The Success of Deep Generative Models

Jakub Tomczak

AMLAB, University of Amsterdam

CERN, 2018

What is AI about?

What is AI about?

Decision making:

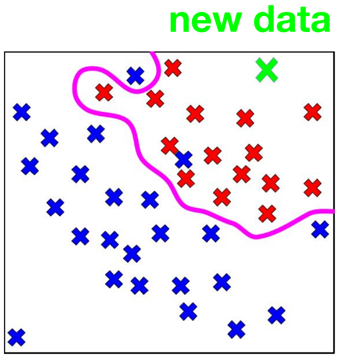
$$p(y|\mathbf{x})$$

What is AI about?

Decision making:

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

High probability
of the **red** label.
=
Highly probable decision!

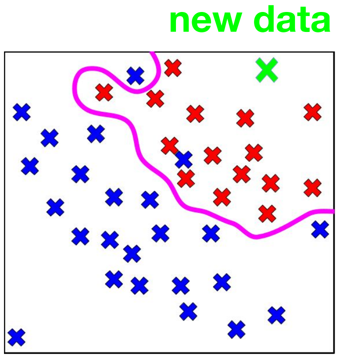


What is AI about?

Decision making:

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

High probability of the **red** label.
=
Highly probable decision!



Understanding:

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

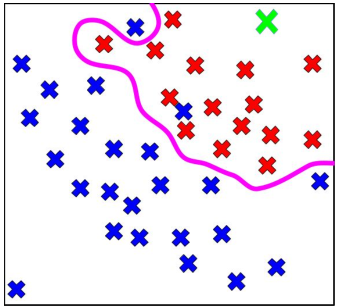
What is AI about?

Decision making:

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

High probability
of the **red** label.
=
Highly probable decision!

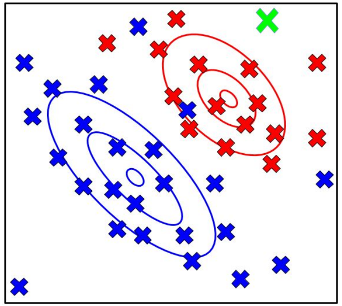
new data



Understanding:

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

High probability
of the **red** label.
x
Low probability
of the **object**
=
Uncertain decision!



What is generative modeling about?

Understanding:

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

finding underlying factors (**discovery**)

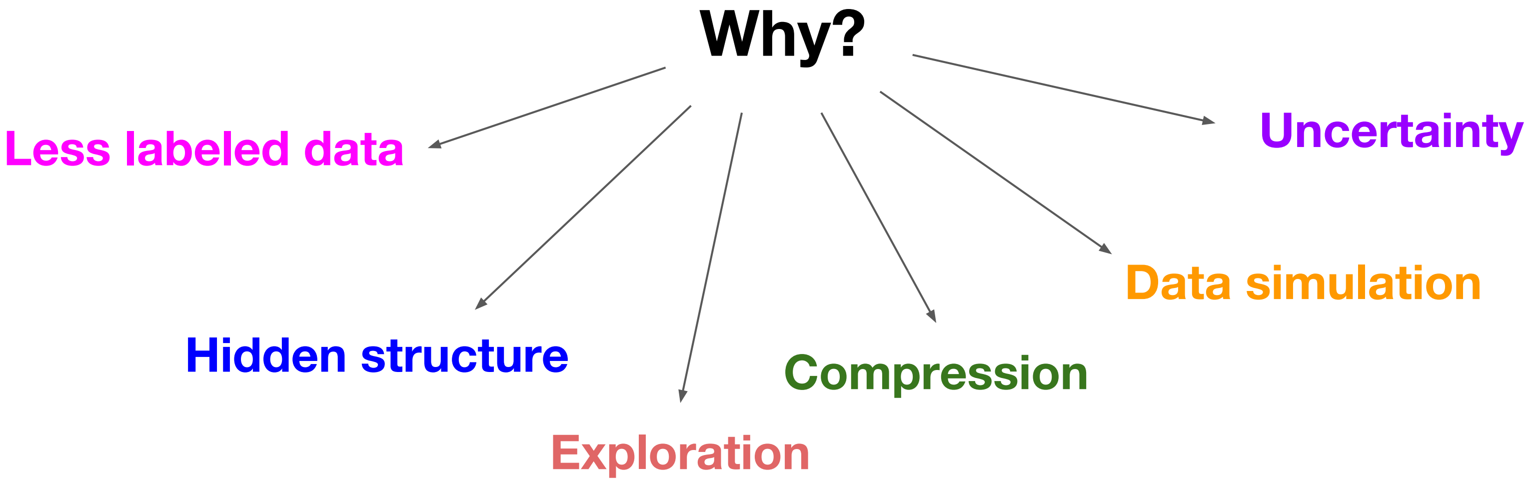
predicting and anticipating future events (**planning**)

finding analogies (**transfer learning**)

detecting rare events (**anomaly detection**)

decision making

Why generative modeling?

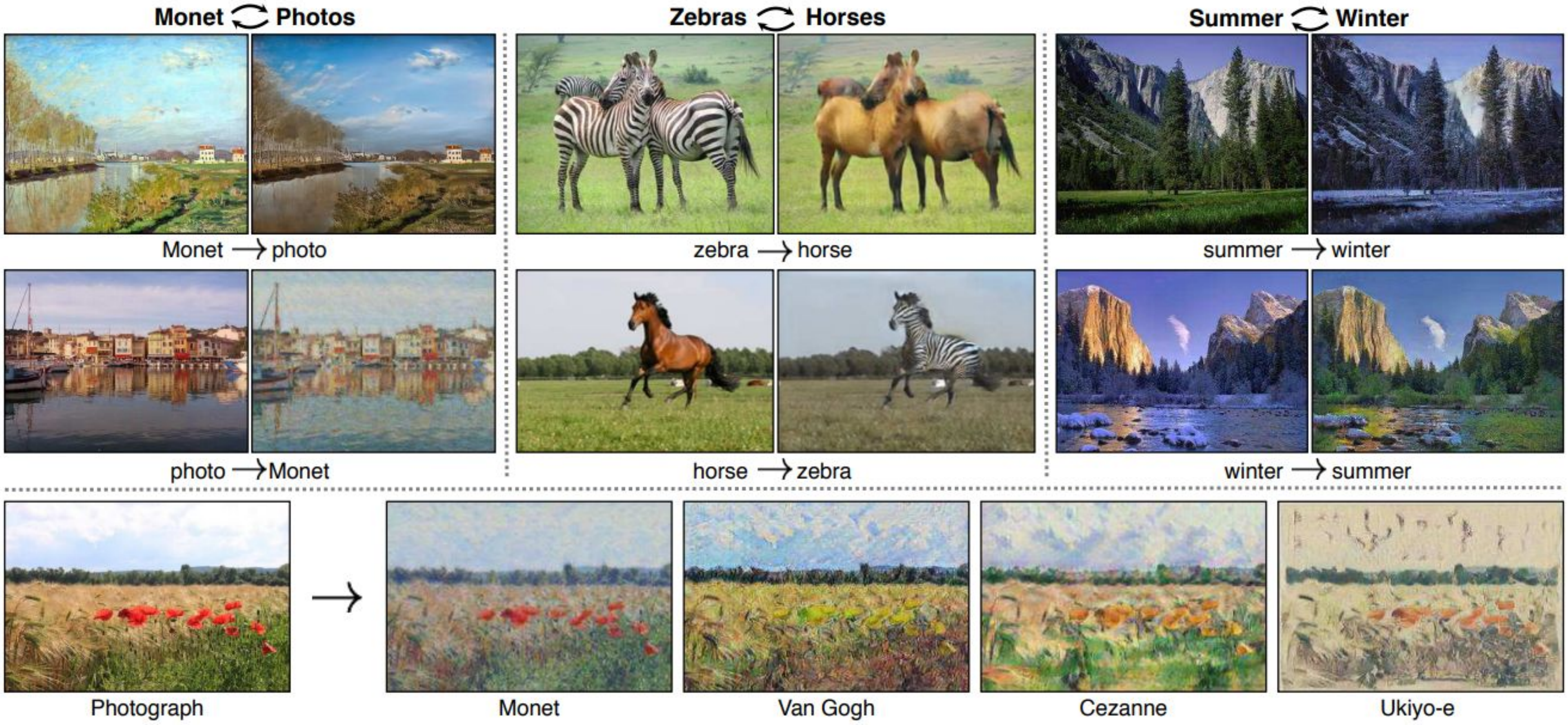


Generative modeling: **How**?

How?

- **Fully-observed (*e.g.*, PixelCNN)**
- **Latent variable models**
  - **Implicit models (*e.g.*, GANs)**
  - **Prescribed models (*e.g.*, VAE)**

Recent successes: **Style transfer**

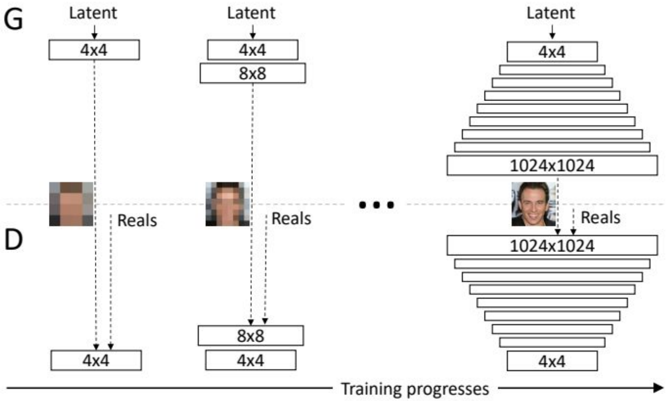


Zhu, J. Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. *CVPR 2017*.

Recent successes: **Image generation**

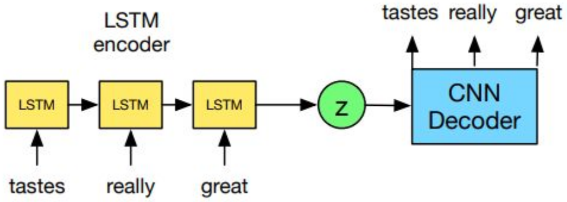
generated

real

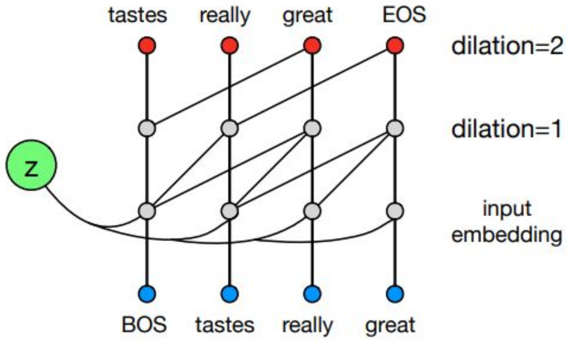


Karras, T., Aila, T., Laine, S., & Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation. *ICLR 2017*.

Recent successes: **Text generation**

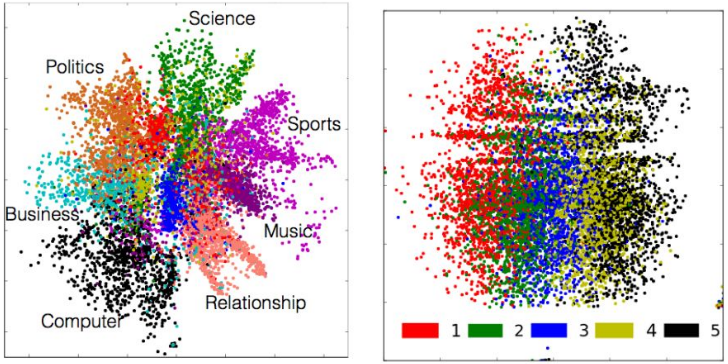


(a) VAE training graph using a dilated CNN decoder.



(b) Digram of dilated CNN decoder.

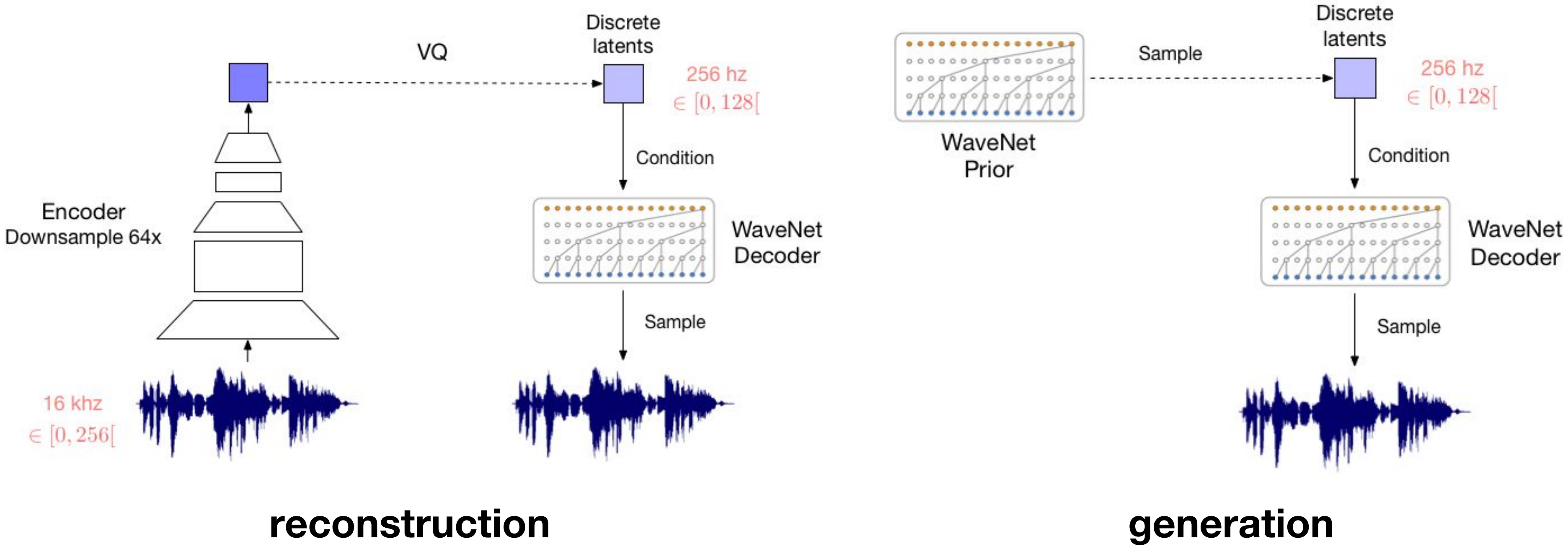
1 star	the food was good but the service was horrible . took forever to get our food . we had to ask twice for our check after we got our food . will not return .
2 star	the food was good , but the service was terrible . took forever to get someone to take our drink order . had to ask 3 times to get the check . food was ok , nothing to write about .
3 star	came here for the first time last night . food was good . service was a little slow . food was just ok .
4 star	food was good , service was a little slow , but the food was pretty good . i had the grilled chicken sandwich and it was really good . will definitely be back !
5 star	food was very good , service was fast and friendly . food was very good as well . will be back !



