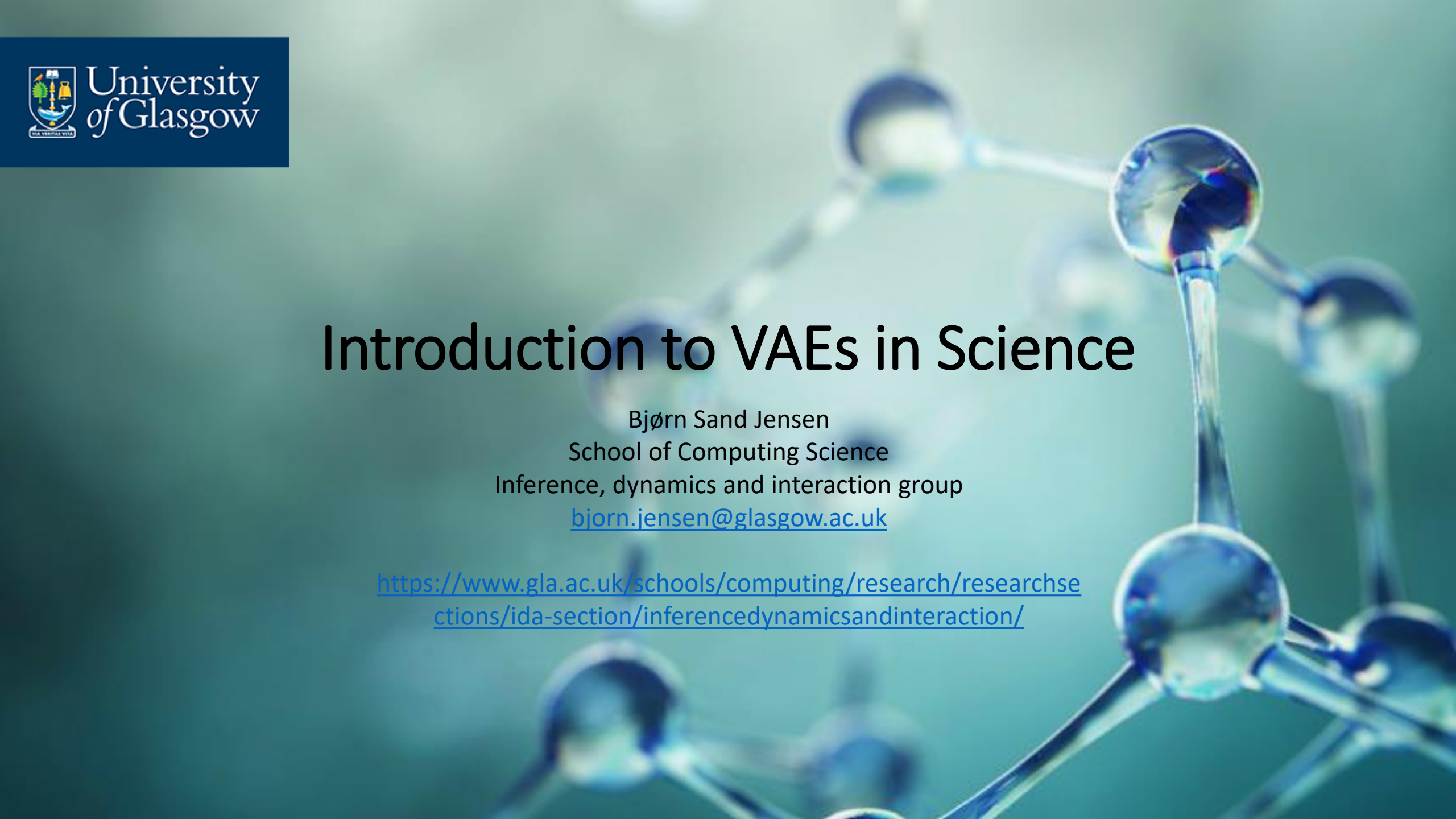


Introduction to VAEs in Science

Bjørn Sand Jensen
School of Computing Science
Inference, dynamics and interaction group
bjorn.jensen@glasgow.ac.uk

https://www.gla.ac.uk/schools/computing/research/researchsections/ida-section/inferencedynamicsandinteraction/

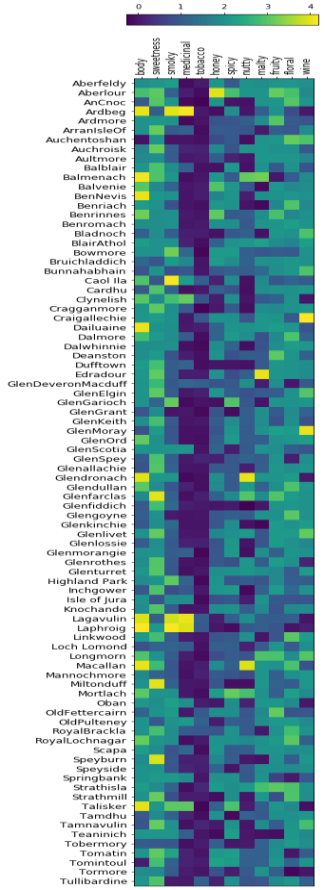
Outline

- Context: Data & Tasks
- The (basic) VAE – *yet another way to model* p(x)
- What can we do with a (basic) VAE?
  - *Representation* learning
  - Synthesis / generation
  - A basic building block (in directed graphical models)
- References

