Probabilistic Graphical Models for Image Analysis - Lecture 9

Stefan Bauer

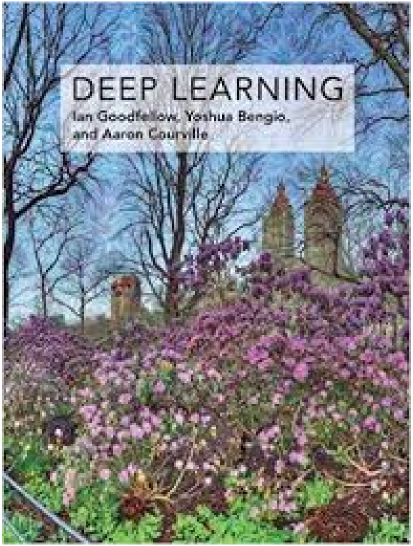
16th November 2018

Max Planck ETH Center for Learning Systems

Overview

1. Repetition
2. Kernel PCA
3. Autoencoding Variational Bayes

New Reference *



*Goodfellow, Bengio and Courville, Deep Learning
https://www.deeplearningbook.org/

Repetition

Factor Analysis Model

$z \in \mathbb{R}^k$ is a latent variable and y is the observed data:

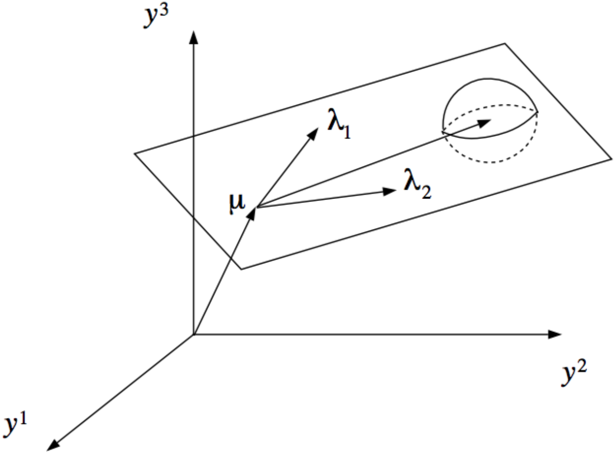
$$z \sim \mathcal{N}(0, 1)$$
$$x|z \sim \mathcal{N}(\mu + \Lambda z, \Psi)$$

Parameters of our model are thus:

- $\mu \in \mathbb{R}^n$
- $\Lambda \in \mathbb{R}^{n \times k}$
- Diagonal matrix $\Psi \in \mathbb{R}^{n \times n}$

Note: Dimensionality reduction since k is chosen smaller than n.

Illustration



where y are the observations.

Equivalent formulation

$$
\begin{aligned}
z &\sim \mathcal{N}(0, 1) \\
\varepsilon &\sim \mathcal{N}(0, \Psi) \\
x &= \mu + \Lambda z + \varepsilon
\end{aligned}
$$

where ε and z are independent.

Joint model:

$$
\begin{bmatrix} z \\ x \end{bmatrix} \sim \mathcal{N}(\mu_{zx}, \Sigma)
$$

Goal: Identify μ_{zx} and Σ.

Factor Analysis

Joint model:

$$\begin{bmatrix} z \\ x \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ \mu \end{bmatrix}, \begin{bmatrix} 1 & \Lambda^\intercal \\ \Lambda & \Lambda\Lambda^\intercal + \Psi \end{bmatrix}\right)$$

Marginal Distribution

$$x \sim \mathcal{N}(\mu, \Lambda\Lambda^\intercal + \Psi)$$

Log-Likelihood of parameters

$$l(\mu, \Lambda, \Psi) = \log \prod_{i=1}^{m} \frac{1}{(2\pi)^{\frac{n}{2}} |\Lambda\Lambda^\intercal + \Psi|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x^{(i)} - \mu)^\intercal (\Lambda\Lambda^\intercal + \Psi)^{-1} (x^{(i)} - \mu)\right)$$

Application to Images - Eigenfaces

Motivation Represent face images efficiently and capture relevant information while removing nuisance factors like lighting conditions, facial expression, occlusion etc.

