Generative Adversarial Networks

personal favourites from 2016-2017

Botond Cseke / VW Datalab

Disclaimer: all figures are copied from the corresponding papers.

Contents

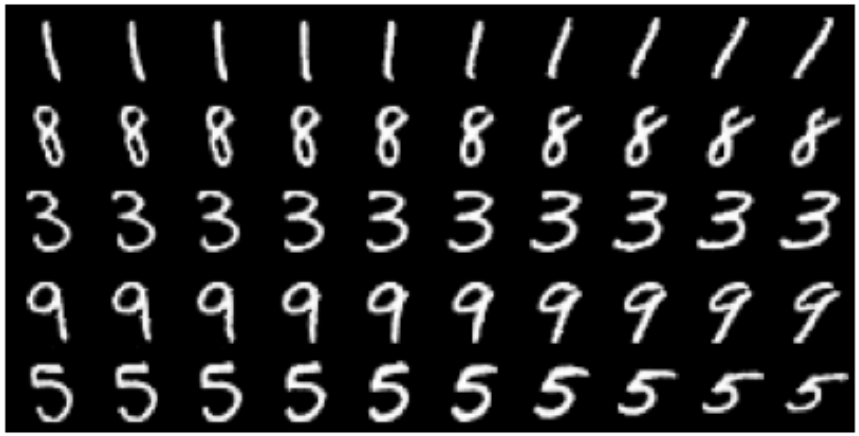
GAN models

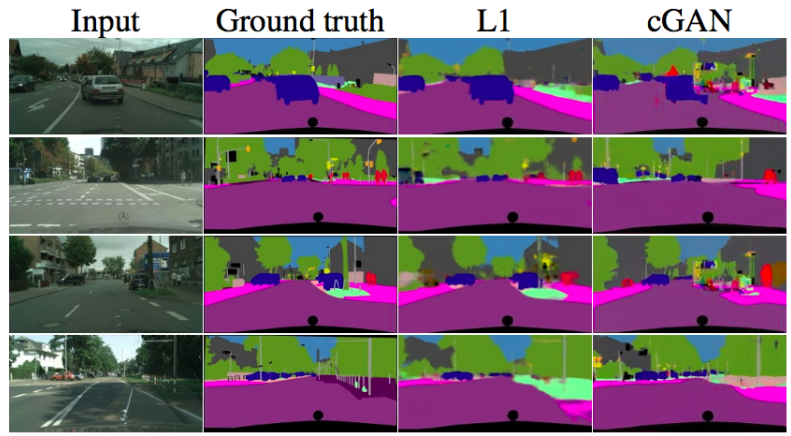
- **GAN: Generative Adversarial Nets**
- InfoGAN: learning semantic features
- CycleGAN: unpaired image-to-image translation
- f-GAN: learning neural samplers by div. min.

GAN models for Bayesian inference

- VAE: Neural variational Bayes
- ALI: Adversarially Learned Inference
- AVB: Adversarial Variational Bayes

Supervised and unsupervised learning





Unsupervised learning

- model

$$p_\theta(x) = \int p_\theta(x \mid z)\, p_0(z) dz$$

- approximation

$$p_\theta(x) \approx \hat{p}(x) = \frac{1}{S} \sum_s \delta(x - x_s)$$

Supervised learning

- model

$$p_\theta(x|u) = \int p_\theta(x \mid u, z)\, p_0(z) dz$$

- approximation

$$p_\theta(x|u) \approx \hat{p}(x|u) = \frac{1}{S} \sum_s \delta((x,u) - (x_s, u_s))$$

