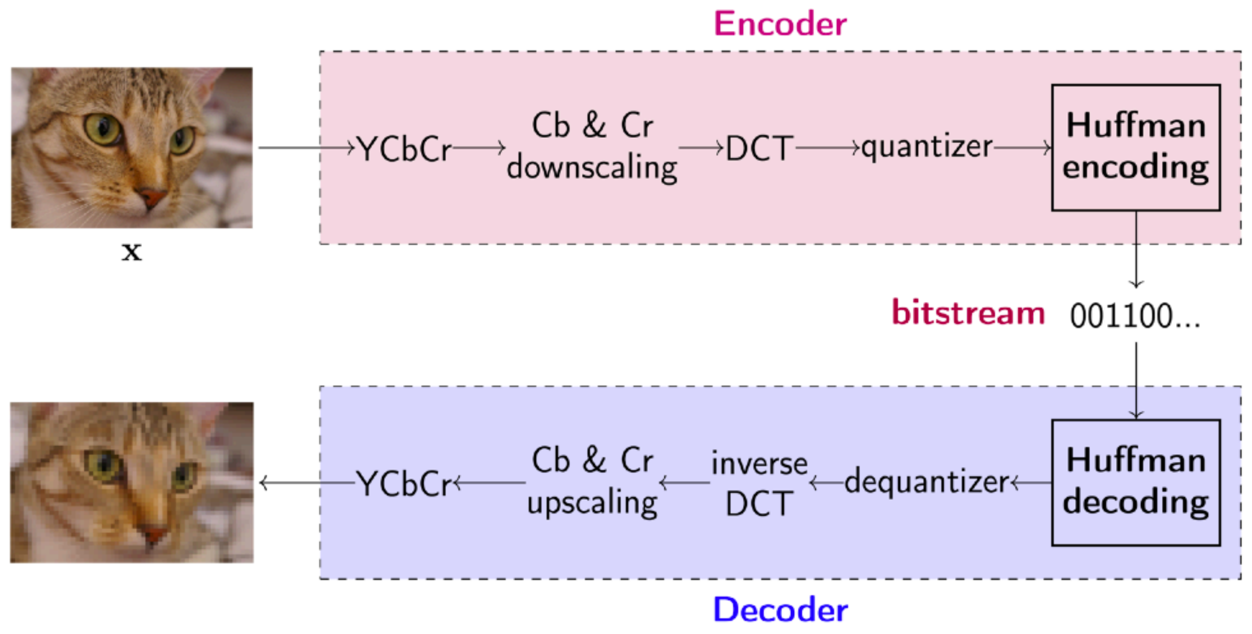


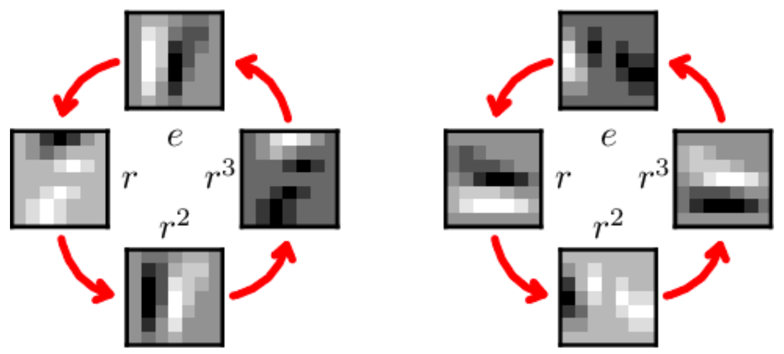


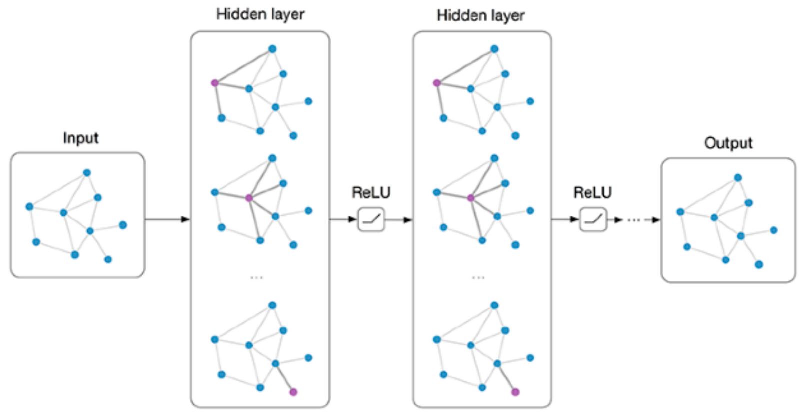
Overview of Generative AI

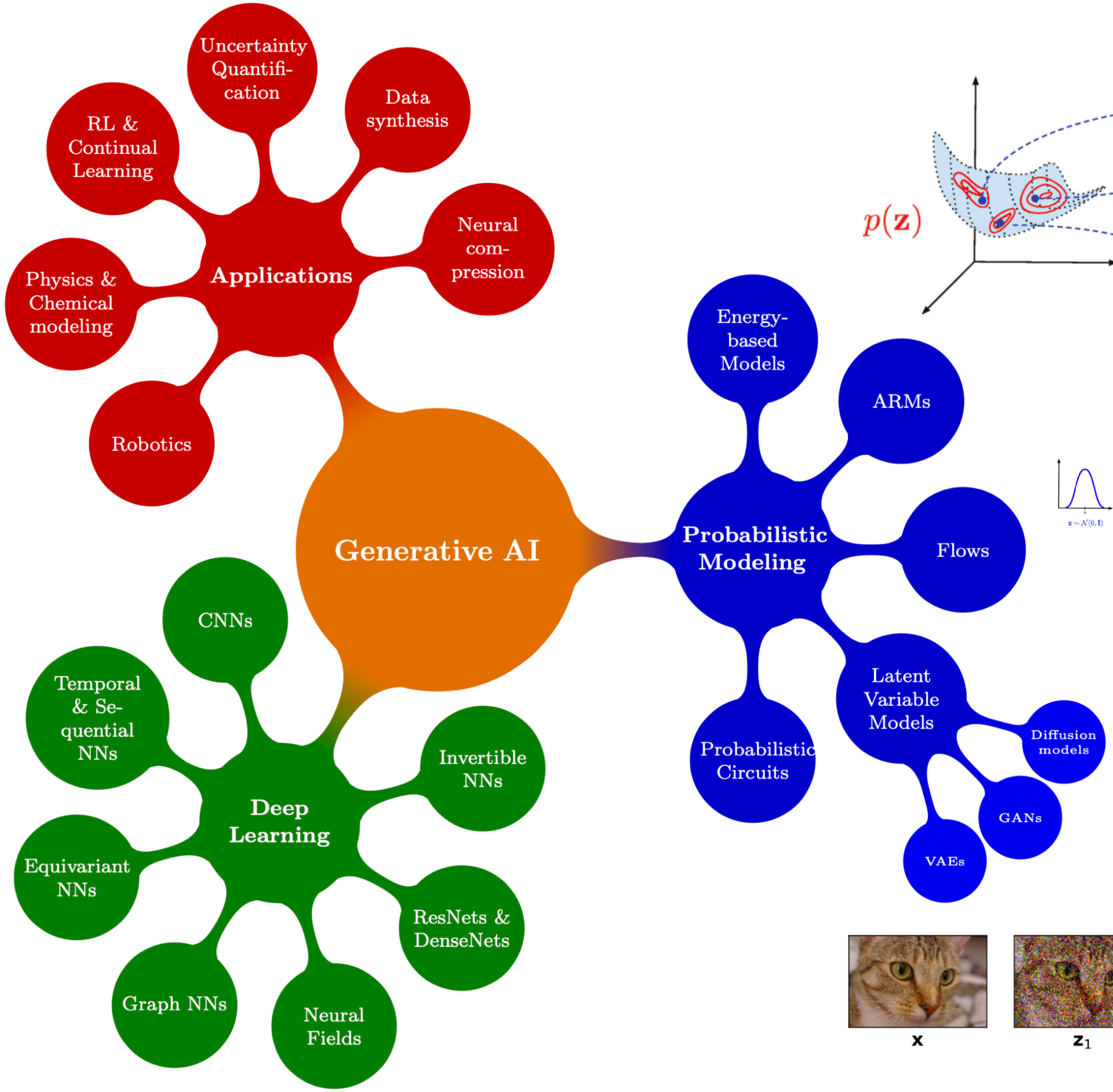


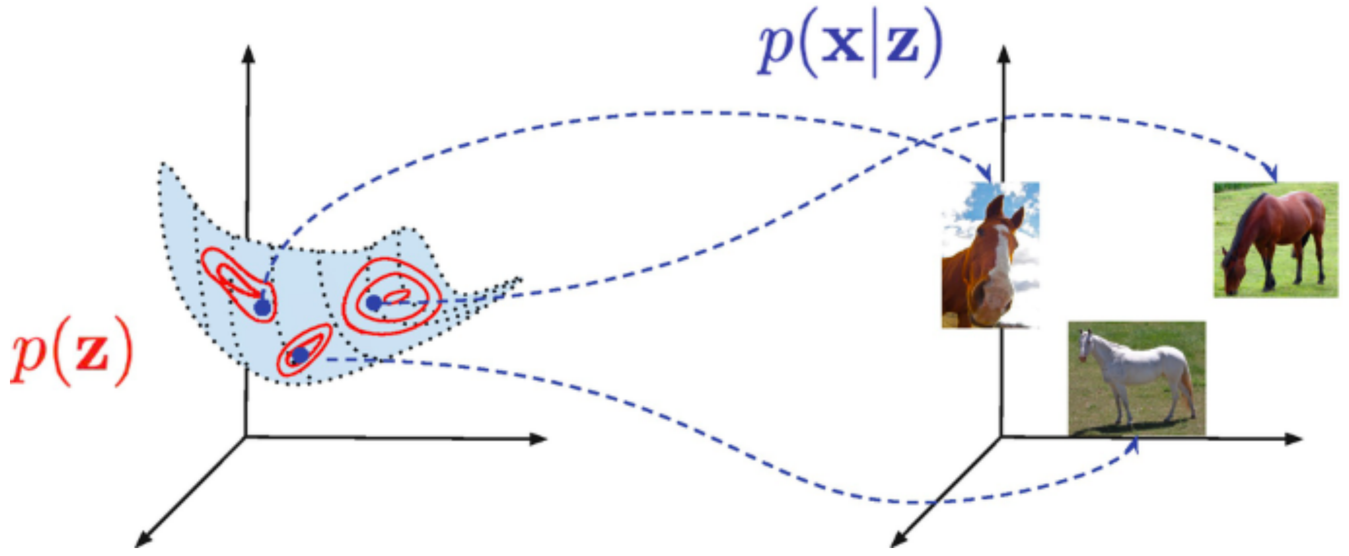


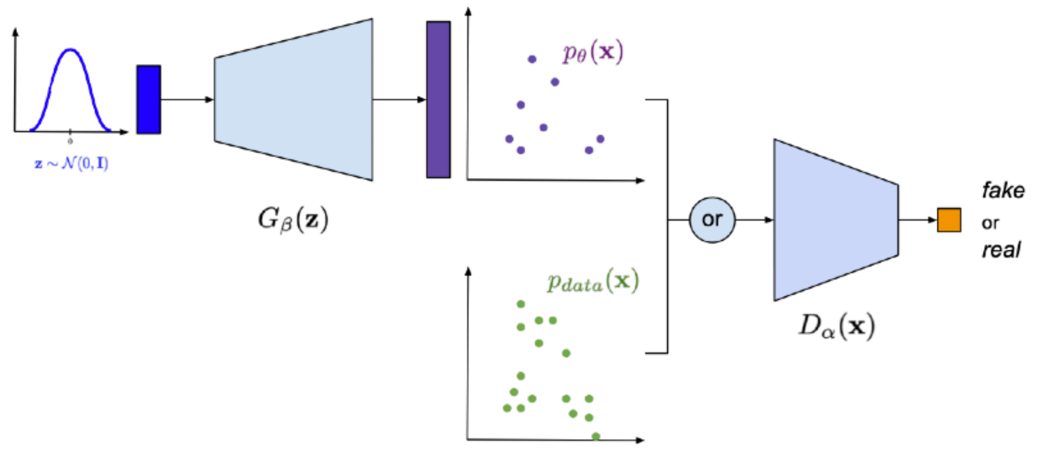


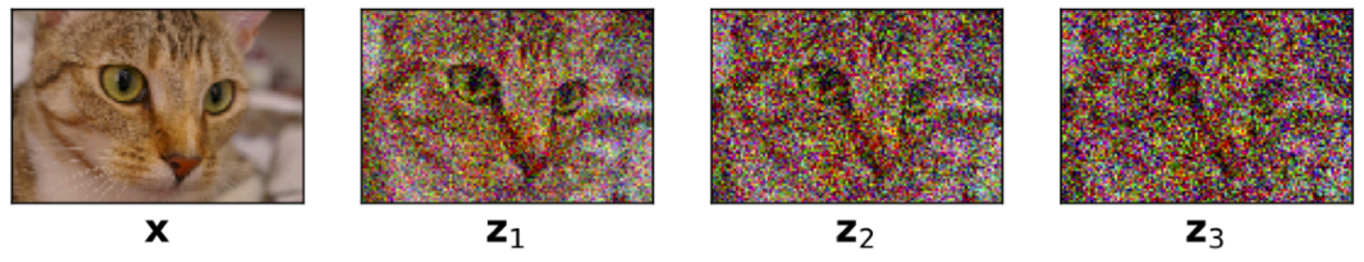












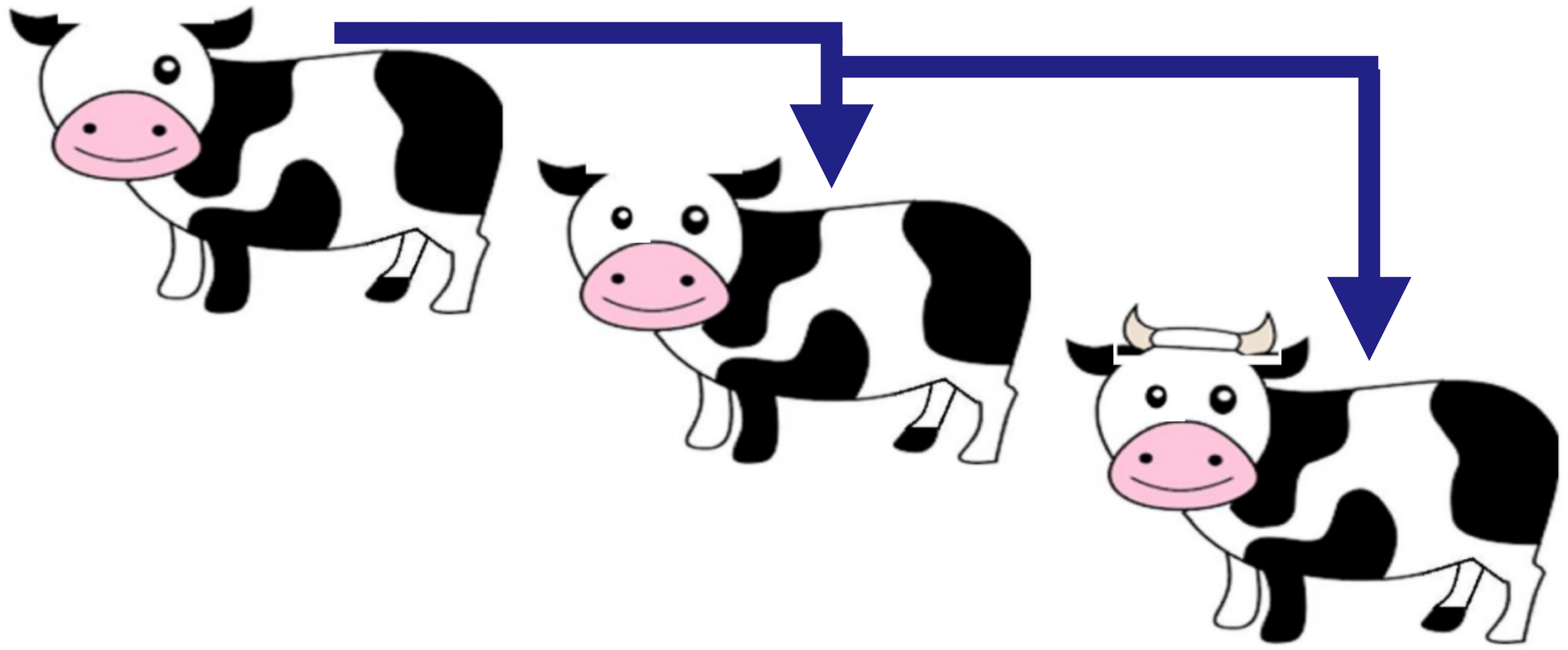
Generative AI and (spherical) cows

A high-dim object

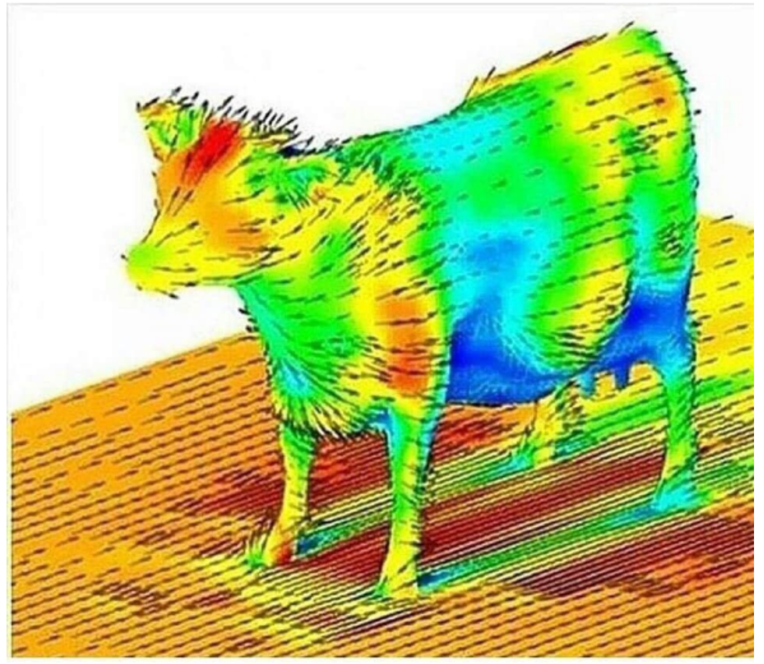
Latent Variable Models

Flow-based models

Goal: $p(\mathbf{x})$



Autoregressive Models



Diffusion models