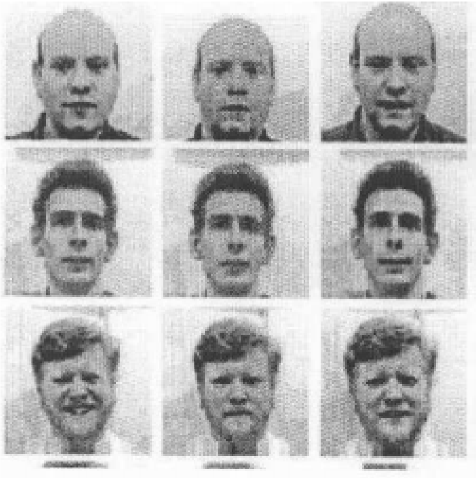
Idea

- Given training set of N images, use PCA to form a basis of K images, K«N.
- PCA for dimensionality reduction: Eigenface = eigenvector of covariance function
- Use lower dimensional features e.g. for face classification

Literature

Sirovich and Kirby, Low-dimensional procedure for the characterization of human face, 1987
Turk and Pentland, Eigenfaces for Recognition, Journal of Cognitive Neuroscience, 1991
Turk and Pentland, Face Recognition using Eigenfaces, CVPR 1991

Data [*]



[*]image from supplement Turk and Pentland, Eigenfaces for Recognition, Journal of Cognitive Neuroscience, 1991, `http://www.vision.jhu.edu/teaching/vision08/Handouts/case_study_pca1.pdf`

Basis*

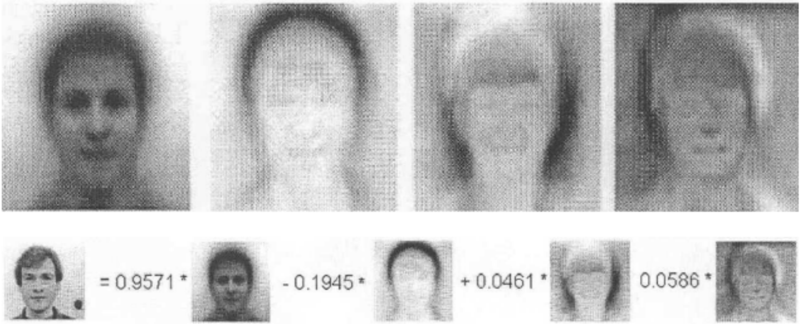


*image from supplement Turk and Pentland, Eigenfaces for Recognition, Journal of Cognitive Neuroscience, 1991, http://www.vision.jhu.edu/teaching/vision08/Handouts/case_study_pca1.pdf

Reconstruction*

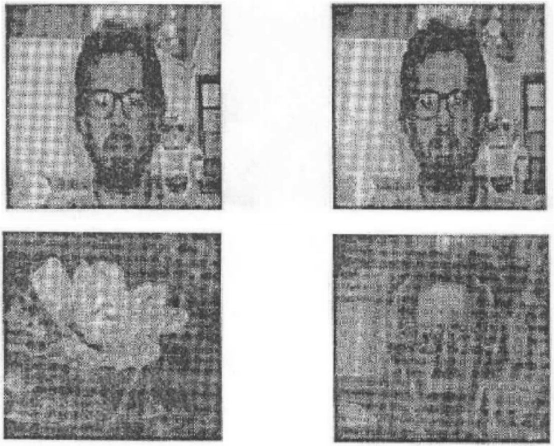
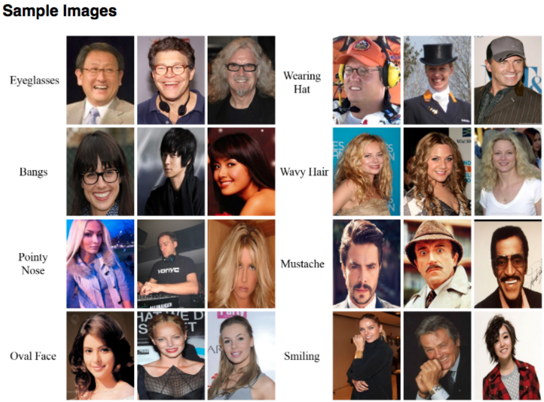


*image from supplement Turk and Pentland, Eigenfaces for Recognition, Journal of Cognitive Neuroscience, 1991, `http://www.vision.jhu.edu/teaching/vision08/Handouts/case_study_pca1.pdf`