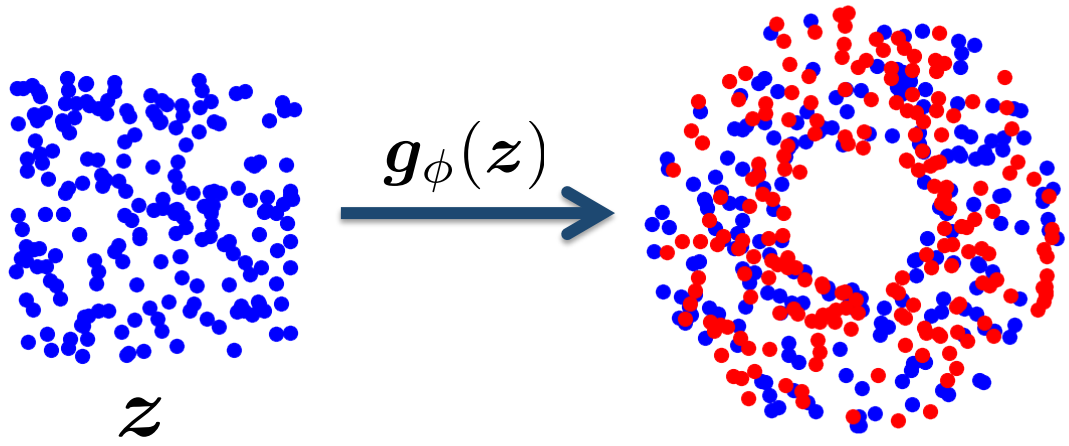


Figure: Mapping a standard distribution (blue) to approx. a data distribution (red)

Learn a random number generator G_θ to match $\hat{p}(x)$

$$x' = G_\theta(z), \quad z \sim p_0(z) \quad \text{where} \quad p_\theta(x') = \int dz\, p_0(z)\, \delta(x' - G_\theta(z))$$

Constraint on training: have access to p and p_θ only through samples

$$x \sim \hat{p}(x) \quad \text{and} \quad x' \sim p_\theta(x')$$

Limitations in learning / training

Ways to define the objectives

1. minimising any loss that at the minimum matches two distributions that is

$$\min_\theta L(p, p_\theta) \;\rightarrow\; p_\theta = p$$

2. sample based divergence minimisation approach $\min_\theta D[p_\theta(x) \,\|\, \hat{p}(x)]$

To fit θ in $p_\theta(x)$ we need the following components

1. loss functions that *use only expectations w.r.t. p_θ / MC samples*

$$E_{\hat{p}}[l(x)] \approx \frac{1}{S}\sum_s l(x_s) \quad \textit{and} \quad E_{p_\theta}[l(x')] = E_{p_0}[l(G_\theta(z))] \approx \frac{1}{S}\sum_s l(G_\theta(z_s)))$$

2. *an* autodiff framework to propagate differentials through x'

$$\frac{\partial}{\partial\theta} E_{q_\theta}[l(x')] \approx \frac{1}{S}\sum_s \frac{\partial}{\partial x} l(G_\theta(z_s)))\frac{\partial}{\partial\theta} G_\theta(z_s)$$

Generative Adversarial Nets

Ian J. Goodfellow,* Jean Pouget-Abadie,† Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair,‡ Aaron Courville, Yoshua Bengio§
Département d'informatique et de recherche opérationnelle
Université de Montréal
Montréal, QC H3C 3J7

Inference as classification

1. construct two classes with real and generated data distributions

$$\mathcal{C}_1 = \{(x, 1) : \; x \sim \hat{p}(x)\} \qquad \text{and} \qquad \mathcal{C}_0 = \{(x', 0) : \; x' = G_\theta(z), \; z \sim p_0(z)\}$$

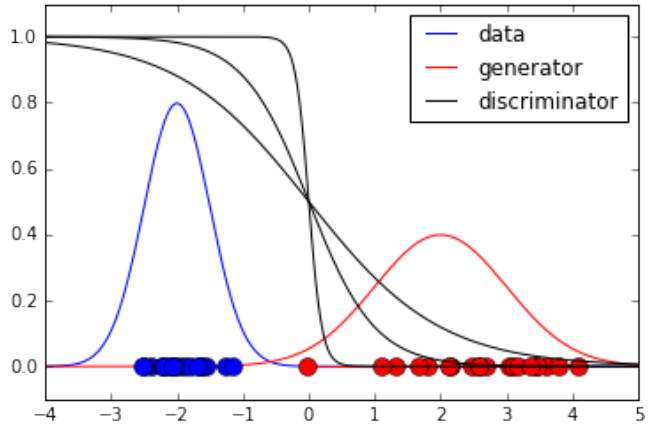
2. define the classification (quasi) empirical loss

$$L(\theta, \phi) = E_{(x,y)\sim\mathcal{C}_1}[l(D_\phi(x), y)] + E_{(x',y')\sim\mathcal{C}_0}[l(D_\phi(x'), y')]$$