Context

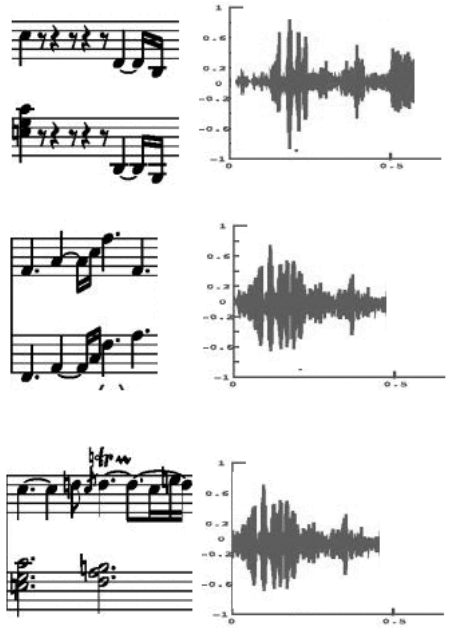
- the data, x

Vectors: Whisky tasteprofiles, LHC events,…

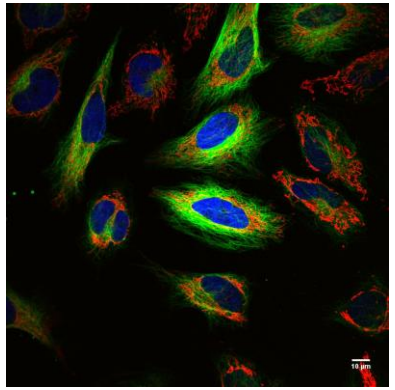


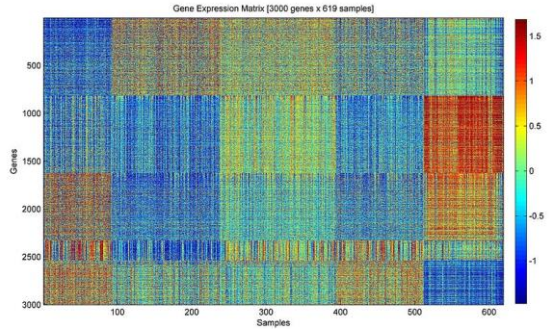
https://www.mathstat.strath.ac.uk/outreach/nessie/nessie_whisky.html

Timeseries/sequences: Text, epidemiology, weather/climate, economics (e.g. stock prices), audio

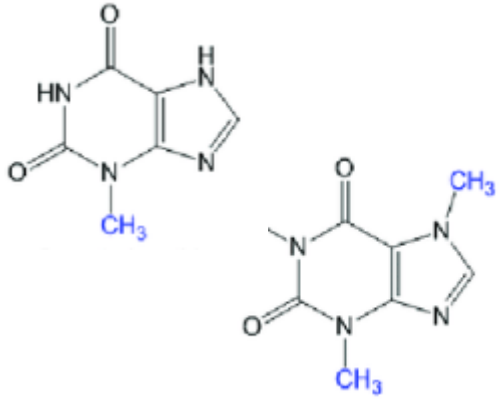


Matrices/tensors: Natural images, Microscopy, scRNA-seq,…





Graphs: Molecules, social networks, knowledge graphs…



Context

- properties

- High-dimensional (e.g. images, scRNA-seq)
- Complex: Multiple factors, several noise sources
- Low signal-to-noise ratio
- Large data sets with no, few or weak ”labels”

Tasks

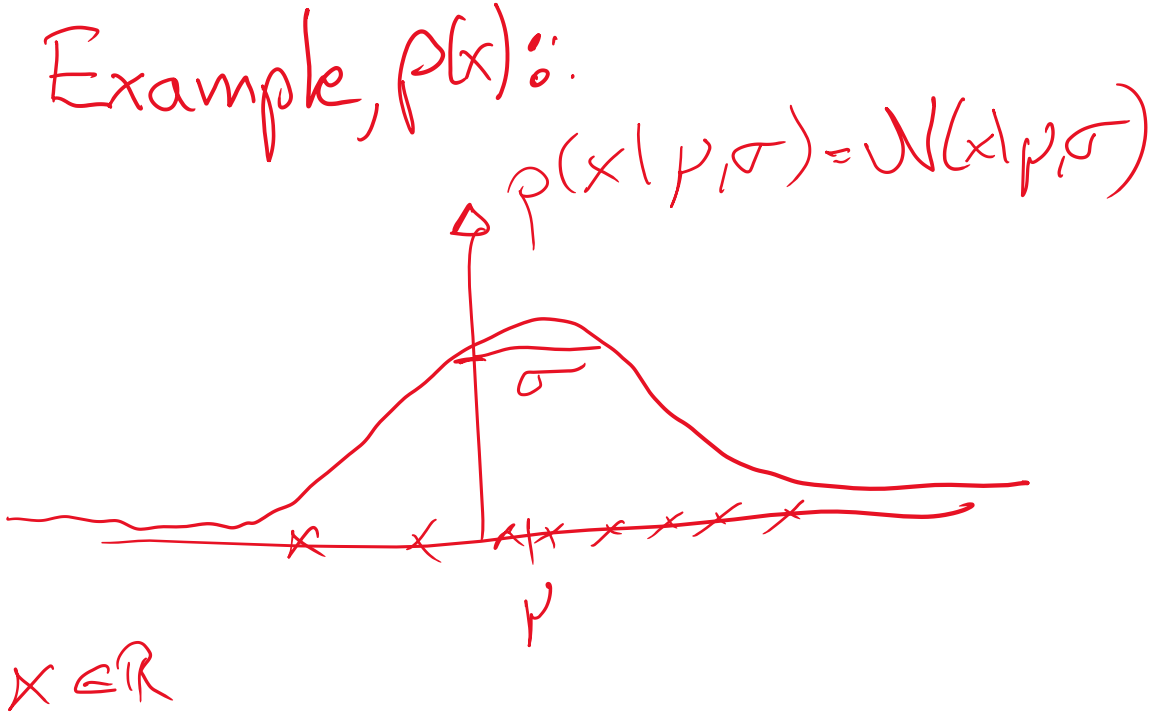
The classical ML setup

- Unsupervised, p(x)
- Supervised, p(x,y) or p(y|x) (regression, classification, ranking)
- Reinforcement learning, p(action|x)
- + combinations of the above.

the basic VAE

- … we also need to consider:
  - Experimental design
  - Data pre-processing and curation
  - Hypothesis testing
  - Explainability/interpretability

Example, p(x):



Taxonomy

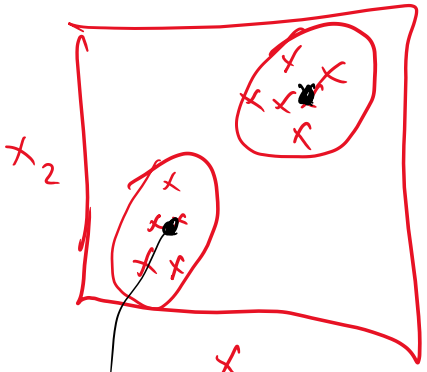
- How can we model p(x)?