(a) Yahoo (b) Yelp

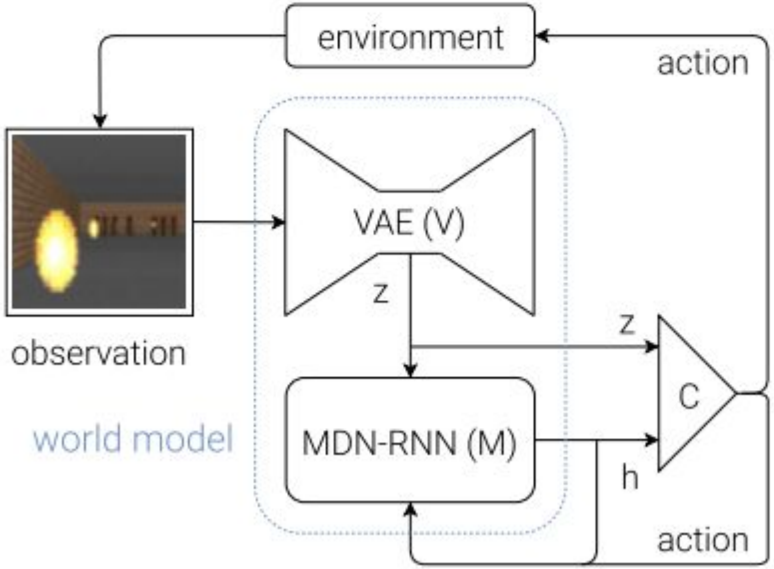
Yang, Z., Hu, Z., Salakhutdinov, R., & Berg-Kirkpatrick, T. (2017). Improved variational autoencoders for text modeling using dilated convolutions. *ICML 2017*

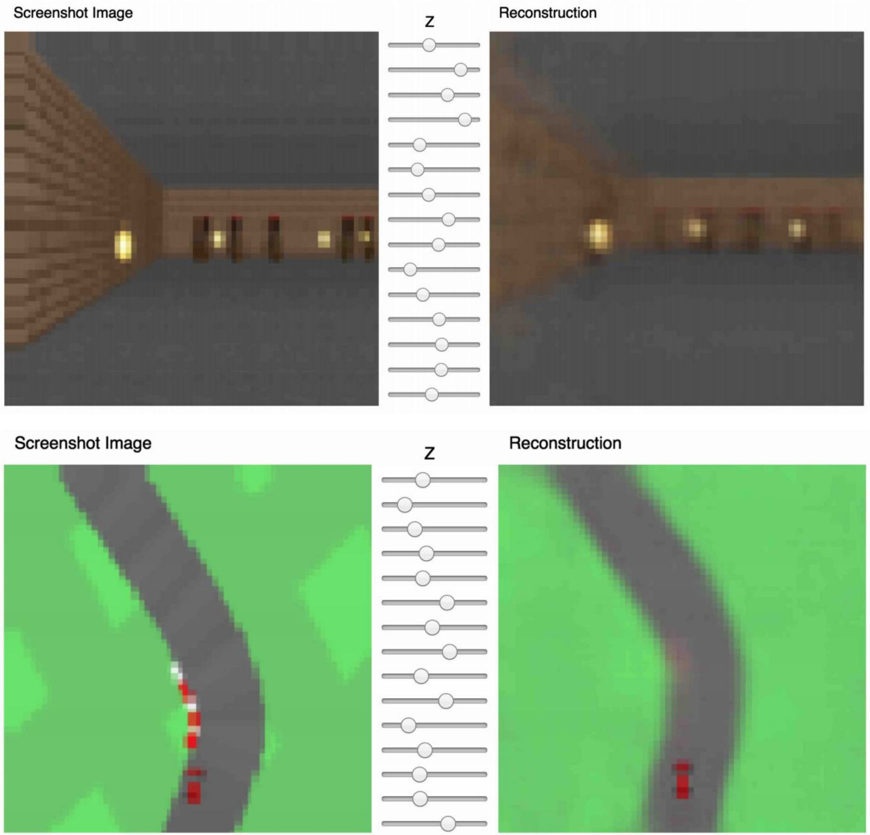
Recent successes: **Audio generation**



van den Oord, A., & Vinyals, O. (2017). Neural discrete representation learning. *NIPS 2017*.

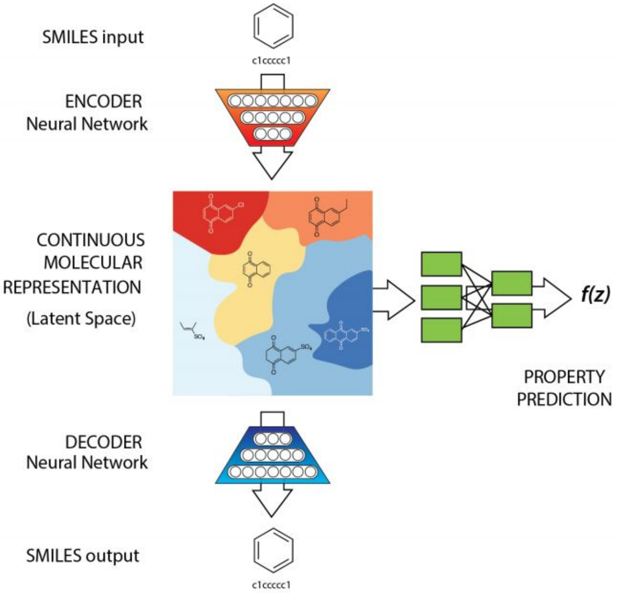
Recent successes: **Reinforcement learning**

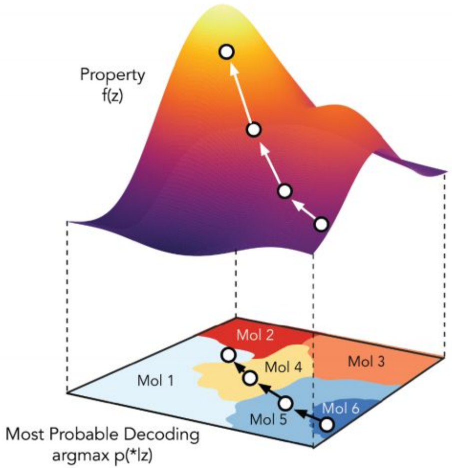


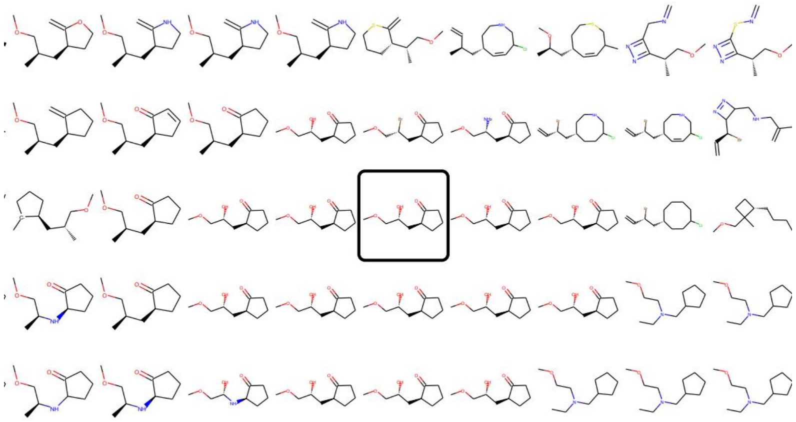


Ha, D., & Schmidhuber, J. (2018). World models. arXiv preprint. *arXiv preprint arXiv:1803.10122*.

Recent successes: **Drug discovery**



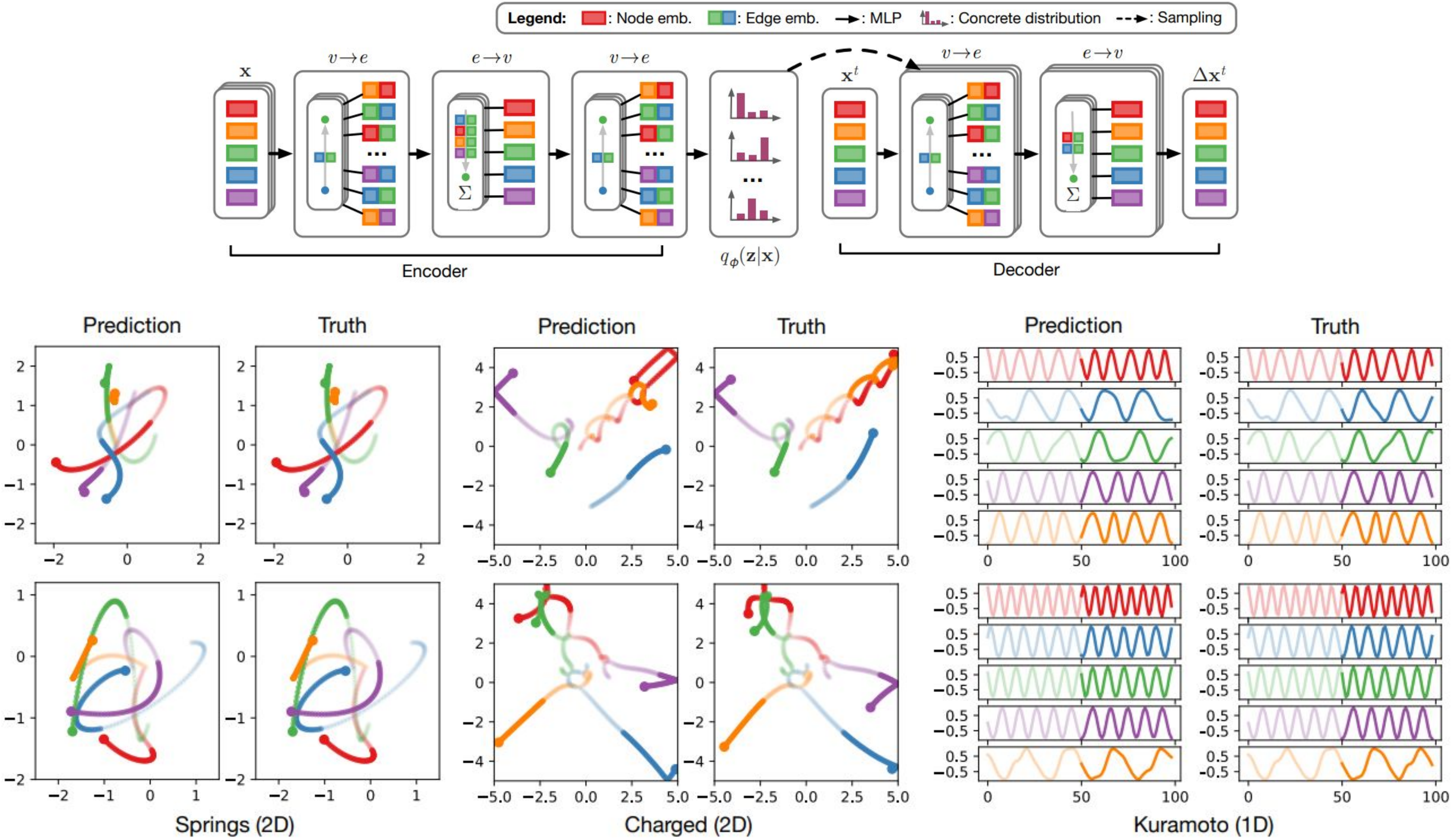




Gómez-Bombarelli, R., et al. (2018). Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules ACS Cent.

Kusner, M. J., Paige, B., & Hernández-Lobato, J. M. (2017). Grammar variational autoencoder. *arXiv preprint arXiv:1703.01925*.

Recent successes: **Physics (interacting systems)**



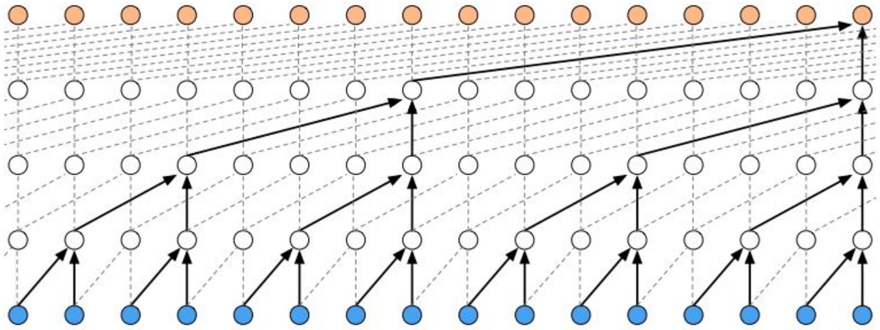
Kipf, T., Fetaya, E., Wang, K. C., Welling, M., & Zemel, R. (2018). Neural relational inference for interacting systems. *ICML 2018*.

Generative modeling: **Auto-regressive models**

General idea is to factorise the joint distribution:

$$p(\mathbf{x}) = p(x_1) \prod_{d=2}^{D} p(x_d | \mathbf{x}_{1:d-1})$$

and use neural networks (e.g., convolutional NN) to model it efficiently:



Van Den Oord, A., et al. (2016). Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.

Generative modeling: **Latent Variable Models**

We assume data lies on a low-dimensional manifold so the generator is:

$$\mathbf{x} = f_\theta(\mathbf{z})$$

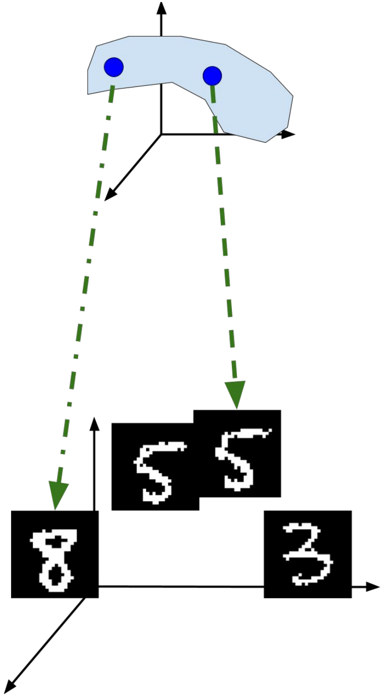
where:

$$\mathbf{x} \in \mathcal{X} \;\; (\text{e.g. } \; \mathcal{X} = \mathbb{R}^D \;) \quad \text{and } \mathbf{z} \in \mathbb{R}^d$$

Two main approaches:

→ **Generative Adversarial Networks** (GANs)

→ **Variational Auto-Encoders** (VAEs)



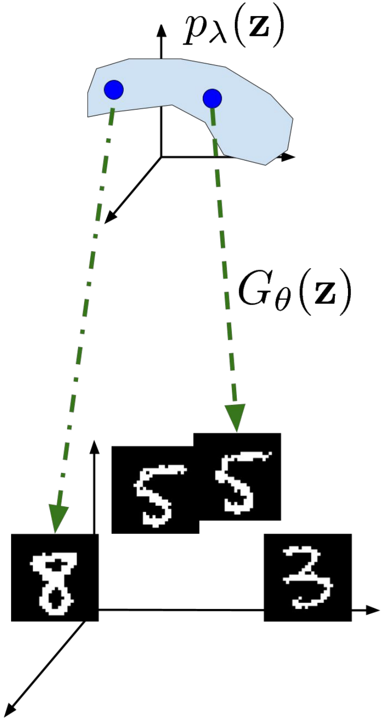
Generative modeling: **GANs**

We assume a **deterministic generator**:

$$\mathbf{x} = G_\theta(\mathbf{z})$$

and **a prior** over latent space:

$$\mathbf{z} \sim p_\lambda(\mathbf{z})$$



Generative modeling: **GANs**

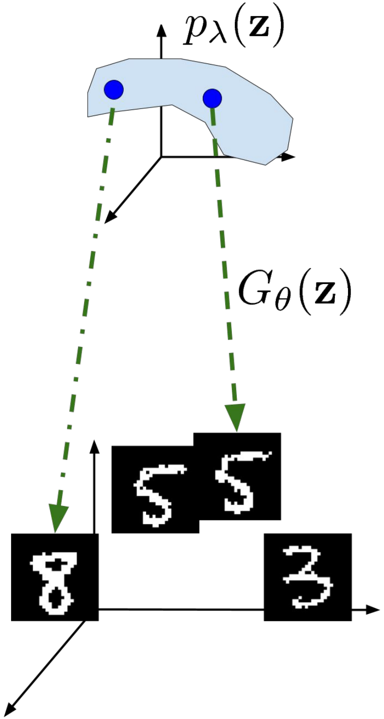
We assume a **deterministic generator**:

$$\mathbf{x} = G_{\theta}(\mathbf{z})$$

and **a prior** over latent space:

$$\mathbf{z} \sim p_{\lambda}(\mathbf{z})$$

How to train it?