Training

1. train a classifier $D_\phi(x)$ minimise the loss
2. train generator $G_\theta(z)$ to maximise loss

GAN / Adversarial classification



Inference as classification

1. construct two classes with real and generated data distributions

$$\mathcal{C}_1 = \{(x, 1) : \; x \sim p(x)\} \qquad \text{and} \qquad \mathcal{C}_0 = \{(x', 0) : \; x' = G_\theta(z), \; z \sim p(x)\}$$

2. define the classification (quasi) empirical loss

$$\begin{aligned} L(\theta, \phi) &= E_{(x,y)\sim\mathcal{C}_1}[l(D_\phi(x), y)] + E_{(x',y')\sim\mathcal{C}_0}[l(D_\phi(x'), y')] \\ &= E_{(x,y)\sim\mathcal{C}_1}[l(D_\phi(x), y)] + E_{(z',y')\sim\mathcal{C}_0}[l(D_\phi(G_\theta(z)), y')] \end{aligned}$$

Training

1. minimise the loss

$$\min_\phi L(\theta, \phi)$$

2. maximise loss

$$\max_\theta \min_\phi L(\theta, \phi)$$

show video

GAN / Possible loss functions

1. Bernoulli classification likelihood

$$L(p, p_\theta) = E_{(x,y)\sim\mathcal{C}_1}[-y \log D_\phi(x) - \cancel{(1-y)\log(1 - D_\phi(x))}]$$
$$\cancel{E_{(x',y')\sim\mathcal{C}_0}[-y' \log D_\phi(x')} - (1-y')\log\big(1 - D_\phi(x')\big)]$$

$$= E_{x\sim p(x)}[-\log D_\phi(x)] + E_{x'\sim p_\theta(x)}[-\log(1 - D_\phi(x')]$$

where we have $D^* = p/(p + p_\theta)$ and thus $L(p, p_\theta) = D_{JS}[p_\theta \,||p] + \text{const.}$

For $D_\phi(x) = \sigma(r_\phi(x))$ we get $r^*_\phi = \log p - \log p_\theta$.

2. classification as regression based likelihoods (LS-GAN)

$$L(p, p_\theta) = E_{(x,y)\sim\mathcal{C}_1}[\big(D_\phi(x) - y\big)^2] + E_{(x',y')\sim\mathcal{C}_0}[(D_\phi(x') - y'])^2]$$
$$= E_{x\sim\hat{p}(x)}[\big(D_\phi(x) - 1\big)^2] + E_{x'\sim p_\theta(x')}[D_\phi(x')^2]$$

Similar iff. solution $D^* = p/(p + p_\theta)$.

GAN / Training algorithm

Optimisation

$$\min_{\theta} \max_{\phi} E_{x \sim \hat{p}(x)}[\log D_{\phi}(x)] + E_{z \sim p_0(z)}[\log(1 - D_{\phi}(G_{\theta}(z))]$$

Stochastic gradient ascent-descent

1. sample $x_s \sim \hat{p}(x)$ and $z_s \sim \mathcal{N}(0, I_d)$ or $z_s \sim \mathcal{U}_d(0, 1)$
2. MC approximation of the objective

$$L = \frac{1}{N} \sum_i \log D_{\phi}(x_i) + \log(1 - D_{\phi}(G_{\theta}(z_i))$$