Energy-based models

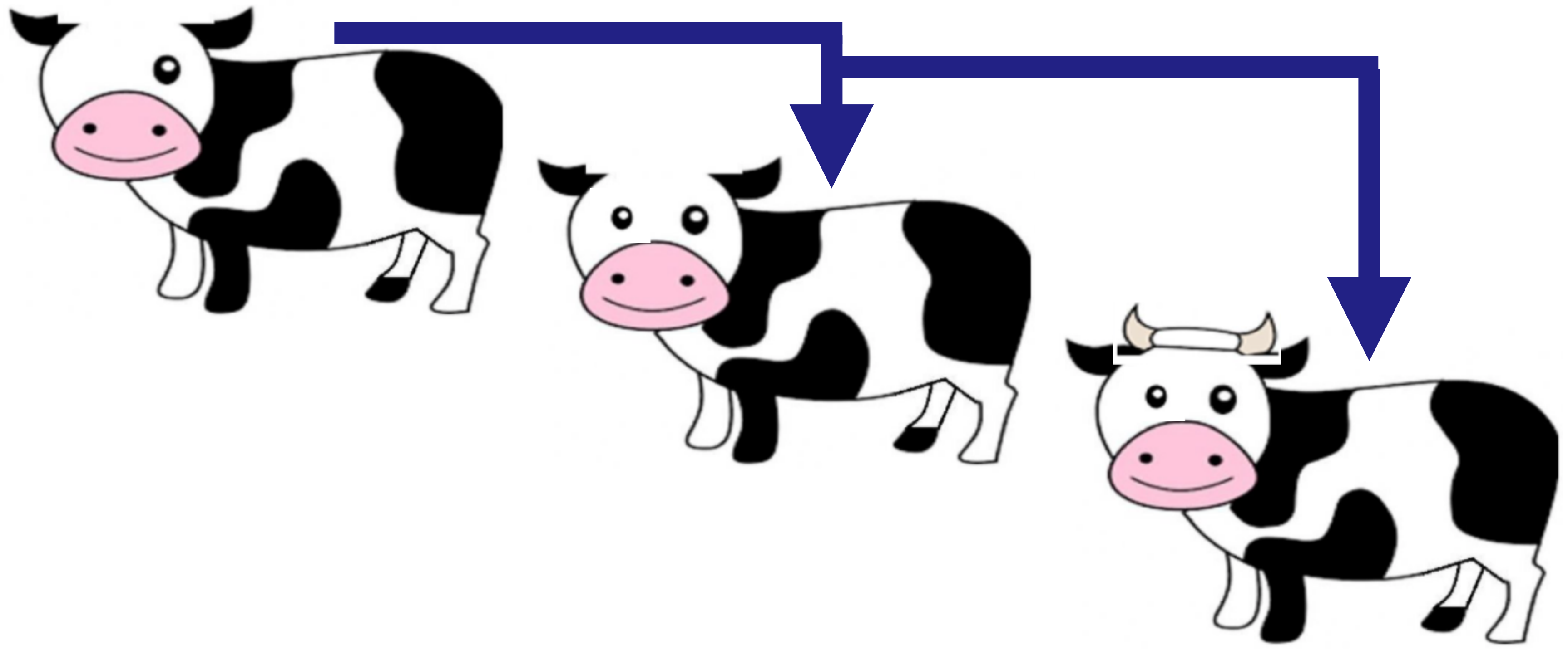
Generative AI and (spherical) cows

A high-dim object

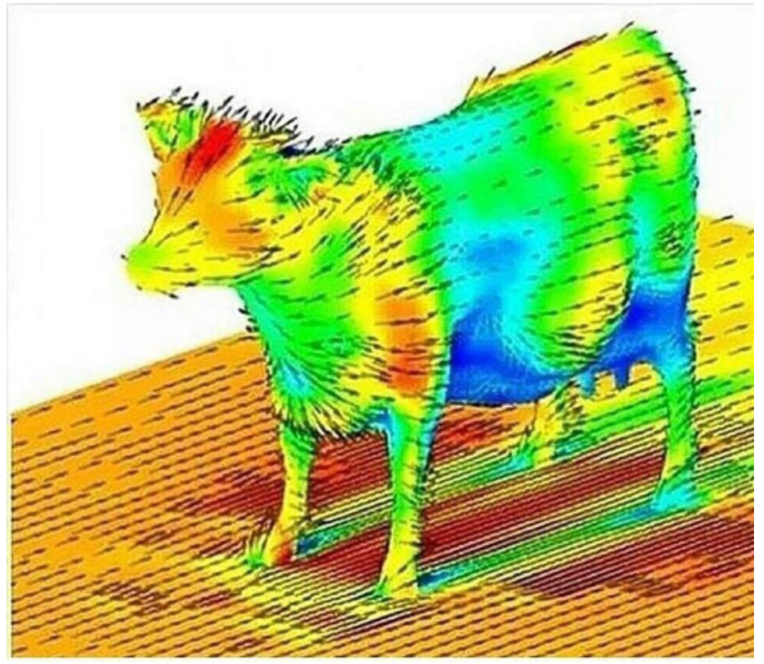
Latent Variable Models

Flow-based models

Goal: $p(\mathbf{x})$



Autoregressive Models



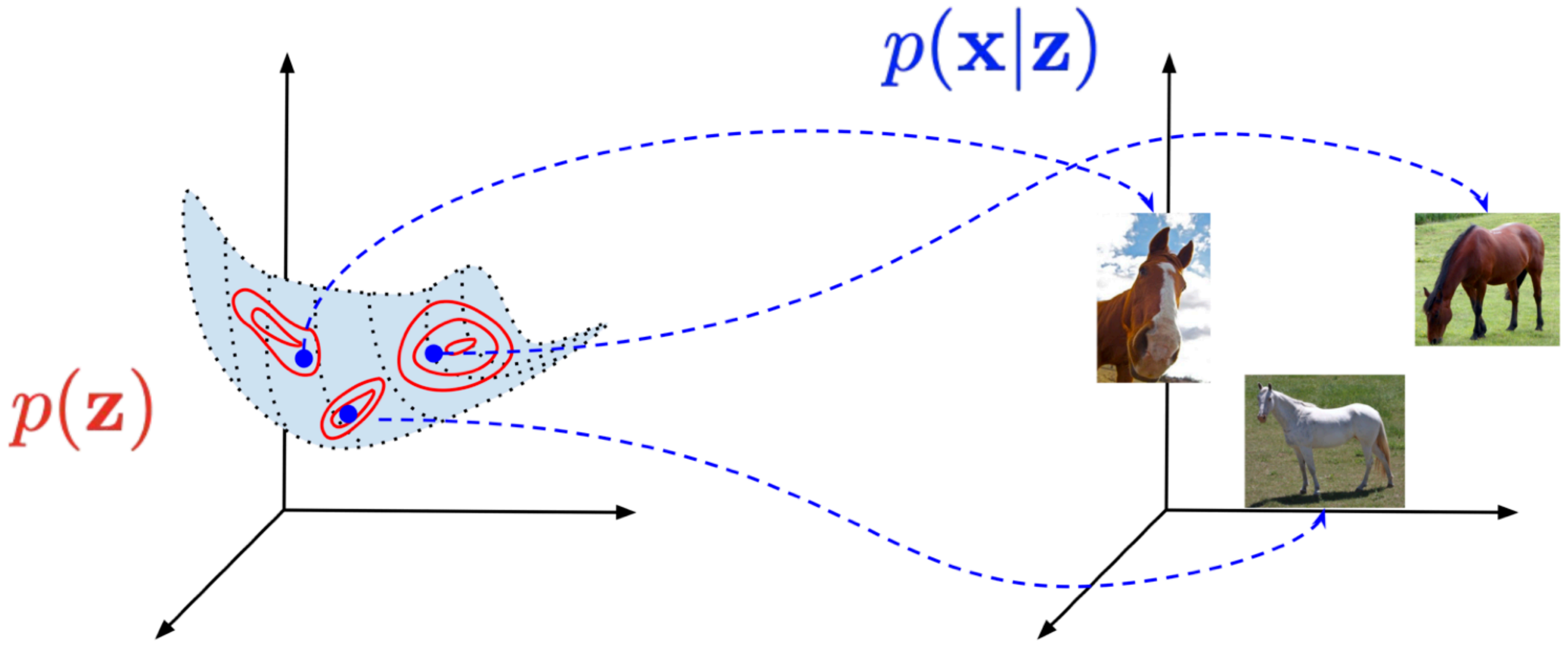
Diffusion models
Energy-based models

Variational Auto-encoders

Let's consider a latent variable model where we distinguish:

- latent variables $\mathbf{z} \in \mathscr{Z}^M$
- observable variables $\mathbf{x} \in \mathscr{X}^D$

Latent variables lie on a low-dimensional manifold.