Example - tractable density & latent variable model (note: also a generative model!)

- Gaussian mixture (see e.g. Bishop 2006)

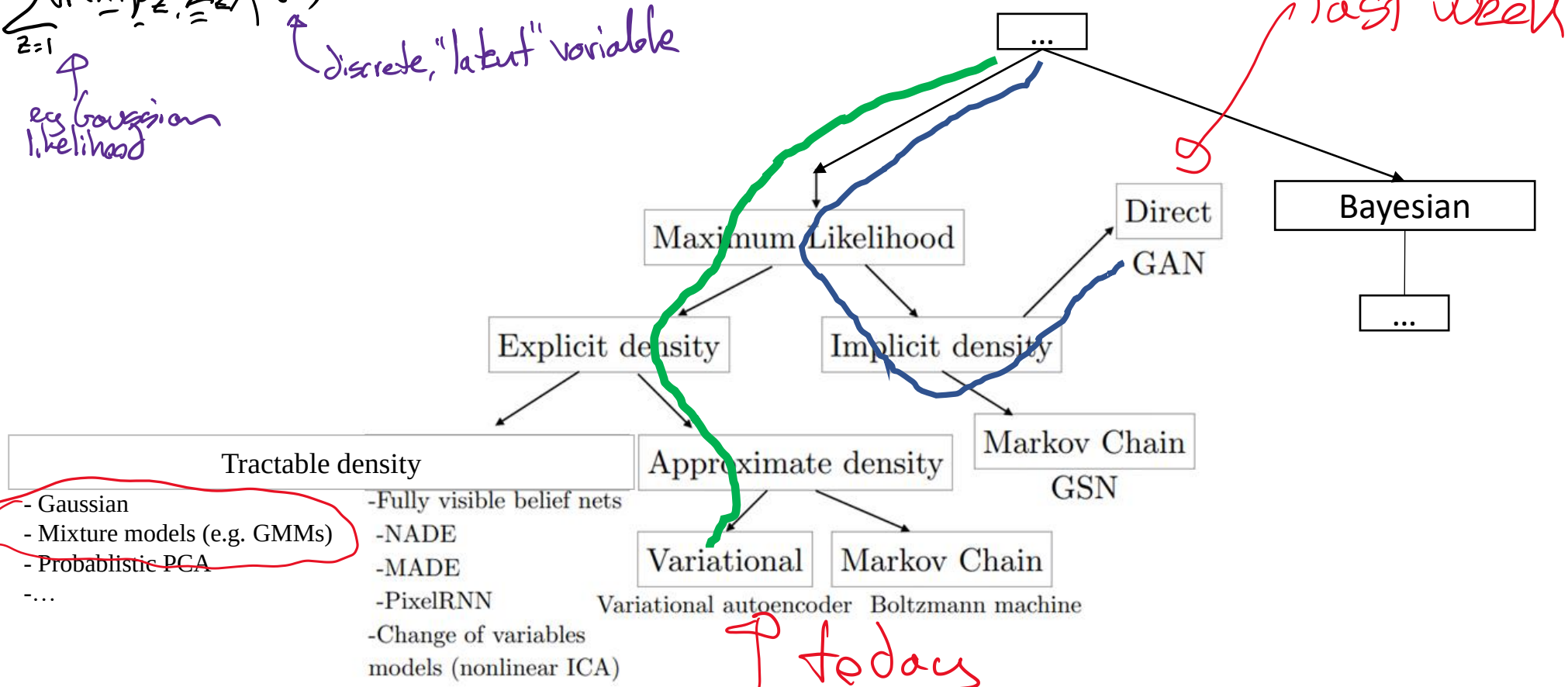
$$p(x|\{\mu_z, \Sigma_z\}_{z=1:N_z}) = \sum_{z=1}^{N_z} N(x|\mu_z, \Sigma_z) p(z)$$

parameters; e.g. Gaussian likelihood; discrete, "latent" variable



$\mu_{z=1} = [-1, -0.1]^T$

$\Sigma_{z=1} = \begin{bmatrix} 0.9 & 0.3 \\ 0.3 & 0.5 \end{bmatrix}$



The basic VAE

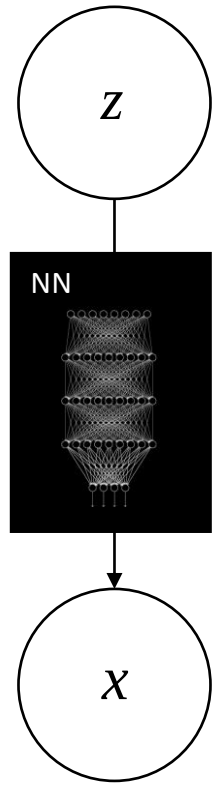
- the generative model

Old idea: Design latent variables models with continuous latent variables to model high-dimensional and complex data.

First step: Parametrise the mean and (co-)variance of the data likelihood using a flexible neural network

The NN can be any std form, e.g.:
- Multilayer perceptrons
- Convolutional neural network
- Recurrent neural network (and LSTMs)
- …

$$p(z^{(i)}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$$



$$p(x^{(i)} \mid z^{(i)}) \overset{\text{for example}}{=} \mathcal{N}\left(x^{(i)} \mid \mu_p^{(i)} = NN_{\theta_1}(z^{(i)}), \sigma_p^{(i)} = NN_{\theta_2}(z^{(i)})\right)$$

the ith observation — likelihood — the mean is parametrised by a neural network (and σ)

The generative story:

- Map from low dim $z \in \mathbb{R}$ to "high" dim $x \in \mathbb{R}^2$

1. Draw $z^{(i)}$ from $p(z)$
   E.g. with $z \in \mathbb{R}$

$z^{(i)} = -1.1$

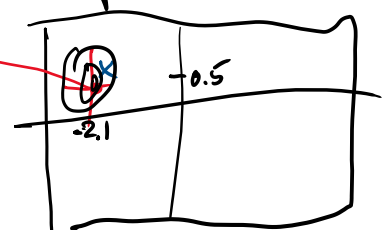
2. Squeeze $z^{(i)}$ through a NN with parameters θ_1, θ_2
   E.g.