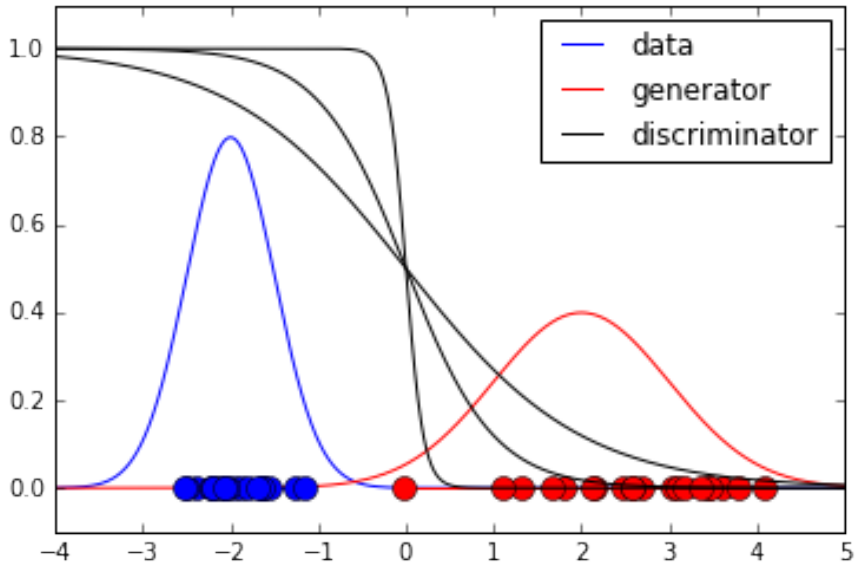
3. gradient ascent on ϕ

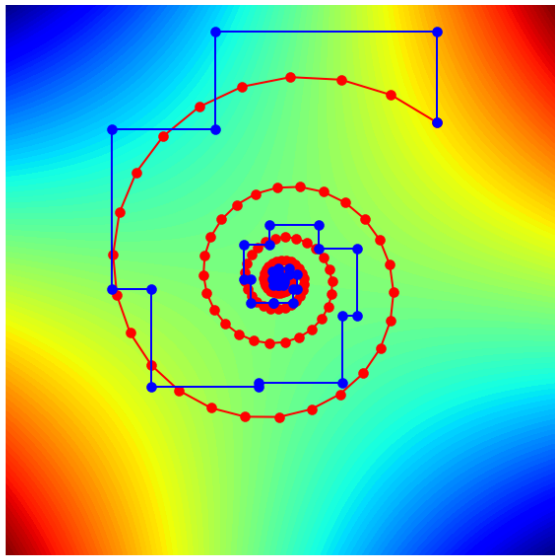
$$\phi^{(t+1)} = \phi^{(t)} + \eta_{\phi} \frac{1}{N} \sum_i \partial_{\phi} \log D_{\phi}(x_i) + \partial_{\phi} \log(1 - D_{\phi}(G_{\theta}(z_i))$$

4. gradient descent on θ

$$\theta^{(t+1)} = \theta^{(t)} - \eta_{\theta} \frac{1}{N} \sum_i \partial_{\theta} \log(1 - D_{\phi}(G_{\theta}(z_i))$$

GAN / Training Algorithm





1. sample $x_i \sim \hat{p}(x)$ and $z_i \sim \mathcal{N}(0, I_d)$ or $z_i \sim \mathcal{U}_d(0, 1)$
2. MC approximation of the objective

$$L(\theta, \phi) \approx \frac{1}{N} \sum_i \log D_\phi(x_i) + \log(1 - D_\phi(G_\theta(z_i))$$

3. gradient ascent on ϕ (K-steps)

$$\phi^{(t+1)} = \phi^{(t)} + \eta_\phi \frac{1}{N} \sum_i \frac{\partial_\phi D_\phi(x_i)}{D_\phi(x_i)} + \frac{\partial_\phi D_\phi(G_\theta(z_i))}{1 - D_\phi(G_\theta(z_i))}$$

4. gradient descent on θ

$$\theta^{(t+1)} = \theta^{(t)} + \eta_\theta \frac{1}{N} \sum_i \frac{\partial_x D_\phi(G_\theta(z_i))}{1 - D_\phi(G_\theta(z_i))} \cdot \partial_\theta G_\theta(z_i) \quad \text{flip to fix}$$

GAN / Details

1. convincing results on digits and faces dataset
2. show interpolation results as a proof that the network is not memorizing

Contents

GAN models

- GAN: Generative Adversarial Nets
- **InfoGAN: learning semantic features**
- CycleGAN: unpaired image-to-image translation
- f-GAN: learning neural samplers by div. min.