Generative process:

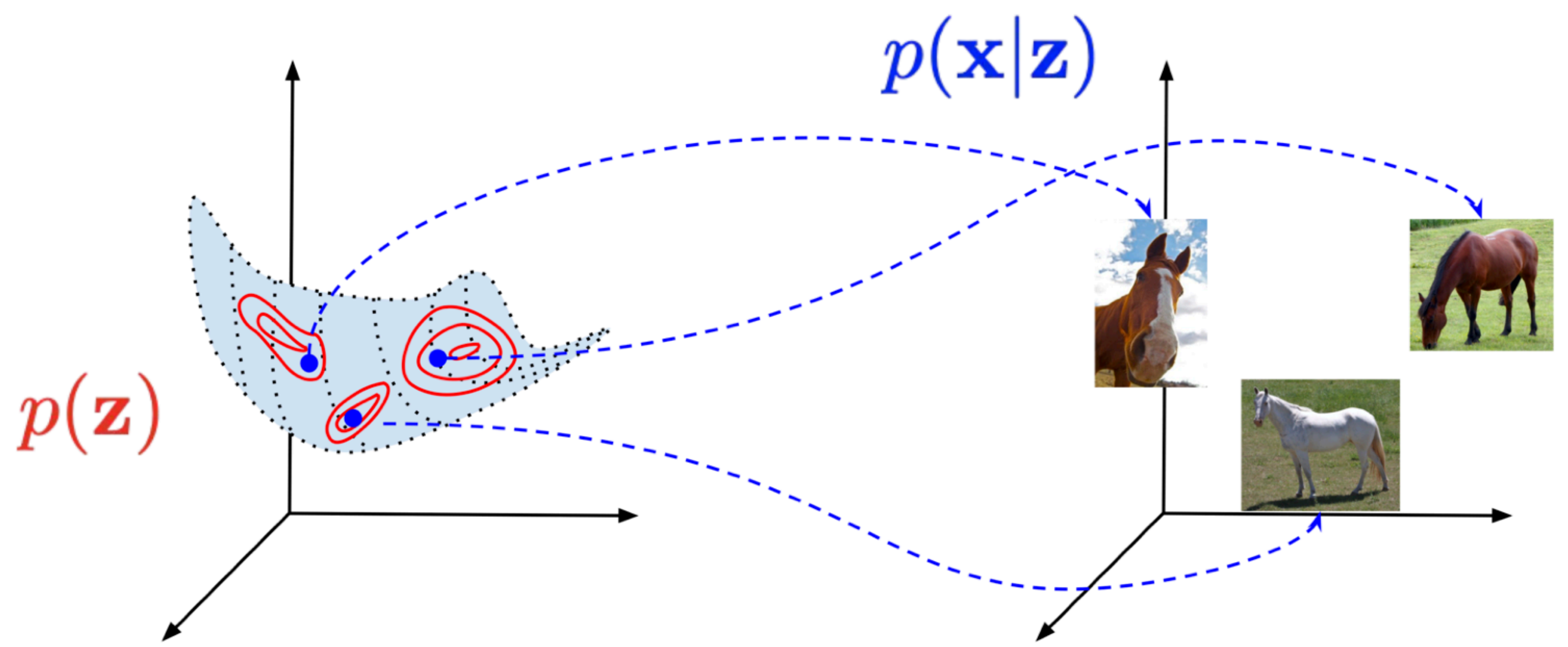
1. $\mathbf{z} \sim p(\mathbf{z})$
2. $\mathbf{x} \sim p(\mathbf{x}\,|\,\mathbf{z})$

Variational Auto-encoders

Let's consider a latent variable model where we distinguish:

- latent variables $\mathbf{z} \in \mathcal{Z}^M$
- observable variables $\mathbf{x} \in \mathcal{X}^D$

Latent variables lie on a low-dimensional manifold.



Generative process:

1. $\mathbf{z} \sim p(\mathbf{z})$
2. $\mathbf{x} \sim p(\mathbf{x}\,|\,\mathbf{z})$

The objective function:

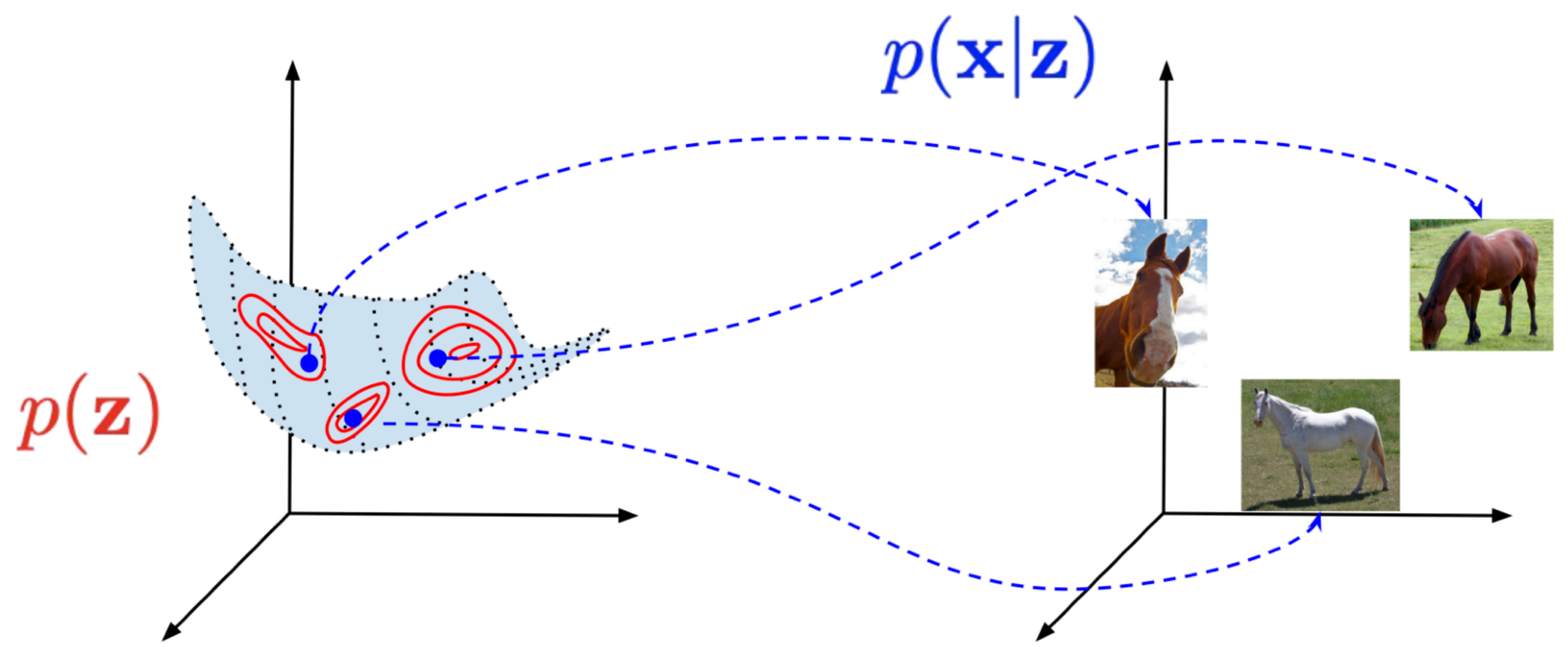
$$\ln p(\mathbf{x}) = \ln \int p(\mathbf{x}\,|\,\mathbf{z}) p(\mathbf{z}) \, \mathrm{d}\mathbf{z}$$

Variational Auto-encoders

Let's consider a latent variable model where we distinguish:

- latent variables $\mathbf{z} \in \mathscr{Z}^M$
- observable variables $\mathbf{x} \in \mathscr{X}^D$

Latent variables lie on a low-dimensional manifold.



Generative process:

1. $\mathbf{z} \sim p(\mathbf{z})$
2. $\mathbf{x} \sim p(\mathbf{x}\,|\,\mathbf{z})$

The objective function:

$$\ln p(\mathbf{x}) = \ln \int p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})\ \mathrm{d}\mathbf{z}$$

The integral is intractable…

Variational Auto-encoders

$$\ln p(\mathbf{x}) \geq \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln p(\mathbf{x}\,|\,\mathbf{z})\right] - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln q_\phi(\mathbf{z}\,|\,\mathbf{x}) - \ln p(\mathbf{z})\right]$$

ELBO: Evidence Lower Bound

Kingma, D.P., and Welling, M. "Auto-encoding variational bayes." ICLR 2014

Variational Auto-encoders

$$\ln p(\mathbf{x}) \geq \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})} \left[\ln p(\mathbf{x} \,|\, \mathbf{z}) \right] - \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})} \left[\ln q_\phi(\mathbf{z} \,|\, \mathbf{x}) - \ln p(\mathbf{z}) \right]$$

We consider amortized inference: $q_\phi(\mathbf{z} \,|\, \mathbf{x})$

In other words, a single parameterization for each new input **x**.

Kingma, D.P., and Welling, M. "Auto-encoding variational bayes." ICLR 2014

Variational Auto-encoders

$$\ln p(\mathbf{x}) \geq \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln p(\mathbf{x}\,|\,\mathbf{z})\right] - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln q_\phi(\mathbf{z}\,|\,\mathbf{x}) - \ln p(\mathbf{z})\right]$$

We consider amortized inference: $q_\phi(\mathbf{z}\,|\,\mathbf{x})$

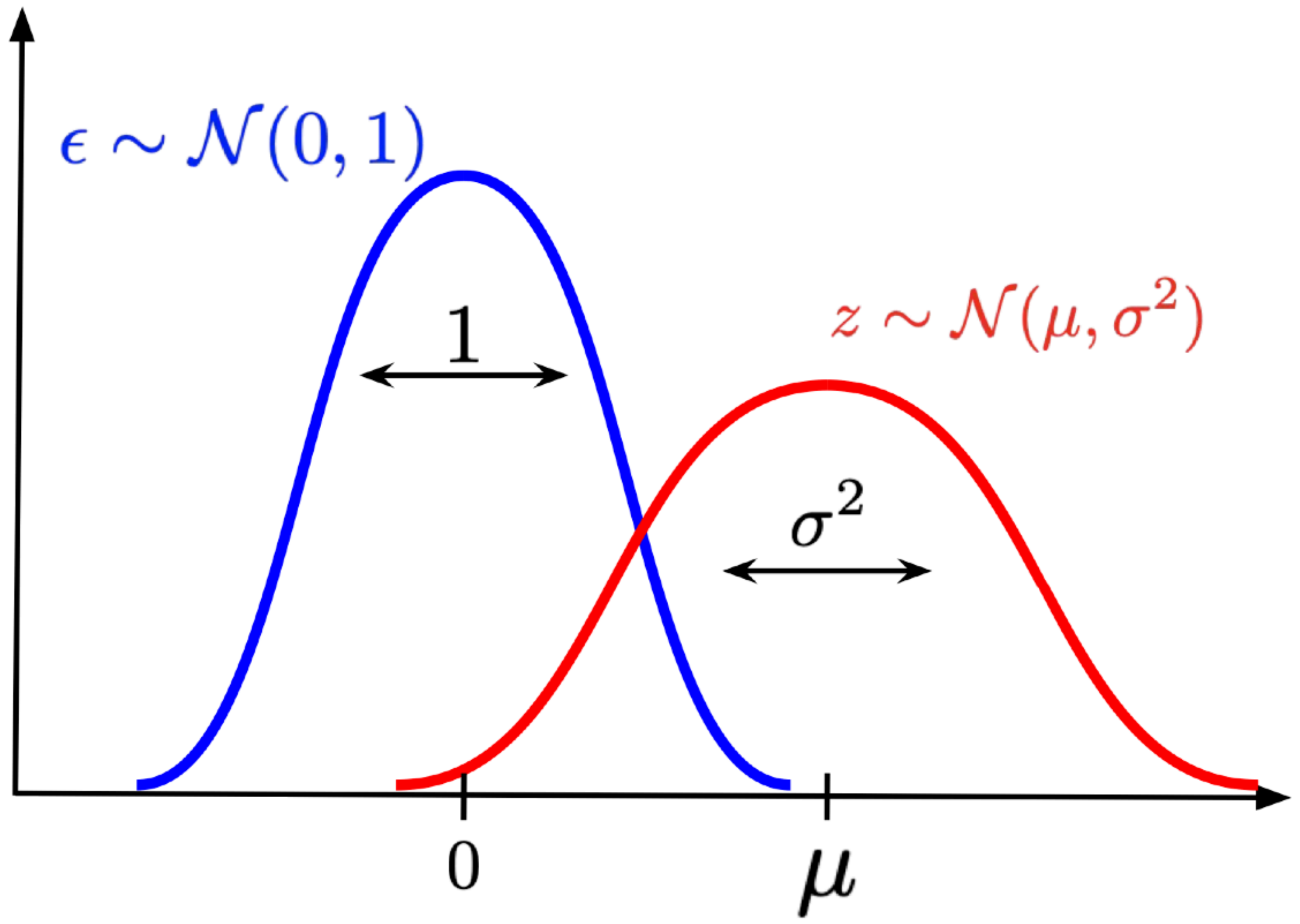
In other words, a single parameterization for each new input **x**.

Moreover, we use the **reparameterization trick**.