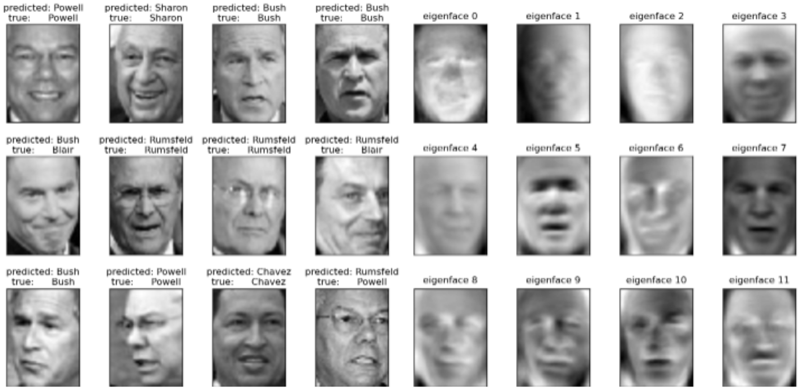
Recall Exercise

Coding Exercise: Eigenfaces using CelebA dataset (> 200K celebrity images).



Report and discuss (mean, speed, rotation, scaling, etc.) using piazza.

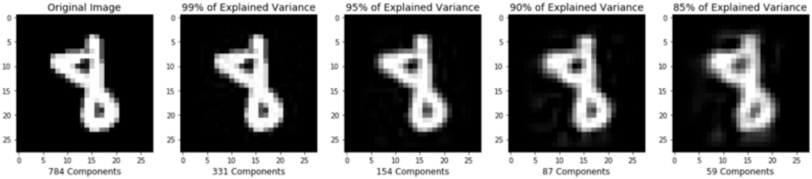
Solution to exercise



Code: `http://scikit-learn.sourceforge.net/0.8/auto_examples/applications/face_recognition.html`

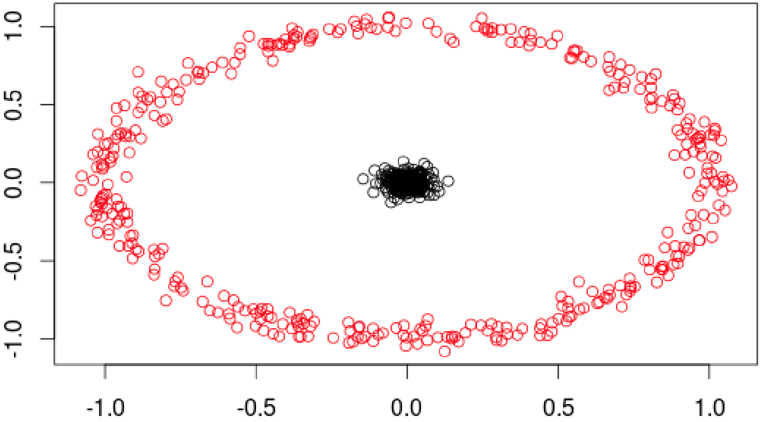
Problem Model Selection

Question: How to choose the number of components?



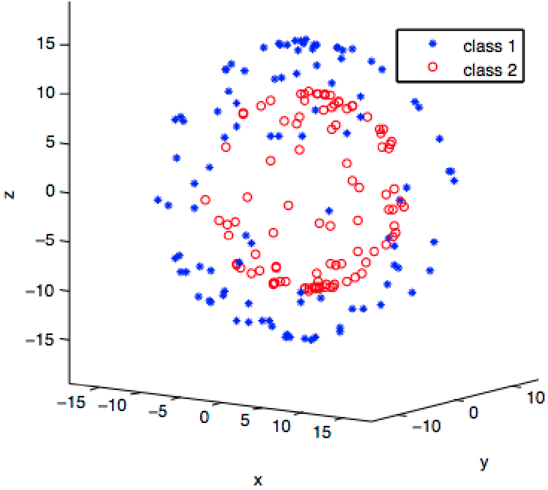
- Number of components might be constrained by problem goal, computational or storage resources e.g. typically choose only 2 or 3 components for visualization problems.
- Eigenvalues magnitudes determine explained variance (recall Lec. 7). Search for elbow criterion.
- Each spring, lecture Statistical Learning Theory `https://ml2.inf.ethz.ch/courses/slt/`.