$\mu_z^{(i)} = NN_{\theta_1}(z^{(i)} = -1.1) = [-2.1, 0.5]^T$ (here $x \in \mathbb{R}^2$ so we need 2D mean)

$\sigma_z^{(i)} = NN_{\theta_2}(z^{(i)} = -1.1) = 0.2$

3. Draw $x^{(i)}$ from parametrised likelihood $p(x^{(i)} \mid \mu_z^{(i)}, \sigma_z^{(i)})$
   e.g.

μ_z
σ_z



$x^{(i)} = [-2.0, 0.59]^T$

The basic VAE model

- inference

Problem: How can we estimate the NN parameters and compute the relevant distributions from observed data?

- Evaluation: $p(x \mid \theta)$

$$p(x \mid \theta) = \int p(x \mid z, \theta)\, p(z) dz$$

$$= \int p\left(x \mid NN_{\theta_1}(z), NN_{\theta_2}(z)\right) p(z) dz$$

$$\overset{\text{for example}}{=} \int \mathcal{N}\left(x \mid \mu = NN_{\theta_1}(z), \sigma = NN_{\theta_2}(z)\right) p(z) dz$$

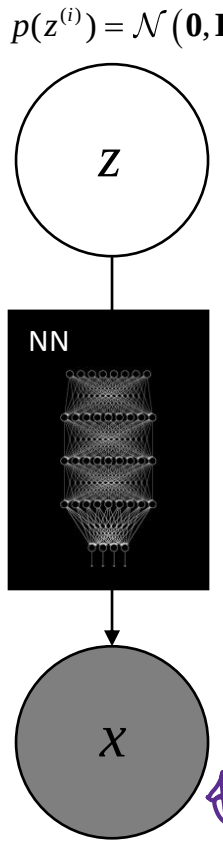
=> Intractable in many cases (e.g. NN with a non-linearity)!

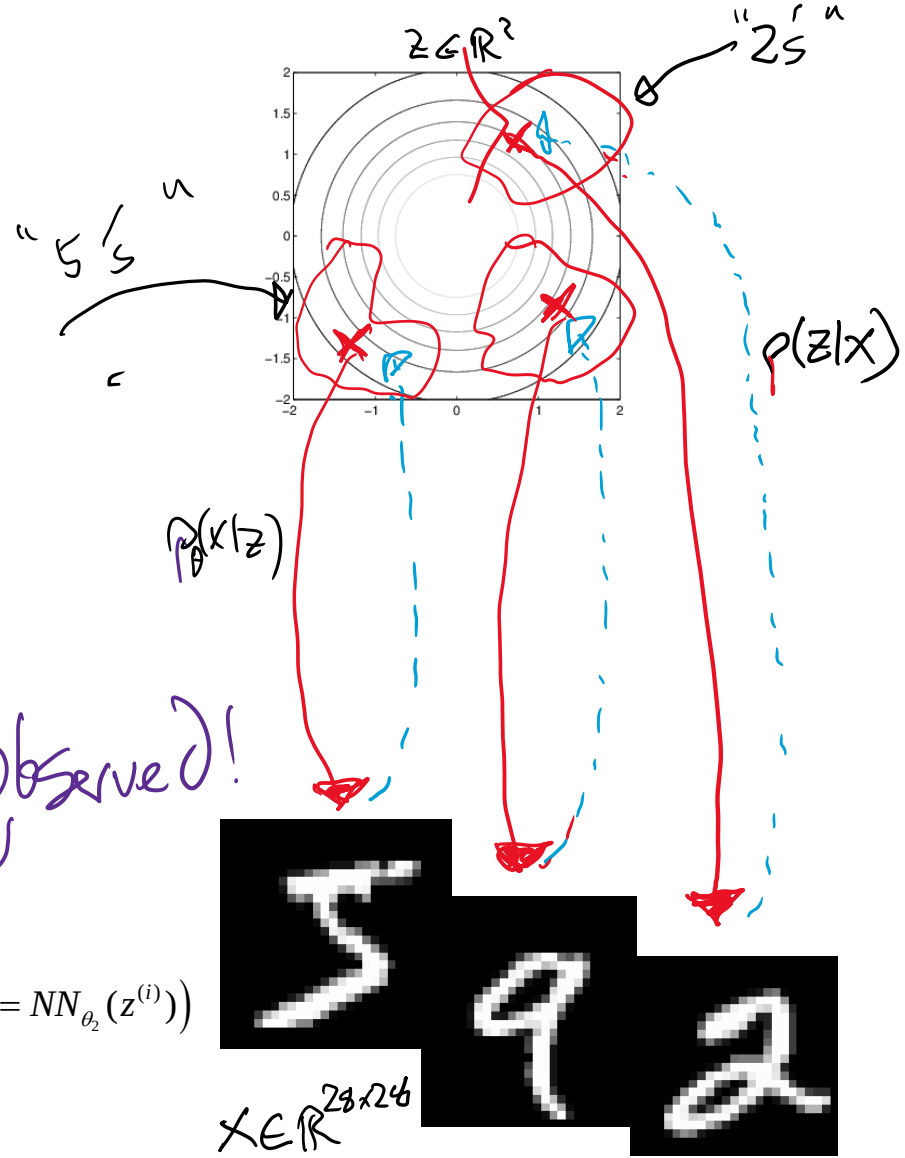
- Representation and coding : $p(z \mid x, \theta)$

=> Intractable in many cases (std EM does not apply)

- Interpretability: $p(x \mid z, \theta)$

=> Readily available given model specification.





$$p(x^{(i)} \mid z^{(i)}) \overset{\text{for example}}{=} \mathcal{N}\left(x^{(i)} \mid \mu_p^{(i)} = NN_{\theta_1}(z^{(i)}), \sigma_p^{(i)} = NN_{\theta_2}(z^{(i)})\right)$$

an image

The basic VAE model

- inference that scales (Kingma et al, 2014)

How can we estimate the parameters and compute the relevant distributions from observed data?

Idea: Let's introduce an distribution q(z|x) parametrised by another NN to approximate the true p(z|x) aka an inference network.