Every Gaussian variable could be defined as:

$z = \mu + \sigma \cdot \varepsilon$

where $\varepsilon \sim \mathcal{N}(0,1)$



Kingma, D.P., and Welling, M. "Auto-encoding variational bayes." ICLR 2014

Variational Auto-encoders

$$\ln p(\mathbf{x}) \geq \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln p(\mathbf{x}\,|\,\mathbf{z})\right] - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln q_\phi(\mathbf{z}\,|\,\mathbf{x}) - \ln p(\mathbf{z})\right]$$

We consider amortized inference: $q_\phi(\mathbf{z}\,|\,\mathbf{x})$

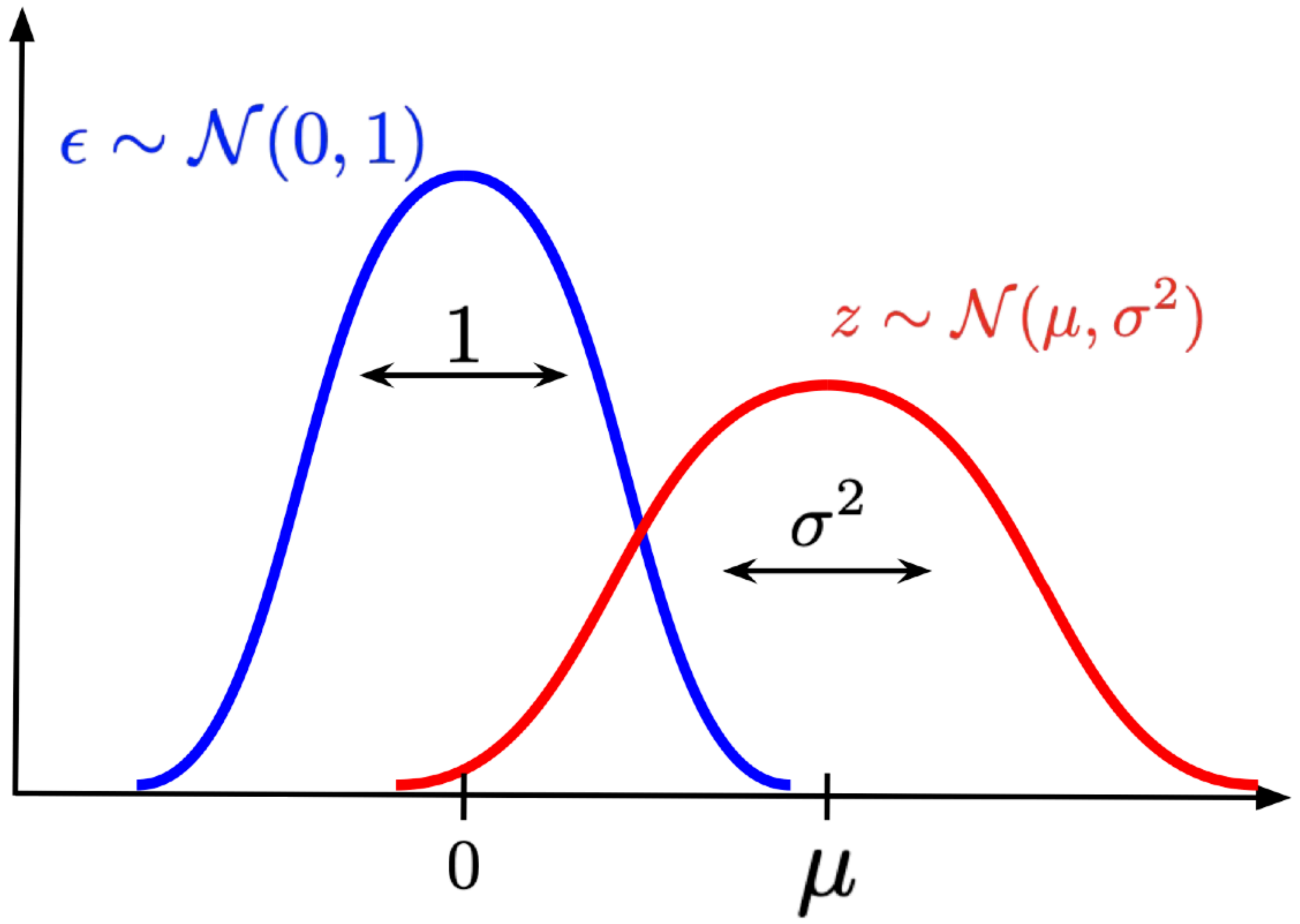
In other words, a single parameterization for each new input **x**.

Moreover, we use the **reparameterization trick**.

It reduces the variance of the gradients.

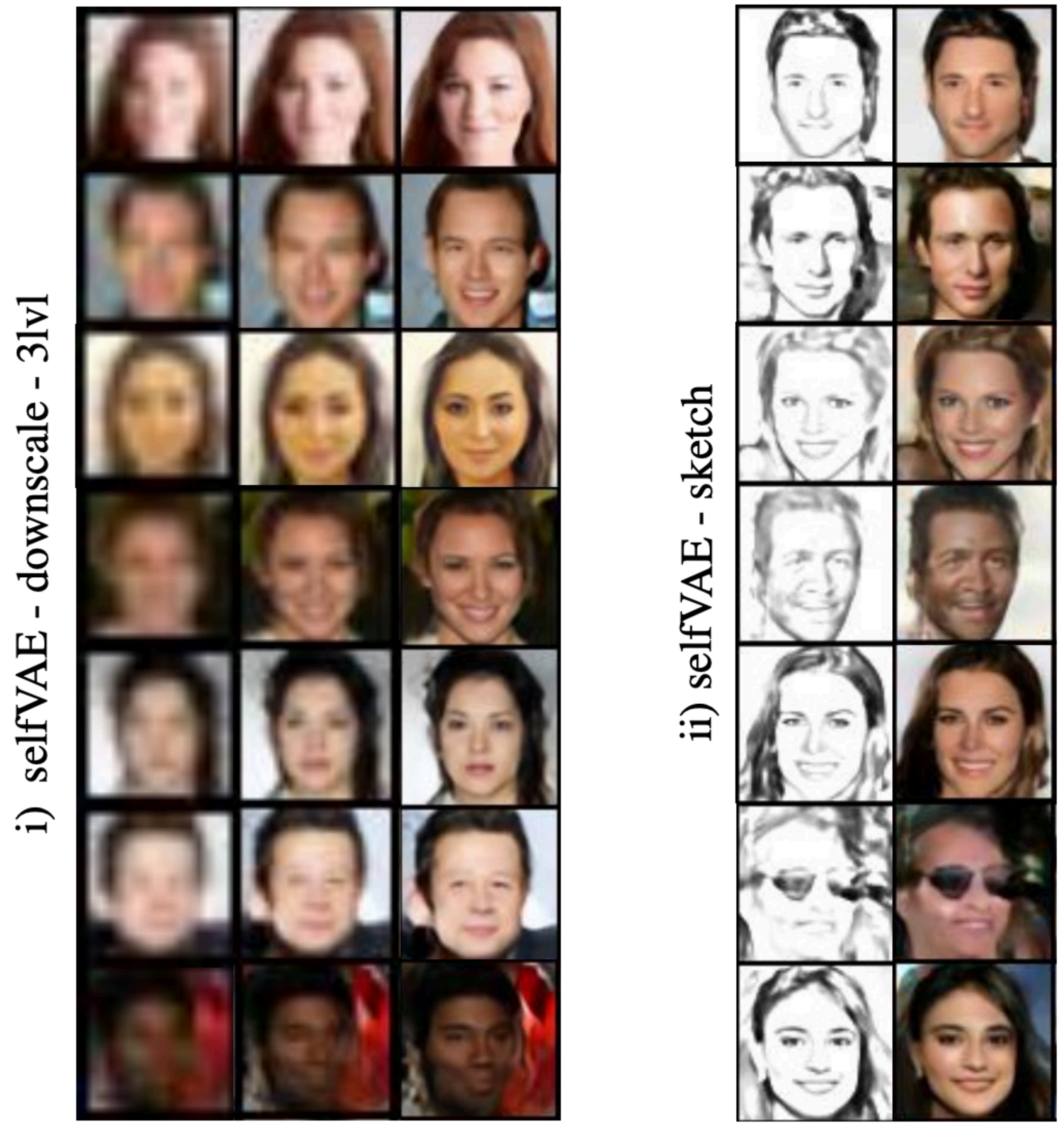
It allows to get randomness outside **z**.

$z = \mu + \sigma \cdot \varepsilon$

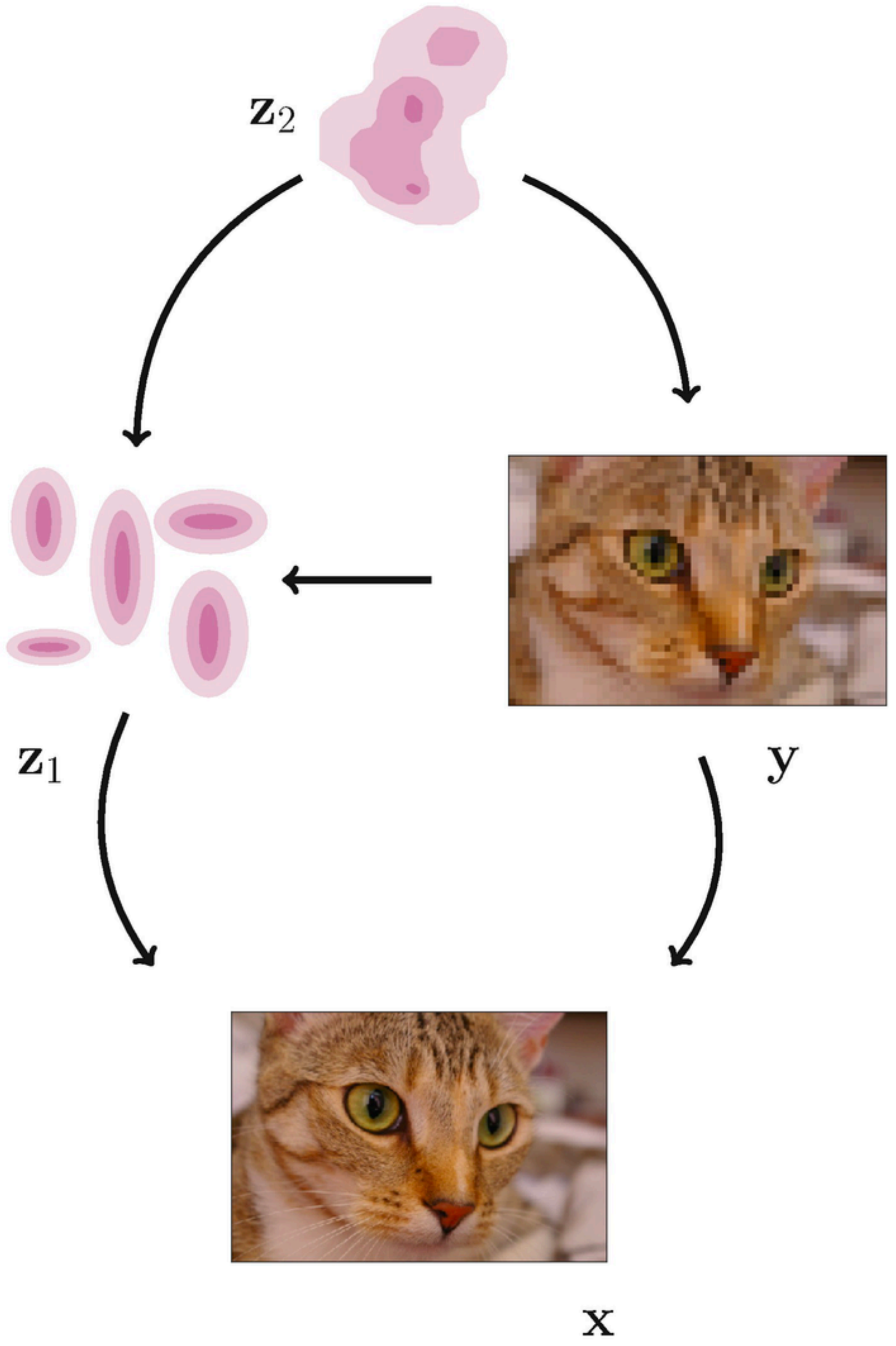


Kingma, D.P., and Welling, M. "Auto-encoding variational bayes." ICLR 2014

Hierarchical Variational Auto-encoders



Generations

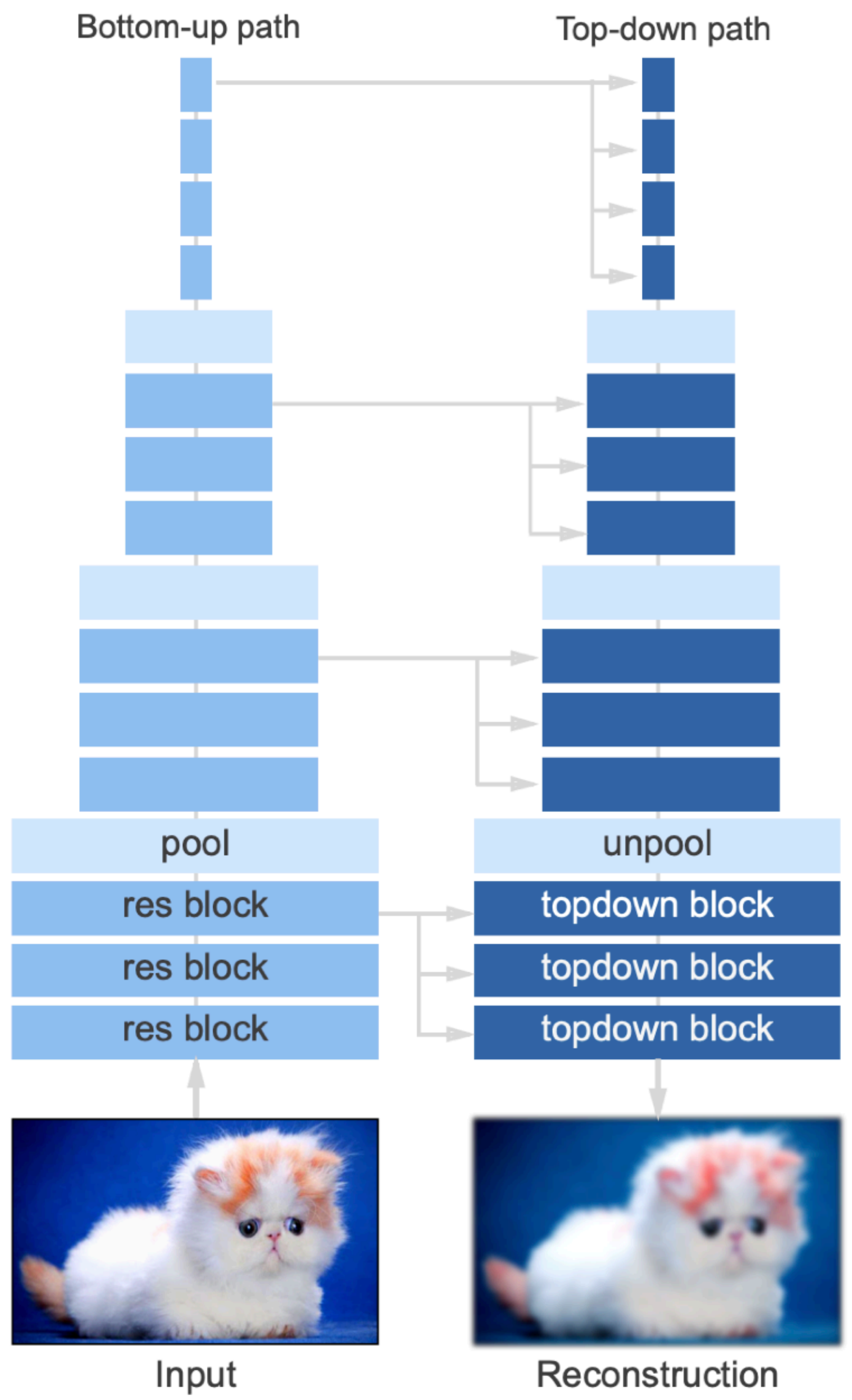


Hierarchical VAE

Gatopoulos, I., and Tomczak, J.M., "Self-Supervised Variational Auto-Encoders." Entropy (2021).