Generative modeling: **GANs**

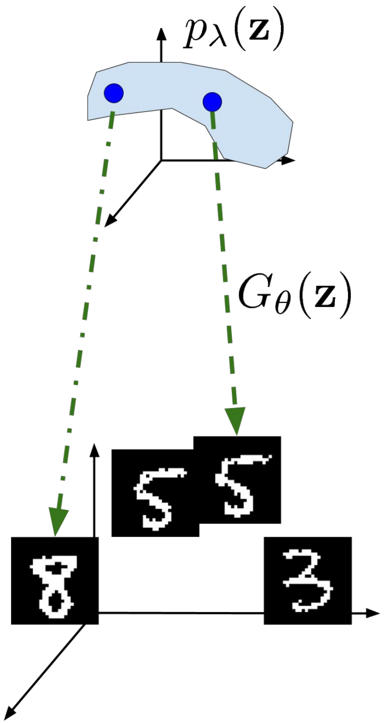
We assume a **deterministic generator**:

$$\mathbf{x} = G_\theta(\mathbf{z})$$

and **a prior** over latent space:

$$\mathbf{z} \sim p_\lambda(\mathbf{z})$$

How to train it? By using a game!



Generative modeling: **GANs**

We assume a **deterministic generator**:

$$\mathbf{x} = G_{\theta}(\mathbf{z})$$

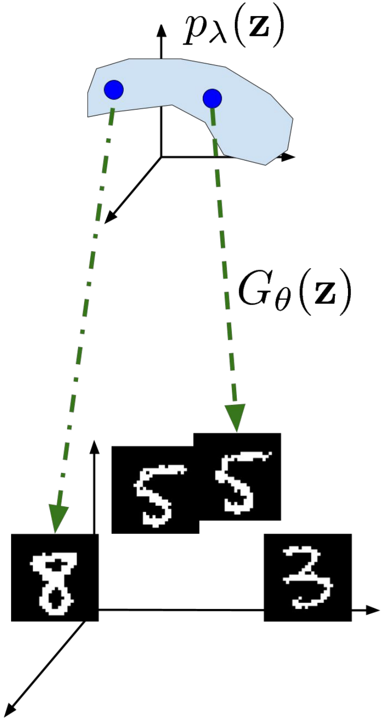
and **a prior** over latent space:

$$\mathbf{z} \sim p_{\lambda}(\mathbf{z})$$

How to train it? By using a game!

For this purpose, we assume a discriminator:

$$D_{\psi}(\mathbf{x}) \in [0, 1]$$



Generative modeling: **GANs**

The learning process is as follows:

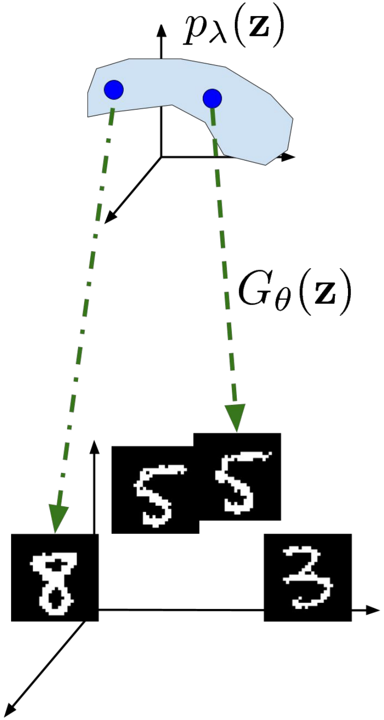
→ the generator tries to **fool** the discriminator;

→ the discriminator tries to **distinguish** between the real and fake images.

We define the learning problem as a min-max problem:

$$\min_{\theta} \max_{\psi} \mathbb{E}_{\mathbf{x}\sim p_{data}} \Big[\ln D_{\psi}(\mathbf{x}) \Big] - \mathbb{E}_{\mathbf{z}\sim p_{\lambda}(\mathbf{z})} \Big[\ln \big(1 - D_{\psi}(G(\mathbf{z}))\big) \Big]$$

In fact, we have a **learnable loss** function!



Goodfellow, I., et al. (2014). Generative adversarial nets. *NIPS 2014*

Generative modeling: **GANs**

The learning process is as follows:

→ the **generator** tries to **fool** the **discriminator**;

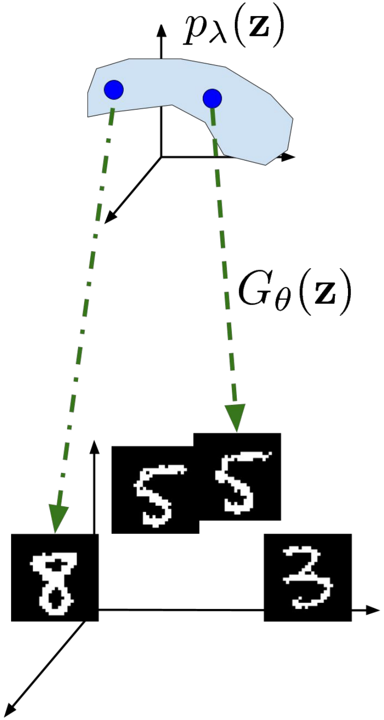
→ the **discriminator** tries to **distinguish** between the **real** and **fake** images.

We define the learning problem as a min-max problem:

$$\min_{\theta} \max_{\psi} \mathbb{E}_{\mathbf{x}\sim p_{data}} \Big[\ln D_{\psi}(\mathbf{x}) \Big] - \mathbb{E}_{\mathbf{z}\sim p_{\lambda}(\mathbf{z})} \Big[\ln \big(1 - D_{\psi}(G(\mathbf{z}))\big) \Big]$$

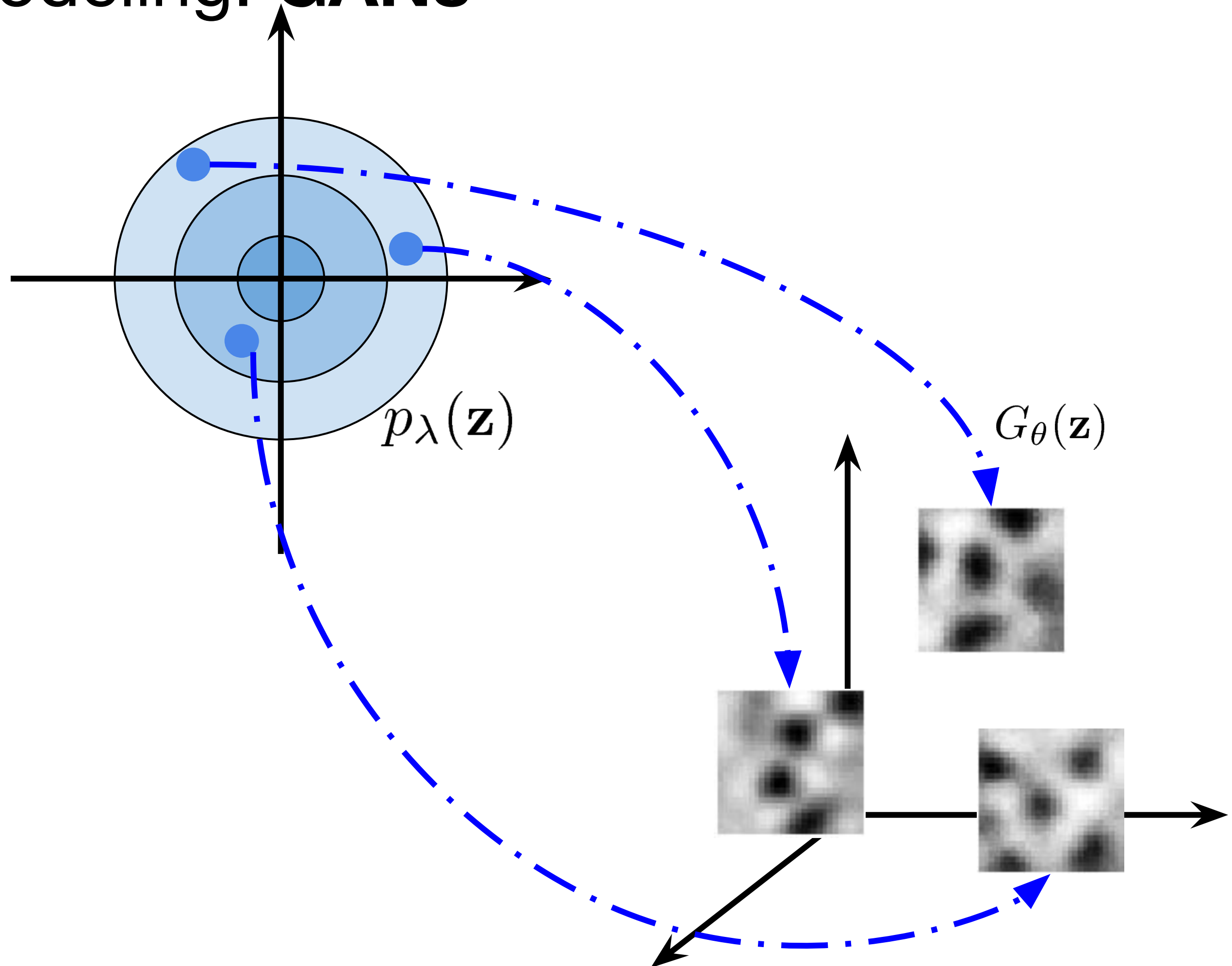
In fact, we have a **learnable loss** function!

→It learns high-order statistics.



Goodfellow, I., et al. (2014). Generative adversarial nets. *NIPS 2014*

Generative modeling: **GANs**



Generative modeling: **GANs**

Pros:

→ we don't need to specify a likelihood function;

→ very flexible;

→ the loss function is trainable;

→ perfect for data simulation.

Cons:

→ we don't know the distribution;

→ training is highly unstable (min-max objective);

→ missing mode problem.

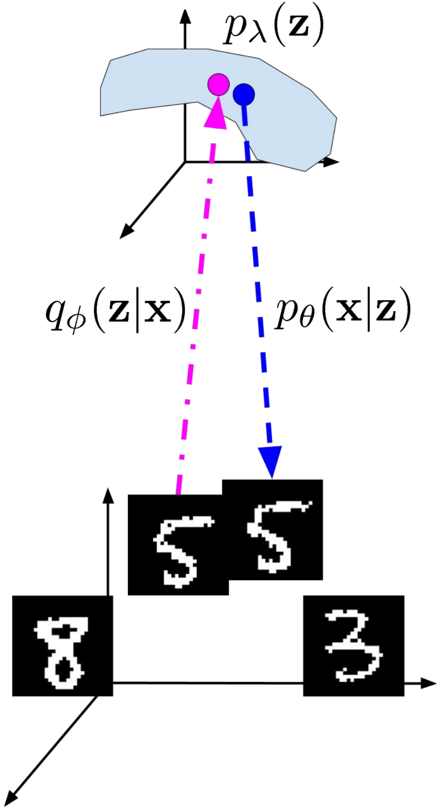
Generative modeling: **VAEs**

We assume **a stochastic generator** (decoder):

$$\mathbf{x} \sim p_\theta(\mathbf{x}|\mathbf{z})$$

and **a prior** over latent space:

$$\mathbf{z} \sim p_\lambda(\mathbf{z})$$



Generative modeling: **VAEs**

We assume **a stochastic generator** (decoder):

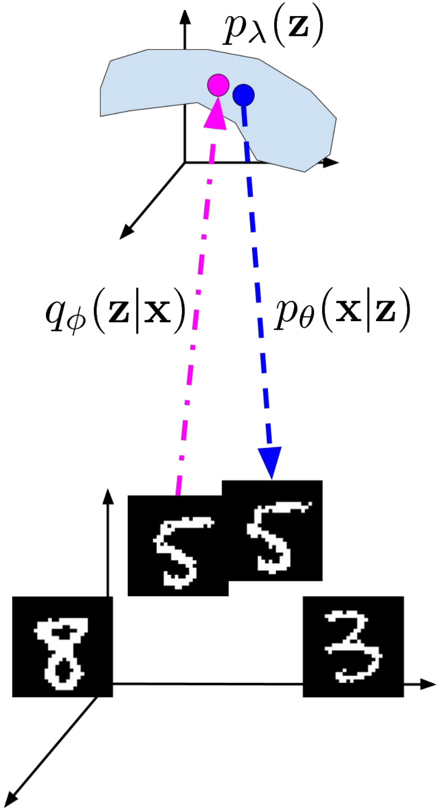
$$\mathbf{x} \sim p_\theta(\mathbf{x}|\mathbf{z})$$

and **a prior** over latent space:

$$\mathbf{z} \sim p_\lambda(\mathbf{z})$$

Additionally, we use **a variational posterior** (encoder):

$$\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})$$



Generative modeling: **VAEs**

We assume **a stochastic generator** (decoder):

$$\mathbf{x} \sim p_\theta(\mathbf{x}|\mathbf{z})$$

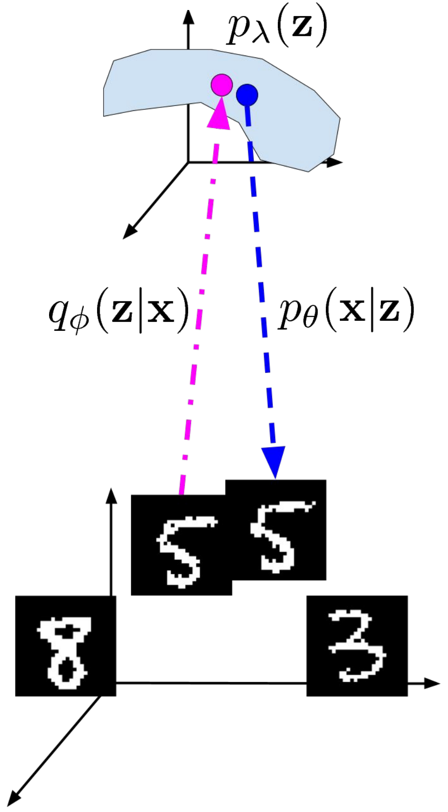
and **a prior** over latent space:

$$\mathbf{z} \sim p_\lambda(\mathbf{z})$$

Additionally, we use **a variational posterior** (encoder):

$$\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})$$

How to train it?



Generative modeling: **VAEs**

We assume **a stochastic generator** (decoder):

$$\mathbf{x} \sim p_{\theta}(\mathbf{x}|\mathbf{z})$$

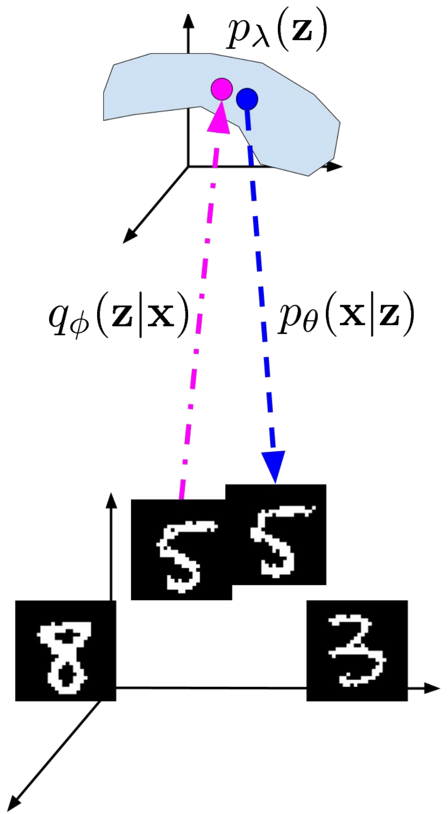
and **a prior** over latent space:

$$\mathbf{z} \sim p_{\lambda}(\mathbf{z})$$

Additionally, we use **a variational posterior** (encoder):

$$\mathbf{z} \sim q_{\phi}(\mathbf{z}|\mathbf{x})$$

How to train it? Using the log-likelihood function!