Objective: maximize the marginal likelihood p(x)

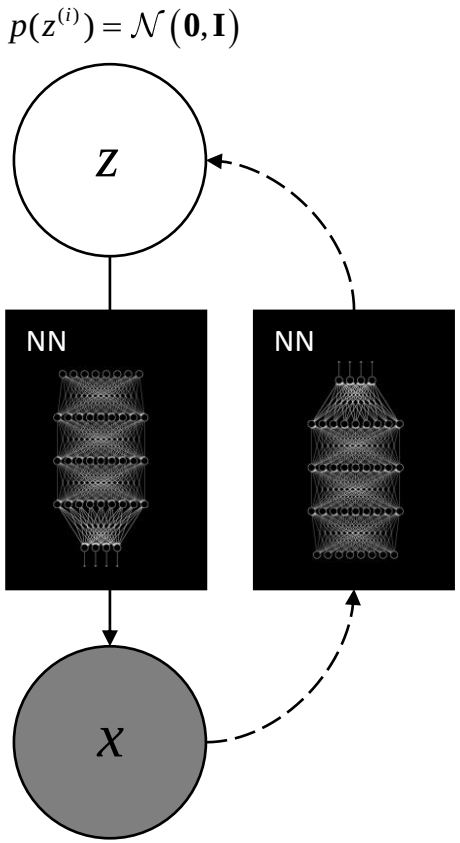
$$\log p(x) = \mathbb{E}_{q_\phi(z|x)}\left[-\log q_\phi(z \mid x) + \log p_\theta(x \mid z) + \log p(z)\right] + \underbrace{KL\left(q_\phi(z \mid x) \,\|\, p(z \mid x)\right)}_{\geq 0}$$

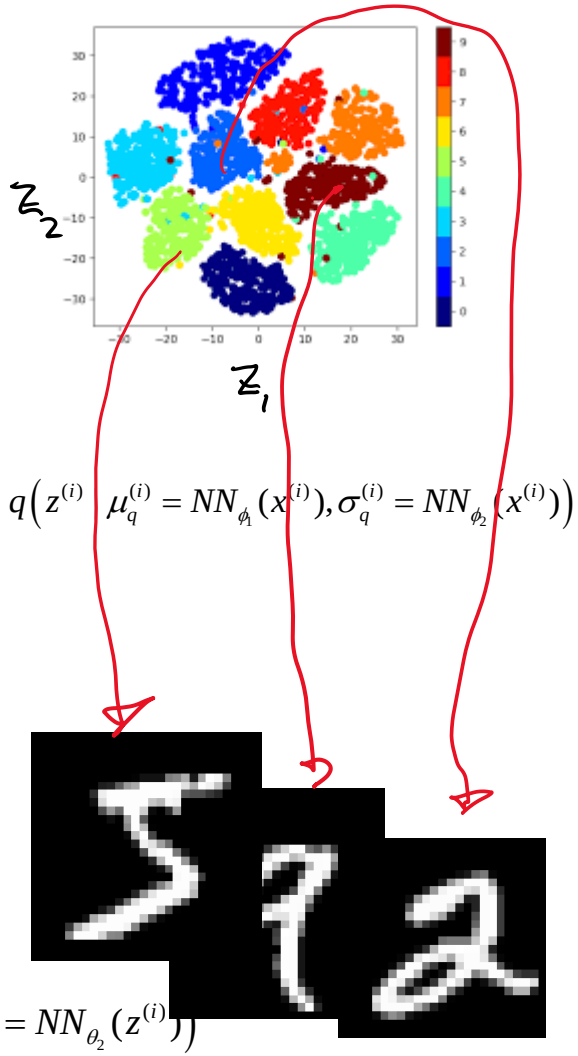
$$\geq \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z)\right] - KL\left(q_\phi(z \mid x) \,\|\, p(z)\right)$$

$$\mathcal{L}(\theta, \phi; x) = \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z)\right] - KL\left[q_\phi(z \mid x) \,\|\, p(z)\right]$$

↑ great we have a loss function!

fit data "regulariser"





$$p(x^{(i)} \mid z^{(i)}) \overset{\text{for example}}{=} \mathcal{N}\left(x^{(i)} \mid \mu_p^{(i)} = NN_{\theta_1}(z^{(i)}), \sigma_p^{(i)} = NN_{\theta_2}(z^{(i)})\right)$$

The basic VAE model

- inference that scales (Kingma et al, 2014)

How can we optimize the loss function $\mathcal{L}(\theta,\phi;x) = \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x\,|\,z)\right] - KL\left[q_\phi(z\,|\,x)\,\|\,p(z)\right]$?

$\nabla_\theta \mathcal{L}(\theta,\phi;x)$: Easy!

$\nabla_\phi \mathcal{L}(\theta,\phi;x)$: Hard! But we can use the so-calld re-parametrisation trick to get a stochastic estimate (see Kingma et al, 2014) for details.

Algorithm 1 Minibatch version of the Auto-Encoding VB (AEVB) algorithm. Either of the two SGVB estimators in section 2.3 can be used. We use settings $M = 100$ and $L = 1$ in experiments.

$\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow$ Initialize parameters
repeat
  $\mathbf{X}^M \leftarrow$ Random minibatch of M datapoints (drawn from full dataset)
  $\boldsymbol{\epsilon} \leftarrow$ Random samples from noise distribution $p(\boldsymbol{\epsilon})$
  $\mathbf{g} \leftarrow \nabla_{\boldsymbol{\theta},\boldsymbol{\phi}} \tilde{\mathcal{L}}^M(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{X}^M, \boldsymbol{\epsilon})$ (Gradients of minibatch estimator (8))
  $\boldsymbol{\theta}, \boldsymbol{\phi} \leftarrow$ Update parameters using gradients $\mathbf{g}$ (e.g. SGD or Adagrad [DHS10])
until convergence of parameters $(\boldsymbol{\theta}, \boldsymbol{\phi})$
return $\boldsymbol{\theta}, \boldsymbol{\phi}$

- Parameters: θ, ϕ
- Evaluation : $p(x\,|\,\theta)$ => we now have a lower bound ✓
- Interpretability: $p(x\,|\,z,\theta)$ => already given (via likelihood and NN) ✓
- Representation and coding: $p(z\,|\,x,\theta)$ => We can use q(z|x) ✓