Top-down Variational Auto-encoders

Generations



Hierarchical VAE

Child, R. "Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images." ICLR 2021

Diffusion-based deep generative models

$$q_\phi(\mathbf{z}_i \mid \mathbf{z}_{i-1}) = \mathcal{N}\left(\mathbf{z}_i \mid \sqrt{1-\beta_i}\mathbf{z}_{i-1}, \beta_i \mathbf{I}\right)$$

Forward diffusion (FIXED!)

Backward diffusion (learnable)

Sohl-Dickstein J., Weiss E., Maheswaranathan N., & Ganguli S.. Deep unsupervised learning using nonequilibrium thermodynamics. ICML 2015
Ho, J., Jain, A., & Abbeel, P. Denoising diffusion probabilistic models. NeurIPS 2020

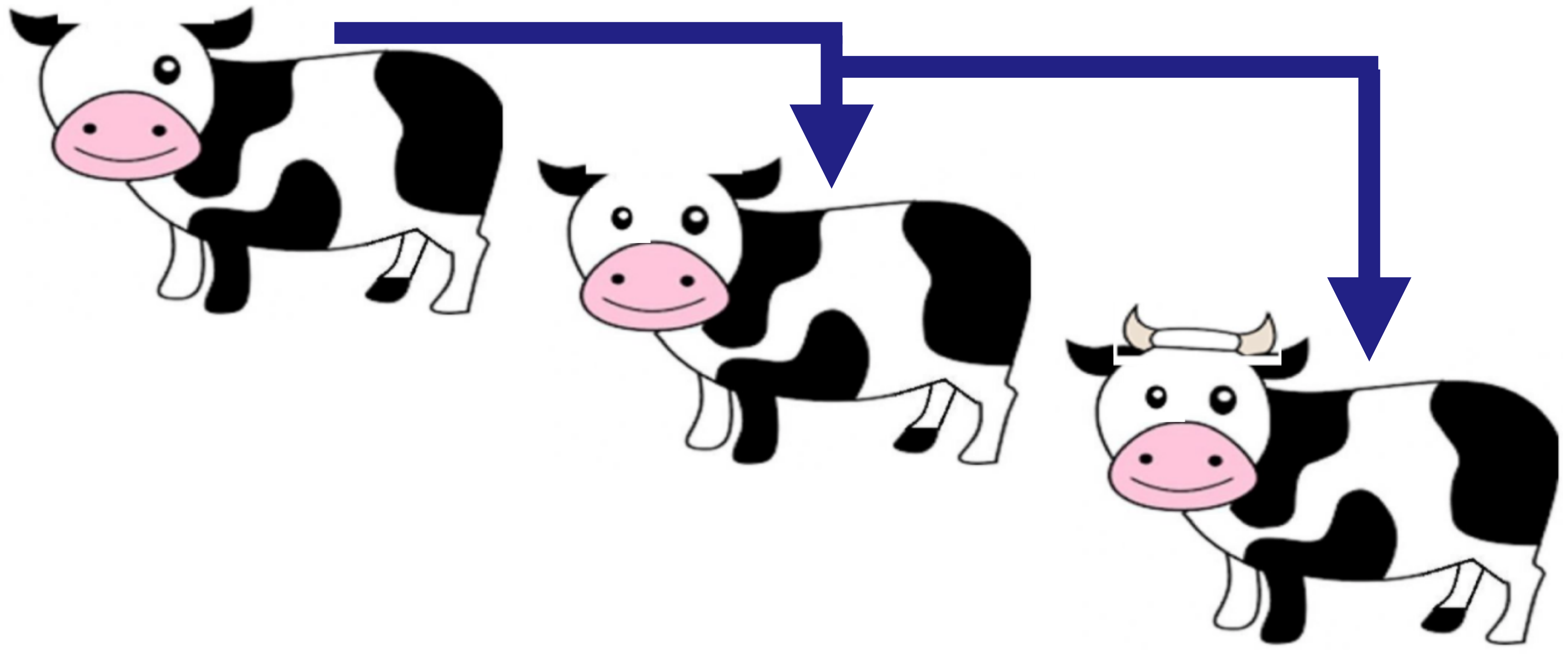
Generative AI and (spherical) cows

A high-dim object

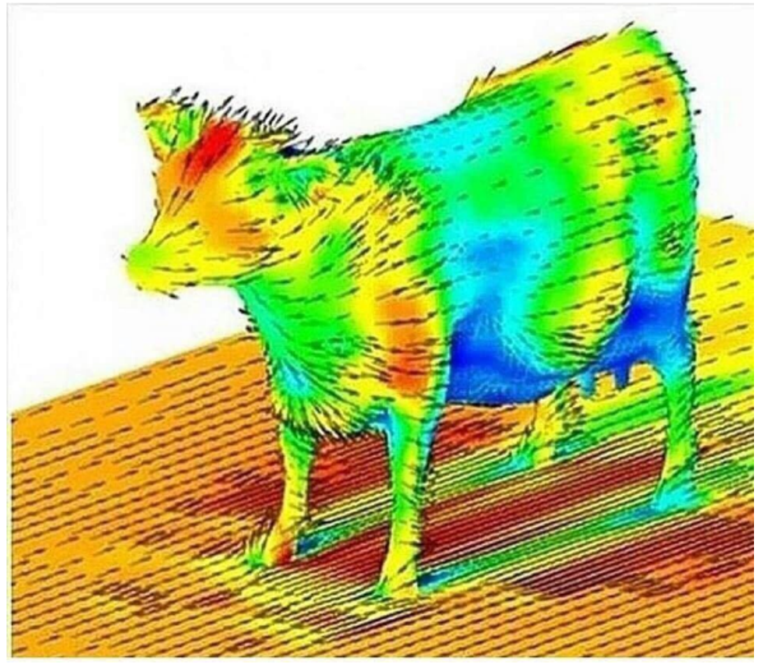
Latent Variable Models

Flow-based models

Goal: $p(\mathbf{x})$



Autoregressive Models



Diffusion models
Energy-based models

Flows (Flow-based models)

- We change a random variable **x** to another random variable **z** using invertible transformations, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^D$:

$$p(\mathbf{x}) = \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) \prod_{i=1}^{K} \left| \mathbf{J}_{f_i}(z_{i-1}) \right|^{-1}$$

Flows (Flow-based models)

- We change a random variable **x** to another random variable **z** using invertible transformations, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^D$:

$$p(\mathbf{x}) = \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) \prod_{i=1}^{K} \left| \mathbf{J}_{f_i}(z_{i-1}) \right|^{-1}$$

Flows (Flow-based models)

- We change a random variable **x** to another random variable **z** using invertible transformations, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^D$:

$$p(\mathbf{x}) = \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) \prod_{i=1}^{K} \left| \mathbf{J}_{f_i}(z_{i-1}) \right|^{-1}$$

Flows (Flow-based models)

- We change a random variable **x** to another random variable **z** using invertible transformations, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^D$:

$$p(\mathbf{x}) = \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) \prod_{i=1}^{K} \left| \mathbf{J}_{f_i}(z_{i-1}) \right|^{-1}$$

Flows (Flow-based models)

- We change a random variable **x** to another random variable **z** using invertible transformations, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^D$: Known (Gaussian)

$$p(\mathbf{x}) = \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) \prod_{i=1}^{K} \left| \mathbf{J}_{f_i}(z_{i-1}) \right|^{-1}$$

Simple distribution

Flows (Flow-based models)

- We change a random variable **x** to another random variable **z** using invertible transformations, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^D$:

$$p(\mathbf{x}) = \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) \prod_{i=1}^{K} \left| \mathbf{J}_{f_i}(z_{i-1}) \right|^{-1}$$

Jacobian must be tractable

Flows (Flow-based models)

- We change a random variable **x** to another random variable **z** using invertible transformations, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^D$:

$$p(\mathbf{x}) = \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) \prod_{i=1}^{K} \left| \mathbf{J}_{f_i}(z_{i-1}) \right|^{-1}$$

- Training objective:

$$\ln p(\mathbf{x}) = \ln \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) - \sum_{i=1}^{K} \ln \left| \mathbf{J}_{f_i}(z_{i-1}) \right|$$

Flows (Flow-based models): Invertible layers

Two main components

1) **Coupling layer**:

$$\mathbf{y}_a = \mathbf{x}_a$$

$$\mathbf{y}_b = \exp\left(s\left(\mathbf{x}_a\right)\right) \odot \mathbf{x}_b + t\left(\mathbf{x}_a\right)$$

is invertible by design:

$$\mathbf{x}_b = \left(\mathbf{y}_b - t(\mathbf{y}_a)\right) \odot \exp\left(-s(\mathbf{y}_a)\right)$$

$$\mathbf{x}_a = \mathbf{y}_a$$

2) **Permutation layer**

Flows (Flow-based models): Invertible layers

Two main components

1) **Coupling layer**:

$$\mathbf{y}_a = \mathbf{x}_a$$

$$\mathbf{y}_b = \exp\left(s\left(\mathbf{x}_a\right)\right) \odot \mathbf{x}_b + t\left(\mathbf{x}_a\right)$$

is invertible by design:

$$\mathbf{x}_b = \left(\mathbf{y}_b - t(\mathbf{y}_a)\right) \odot \exp\left(-s(\mathbf{y}_a)\right)$$

$$\mathbf{x}_a = \mathbf{y}_a$$

2) **Permutation layer**

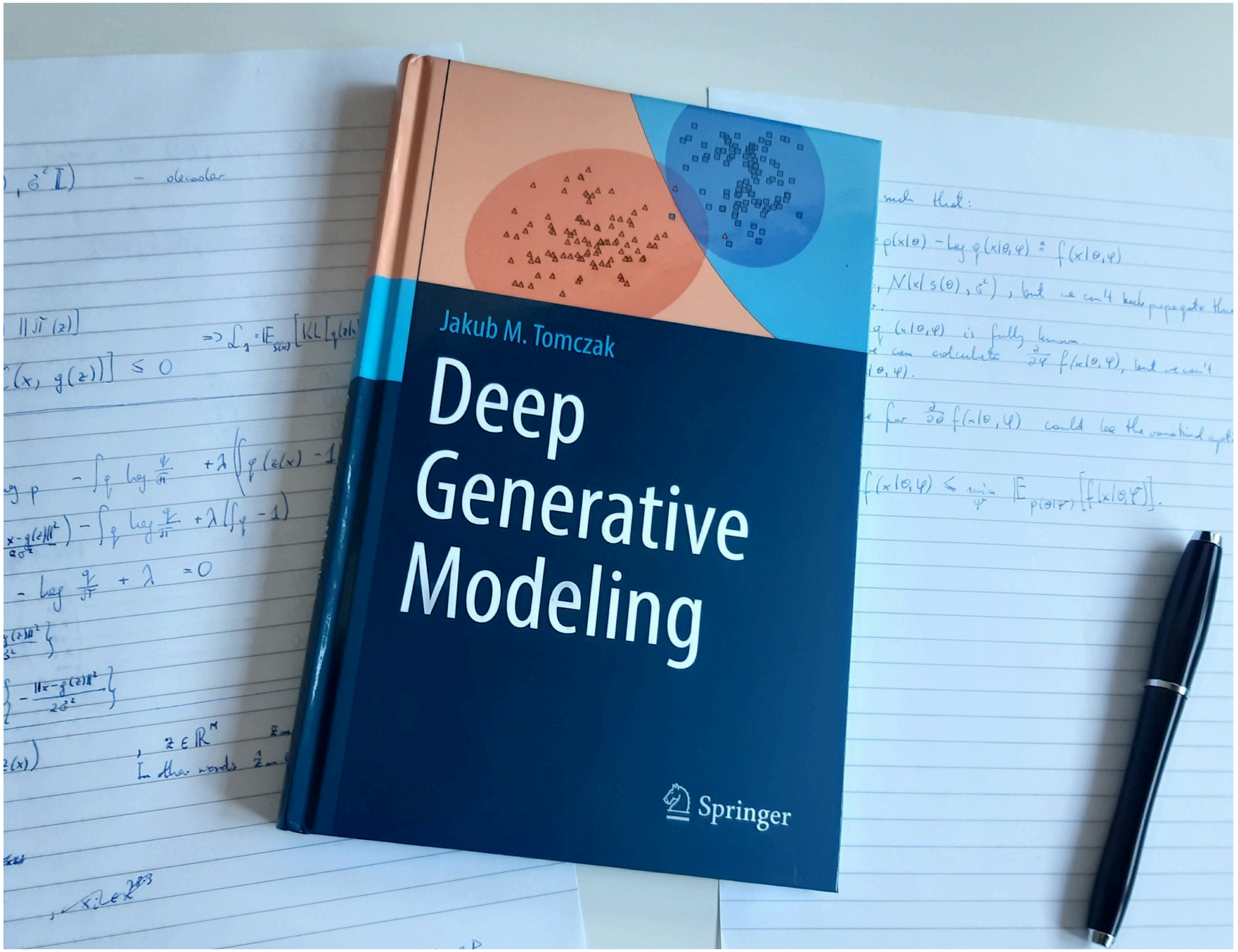
Jacobian is tractable!

$$\det(\mathbf{J}) = \prod_{j=1}^{D-d} \exp\left(s\left(\mathbf{x}_a\right)\right)_j = \exp\left(\sum_{j=1}^{D-d} s\left(\mathbf{x}_a\right)_j\right)$$

$$\det(\mathbf{J}) = 1$$

Flows (Flow-based models)

Kingma, D.P., and Prafulla D. "**Glow**: Generative flow with invertible 1x1 convolutions." NeurIPS, 2018



Deep Generative Modeling

The first comprehensive book on Generative AI.