Problem Non-linear Structure vs Random



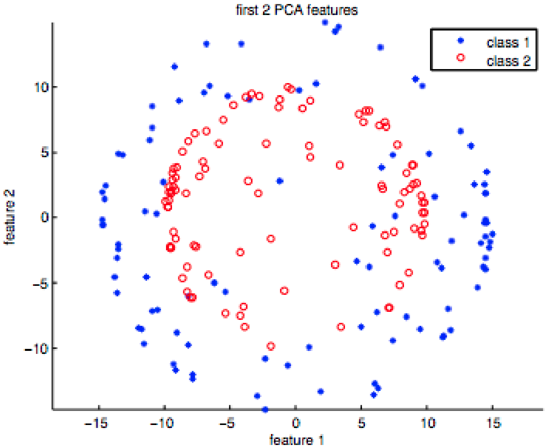
Kernel PCA

Kernel PCA*



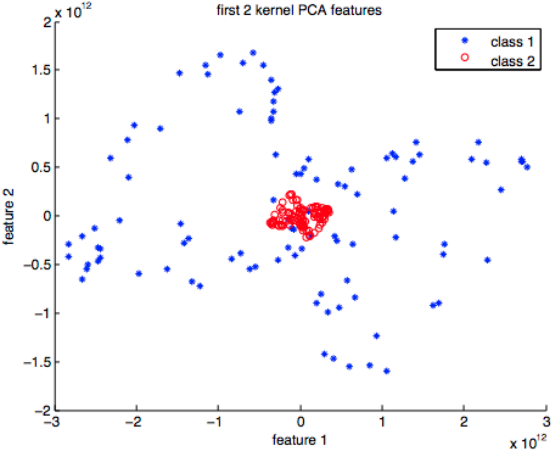
*Wang, Kernel Principal Component Analysis and its Applications in Face Recognition and Active Shape Models, 2012
`https://arxiv.org/pdf/1207.3538.pdf`

Solution using PCA*



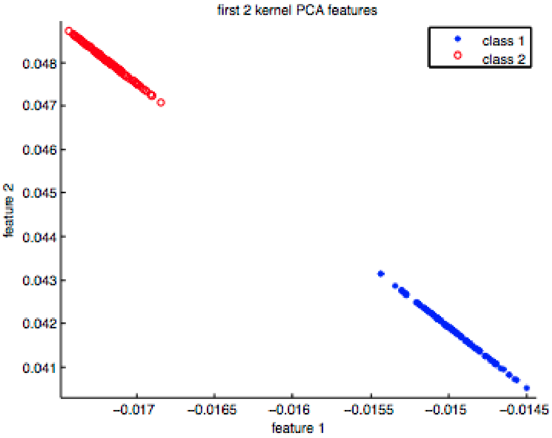
*Wang, Kernel Principal Component Analysis and its Applications in Face Recognition and Active Shape Models, 2012
`https://arxiv.org/pdf/1207.3538.pdf`

Solution using Kernel PCA (polynomial)*



*Wang, Kernel Principal Component Analysis and its Applications in Face Recognition and Active Shape Models, 2012 https://arxiv.org/pdf/1207.3538.pdf

Solution using Kernel PCA (Gaussian)*



*Wang, Kernel Principal Component Analysis and its Applications in Face Recognition and Active Shape Models, 2012
`https://arxiv.org/pdf/1207.3538.pdf`

Robustness to noise

Original Data

Data with one random noise initialization

Linear PCA

Kernel PCA

Autoencoding Variational Bayes

Deep Generative Models

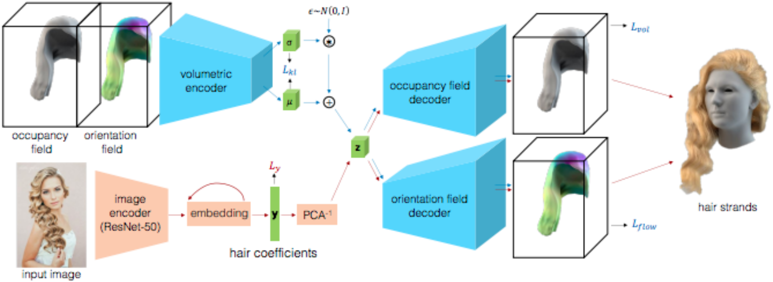
Implicit probabilistic models

- Do not specify the distribution of the data itself but rather a stochastic process which simulates the data
- Since they do not specify a distribution, they do not require a tractable likelihood
- One example are GAN’s (last week)

Explicit probabilistic models

- Specify model and use maximum likelihood
- Everything we have seen so far
- One example are Variational Autoencoders (today).