Variational inference for Latent Variable Models

$$
\begin{aligned}
\log p(\mathbf{x}) &= \log \int p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&= \log \int \frac{q_\phi(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})}\ p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&\geq \int q_\phi(\mathbf{z}|\mathbf{x})\ \log \frac{p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\ \mathrm{d}\mathbf{z} \\
&= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \mathrm{KL}[q_\phi(\mathbf{z}|\mathbf{x})||p_\lambda(\mathbf{z})]
\end{aligned}
$$

Variational inference for Latent Variable Models

Variational posterior

$$
\begin{aligned}
\log p(\mathbf{x}) &= \log \int p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&= \log \int \frac{q_\phi(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})}\ p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&\geq \int q_\phi(\mathbf{z}|\mathbf{x})\ \log \frac{p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\ \mathrm{d}\mathbf{z} \\
&= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \mathrm{KL}[q_\phi(\mathbf{z}|\mathbf{x})||p_\lambda(\mathbf{z})]
\end{aligned}
$$

Variational inference for Latent Variable Models

$$
\begin{aligned}
\log p(\mathbf{x}) &= \log \int p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&= \log \int \frac{q_\phi(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})}\ p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&\geq \int q_\phi(\mathbf{z}|\mathbf{x}) \log \frac{p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\ \mathrm{d}\mathbf{z} \\
&= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \mathrm{KL}[q_\phi(\mathbf{z}|\mathbf{x})||p_\lambda(\mathbf{z})]
\end{aligned}
$$

Jensen's inequality

Variational inference for Latent Variable Models

$$
\begin{aligned}
\log p(\mathbf{x}) &= \log \int p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&= \log \int \frac{q_\phi(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})}\ p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&\geq \int q_\phi(\mathbf{z}|\mathbf{x})\ \log \frac{p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\ \mathrm{d}\mathbf{z} \\
&= \underbrace{\mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]}_{\text{Reconstruction error}} - \underbrace{\mathrm{KL}[q_\phi(\mathbf{z}|\mathbf{x})||p_\lambda(\mathbf{z})]}_{\text{Regularization}}
\end{aligned}
$$

Variational inference for Latent Variable Models

decoder

encoder

prior

$$
\begin{aligned}
\log p(\mathbf{x}) &= \log \int p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&= \log \int \frac{q_\phi(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})}\ p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&\geq \int q_\phi(\mathbf{z}|\mathbf{x})\ \log \frac{p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\ \mathrm{d}\mathbf{z} \\
&= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \mathrm{KL}[q_\phi(\mathbf{z}|\mathbf{x})||p_\lambda(\mathbf{z})]
\end{aligned}
$$

Variational inference for Latent Variable Models

decoder (Neural Net)

encoder (Neural Net)

prior

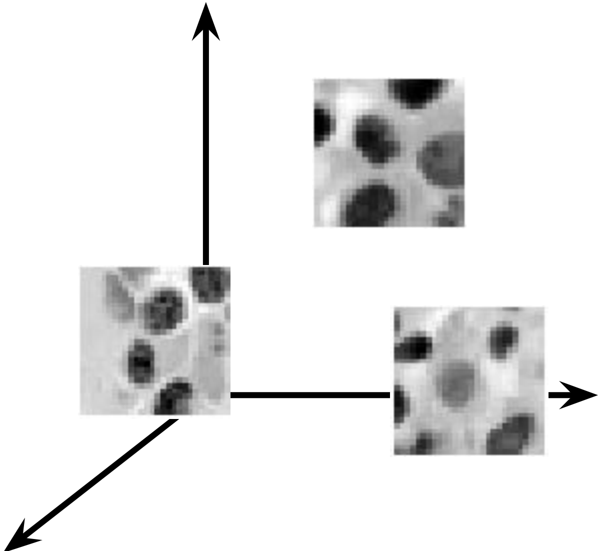
$$\begin{aligned}\log p(\mathbf{x}) &= \log \int p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})\ \mathrm{d}\mathbf{z} \\ &= \log \int \frac{q_\phi(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})}\ p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})\ \mathrm{d}\mathbf{z} \\ &\geq \int q_\phi(\mathbf{z}|\mathbf{x})\ \log \frac{p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\ \mathrm{d}\mathbf{z} \\ &= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \mathrm{KL}[q_\phi(\mathbf{z}|\mathbf{x})||p_\lambda(\mathbf{z})]\end{aligned}$$

\+ reparameterization trick

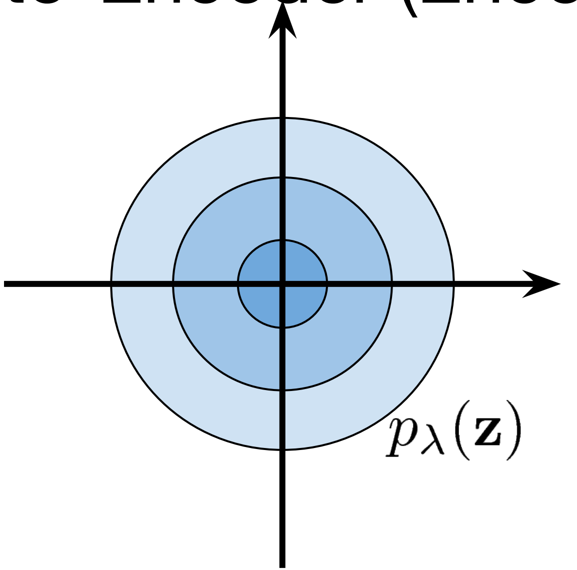
= Variational Auto-Encoder

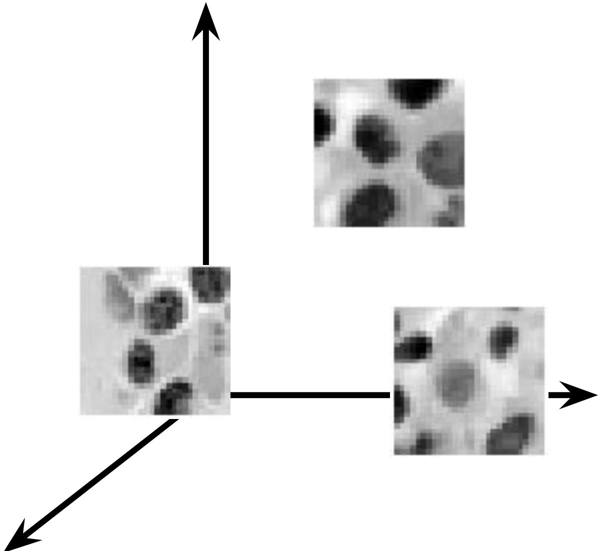
Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114. (*ICLR 2014*)

Variational Auto-Encoder (Encoding-Decoding)

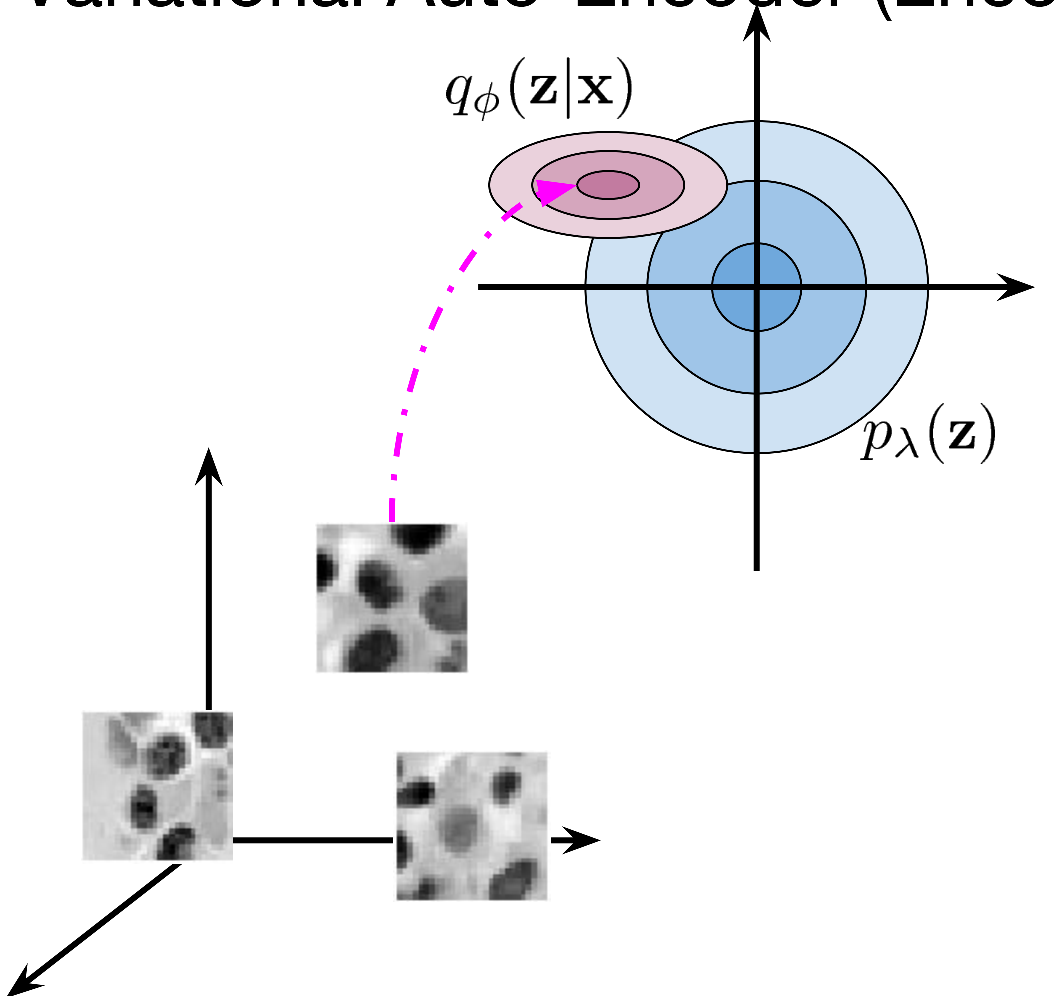


Variational Auto-Encoder (Encoding-Decoding)

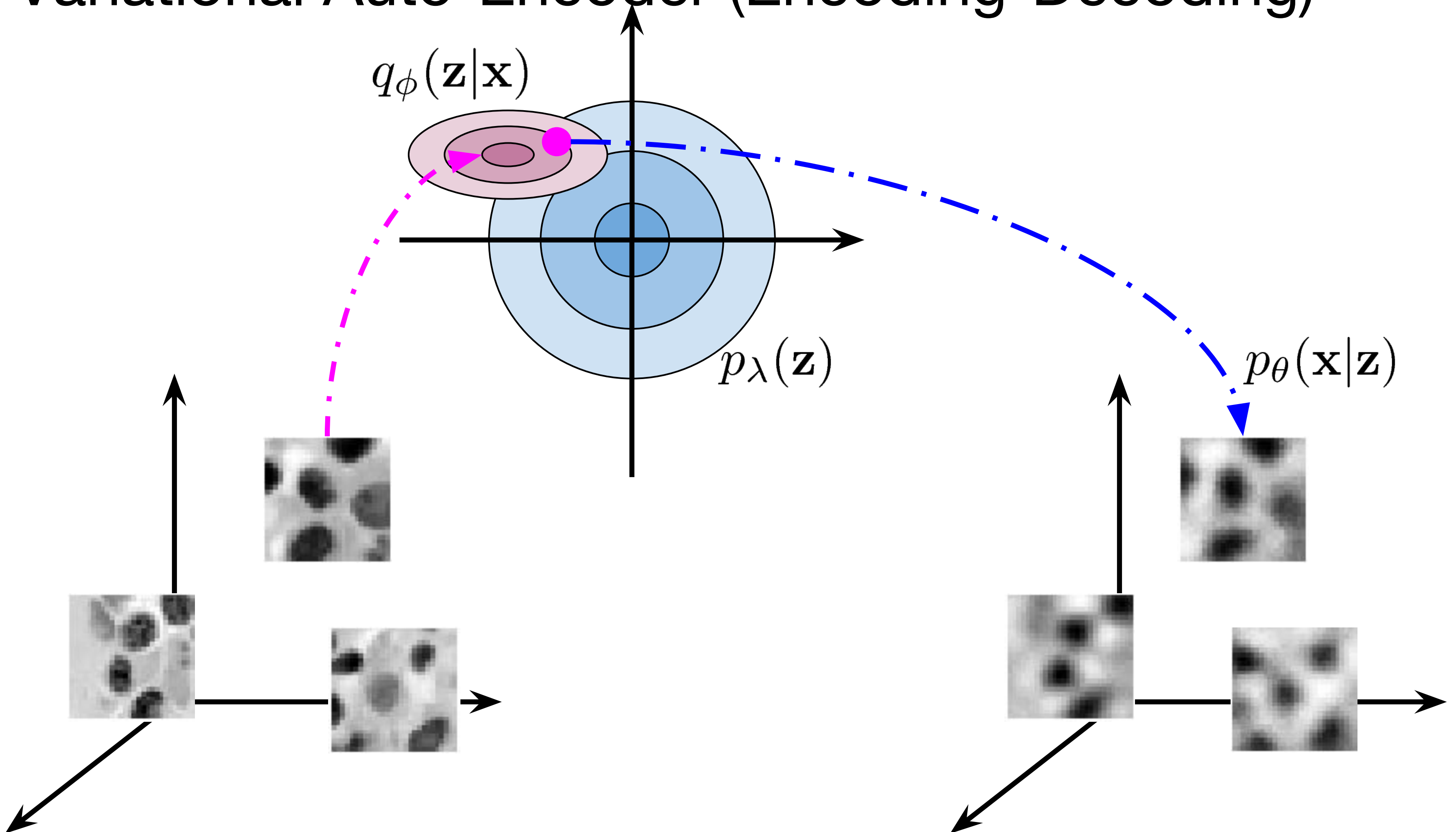




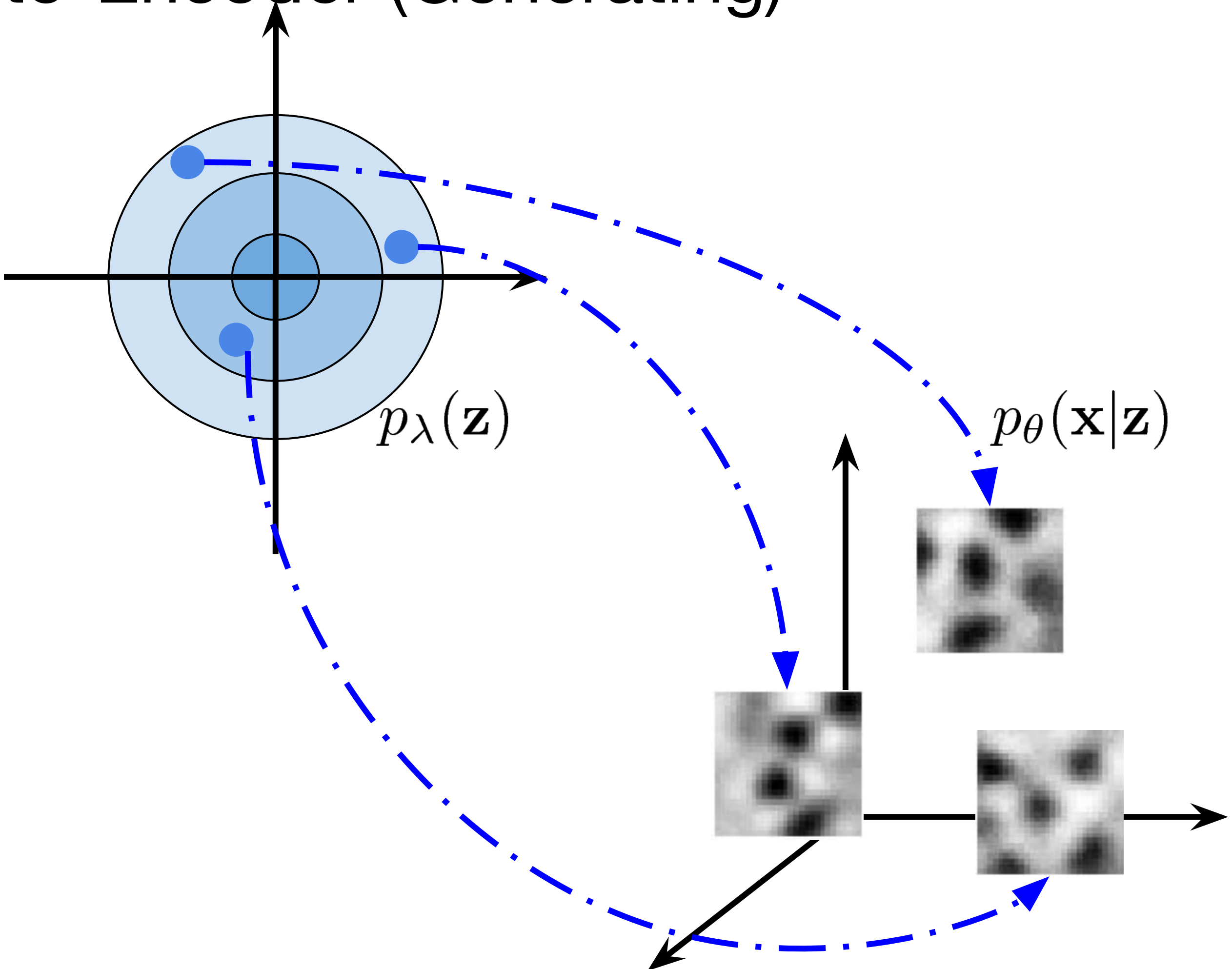
Variational Auto-Encoder (Encoding-Decoding)