Tomczak, J.M., (2022), “Deep Generative Modeling”, Springer Cham

Invertible DenseNets with Concatenated LipSwish

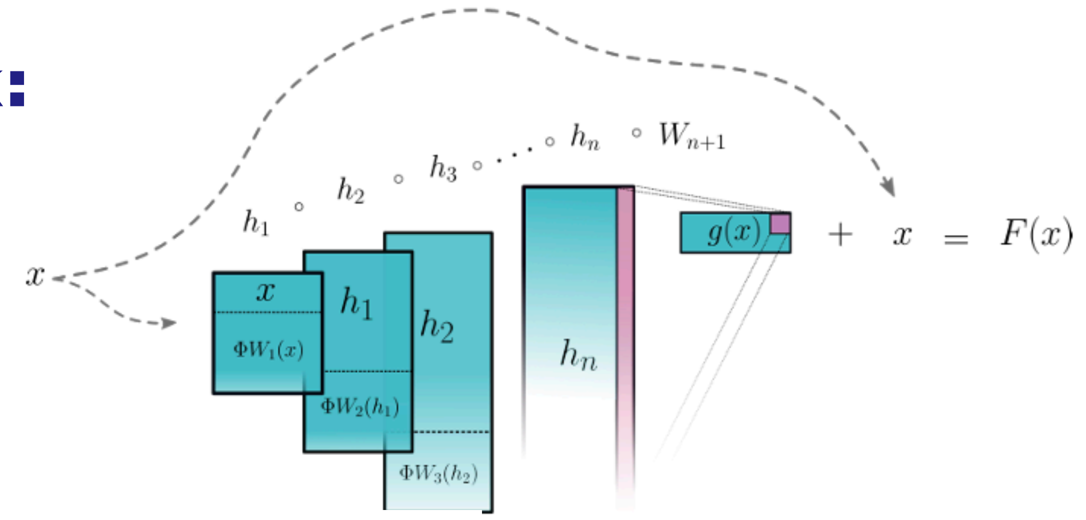


Yura Perugachi-Diaz
Sandjai Bhulai
Jakub M. Tomczak

The change-of-variables formula:

$$\ln p_X(x) = \ln p_Z(z) + \ln|\det J_F(x)|$$

DenseNet block:



$$g(x) = W_{n+1} \circ h_n \circ \cdots \circ h_1(x)$$

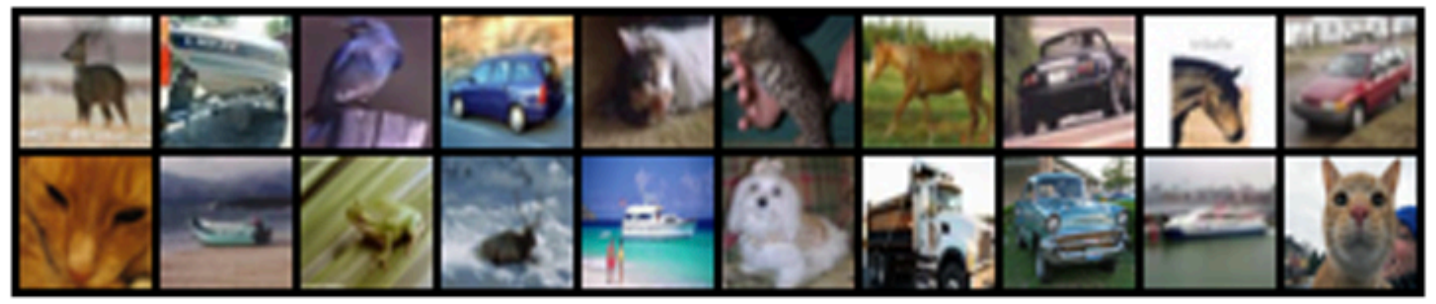
$$h_1(x) = \begin{bmatrix} x \\ \phi(W_1 x) \end{bmatrix},\ h_2(h_1(x)) = \begin{bmatrix} h_1(x) \\ \phi(W_2 h_1(x)) \end{bmatrix}$$

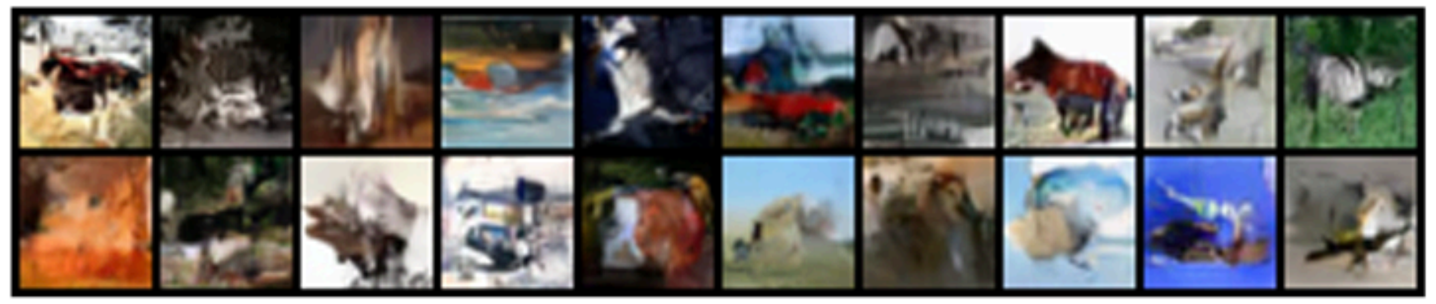
Concatenated LipSwish:

$$\Phi(x) = \begin{bmatrix} \phi_1(x) \\ \phi_2(x) \end{bmatrix} = \begin{bmatrix} \text{LipSwish}(x) \\ \text{LipSwish}(-x) \end{bmatrix}, \quad \text{CLipSwish(x)} = \Phi(\text{x})/\text{Lip}(\Phi)$$

$$\sup_x \sqrt{\left(\frac{\partial \phi_1(x)}{\partial x}\right)^2 + \left(\frac{\partial \phi_2(x)}{\partial x}\right)^2}$$

Example generations:


(a) Real CIFAR10 images.


(b) Samples of i-DenseNets trained on CIFAR10.


(c) Real ImageNet32 images.


(d) Samples of i-DenseNets trained on ImageNet32.

Classification:

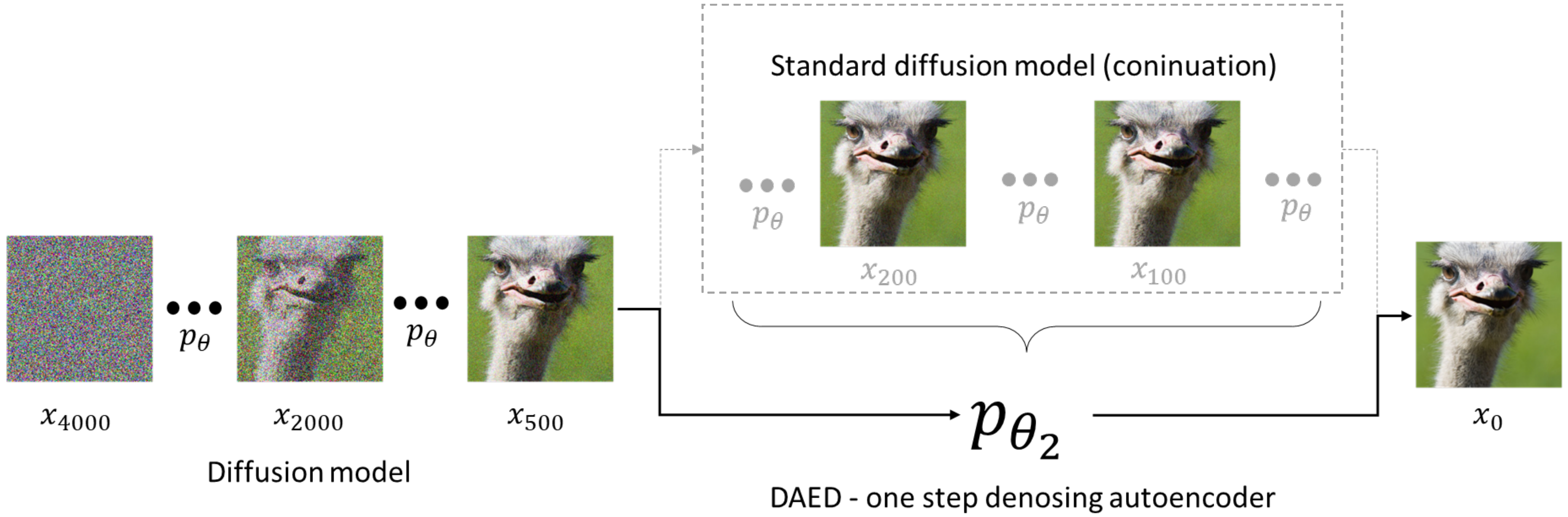
	$\lambda = 0$	$\lambda = \frac{1}{D}$		$\lambda = 1$	
Model \Evaluation	Acc ↑	Acc ↑	bpd ↓	Acc ↑	bpd ↓
Coupling	89.77%	87.58%	4.30	67.62%	3.54
+ 1×1 conv	90.82%	87.96%	4.09	67.38%	3.47
Residual Blocks (full)	91.78%	90.47%	3.62	70.32%	3.39
Dense Blocks (full)	**92.40%**	**90.79%**	**3.49**	**75.67%**	**3.31**

Perugachi-Diaz, Y., Tomczak, J.M., & Bhulai, S. (2021). Invertible densenets with concatenated lipswish. **NeurIPS 2021**

On Analyzing Generative and Denoising Capabilities of Diffusion-based Deep Generative Models

Kamil Deja
Anna Kuzina
Tomasz Trzciński
Jakub M. Tomczak

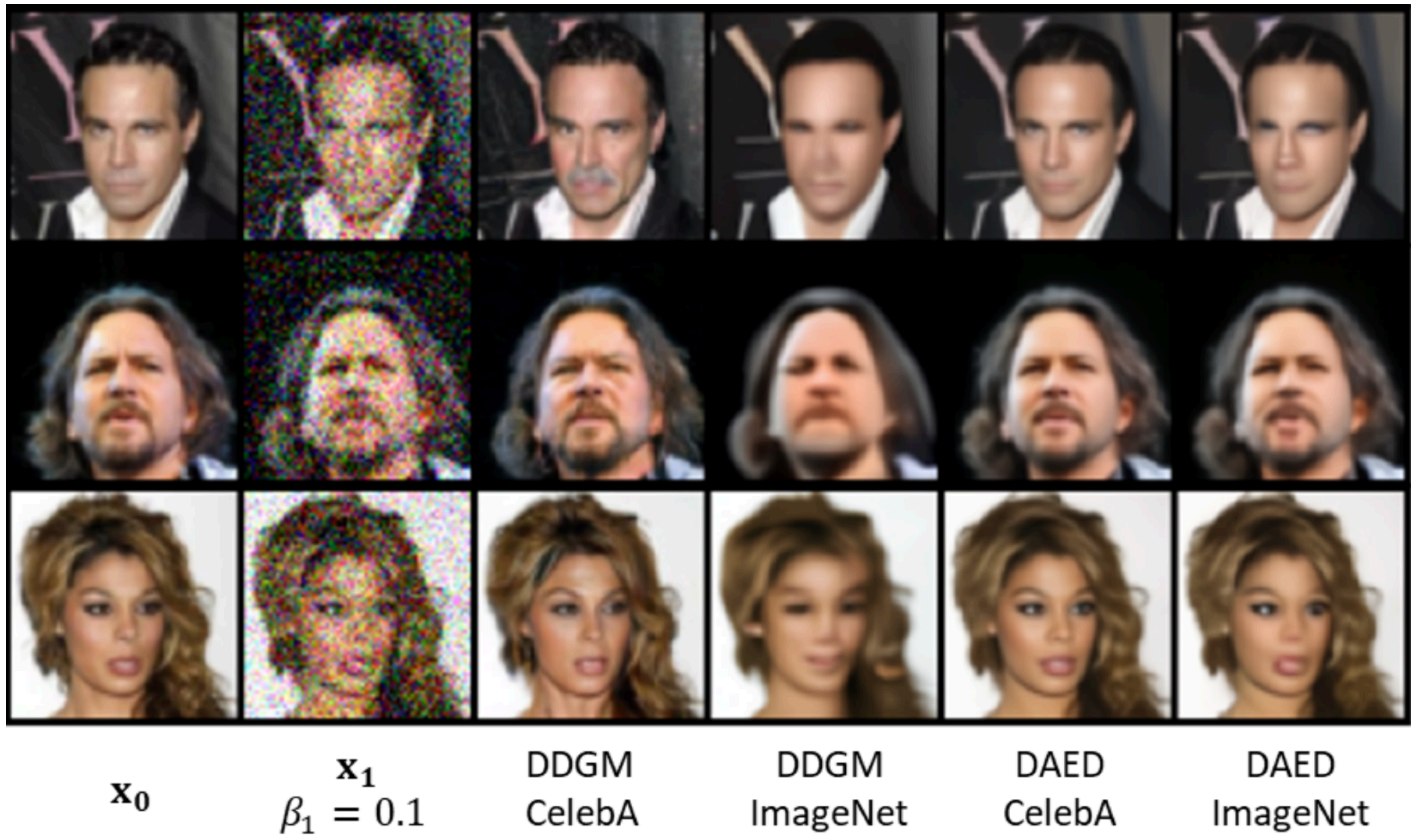
Dividing a diffusion model into a denoiser and a generator:



$$\bar{\ell}(\mathbf{x}_0;\varphi,\theta) = \mathbb{E}_{\mathbf{x}_1\sim q(\mathbf{x}_1|\mathbf{x}_0)}\left[\ln p\left(\mathbf{x}_0|f_\varphi\left(\mathbf{x}_1\right)\right) + \ln p_\theta(\mathbf{x}_1)\right]$$

$$\geq \underbrace{\mathbb{E}_{\mathbf{x}_1\sim q(\mathbf{x}_1|\mathbf{x}_0)}\left[\ln p\left(\mathbf{x}_0|f_\varphi\left(\mathbf{x}_1\right)\right)\right]}_{\ell_{\mathrm{DAE}}(\mathbf{x}_0;\varphi)} + \underbrace{\mathbb{E}_{q(\mathbf{x}_2,\ldots,\mathbf{x}_T|\mathbf{x}_1)}\left[\frac{\ln p_\theta(\mathbf{x}_1,\ldots,\mathbf{x}_T)}{q(\mathbf{x}_1,\ldots,\mathbf{x}_T|\mathbf{x}_0)}\right]}_{\ell_{\mathrm{D}}(\mathbf{x}_0;\theta)}$$