$= p_\theta(x|z)$

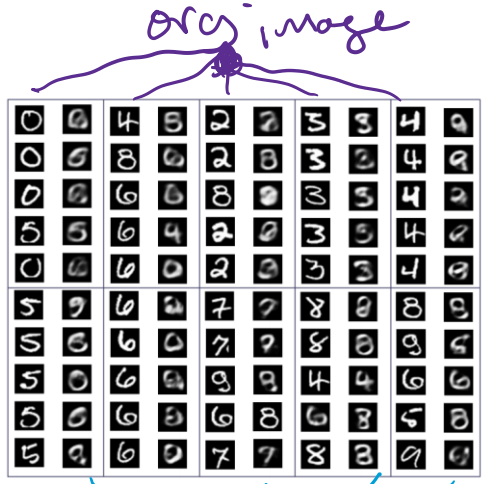
Tips/warnings:
- still quite tricky to train VAE's!
- eg. look out for posterior collapse (where the model ignores z and $KL(q(z|x)\|p(z)) \to 0$)
(see eg. Sønderby et al, 2016)

What can we do with a VAE?

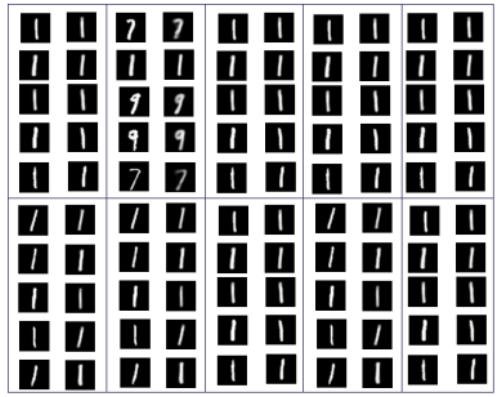
- p(x)

What can we use (a lower bound) on p(x) for?

- Density modelling
- Anomality / outliner detection (e.g. detecting novel events in the LHC, detecting issues with data collection equipment) See e.g. An et al, 2018.
- Challenge: We do not directly have access to p(x), only a lower bound or alternatively the reconstruction probability $\mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x \mid z)\right]$



low reconstruction probability (i.e. likely outlier)



high reconstruction prob (not outlier)

What can we do with a VAE?
- *representation learning*

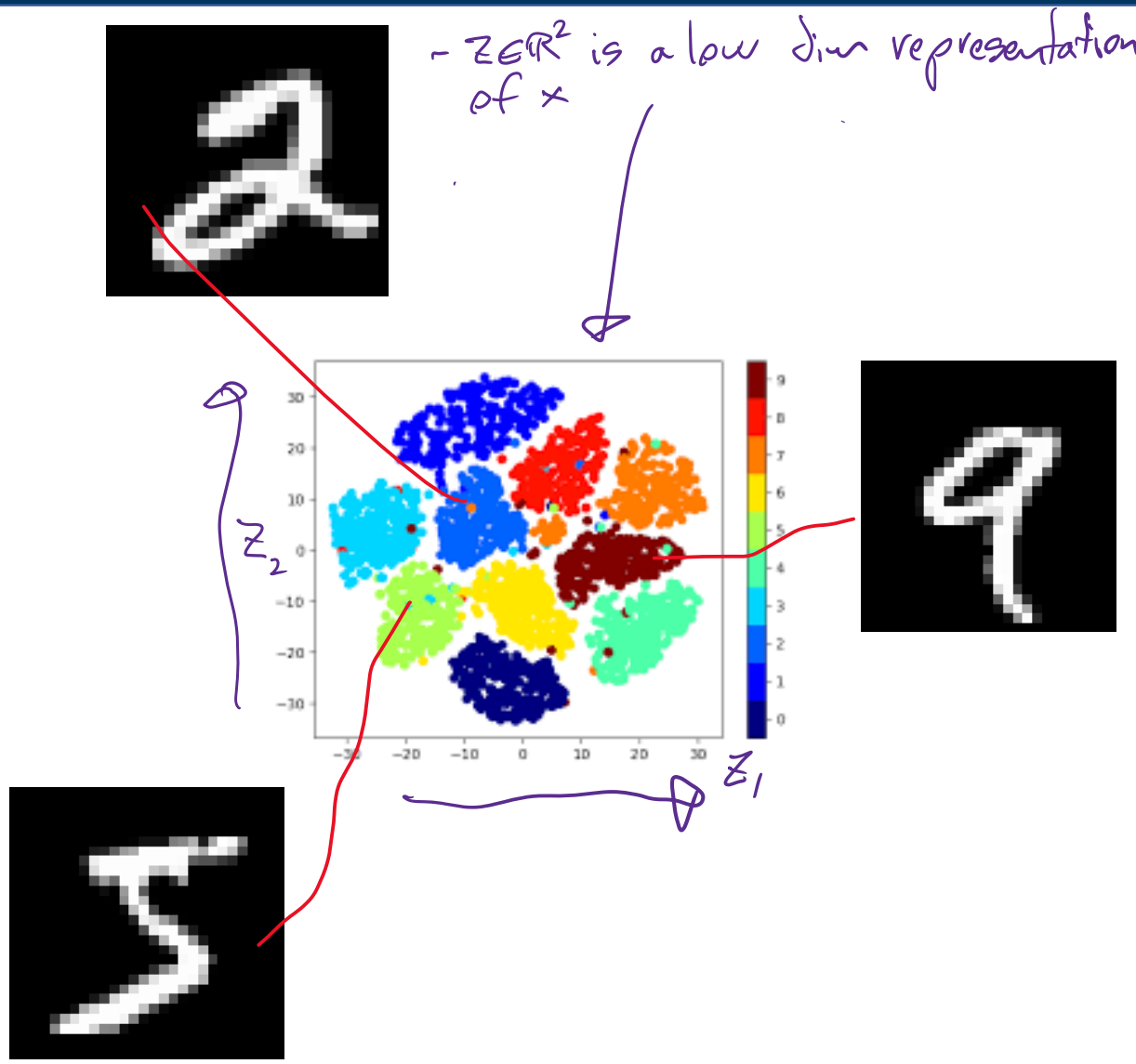
What is a representation?