Variational Auto-Encoder (Encoding-Decoding)



Variational Auto-Encoder (Generating)



Variational Auto-Encoder: Extensions

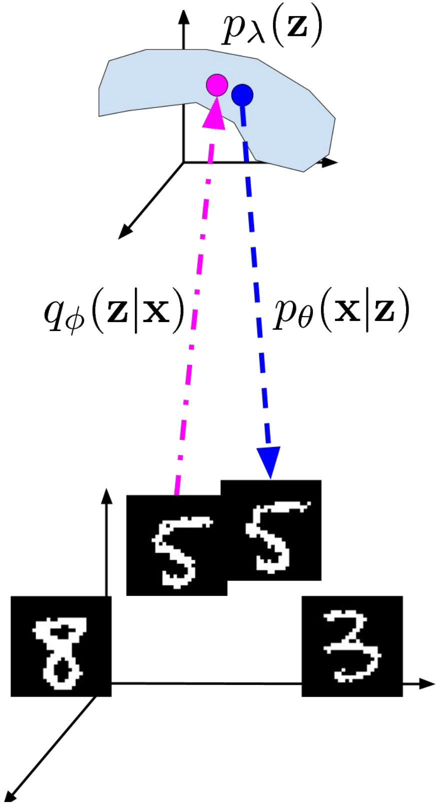
$$q_\phi(\mathbf{z}|\mathbf{x}) \propto p_\theta(\mathbf{x}|\mathbf{z})\, p_\lambda(\mathbf{z})$$

Normalizing flows
Volume-preserving flows
non-Gaussian distributions

Fully-connected
ConvNets
PixelCNN
Other

Autoregressive Prior
Objective Prior
Stick-Breaking Prior
VampPrior

Importance Weighted AE
Renyi Divergence
Stein Divergence



Tomczak, J. M., & Welling, M. (2016). Improving variational auto-encoders using householder flow. *NIPS Workshop 2016.*
Berg, R. V. D., Hasenclever, L., **Tomczak, J. M.**, & Welling, M. (2018). Sylvester Normalizing Flows for Variational Inference. *UAI 2018*.
Tomczak, J. M., & Welling, M. (2017). VAE with a VampPrior. *arXiv preprint arXiv:1705.07120*. (*AISTATS 2018*)
Davidson, T. R., Falorsi, L., De Cao, N., Kipf, T., & **Tomczak, J. M.** (2018). Hyperspherical Variational Auto-Encoders. *UAI 2018.*

Generative modeling: **VAEs**

Pros:

→ we know the distribution and can calculate the likelihood function;

→ we can encode an object in a low-dim manifold (compression);

→ training is stable;

→ no missing modes.

Cons:

→ we need know the distribution;

→ we need a flexible encoder and prior;

→ blurry images (so far…).

Generative modeling: **VAEs (extensions)**

- Normalizing flows
  - Intro
  - Householder flow
  - Sylvester flow
- VampPrior

Conclusion

Generative modeling: the way to go to achieve AI.

Deep generative modeling: very successful in recent years in many domains.

Two main approaches: **GANs** and **VAEs.**

Next steps: **video processing**, **better priors** and **decoders**, **geometric methods**, …

Conclusion

Generative modeling: the way to go to achieve AI.

Deep generative modeling: very successful in recent years in many domains.

Two main approaches: **GANs** and **VAEs.**

Next steps: **video processing**, **better priors** and **decoders**, **geometric methods**, …

Conclusion

Generative modeling: the way to go to achieve AI.

Deep generative modeling: very successful in recent years in many domains.

Two main approaches: **GANs** and **VAEs.**

Next steps: **video processing**, **better priors** and **decoders**, **geometric methods**, …

Conclusion

Generative modeling: the way to go to achieve AI.

Deep generative modeling: very successful in recent years in many domains.

Two main approaches: **GANs** and **VAEs.**

Next steps: **video processing**, **better priors** and **decoders**, **geometric methods**, …

Conclusion

Generative modeling: the way to go to achieve AI.

Deep generative modeling: very successful in recent years in many domains.

Two main approaches: **GANs** and **VAEs**.

Next steps: **video processing**, **better priors** and **decoders**, **geometric methods**, …

Code on github:

https://github.com/jmtomczak

Webpage:

http://jmtomczak.github.io/

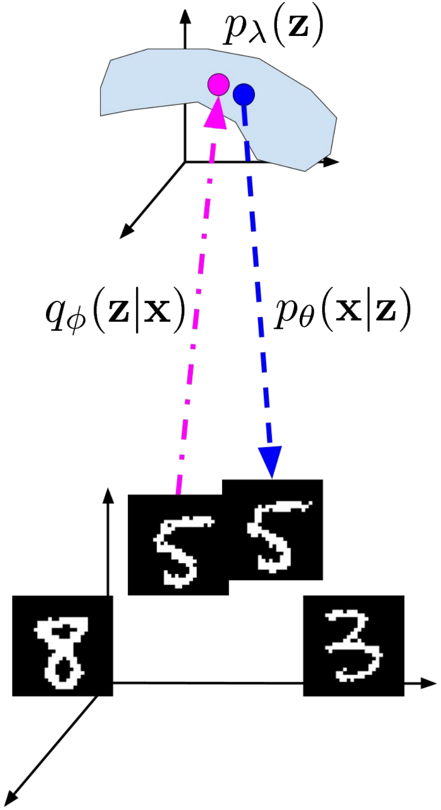
Contact:

jakubmkt@gmail.com

Marie Skłodowska-Curie Actions

The research conducted by Jakub M. Tomczak was funded by the European Commission within the Marie Skłodowska-Curie Individual Fellowship (Grant No. 702666, "Deep learning and Bayesian inference for medical imaging").

APPENDIX

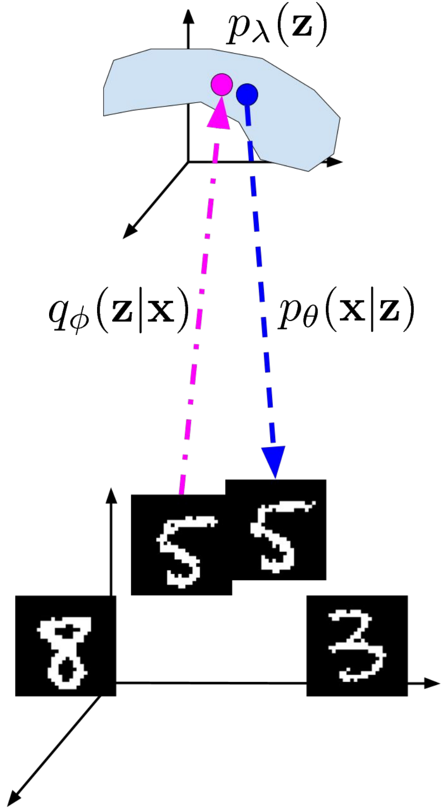


$p_\lambda(\mathbf{z})$
$q_\phi(\mathbf{z}|\mathbf{x})$
$p_\theta(\mathbf{x}|\mathbf{z})$

Variational Auto-Encoder

$$q_\phi(\mathbf{z}|\mathbf{x}) \propto p_\theta(\mathbf{x}|\mathbf{z})\ p_\lambda(\mathbf{z})$$

Normalizing flows
Volume-preserving flows
non-Gaussian distributions



Improving posterior using Normalizing Flows

- Diagonal posterior - **insufficient** and **inflexible**.
- How to get more flexible posterior?
  - Apply a series of T invertible transformations $\mathbf{f}^{(t)}$ to $\mathbf{z}^{(0)} \sim q(\mathbf{z}|\mathbf{x})$
- New objective:

$$\ln p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z}^{(0)}|\mathbf{x})}\Big[\ln p(\mathbf{x}|\mathbf{z}^{(T)}) + \sum_{t=1}^{T} \ln \Big|\det \frac{\partial \mathbf{f}^{(t)}}{\partial \mathbf{z}^{(t-1)}}\Big|\Big] - \mathrm{KL}(q(\mathbf{z}^{(0)}|\mathbf{x})||p(\mathbf{z}^{(T)})).$$

Rezende, D. J., & Mohamed, S. (2015). Variational inference with normalizing flows. arXiv preprint arXiv:1505.05770. ICML 2015

Improving posterior using Normalizing Flows

- Diagonal posterior - **insufficient** and **inflexible**.
- How to get more flexible posterior?
  - ➢ Apply a series of T invertible transformations $\mathbf{f}^{(t)}$ to $\mathbf{z}^{(0)} \sim q(\mathbf{z}|\mathbf{x})$
- New objective:

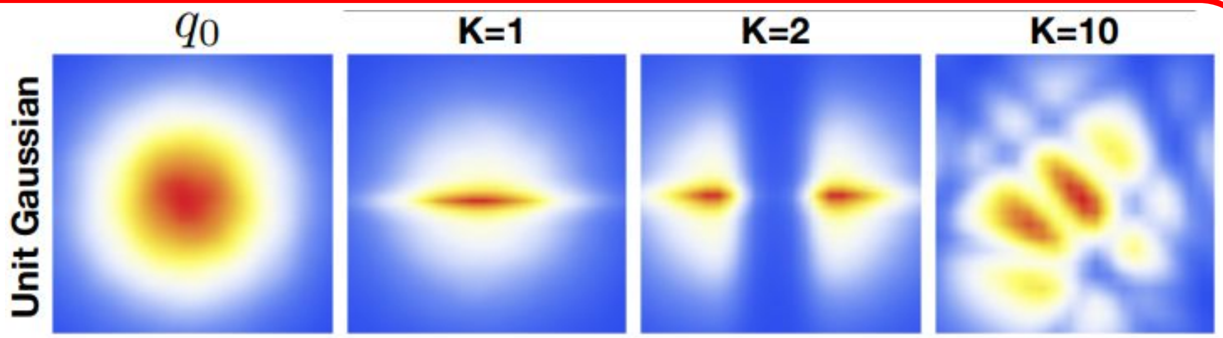
Change of variables:

$$q(\mathbf{z}^{(T)}|\mathbf{x}) = q(\mathbf{z}^{(0)}|\mathbf{x}) \prod_{t=1}^{T} \left| \det \frac{\partial f^{(t)}}{\partial \mathbf{z}^{(t-1)}} \right|^{-1}$$

Rezende, D. J., & Mohamed, S. (2015). Variational inference with normalizing flows. arXiv preprint arXiv:1505.05770. ICML 2015

Improving posterior using Normalizing Flows

- Diagonal posterior - **insufficient** and **inflexible**.
- How to get more flexible posterior?
    - ➢ Apply a series of T invertible transformations $\mathbf{f}^{(t)}$ to $\mathbf{z}^{(0)} \sim q(\mathbf{z}|\mathbf{x})$
- New objective:



Rezende, D. J., & Mohamed, S. (2015). Variational inference with normalizing flows. arXiv preprint arXiv:1505.05770. ICML 2015

Improving posterior using Normalizing Flows

- Diagonal posterior - **insufficient** and **inflexible**.
- How to get more flexible posterior?
    - ➢ Apply a series of *T* invertible transformations $\mathbf{f}^{(t)}$ to $\mathbf{z}^{(0)} \sim q(\mathbf{z}|\mathbf{x})$
- **New objective**:

$$\ln p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z}^{(0)}|\mathbf{x})}\Big[\ln p(\mathbf{x}|\mathbf{z}^{(T)}) + \sum_{t=1}^{T} \ln \Big|\det \frac{\partial \mathbf{f}^{(t)}}{\partial \mathbf{z}^{(t-1)}}\Big|\Big] - \mathrm{KL}\big(q(\mathbf{z}^{(0)}|\mathbf{x})||p(\mathbf{z}^{(T)})\big).$$

Rezende, D. J., & Mohamed, S. (2015). Variational inference with normalizing flows. arXiv preprint arXiv:1505.05770. ICML 2015

Improving posterior using Normalizing Flows

- Diagonal posterior - **insufficient** and **inflexible**.
- How to get more flexible posterior?
  - ➢ Apply a series of *T* invertible transformations $\mathbf{f}^{(t)}$ to $\mathbf{z}^{(0)} \sim q(\mathbf{z}|\mathbf{x})$
- **New objective**:

$$\ln p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z}^{(0)}|\mathbf{x})}\Big[\ln p(\mathbf{x}|\mathbf{z}^{(T)}) + \boxed{\sum_{t=1}^{T} \ln\Big|\det \frac{\partial \mathbf{f}^{(t)}}{\partial \mathbf{z}^{(t-1)}}\Big|}\Big] - \mathrm{KL}\big(q(\mathbf{z}^{(0)}|\mathbf{x})||p(\mathbf{z}^{(T)})\big).$$

Jacobian determinant: (i) general normalizing flow (|det J| is **easy** to calculate);

(ii) volume-preserving flow, *i.e.*, |det J| = 1.

Rezende, D. J., & Mohamed, S. (2015). Variational inference with normalizing flows. arXiv preprint arXiv:1505.05770. ICML 2015

BACK

Volume-preserving flows

Improving Variational Auto-Encoders using Householder Flow

Jakub M. Tomczak, Max Welling
University of Amsterdam
J.M.Tomczak@uva.nl, M.Welling@uva.nl

Abstract

Variational auto-encoders (VAE) are scalable and powerful generative models. However, the choice of the variational posterior determines tractability and flexibility of the VAE. Commonly, latent variables are modeled using the normal distribution with a diagonal covariance matrix. This results in computational effi-

Tomczak, J. M., & Welling, M. (2016). Improving Variational Inference with Householder Flow. arXiv preprint arXiv:1611.09630. NIPS Workshop on Bayesian Deep Learning 2016

Householder Flow

- How to obtain more **flexible** posterior and preserve |det J|=1?
- Model **full-covariance posterior** using **orthogonal matrices**.
- **Proposition**: Apply a linear transformation:

$$\mathbf{z}^{(1)} = \mathbf{U}\mathbf{z}^{(0)}, \ \mathbf{z}^{(1)} \sim \mathcal{N}(\mathbf{U}\mu, \mathbf{U} \ \mathrm{diag}(\sigma^2) \ \mathbf{U}^\top)$$

and since $\mathbf{U}$ is orthogonal, Jacobian-determinant is 1.

- **Question**: *Is it possible to model an orthogonal matrix efficiently?*

Tomczak, J. M., & Welling, M. (2016). Improving Variational Inference with Householder Flow. arXiv preprint arXiv:1611.09630. NIPS Workshop on Bayesian Deep Learning 2016

Householder Flow

Theorem
Any orthogonal matrix with the basis acting on the K-dimensional subspace can be expressed as a product of exactly K Householder transformations.