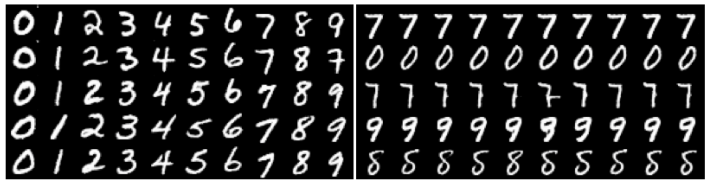
GAN models for Bayesian inference

- VAE: Neural variational Bayes
- ALI: Adversarially Learned Inference
- AVB: Adversarial Variational Bayes

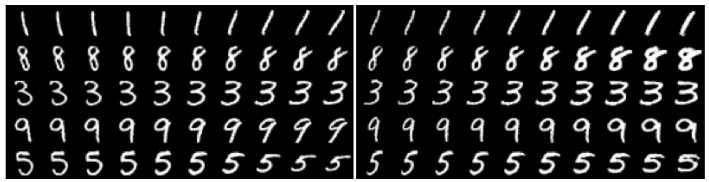
InfoGAN

InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets

Xi Chen[†‡], **Yan Duan**[†‡], **Rein Houthooft**[†‡], **John Schulman**[†‡], **Ilya Sutskever**[‡], **Pieter Abbeel**[†‡]
† UC Berkeley, Department of Electrical Engineering and Computer Sciences
‡ OpenAI



(a) Varying c_1 on InfoGAN (Digit type) (b) Varying c_1 on regular GAN (No clear meaning)



(c) Varying c_2 from -2 to 2 on InfoGAN (Rotation) (d) Varying c_3 from -2 to 2 on InfoGAN (Width)

A version of GAN that

1. maximises the mutual information between a small subset of the latent variables and the observation
2. disentangles writing styles from digit shapes on the MNIST dataset, pose from lighting of 3D rendered images

InfoGAN / Loss function

Add mutual entropy regularisation to the GAN objective

1. main idea: $z' = (z, c) \sim p(z)p(c), x' = G(z') = G_\theta(z, c)$ such that
   1. z comes from the usual base distribution
   2. "code distribution": c encodes semantic features by ME

$$I(c, x') = \int p(x'|c)p(x') \log \frac{p(x'|c)}{p(x')p(c)}$$

2. bounds for the intractable parts–use a KL bound

$$\begin{aligned} I(c, G_\theta(z, c)) &= H(c) + E_{x \sim G_\theta(z,c)}[E_{c' \sim p(c|x')}[\log p(c'|x')]] \\ &= H(c) + E_{x' \sim G_\theta(z,c)}[D[p(c|x') \| Q_\eta(c|x')]] \\ &\quad + E_{x' \sim G_\theta(z,c)}[E_{c' \sim p(c|x')}[\log Q_\eta(c'|x')]] \\ &\geq H(c) + E_{x' \sim G_\theta(z,c)}[E_{c' \sim p(c|x')}[\log Q_\eta(c'|x')]] \\ &\geq H(c) + E_{c \sim p(c), x' \sim G_\theta(z,c)}[\log Q_\eta(c'|x')]] \qquad (\equiv L_{MI}(\theta, \eta)) \end{aligned}$$

InfoGAN / optimisation

Augmented GAN objective

$$L_{iGAN}(\theta, \phi) = E_{x \sim \hat{p}(x)}[\log D_\phi(x)] + E_{(z,c) \sim p(z,c)}[\log(1 - D_\phi(G_\theta(z, c)))] - \lambda I(c, G_\theta(z, c))$$

Joint objective and optimisation

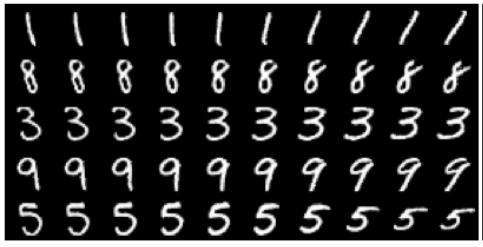
1. objective
$$L(\theta, \eta, \phi) = L_{GAN}(\phi, \theta) - \lambda \, L_{MI}(\theta, \eta)$$
2. optimisation
$$\min_{\theta, \eta} \max_{\phi} \left\{ L_{GAN}(\phi, \theta) - \lambda \, L_{MI}(\theta, \eta) \right\}$$
3. training tricks
    1. D_ϕ and Q_η share an image feature layer that only D_ϕ trains!
    2. $\lambda \approx 1$ larger learning rate for G_θ than for D_ϕ
    3. trying to keep the objective and regulariser on the same scale help is most cases