Example results:



Deja, K., Kuzina, A., Trzciński, T., & Tomczak, J.M., (2022). On Analyzing Generative and Denoising Capabilities of Diffusion-based Deep Generative Models. **(Under review)**

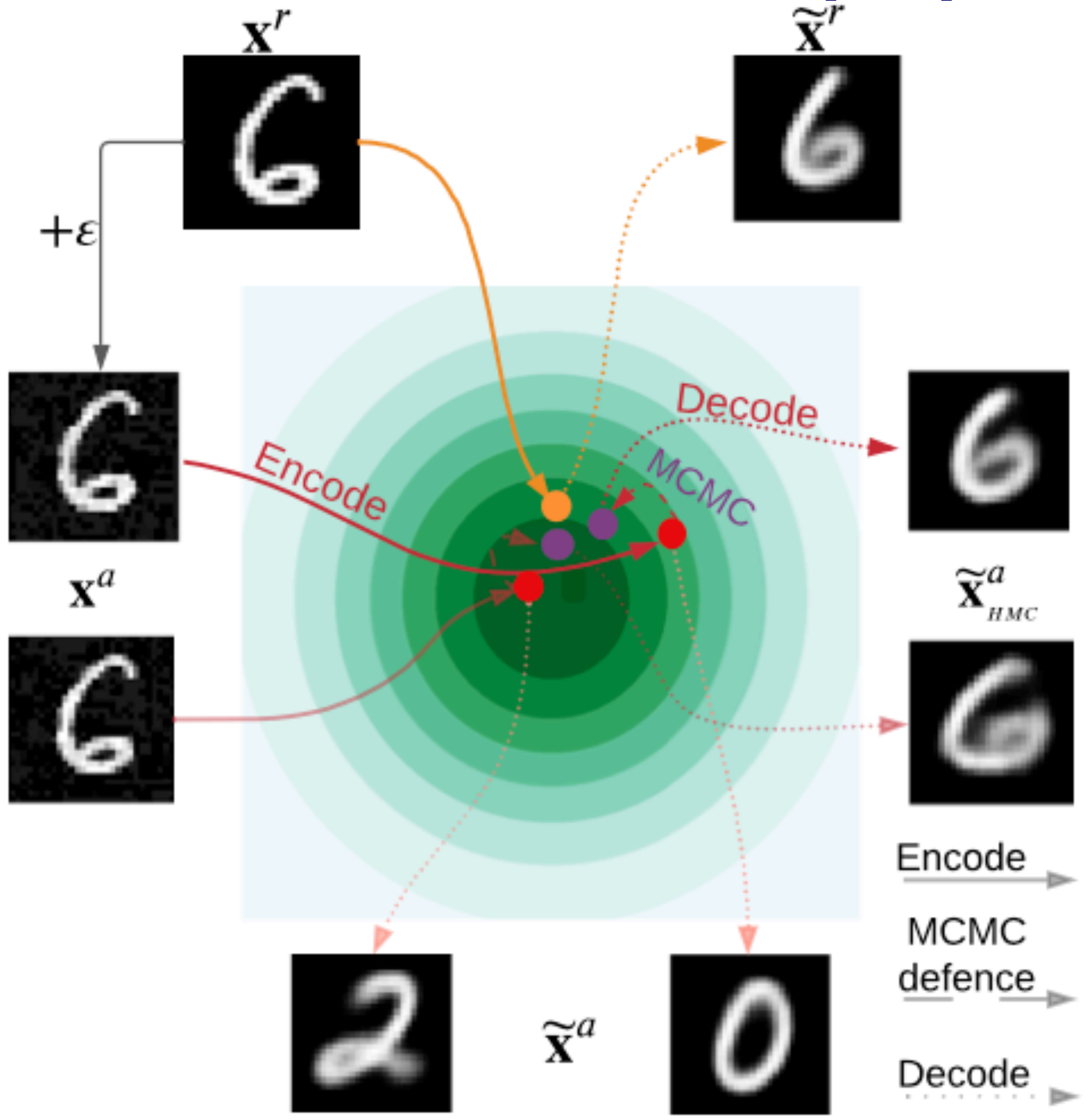
Alleviating Adversarial Attacks on Variational Autoencoders with MCMC


Anna Kuzina


Max Welling


Jakub M. Tomczak

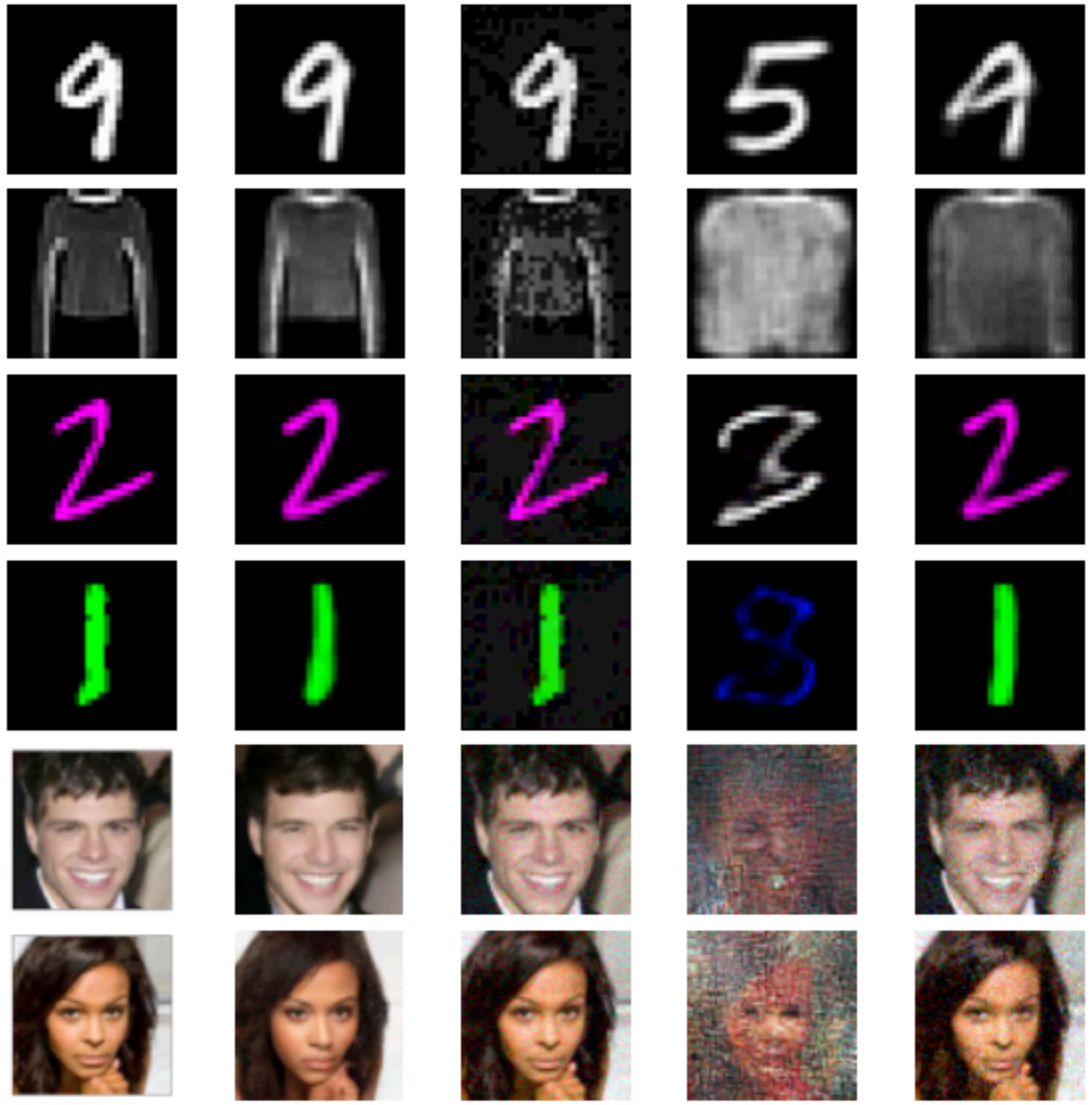
Unsupervised attacks on VAE and the proposed defense:



Theorem 1 *The upper bound on the total variation distance between samples from MCMC for a given corrupted point* $\mathbf{x}^a$, $q^{(t)}(\mathbf{z}|\mathbf{x}^a)$, *and the variational posterior for the given real point* $\mathbf{x}^r$, $q_\phi(\mathbf{z}|\mathbf{x}^r)$, *is the following:*

$$\mathrm{TV}\left[q^{(t)}(\mathbf{z}|\mathbf{x}^a), q_\phi(\mathbf{z}|\mathbf{x}^r)\right] \leq \mathrm{TV}\left[q^{(t)}(\mathbf{z}|\mathbf{x}^a), p_\theta(\mathbf{z}|\mathbf{x}^a)\right] + \sqrt{\tfrac{1}{2} D_{\mathrm{KL}}\left[q_\phi(\mathbf{z}|\mathbf{x}^r) \| p_\theta(\mathbf{z}|\mathbf{x}^r)\right]} + o(\sqrt{\|\varepsilon\|}).$$

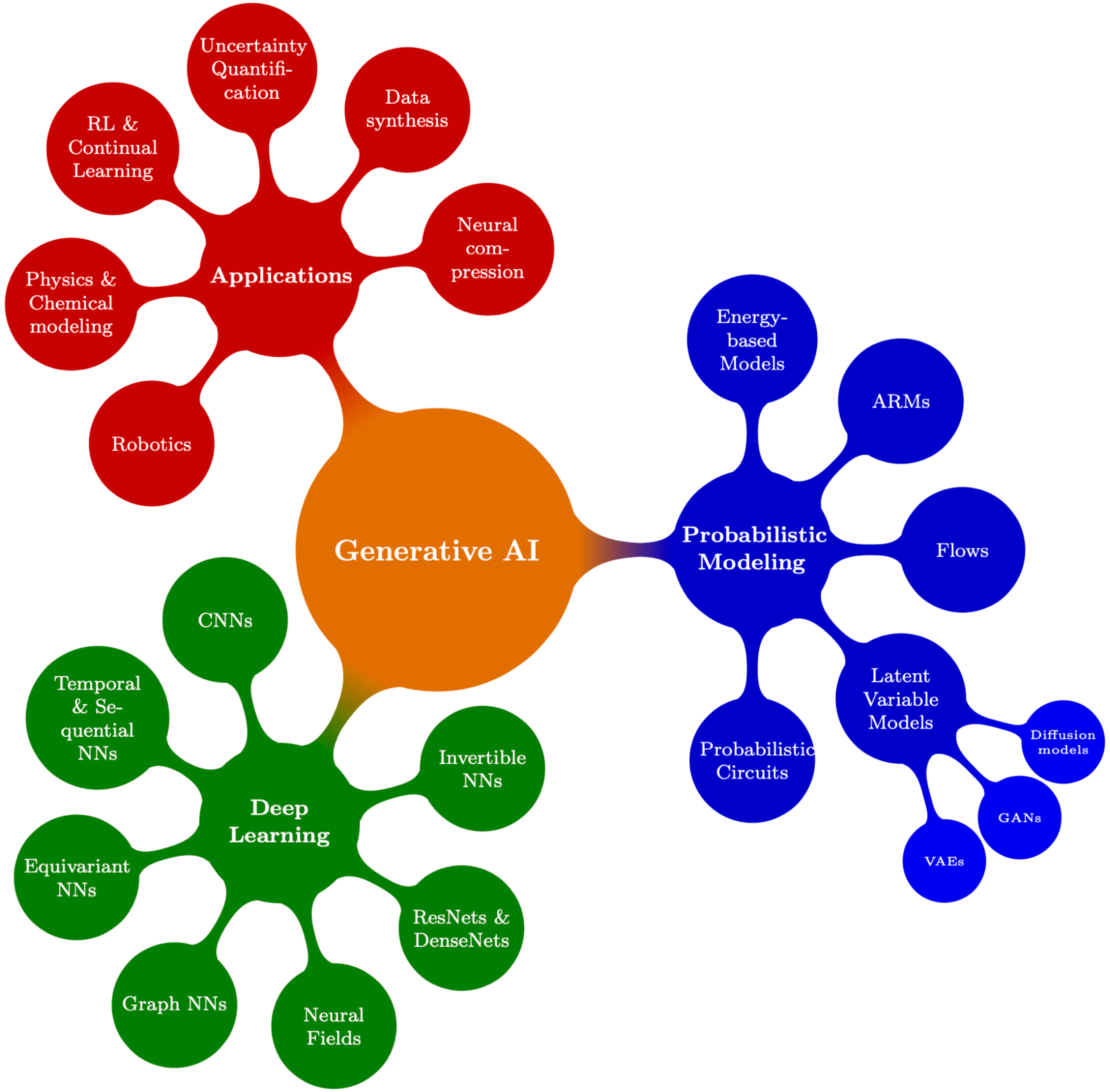
Example results:



(a) $\mathbf{x}^r$ (b) $\tilde{\mathbf{x}}^r$ (c) $\mathbf{x}^a$ (d) $\tilde{\mathbf{x}}^a$ (e) $\tilde{\mathbf{x}}^a_{\mathrm{HMC}}$

Kuzina, A., Welling, M., & Tomczak, J.M., (2022). Alleviating Adversarial Attacks on Variational Autoencoders with MCMC. **(Under review)**

Challenges in Generative AI



Challenges in Generative AI

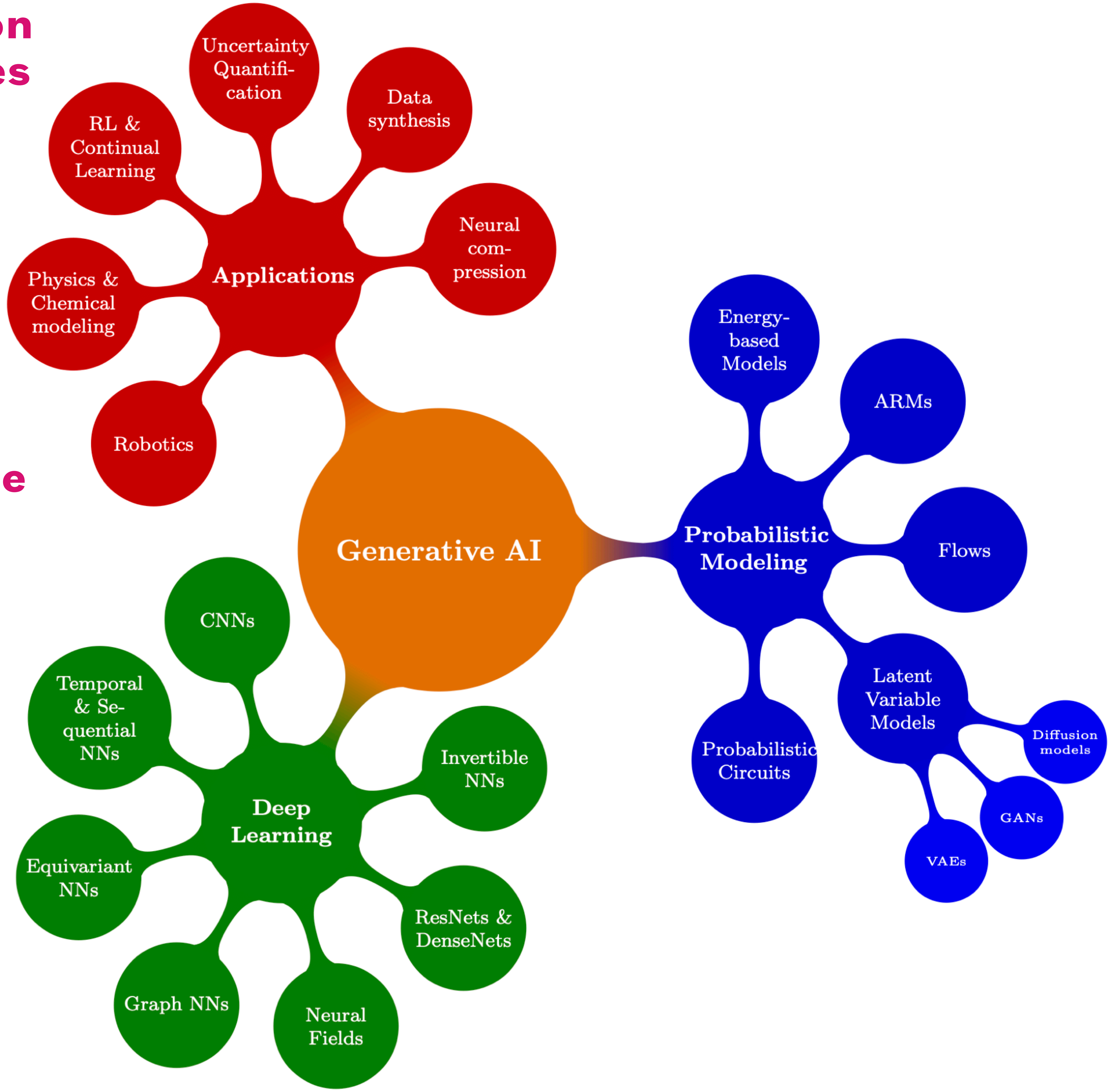
Deploying on edge devices

Reliability

Reproducibility

Inference time

Model size



Challenges in Generative AI

Deploying on edge devices

Reliability

Reproducibility

Inference time

Model size

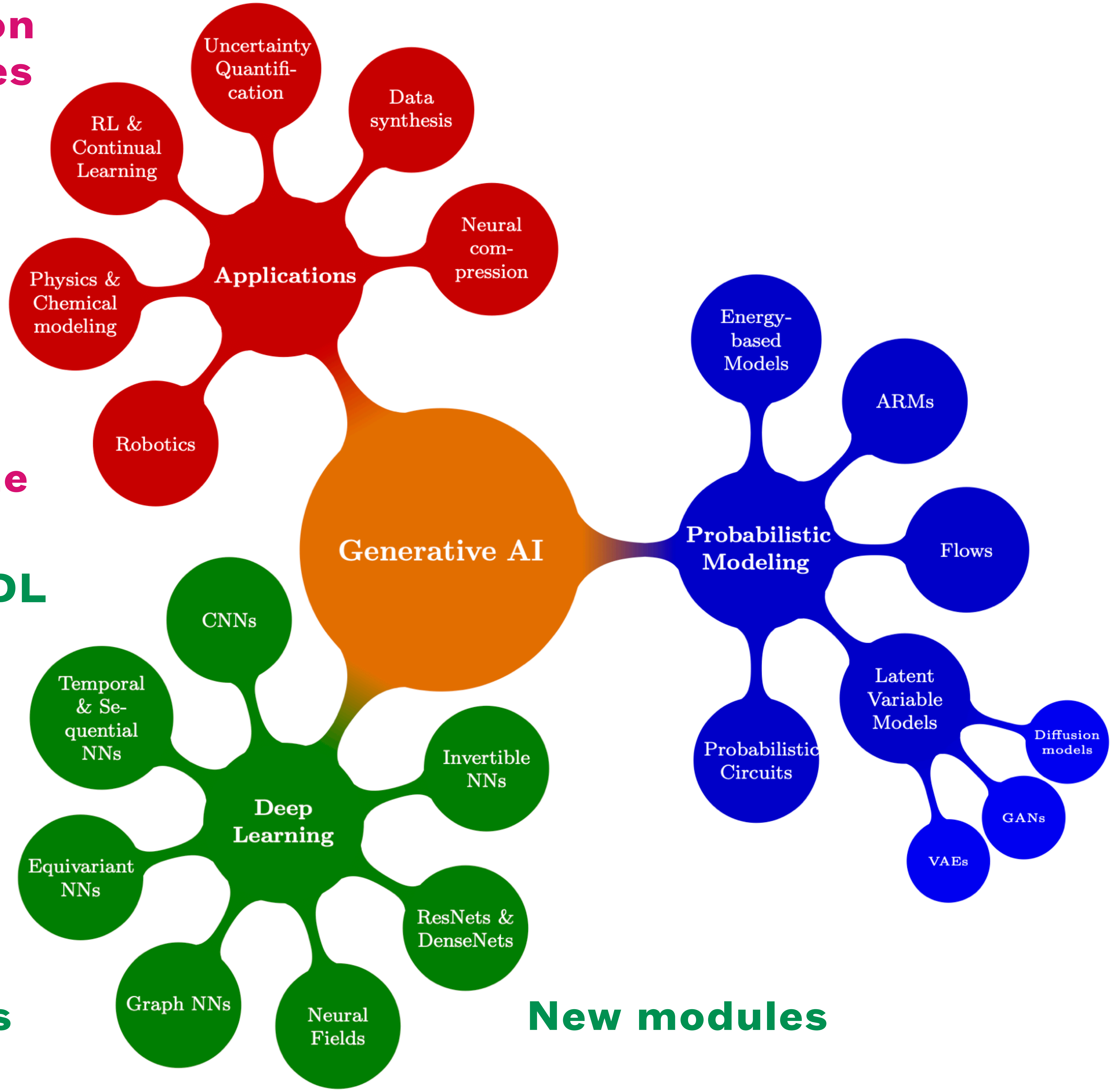
Tiny DL

Quantization

New computing paradigms

New learning methods

New architectures



New modules

Challenges in Generative AI

Deploying on edge devices

Reliability

Reproducibility

Inference time

Model size

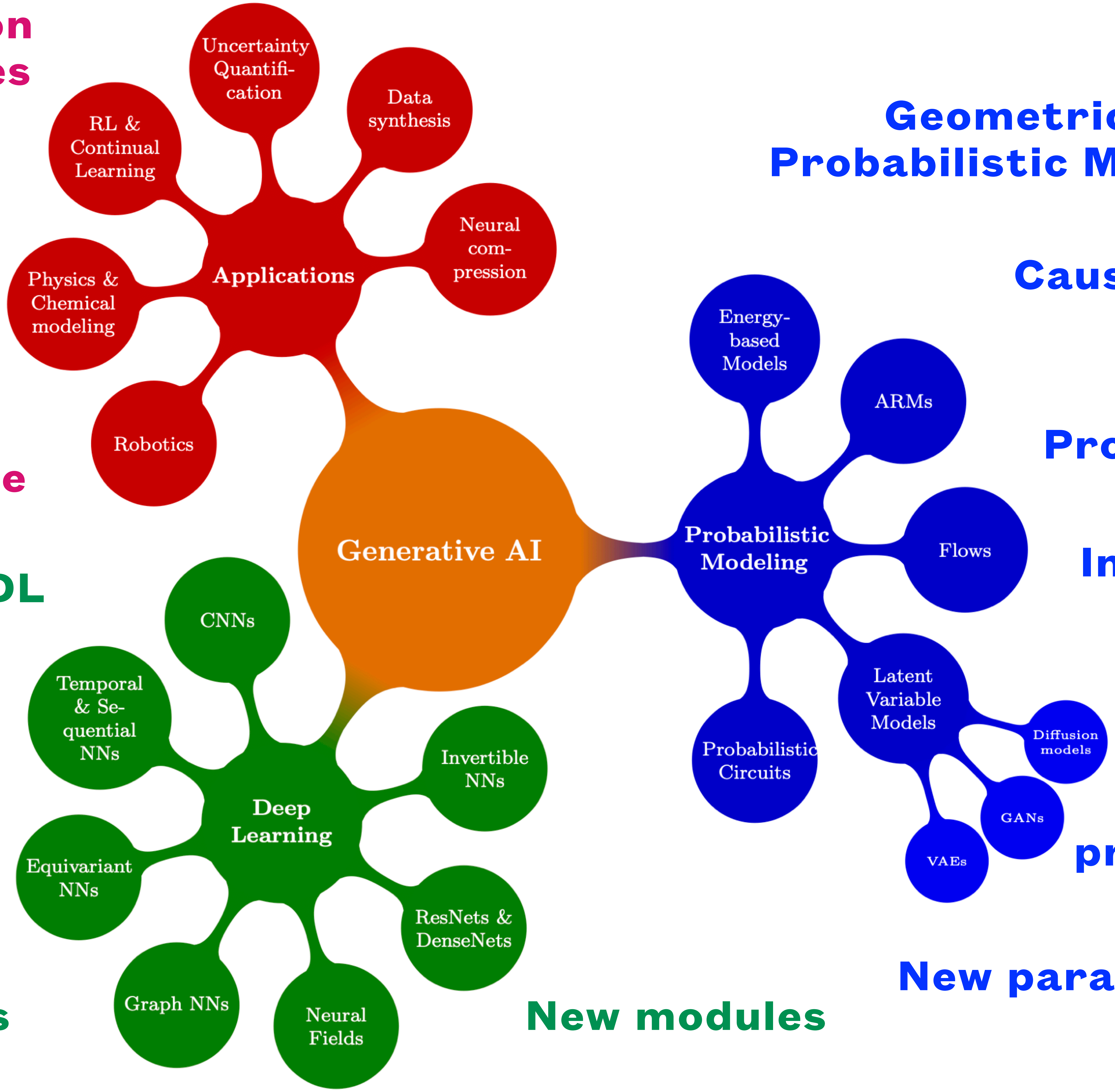
Tiny DL

Quantization

New computing paradigms

New learning methods

New architectures



New modules

Geometric Probabilistic Models

Causality theory

Discrete Probabilistic Models

Inference methods

Tractable models

Hybrid probabilistic models

New parameterizations

Thank you!

Jakub M. Tomczak
assistant professor of AI
Vrije Universiteit Amsterdam

jmk.tomczak@gmail.com
https://jmtomczak.github.io/
https://github.com/jmtomczak