Sun, X., & Bischof, C. (1995). A basis-kernel representation of orthogonal matrices. SIAM Journal on Matrix Analysis and Applications, 16(4), 1184-1196.

- **Question**: *Is it possible to model an orthogonal matrix efficiently?* **YES**

Tomczak, J. M., & Welling, M. (2016). Improving Variational Inference with Householder Flow. arXiv preprint arXiv:1611.09630. NIPS Workshop on Bayesian Deep Learning 2016

Householder Flow

In the **Householder transformation** we reflect a vector around a hyperplane defined by a **Householder vector** $\mathbf{v}_t \in \mathbb{R}^M$

$$\mathbf{z}^{(t)} = \underbrace{\left(\mathbf{I} - 2\frac{\mathbf{v}_t \mathbf{v}_t^\top}{||\mathbf{v}_t||^2}\right)}_{Householder\ matrix} \mathbf{z}^{(t-1)} = \mathbf{H}_t\ \mathbf{z}^{(t-1)}.$$

Very efficient: small number of parameters, |J|=1, easy amortization (!).

Tomczak, J. M., & Welling, M. (2016). Improving Variational Inference with Householder Flow. arXiv preprint arXiv:1611.09630. NIPS Workshop on Bayesian Deep Learning 2016

Householder Flow (MNIST)

Method	ELBO
VAE	-93.9
VAE+HF(T=1)	-87.8
VAE+HF(T=10)	**-87.7**
VAE+NICE(T=10)	-88.6
VAE+NICE(T=80)	**-87.2**
VAE+HVI(T=1)	-91.7
VAE+HVI(T=8)	-88.3
VAE+PlanarFlow(T=10)	**-87.5**
VAE+PlanarFlow(T=80)	**-85.1**

Volume-preserving

Non-linear

Tomczak, J. M., & Welling, M. (2016). Improving Variational Inference with Householder Flow. arXiv preprint arXiv:1611.09630. NIPS Workshop on Bayesian Deep Learning 2016

BACK

General normalizing flow

Sylvester Normalizing Flows for Variational Inference

Rianne van den Berg* University of Amsterdam

Leonard Hasenclever* University of Oxford

Jakub M. Tomczak University of Amsterdam

Max Welling University of Amsterdam

Abstract

Variational inference relies on flexible approximate posterior distributions. Normalizing flows provide a general recipe to con-

Variational inference searches for the best posterior approximation within a parametric family of distributions. Hence, the true posterior distribution can only be recovered exactly if it happens to be in the chosen family. In particular, with widely used simple variational families such as diagonal covariance Gaussian distributions

van den Berg, R., Hasenclever, L., Tomczak, J. M., & Welling, M. (2018). Sylvester Normalizing Flows for Variational Inference, UAI 2018 (oral presentation)

Sylvester Flow

- Can we have a **non-linear** flow with a **simple** Jacobian-determinant?

- Let us consider the following normalizing flow:

$$\mathbf{z}^{(t)} = \mathbf{z}^{(t-1)} + \mathbf{A}h(\mathbf{B}\mathbf{z}^{(t)} + \mathbf{b})$$

where A is *DxM*, B is *MxD*.

- How to calculate the Jacobian-determinant efficiently?

  ➢ **Sylvester's determinant identity**

van den Berg, R., Hasenclever, L., Tomczak, J. M., & Welling, M. (2018). Sylvester Normalizing Flows for Variational Inference, UAI 2018 (oral presentation)

Sylvester Flow

- Can we have a **non-linear** flow with a **simple** Jacobian-determinant?
- Let us consider the following normalizing flow:

$$\mathbf{z}^{(t)} = \mathbf{z}^{(t-1)} + \mathbf{A}h(\mathbf{B}\mathbf{z}^{(t)} + \mathbf{b})$$

where A is *D*x*M*, B is *M*x*D*.

- How to calculate the Jacobian-determinant efficiently?
  - ➢ **Sylvester's determinant identity**

van den Berg, R., Hasenclever, L., Tomczak, J. M., & Welling, M. (2018). Sylvester Normalizing Flows for Variational Inference, UAI 2018 (oral presentation)

Sylvester Flow

- Can we have a **non-linear** flow with a **simple** Jacobian-determinant?

- Let us consider the following normalizing flow:

$$\mathbf{z}^{(t)} = \mathbf{z}^{(t-1)} + \mathbf{A}h(\mathbf{B}\mathbf{z}^{(t)} + \mathbf{b})$$

where A is *DxM*, B is *MxD*.

- How to calculate the Jacobian-determinant efficiently?

  ➢ **Sylvester's determinant identity**

van den Berg, R., Hasenclever, L., Tomczak, J. M., & Welling, M. (2018). Sylvester Normalizing Flows for Variational Inference, UAI 2018 (oral presentation)

Sylvester Flow

Theorem
For all $\mathbf{A} \in \mathbb{R}^{D\times M}, \mathbf{B} \in \mathbb{R}^{M\times D}$

$$\det\left(\mathbf{I}_D + \mathbf{AB}\right) = \det\left(\mathbf{I}_M + \mathbf{BA}\right).$$

- How to calculate the Jacobian-determinant efficiently?
  - ➢ **Sylvester's determinant identity**

van den Berg, R., Hasenclever, L., Tomczak, J. M., & Welling, M. (2018). Sylvester Normalizing Flows for Variational Inference, UAI 2018 (oral presentation)

Sylvester Flow

- How to use the Sylvester's determinant identity?

$$\det \frac{\partial \mathbf{z}^{(t)}}{\partial \mathbf{z}^{(t-1)}} = \det \left(\mathbf{I}_M + \operatorname{diag}\left(h'(\mathbf{B}\mathbf{z}^{(t-1)} + \mathbf{b}) \right) \mathbf{B}\mathbf{A} \right)$$

- How to parameterize matrices **A** and **B**?

$$\mathbf{z}^{(t)} = \mathbf{z}^{(t-1)} + \mathbf{Q}\mathbf{R}_1 h(\mathbf{R}_2 \mathbf{Q}^\top \mathbf{z}^{(t-1)} + \mathbf{b})$$

$\mathbf{Q}$ is orthogonal

$\mathbf{R}_1, \mathbf{R}_2$ are triangular

van den Berg, R., Hasenclever, L., Tomczak, J. M., & Welling, M. (2018). Sylvester Normalizing Flows for Variational Inference, UAI 2018 (oral presentation)

Sylvester Flow

- How to use the Sylvester's determinant identity?

$$\det \frac{\partial \mathbf{z}^{(t)}}{\partial \mathbf{z}^{(t-1)}} = \det \left(\mathbf{I}_M + \operatorname{diag}\left(h'(\mathbf{B}\mathbf{z}^{(t-1)} + \mathbf{b}) \right) \mathbf{B}\mathbf{A} \right)$$

- How to parameterize matrices **A** and **B**?

$$\mathbf{z}^{(t)} = \mathbf{z}^{(t-1)} + \mathbf{Q}\mathbf{R}_1 h(\mathbf{R}_2 \mathbf{Q}^\top \mathbf{z}^{(t-1)} + \mathbf{b})$$

$\mathbf{Q}$ is orthogonal **Householder matrices**, **permutation matrix**, **orthogonalization procedure**

$\mathbf{R}_1, \mathbf{R}_2$ are triangular

van den Berg, R., Hasenclever, L., Tomczak, J. M., & Welling, M. (2018). Sylvester Normalizing Flows for Variational Inference, UAI 2018 (oral presentation)

Sylvester Flow

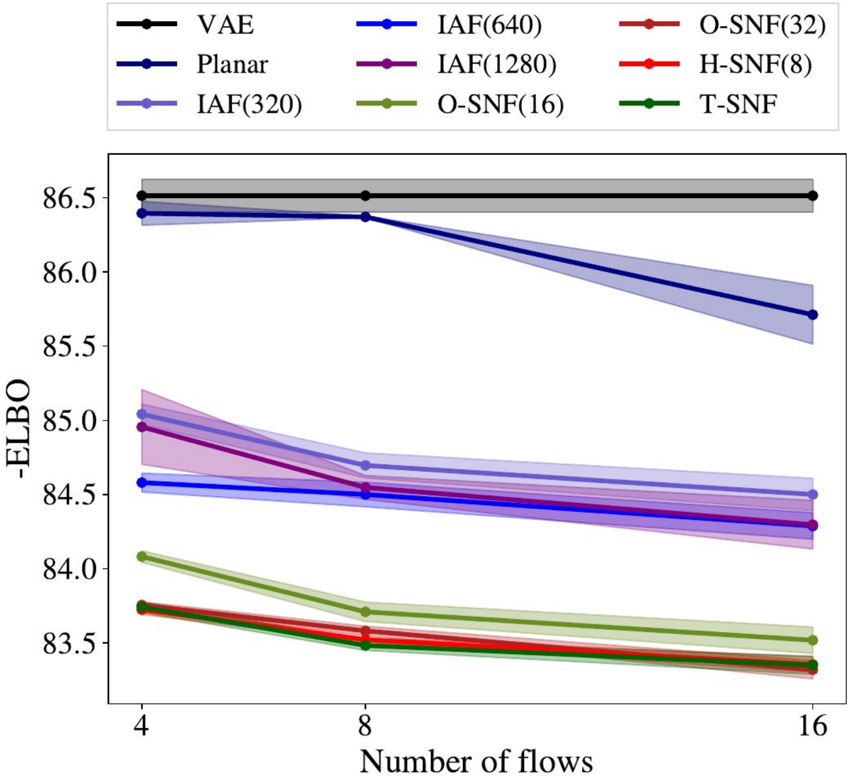
- The Jacobian-determinant:

$$\det\left(\frac{\partial \mathbf{z}^{(t)}}{\partial \mathbf{z}^{(t-1)}}\right) = \det\left(\mathbf{I}_M + \operatorname{diag}\left(h'(\mathbf{R}_2\mathbf{Q}^T\mathbf{z}^{(t-1)} + \mathbf{b})\right)\mathbf{R}_2\mathbf{Q}^T\mathbf{Q}\mathbf{R}_1\right)$$
$$= \det\left(\mathbf{I}_M + \operatorname{diag}\left(h'(\mathbf{R}_2\mathbf{Q}^T\mathbf{z}^{(t-1)} + \mathbf{b})\right)\mathbf{R}_2\mathbf{R}_1\right)$$

- As a result, for properly chosen *h*, the determinant is **upper-triangular** and, thus, **easy to calculate**.

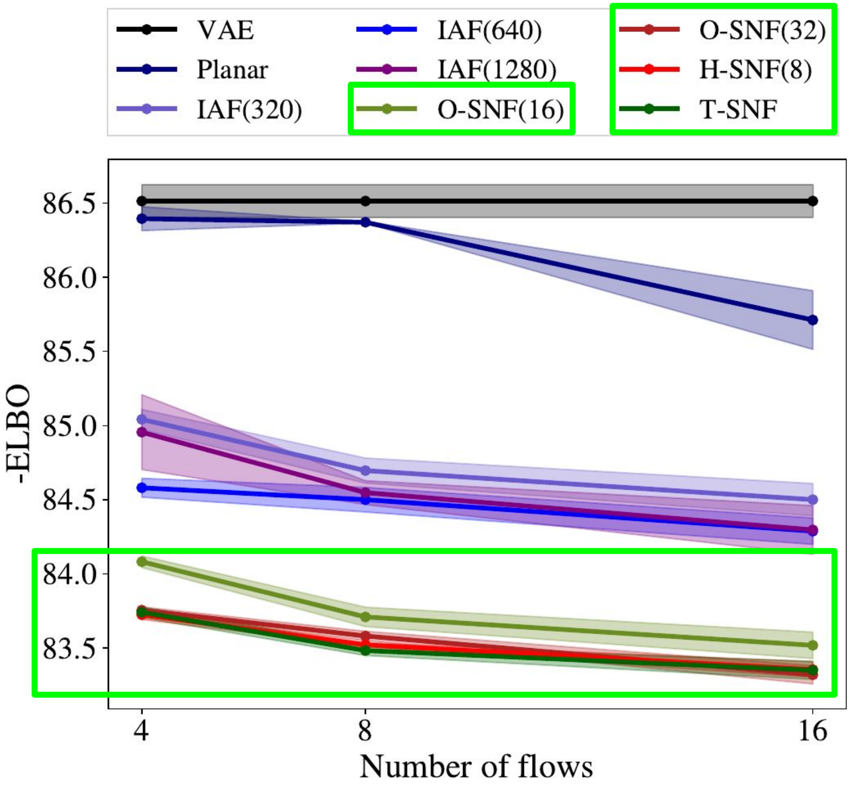
van den Berg, R., Hasenclever, L., Tomczak, J. M., & Welling, M. (2018). Sylvester Normalizing Flows for Variational Inference, UAI 2018 (oral presentation)

Sylvester Flow (MNIST)



van den Berg, R., Hasenclever, L., Tomczak, J. M., & Welling, M. (2018). Sylvester Normalizing Flows for Variational Inference, UAI 2018 (oral presentation)

Sylvester Flow (MNIST)



van den Berg, R., Hasenclever, L., Tomczak, J. M., & Welling, M. (2018). Sylvester Normalizing Flows for Variational Inference, UAI 2018 (oral presentation)

Sylvester Flow

Model	Freyfaces		Omniglot		Caltech 101	
	-ELBO	NLL	-ELBO	NLL	-ELBO	NLL
VAE	4.53 ± 0.02	4.40 ± 0.03	104.28 ± 0.39	97.25 ± 0.23	110.80 ± 0.46	99.62 ± 0.74
Planar	$\mathbf{4.40 \pm 0.06}$	$\mathbf{4.31 \pm 0.06}$	102.65 ± 0.42	96.04 ± 0.28	109.66 ± 0.42	98.53 ± 0.68
IAF	4.47 ± 0.05	4.38 ± 0.04	102.41 ± 0.04	96.08 ± 0.16	111.58 ± 0.38	99.92 ± 0.30
O-SNF	4.51 ± 0.04	4.39 ± 0.05	99.00 ± 0.29	93.82 ± 0.21	106.08 ± 0.39	94.61 ± 0.83
H-SNF	4.46 ± 0.05	4.35 ± 0.05	$\mathbf{99.00 \pm 0.04}$	$\mathbf{93.77 \pm 0.03}$	$\mathbf{104.62 \pm 0.29}$	$\mathbf{93.82 \pm 0.62}$
T-SNF	4.45 ± 0.04	4.35 ± 0.04	99.33 ± 0.23	93.97 ± 0.13	105.29 ± 0.64	94.92 ± 0.73

van den Berg, R., Hasenclever, L., Tomczak, J. M., & Welling, M. (2018). Sylvester Normalizing Flows for Variational Inference, UAI 2018 (oral presentation)

Sylvester Flow

Model	Freyfaces		Omniglot		Caltech 101	
	-ELBO	NLL	-ELBO	NLL	-ELBO	NLL
VAE	4.53 ± 0.02	4.40 ± 0.03	104.28 ± 0.39	97.25 ± 0.23	110.80 ± 0.46	99.62 ± 0.74
Planar	$\mathbf{4.40 \pm 0.06}$	$\mathbf{4.31 \pm 0.06}$	102.65 ± 0.42	96.04 ± 0.28	109.66 ± 0.42	98.53 ± 0.68
IAF	4.47 ± 0.05	4.38 ± 0.04	102.41 ± 0.04	96.08 ± 0.16	111.58 ± 0.38	99.92 ± 0.30
O-SNF	4.51 ± 0.04	4.39 ± 0.05	99.00 ± 0.29	93.82 ± 0.21	106.08 ± 0.39	94.61 ± 0.83
H-SNF	4.46 ± 0.05	4.35 ± 0.05	$\mathbf{99.00 \pm 0.04}$	$\mathbf{93.77 \pm 0.03}$	$\mathbf{104.62 \pm 0.29}$	$\mathbf{93.82 \pm 0.62}$
T-SNF	4.45 ± 0.04	4.35 ± 0.04	99.33 ± 0.23	93.97 ± 0.13	105.29 ± 0.64	94.92 ± 0.73