Saito et.al, 3D Hair Synthesis Using Volumetric Variational Autoencoders, SIGGRAPH Asia 2018



https://www.youtube.com/watch?v=UT2EiLG4Mrg

Framework

x observations, Z hidden variables, α additional parameters

$$p(z \mid x, \alpha) = \frac{p(z, x \mid \alpha)}{\int p(z, x \mid \alpha)} \tag{1}$$

Idea: Pick family of distributions over latent variables with its own variational parameter

$$q(z \mid \nu) = \ldots ?$$

and find variational parameters ν such that q and p are "close".

Using Jensen's inequality to obtain a lower bound

$$
\begin{aligned}
\log p(x) &= \log \int_Z p(z,x) = \\
&= \log \int_Z p(z,x) \frac{q(z)}{q(z)} \\
&= \log \mathbb{E}_q \left(\frac{p(x,Z)}{q(Z)} \right) \\
&\geq \mathbb{E}_q \left(\log p(x,Z) \right) - \mathbb{E}_q \left(\log q(Z) \right)
\end{aligned}
$$

Proposal: Choose/design variational family Q such that the expectations are easily computable.

Relation with KL

$$\begin{aligned}
\mathrm{KL}[q(z) \,||\, p(z \mid y)] &= \mathbb{E}_q \left[\log \frac{q(Z)}{p(Z \mid y)}\right] \\
&= \mathbb{E}_q[\log q(Z)] - \mathbb{E}_q[\log p(Z \mid y)] \\
&= \mathbb{E}_q[\log q(Z)] - \mathbb{E}_q[\log p(Z, y)] + \log p(y) \\
&= -\left(\mathbb{E}_q[\log p(Z, y)] - \mathbb{E}_q[\log q(Z)]\right) + \log p(y)
\end{aligned}$$

Difference between KL and ELBO is precisely the log normalizer, which does not depend on q and is bounded by the ELBO.

Mean-field for conjugates

Mean-field: $q(z, \beta) = q_\lambda(\beta) \prod_{i=1}^{k} q_{\varphi_i}(z_i)$

- λ global variational parameter
- φ local variational parameter

Local update $\varphi_i \leftarrow \mathbb{E}_\lambda[\eta_l(\beta, x_i)]$

Global update: $\lambda \leftarrow \mathbb{E}_\varphi[\eta_g(x, z)]$

Note: Coordinate ascent iterates between local and global updates.

Summary

We use variational inference to approximate the posterior distribution

$$\log p(x, \theta) = \mathrm{ELBO}(q, \theta) + \mathrm{KL}(q(z)||p(z|x, \theta)),$$

$$\log p(x, \theta) \geq \mathbb{E}_q[\log p(Z, x)] - \mathbb{E}_q[\log q(Z)]$$

To optimize the lower bound, we can use coordinate ascent!

Problems:

- In each iteration we go over all the data!
- Computing the gradient of the expectations above.

Solution: Stochastic and Black Box Variational Inference

Do not worry - It is easy to use!

Black-Box Stochastic Variational Inference in Five Lines of Python

David Duvenaud
dduvenaud@seas.harvard.edu
Harvard University

Ryan P. Adams
rpa@seas.harvard.edu
Harvard University

Abstract

Several large software engineering projects have been undertaken to support black-box inference methods. In contrast, we emphasize how easy it is to construct scalable and easy-to-use automatic inference methods using only automatic differentiation. We present a small function which computes stochastic gradients of the evidence lower bound for any differentiable posterior. As an example, we perform stochastic variational inference in a deep Bayesian neural network.

Gradient Estimates of the ELBO

$$\mathrm{ELBO} = \mathbb{E}_{q_\nu}[\log p_\theta(z, x)] - \mathbb{E}_q[\log q_\nu(z)]$$

where ν are the parameters of the variational distribution and θ the parameters of the model (as before).

Aim: Maximize the ELBO

Problem: Need unbiased estimates of $\nabla_{\nu,\theta}\mathrm{ELBO}$.

Optimizing the ELBO

$\mathbb{E}_{q(\lambda)}[\log p(z,x) - \log q(z)] =: \mathbb{E}[g(z)]$

Exercise:

$\nabla_\lambda \mathrm{ELBO} = \nabla_\lambda \mathbb{E}[g(z)] = \mathbb{E}[g(z) \nabla \log q(z)] + \mathbb{E}[\nabla g(z)]$

where $\nabla \log q(z)$ is called the *score function*.