Applications of q(z|x):

- Non-linear dimensionality reduction / compression
- Feature extraction (for other tasks)
- Factor analysis: Identification and analysis the latent factors giving rise the variability in the data.

Evaluation metrics:

- *Reading tea leaves (*you will always get a q(z|x), but is it really useful?)
- Linear probing (use e.g. a simple classifier to classify an auxiliary task based on z)
- Correlation / mutual information (e.g. using auxiliary data, groundtruth)
- Synthesis…



What can we do with a VAE?

- synthesis

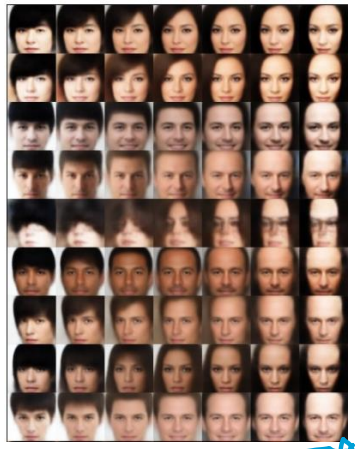
Idea: Given p(x) or p(x|z) we can draw a sample from it!

- Sample from z, then draw from p(x|z) (or use the mean)
- Sample in the neighbourhood of a known x; e.g. encode a x using q(z|x) and sample a z close to z|x.
- Interpolate between two known observations to explore the space.

Usefulness in science….?

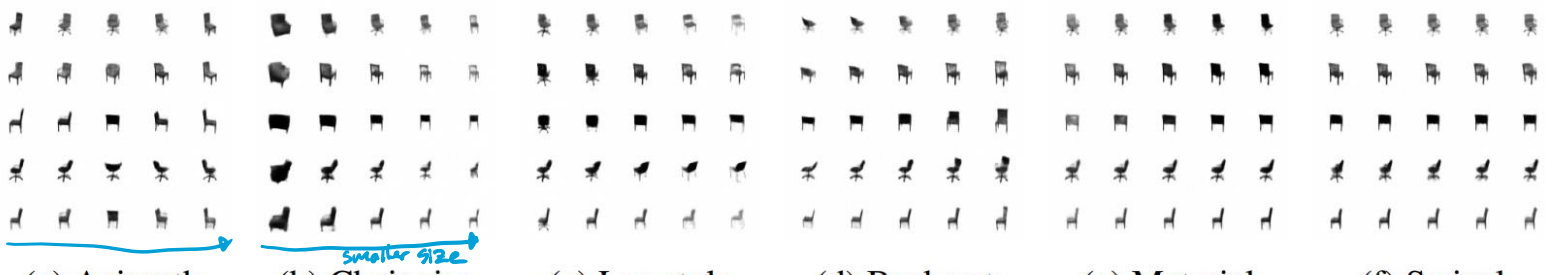
- Controlled generation to explore meaning of the z variables, i.e. the latent factors (e.g. [Chen et al, 2019])
- Explore the variability of the data (e.g. for human learners to learn new concepts)
- Design, screen and evaluate material objects (e.g. drugs, biomaterials, biological cells, etc) [Gómez-Bombarelli et al, 2017]
- …

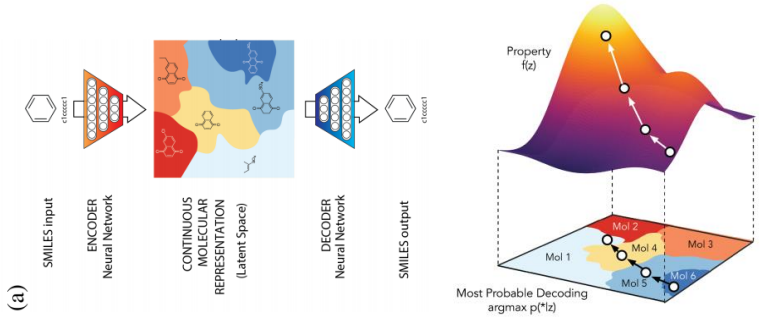
Issues: Quality of samples is sometimes a problem for VAE although large improvement with for example PixelVAE [Gulrajani, 2016]



[Chen et al, 2019, beta-TCVAE]

Warning: Obtaining these "useful" representations often require extensive training/evaluation and fine-tuning. Specific methods (e.g. beta-TCVAE) can sometimes help although there is no free lunch. See e.g. [Mathieu et al, 2018], [Chen, 2018] and [Tonolini et al, 2019] for discussion of informative representations and generation with VAEs.





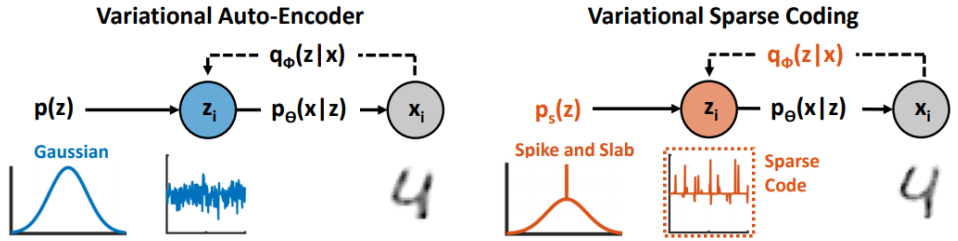
[Gómez-Bombarelli et al, 2017]

What can we do with a VAE?

- a basic building block

Idea: Use the VAE – especially the inference techniques - to build even more interesting models / applications.

Key advantage of VAEs: The graphical model allows explicit statements about assumptions using the language of probability.