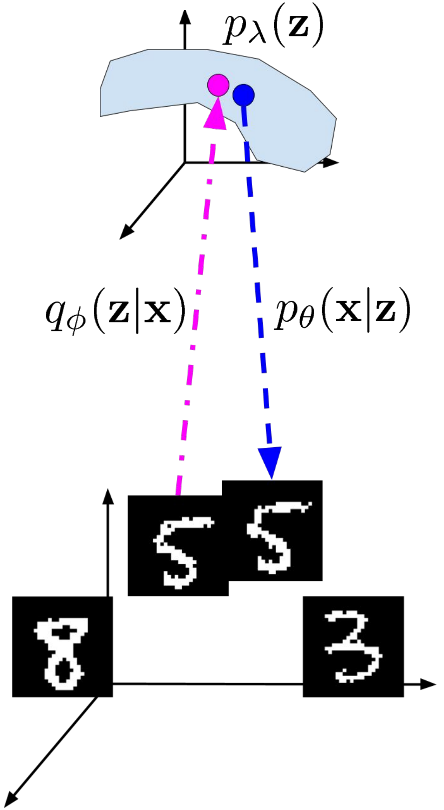
van den Berg, R., Hasenclever, L., Tomczak, J. M., & Welling, M. (2018). Sylvester Normalizing Flows for Variational Inference, UAI 2018 (oral presentation)

BACK

Variational Auto-Encoder

$$q_\phi(\mathbf{z}|\mathbf{x}) \propto p_\theta(\mathbf{x}|\mathbf{z})\, p_\lambda(\mathbf{z})$$

Autoregressive Prior
Objective Prior
Stick-Breaking Prior
VampPrior



New Prior

VAE with a VampPrior

Jakub M. Tomczak
University of Amsterdam

Max Welling
University of Amsterdam

Abstract

Many different methods to train deep generative models have been introduced in the past. In this paper, we propose to extend the varia-

efficient through the application of the *reparameterization trick* resulting in a highly scalable framework now known as the *variational auto-encoders* (VAE) [19, 33]. Various extensions to deep generative models have been proposed that aim to enrich the variational

Tomczak, J. M., & Welling, M. (2018). VAE with a VampPrior, AISTATS 2018 (oral presentation, 14% of accepted papers)

New Prior

- Let's re-write the ELBO:

$$
\begin{aligned}
\mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\big[\ln p(\mathbf{x})\big] \geq & \mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\big[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\ln p_\theta(\mathbf{x}|\mathbf{z})]\big] + \\
& + \mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\big[\mathbb{H}[q_\phi(\mathbf{z}|\mathbf{x})]\big] + \\
& - \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}[-\ln p_\lambda(\mathbf{z})]
\end{aligned}
$$

Tomczak, J. M., & Welling, M. (2018). VAE with a VampPrior, AISTATS 2018

New Prior

- Let’s re-write the ELBO:

$$\mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\big[\ln p(\mathbf{x})\big] \geq \mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\big[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\ln p_\theta(\mathbf{x}|\mathbf{z})]\big] + \\ + \mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\big[\mathbb{H}[q_\phi(\mathbf{z}|\mathbf{x})]\big] + \\ - \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}[-\ln p_\lambda(\mathbf{z})]$$

Empirical distribution

Tomczak, J. M., & Welling, M. (2018). VAE with a VampPrior, AISTATS 2018

New Prior

- Let's re-write the ELBO:

$$
\begin{aligned}
\mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\big[\ln p(\mathbf{x})\big] \geq & \mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\big[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\ln p_\theta(\mathbf{x}|\mathbf{z})]\big] + \\
& + \mathbb{E}_{\mathbf{x}\sim q(\mathbf{x})}\big[\mathbb{H}[q_\phi(\mathbf{z}|\mathbf{x})]\big] + \\
& - \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}[-\ln p_\lambda(\mathbf{z})]
\end{aligned}
$$

Aggregated posterior

$$
\begin{aligned}
q(\mathbf{z}) &= \mathbb{E}_{q(\mathbf{x})}[q_\phi(\mathbf{z}|\mathbf{x})] \\
&= \frac{1}{N}\sum_{n=1}^{N} q_\phi(\mathbf{z}|\mathbf{x}_n)
\end{aligned}
$$

Tomczak, J. M., & Welling, M. (2018). VAE with a VampPrior, AISTATS 2018

New Prior (**Va**riational **M**ixture of **P**osteriors **Prior**)

- We look for **the optimal prior** using the Lagrange function:

$$\max_{p_\lambda(\mathbf{z})} -\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}[-\ln p_\lambda(\mathbf{z})] + \beta\Big(\int p_\lambda(\mathbf{z})\mathrm{d}\mathbf{z} - 1\Big)$$

- The solution is simply **the aggregated posterior**.

- We approximate it using *K* **pseudo-inputs** instead of *N* observations:

$$p_\lambda(\mathbf{z}) = \frac{1}{K}\sum_{k=1}^{K} q_\phi(\mathbf{z}|\mathbf{u}_k)$$

Tomczak, J. M., & Welling, M. (2018). VAE with a VampPrior, AISTATS 2018

New Prior (**Va**riational **M**ixture of **P**osteriors **Prior**)

- We look for **the optimal prior** using the Lagrange function:

$$\max_{p_\lambda(\mathbf{z})} -\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}[-\ln p_\lambda(\mathbf{z})] + \beta\Big(\int p_\lambda(\mathbf{z})\mathrm{d}\mathbf{z} - 1\Big)$$

- The solution is simply **the aggregated posterior**. $\boxed{p^*_\lambda(\mathbf{z}) = \frac{1}{N}\sum_{n=1}^{N} q_\phi(\mathbf{z}|\mathbf{x}_n)}$

- We approximate it using *K* **pseudo-inputs** instead of *N* observations:

$$p_\lambda(\mathbf{z}) = \frac{1}{K}\sum_{k=1}^{K} q_\phi(\mathbf{z}|\mathbf{u}_k)$$

Tomczak, J. M., & Welling, M. (2018). VAE with a VampPrior, AISTATS 2018

New Prior (**Va**riational **M**ixture of **P**osteriors **Prior**)

- We look for **the optimal prior** using the Lagrange function:

$$\max_{p_\lambda(\mathbf{z})} -\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}[-\ln p_\lambda(\mathbf{z})] + \beta\left(\int p_\lambda(\mathbf{z})\mathrm{d}\mathbf{z} - 1\right)$$

- The solution is simply **the aggregated posterior**.

$$\boxed{p^*_\lambda(\mathbf{z}) = \frac{1}{N}\sum_{n=1}^{N} q_\phi(\mathbf{z}|\mathbf{x}_n)}$$

- We approximate it using K **pseudo-inputs** instead of N observations:

infeasible

$$p_\lambda(\mathbf{z}) = \frac{1}{K}\sum_{k=1}^{K} q_\phi(\mathbf{z}|\mathbf{u}_k)$$

Tomczak, J. M., & Welling, M. (2018). VAE with a VampPrior, AISTATS 2018

New Prior (**Va**riational **M**ixture of **P**osteriors **Prior**)

- We look for **the optimal prior** using the Lagrange function:

$$\max_{p_{\lambda}(\mathbf{z})} -\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}[-\ln p_{\lambda}(\mathbf{z})] + \beta\Big(\int p_{\lambda}(\mathbf{z})\mathrm{d}\mathbf{z} - 1\Big)$$

- The solution is simply **the aggregated posterior**.

- We approximate it using *K* **pseudo-inputs** instead of *N* observations:

$$p_{\lambda}(\mathbf{z}) = \frac{1}{K}\sum_{k=1}^{K} q_{\phi}(\mathbf{z}|\mathbf{u}_k)$$

Tomczak, J. M., & Welling, M. (2018). VAE with a VampPrior, AISTATS 2018

New Prior (**Va**riational **M**ixture of **P**osteriors **Prior**)

- We look for **the optimal prior** using the Lagrange function:

$$\max_{p_\lambda(\mathbf{z})} -\mathbb{E}_{\mathbf{z}\sim q(\mathbf{z})}[-\ln p_\lambda(\mathbf{z})] + \beta\Big(\int p_\lambda(\mathbf{z})\mathrm{d}\mathbf{z} - 1\Big)$$

- The solution is simply **the aggregated posterior**.

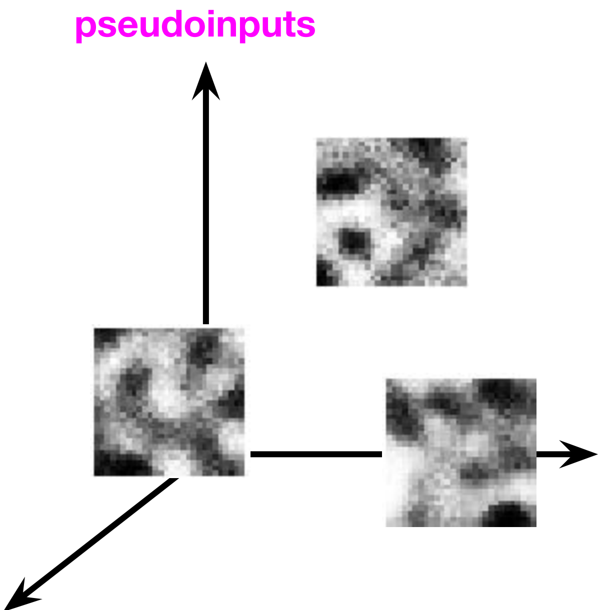
- We approximate it using *K* **pseudo-inputs** instead of *N* observations:

$$p_\lambda(\mathbf{z}) = \frac{1}{K}\sum_{k=1}^{K} q_\phi(\mathbf{z}|\mathbf{u}_k)$$

they are trained from scratch by SGD

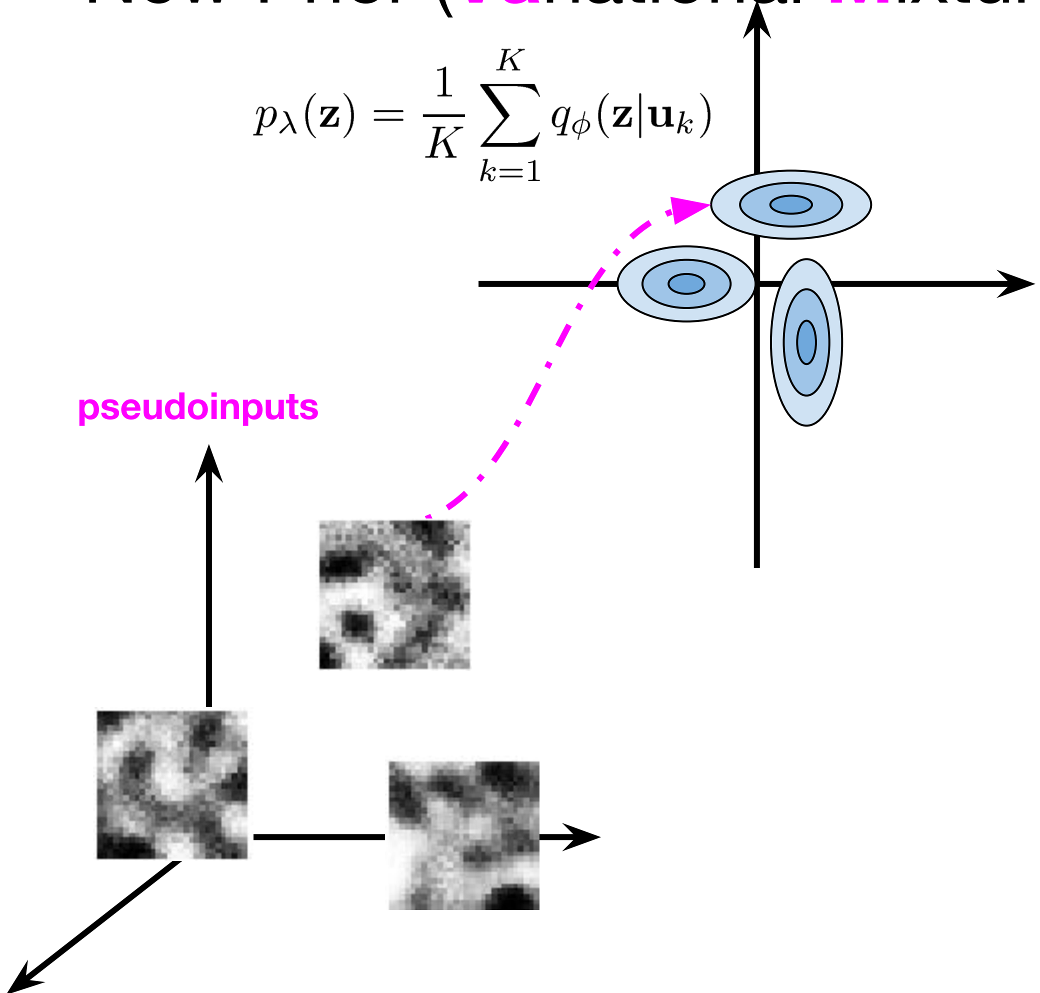
Tomczak, J. M., & Welling, M. (2018). VAE with a VampPrior, AISTATS 2018

New Prior (**Va**riational **M**ixture of **P**osteriors **Prior**)



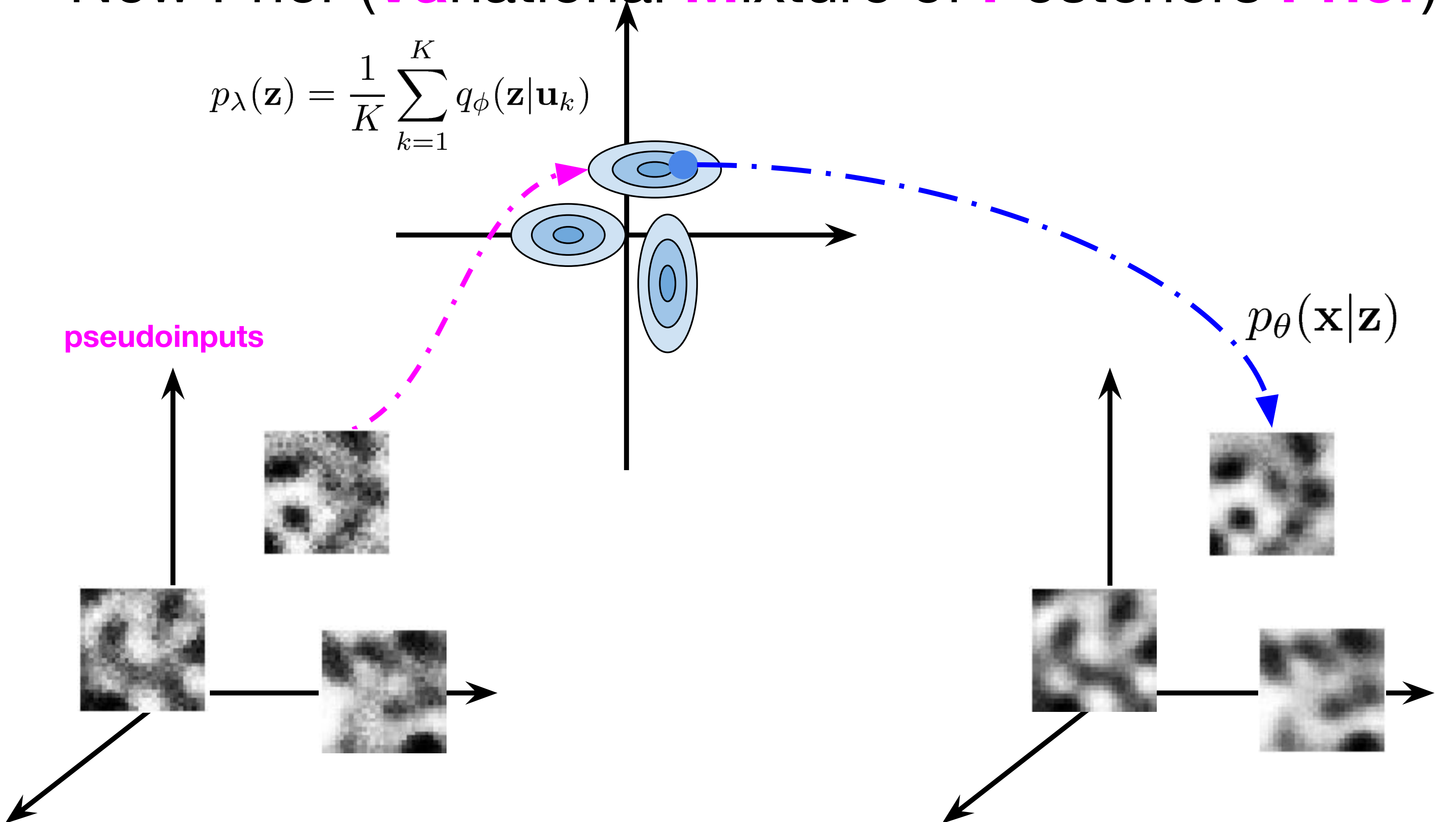
New Prior (**Va**riational **M**ixture of **P**osteriors **Prior**)

$$p_\lambda(\mathbf{z}) = \frac{1}{K}\sum_{k=1}^{K} q_\phi(\mathbf{z}|\mathbf{u}_k)$$