Results

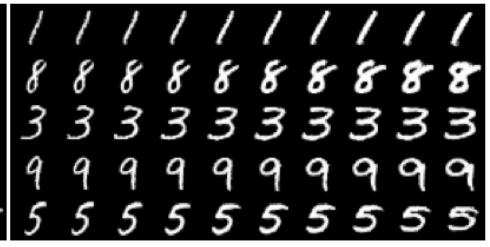
MNIST handwritten digits dataset

(a) Varying c_1 on InfoGAN (Digit type)

(b) Varying c_1 on regular GAN (No clear meaning)



(c) Varying c_2 from -2 to 2 on InfoGAN (Rotation)



(d) Varying c_3 from -2 to 2 on InfoGAN (Width)

Faces dataset

(a) Azimuth (pose)

(b) Elevation

Contents

GAN models

- GAN: Generative Adversarial Nets
- InfoGAN: learning semantic features
- **CycleGAN: unpaired image-to-image translation**
- f-GAN: learning neural samplers by div. min.

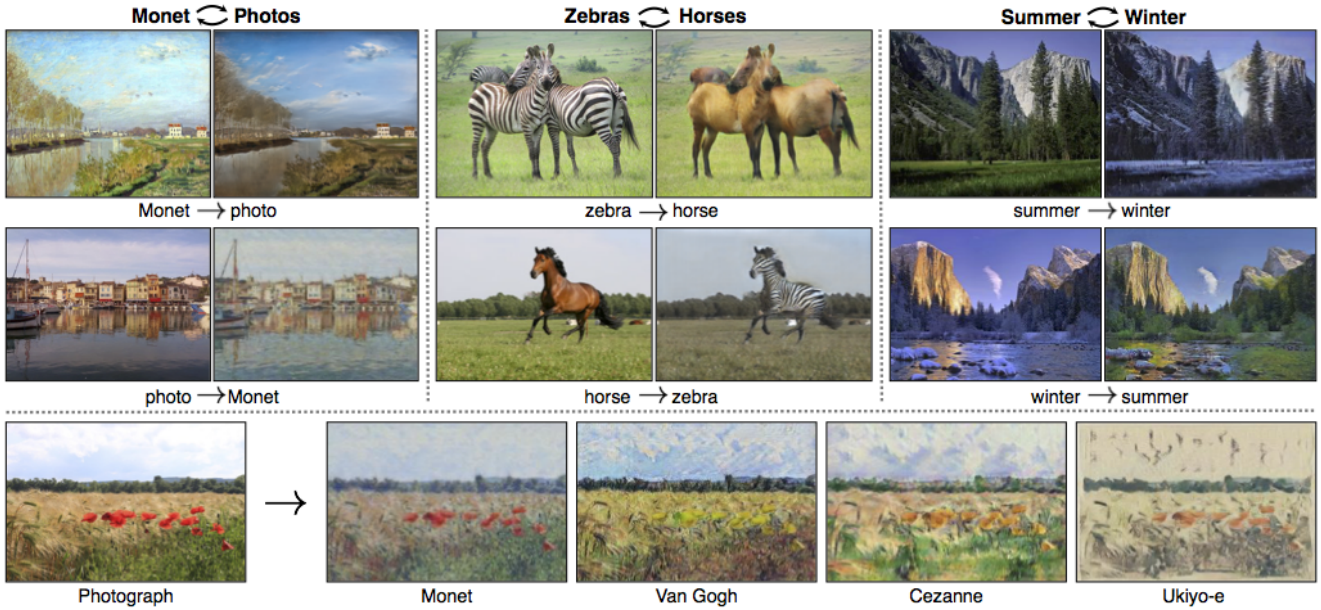
GAN models for Bayesian inference

- VAE: Neural variational Bayes
- ALI: Adversarially Learned Inference
- AVB: Adversarial Variational Bayes

CycleGAN

Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks

Jun-Yan Zhu* Taesung Park* Phillip Isola Alexei A. Efros
Berkeley AI Research (BAIR) laboratory, UC Berkeley