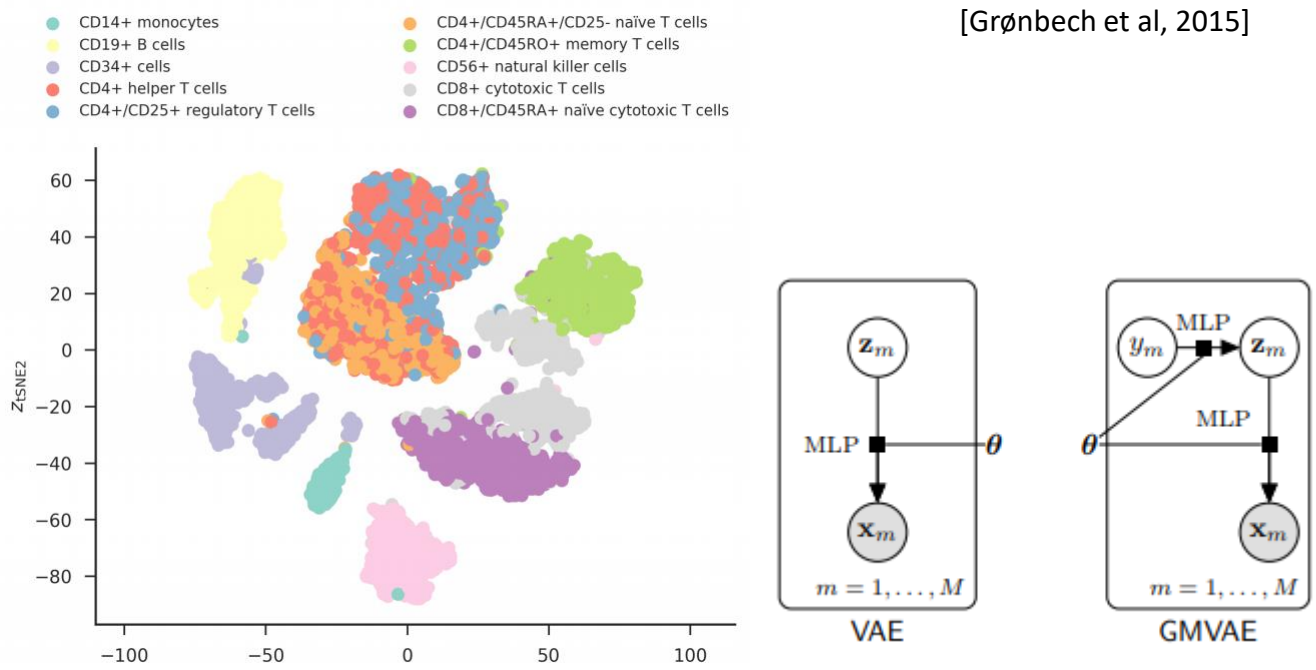


[Tonolini et al, 2019]

A few interesting extentions:

- Explict assumptions about sparsity [Tonolini et al, 2019])
- Conditional VAEs
- (Semi-) Supervised representation learning
- (Hierachical) Clustering [Grønbech et al, 2015]
- …

[Grønbech et al, 2015]



Implicit vs explicit modelling

- GANs vs VAEs

- **GANs**
  - Pros:
    - Very good quality of synthesised example (at least for images)
    - No explicit likelihood required
  - Cons:
    - Representation learning capabilities is not always a given (although see InfoGAN)
    - Difficult to train (a minmax problem)
    - Extensions sometimes difficult to incorporate in a principled fashion
- **VAEs**
  - Pros:
    - Very good likelihood performance
    - Traditionally/often superior representation learning (i.e. useful z's)
    - A principled probabilistic formulation which allows explicit formulation of assumptions (e.g. via graphical model)
  - Cons:
    - Requires explicit formulation of likelihood (implicit in GANs)
    - Requires optimizing a parameterised variational bound, although this can be made a arbitrarily flexible (e.g. normalising flows, ladder networks etc) .
    - Traditionally lower quality of synthesised images than GANs (although some attempts to improve this, e.g. VQ-VAE, PixelVAE, etc)
    - Somewhat difficult to train (but not as hard as GANs).

A few references

- many,more papers and tutorials out there)

The fundamentals / latent variable models (a good starter if you want to understand the details for latent variable models):

[Bishop, 2016] Christopher M Bishop. 2006. Pattern recognition and machine learning. Springer (free online)

Basic VAEs and traning:

[Kingma et al, 2014] Auto-Encoding Variational Bayes. Diederik P Kingma, Max Welling, ICLR 2014, https://arxiv.org/abs/1312.6114

[Rezende et al, 2014] Stochastic Backpropagation and Approximate Inference in Deep Generative Models. ICML, 2014

[Higgins et al, 2017] β-VAE: LEARNING BASIC VISUAL CONCEPTS WITH A. CONSTRAINED VARIATIONAL FRAMEWORK, ICLR 2017, https://openreview.net/pdf?id=Sy2fzU9gl (note: claims to help with training and representation learning although this is not always evident!)

[Sønderby et al, 2016] Sønderby, C. K., Raiko, T., Maaløe, L., Sønderby, S. K., & Winther, O. (2016). How to train deep variational autoencoders and probabilistic ladder networks. In 33rd International Conference on Machine Learning (ICML 2016).

Representations learning (a few core methods):

[Chen, 2018] Isolating Sources of Disentanglement in Variational Autoencoders, 2018, https://arxiv.org/abs/1802.04942 (note: this is beta-TCVAE method which improves on the std beta-VAE)

[Mathieu et al, 2018] Disentangling disentanglement in variational autoencoders. arXiv preprint arXiv:1812.02833 (2018).

[Tonolini et al, 2019] Variational Sparse Coding, UAI, 2019, http://auai.org/uai2019/proceedings/papers/239.pdf

See NeurIPS, ICML, JMLR, AISTATS, UAI, ICLR for the latest VAE techniques

Synthesis (examples papers)

[Gulrajani et al, 2016] PixelVAE: A Latent Variable Model for Natural Images, https://arxiv.org/abs/1611.05013

[Razavi et al, 2019] Generating Diverse High-Fidelity Images with VQ-VAE-2, https://arxiv.org/abs/1906.00446 (see also VQ-VAE)

Some applications of VAEs:

[An et al, 2018] Variational Autoencoder based Anomaly Detection using Reconstruction Probability. Technical Report. SNU Data Mining Center.

[Gómez-Bombarelli et al, 2017] Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules, https://arxiv.org/pdf/1610.02415.pdf

[Grønbech et al, 2015] scVAE: Variational auto-encoders for single-cell gene expression data, https://www.biorxiv.org/content/10.1101/318295v1

A few Toolboxes /code:

Keras: https://keras.io/examples/variational_autoencoder/ Tensorflow: https://www.tensorflow.org/tutorials/generative/cvae PyTorch: https://github.com/pytorch/examples/tree/master/vae