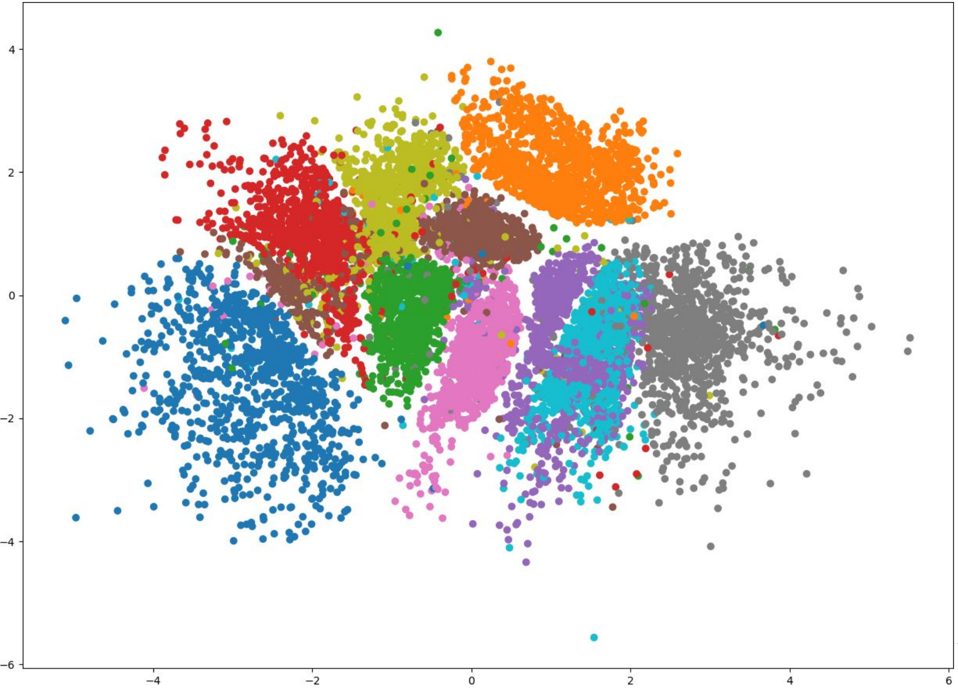
New Prior (**Va**riational **M**ixture of **P**osteriors **Prior**)

$$p_\lambda(\mathbf{z}) = \frac{1}{K}\sum_{k=1}^{K} q_\phi(\mathbf{z}|\mathbf{u}_k)$$

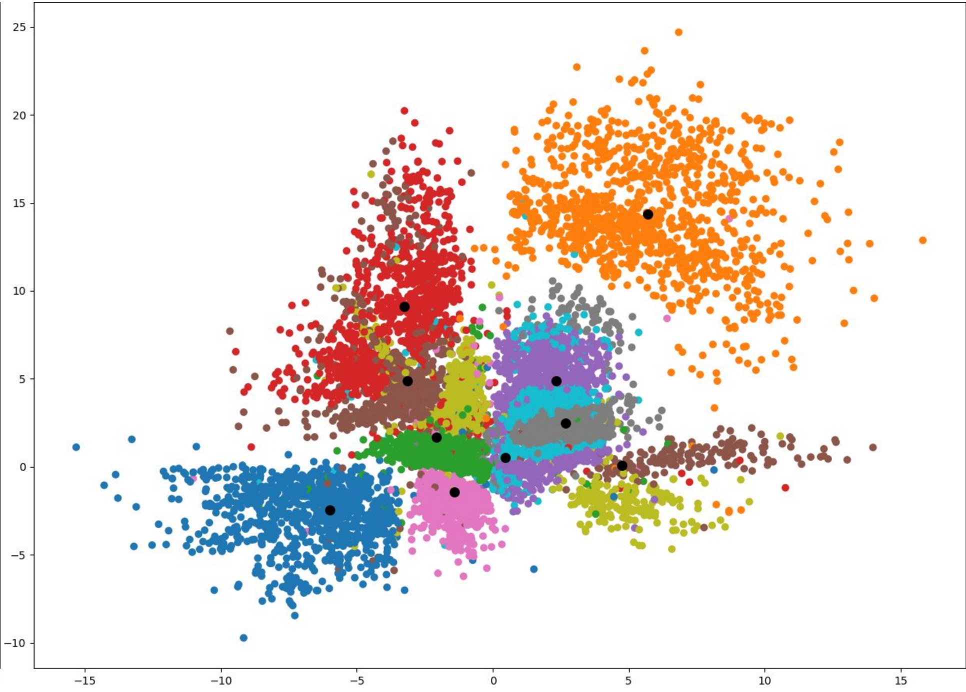


Toy problem (MNIST): VAE with dim(z)=2

Latent space representation + psedoinputs (black dots)



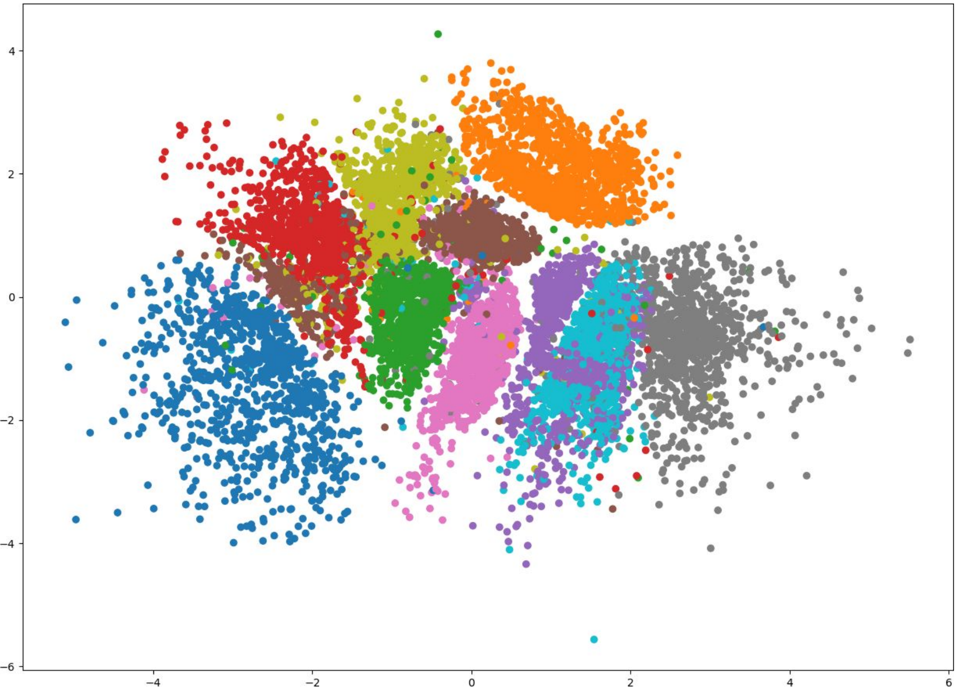
standard



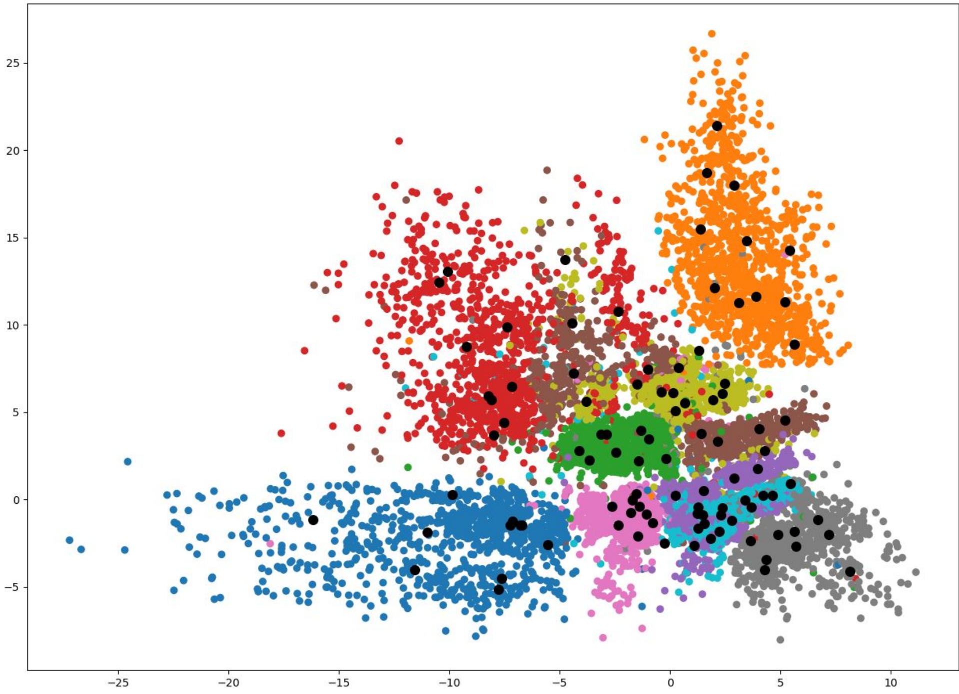
K=10

Toy problem (MNIST): VAE with dim(z)=2

Latent space representation + psedoinputs (black dots)



standard



K=100

Experiments

DATASET	VAE ($L=1$) standard	VAE ($L=1$) VampPrior	HVAE ($L=2$) standard	HVAE ($L=2$) VampPrior	convHVAE ($L=2$) standard	convHVAE ($L=2$) VampPrior	PixelHVAE ($L=2$) standard	PixelHVAE ($L=2$) VampPrior
staticMNIST	−88.56	**−85.57**	−86.05	**−83.19**	−82.41	**−81.09**	−80.58	**−79.78**
dynamicMNIST	−84.50	**−82.38**	−82.42	**−81.24**	−80.40	**−79.75**	−79.70	**−78.45**
Omniglot	−108.50	**−104.75**	−103.52	**−101.18**	−97.65	**−97.56**	−90.11	**−89.76**
Caltech 101	−123.43	**−114.55**	−112.08	**−108.28**	−106.35	**−104.22**	**−85.51**	−86.22
Frey Faces	4.63	**4.57**	4.61	**4.51**	4.49	**4.45**	4.43	**4.38**
Histopathology	6.07	**6.04**	5.82	**5.75**	5.59	**5.58**	4.84	**4.82**

Tomczak, J. M., & Welling, M. (2018). VAE with a VampPrior, AISTATS 2018

BACK

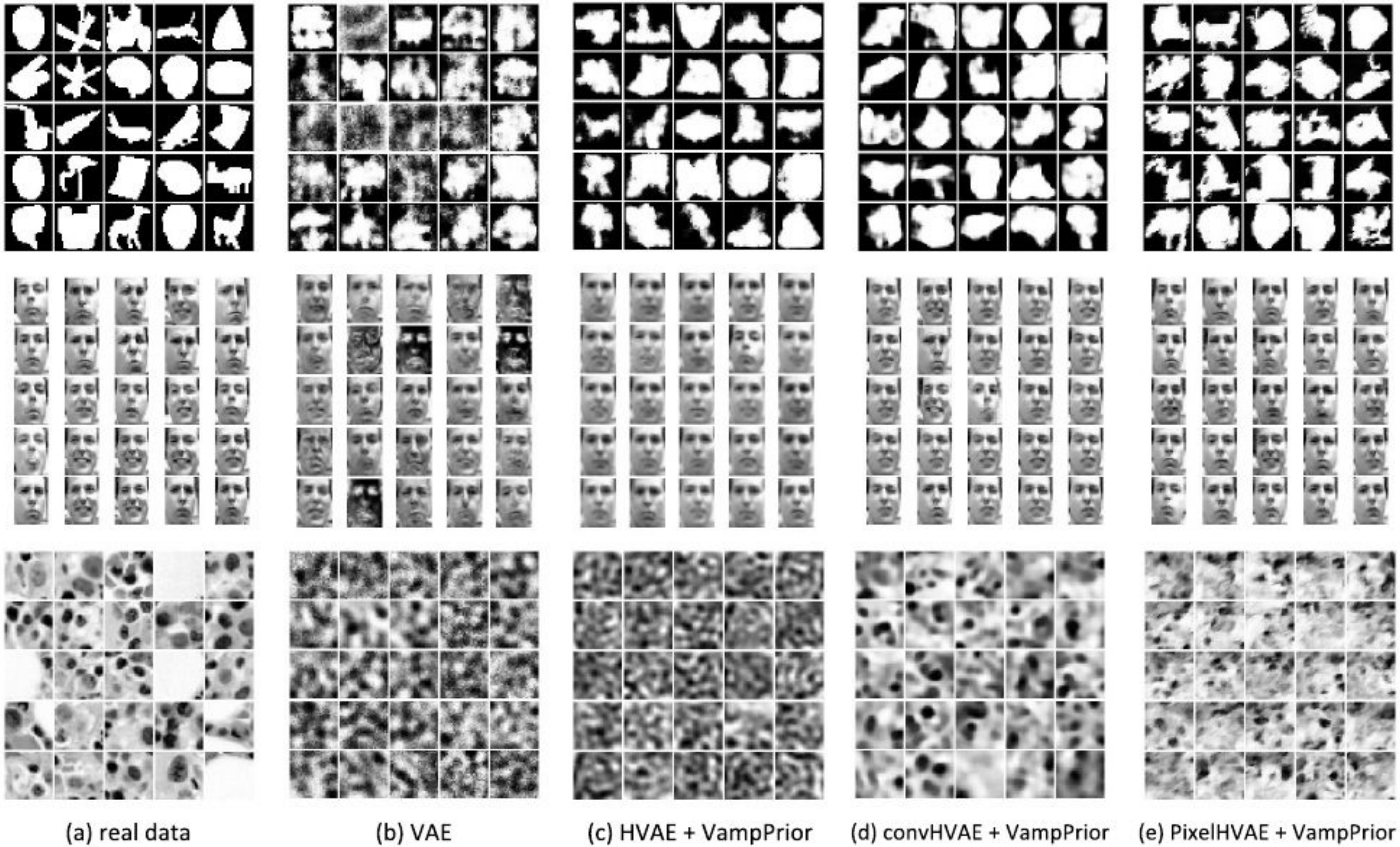


Figure 5: (a) Real images from test sets and images generated by (b) the vanilla VAE, (c) the HVAE ($L = 2$) + VampPrior, (d) the convHVAE ($L = 2$) + VampPrior and (e) the PixelHVAE ($L = 2$) + VampPrior.