Note: The expectation of the score function is zero for any q i.e.

$$\mathbb{E}_q[\nabla \log q(z)] = 0$$

Thus, to compute a noisy gradient of the ELBO

- sample from $q(z)$
- evaluate $\nabla \log q(z)$
- evaluate $\log p(x,z)$ and $\log q(z)$

Algorithm

Algorithm 1 Black Box Variational Inference

1: **Input:** data x, model p(x,z), variational family $q_\varphi(z)$,
2: **while** Stopping criteria is not fulfilled **do**
3: Draw L samples $z_l \sim q_\varphi(z)$
4: Update variational paramater using the collected samples

$$\varphi \leftarrow \varphi + \delta_t \frac{1}{L} \sum_{l=1}^{L} \nabla \log q(z_l)(\log p(x, z_l) - \log q(z_l))$$

5: Check step size and update if required!
6: **end while**

Note: Active research area (problem) is the reduction of the variance of the noisy gradient estimator.

Reparametrization trick

Simplified notation:

$$\nabla_\nu \mathbb{E}_{q_\nu}[f_\nu(z)]$$

Assume that there exists a fixed reparameterization such that

$$\mathbb{E}_{q_\nu}[f_\nu(z)] = \mathbb{E}_q[f_\nu(g_\nu(\varepsilon))]$$

where the expectation on the right does now not depend on ν. Then

$$\nabla_\nu \mathbb{E}_q[f_\nu(g_\nu(\varepsilon))] = \mathbb{E}_q[\nabla_\nu f_\nu(g_\nu(\varepsilon))]$$

Solution: Obtain unbiased estimates by taking a Monte Carlo estimate of the expectation on the right.

Comparison

Score Function (reinforce)

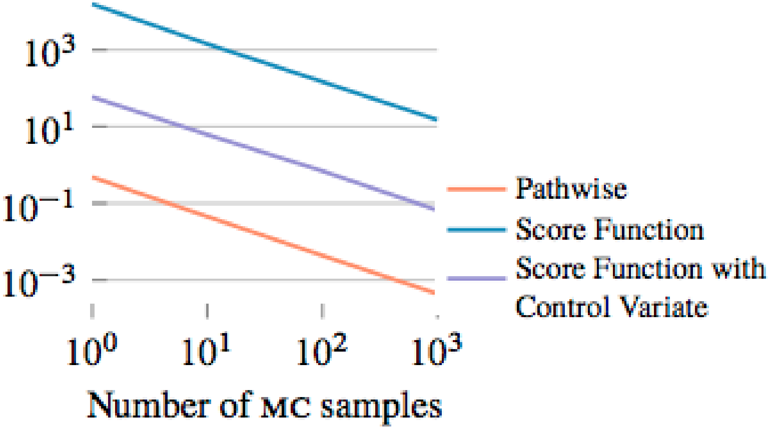
- Differentiates the density $\nabla_\nu q(z, \nu)$
- Works for discrete and continuous models
- Works for large class of variational approximations
- Variance is a big issue

Pathwise (reparameterization)

- Differentiates the function $\nabla_z[\log p(x, z) - \log q(z, \nu)]$
- requires differentiable models
- requires variational models to have special form
- In practice better behaved variance

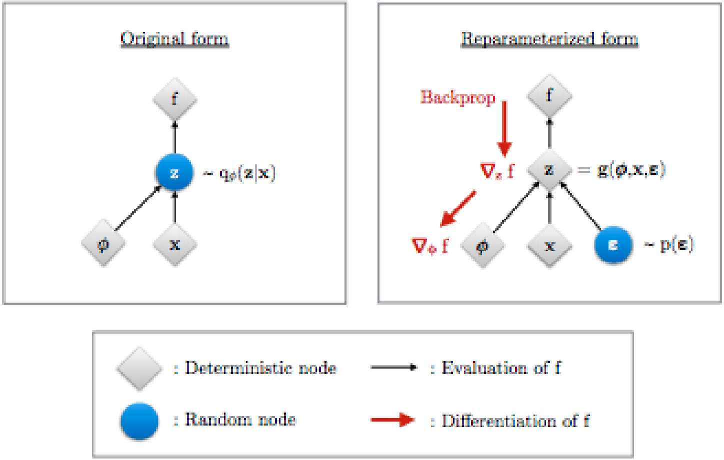
Appendix D in `https://arxiv.org/pdf/1401.4082.pdf` provides a discussion about variance of both approaches.

Variance Comparison*



*NIPS Variational Inference Tutorial 2016
https://media.nips.cc/Conferences/2016/Slides/6199-Slides.pdf

Visualization *

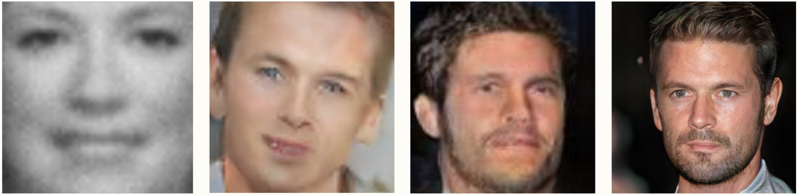


*from D. Kignma, Variational Inference & Deep Learning: A New Synthesis, PhD Thesis 2017 `https://pure.uva.nl/ws/files/17891313/Thesis.pdf`

GANs [*]

[*]Karras et.al. Progressive Growing of GANs for improved Quality, Stability, and Variation, ICLR 2018 `https://arxiv.org/pdf/1710.10196.pdf`

GAN Progress[*]



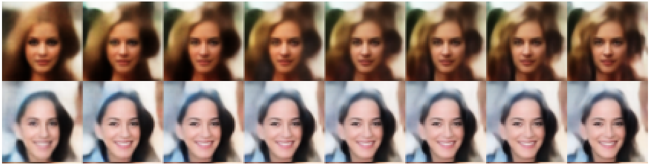
2014 2015 2016 2017

[*]Brundage et.al. 2018 https://img1.wsimg.com/blobby/go/3d82daa4-97fe-4096-9c6b-376b92c619de/downloads/1c6q2kc4v_50335.pdf

Progress GANs vs. State of the art VAE

GAN *

Recent VAE*



*Brundage et.al. 2018 `https://arxiv.org/pdf/1802.07228.pdf`

*Zhao et.al 2017 `https://arxiv.org/pdf/1702.08658.pdf`

Summary

Variational Autoencoders

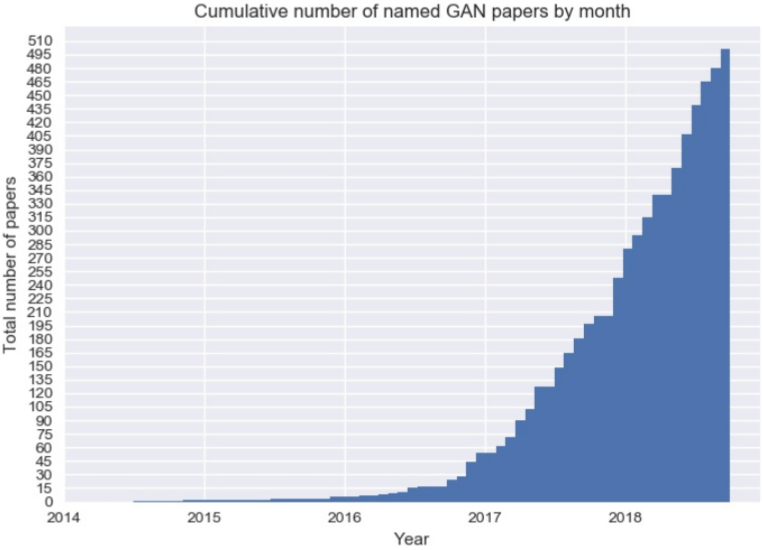
- for image generation, necessity to reconstruct each pixel
- reparametrization is not applicable to discrete latent variables
- usually only allows to use a fixed standard normal as a prior
- images are often blurry compared to high-fidelity samples generated by GANs
- allows for efficient Bayesian inference

Generative Adversarial Networks

- Instability of training
- mode collapse i.e. generated samples are often only from a few modes of the data distribution
- only visual inspection since GANs do not support inference (can additionally train an inference network)
- does not support discrete visible variables

Generally: We are unable to control the attributes of generated samples e.g. aim for regularization which enforces disentangled latent codes.

Gan Zoo*



*https://github.com/hindupuravinash/the-gan-zoo

Next week

So far missing

- Wake-Sleep Algorithm
- Independent Components
- Combination of State Space Models with Autoencoder
- Identification of number of latent components; Bayesian non-parametrics

Plan for Exercise on Tuesday:

Different Derivation based on Tutorial on Variational Autoencoders `https://arxiv.org/abs/1606.05908`

Questions?