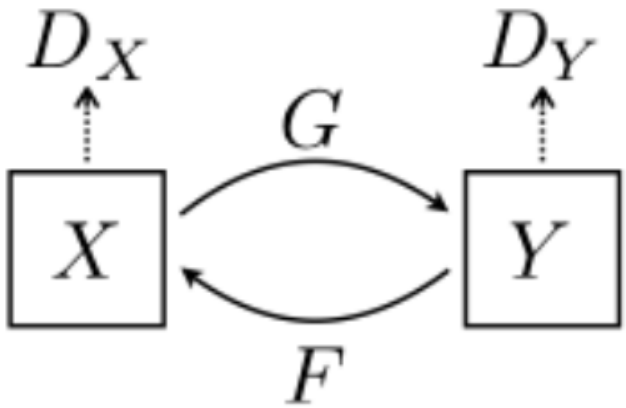


Problem statement

1. find a mapping $G : X \to Y$, such that the distribution of images $G(X)$ is indistinguishable from the distribution Y using an adversarial loss.
2. highly under-constrained mapping: we couple it with an inverse mapping $F : Y \to X$ enforce $F(G(X)) = X$ (and vice-versa)

CycleGAN / Loss functions I

Equivalent domains → symmetric GAN losses on empirical distributions / data domains



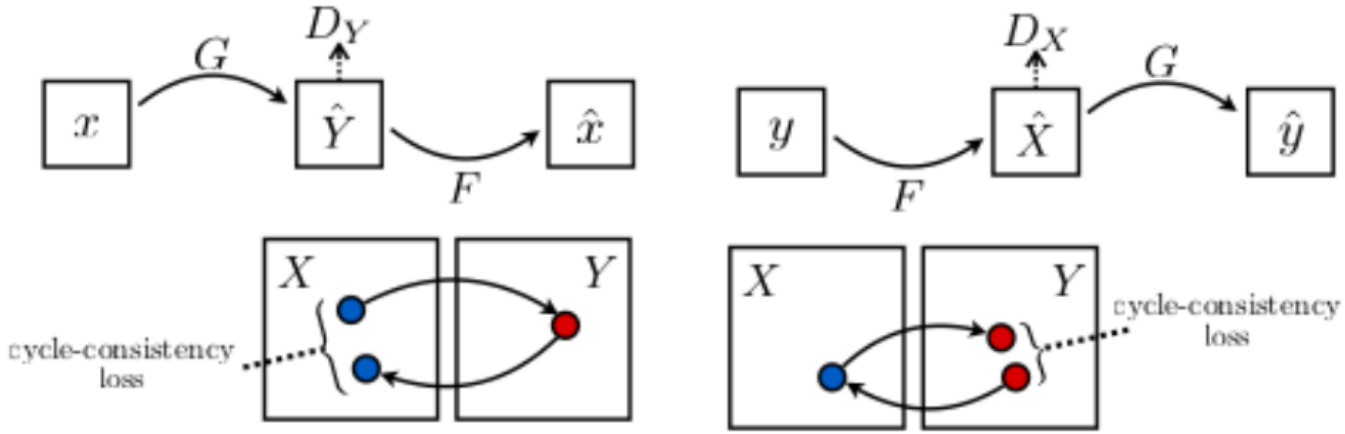
Generator space is a data domain space → a GAN in each domain

$$L_Y(G, D_Y, X, Y) = E_{y \sim \hat{p}_Y(y)}[\log D_Y(y)] + E_{x \sim \hat{p}_X(x)}[\log(1 - D_Y(G(x)))]$$
$$L_X(F, D_X, X, Y) = E_{x \sim \hat{p}_X(x)}[\log D_X(x)] + E_{y \sim \hat{p}_Y(y)}[\log(1 - D_X(F(y)))]$$

Results in

$$\min_{G,F} \max_{D_X, D_Y} L_Y(G, D_Y, X, Y) + L_X(F, D_X, X, Y)$$

CycleGAN / loss functions II



Cycle constraints

1. motivation
   - large complexity $\rightarrow$ network can map the same set of input images to any random permutation
   - adversarial losses alone cannot guarantee that the learned function can map an individual input x_i to a desired output y_i
2. formulation

$$L_{cycle}(G, F) = E_{x \sim \hat{p}_X(x)}[\| F(G(x)) - x \|_1] + E_{y \sim \hat{p}_Y(y)}[\| G(F(y)) - y \|_1]$$

CycleGAN / Loss function III

Joint loss and optimisation

- overall loss

$$L(G, F, D_X, D_Y) = L_Y(G, D_Y, X, Y) + L_X(F, D_X, X, Y) + \lambda\, L_{cycle}(G, F)$$

- total loss

$$\min_{G,F} \max_{D_X, D_Y} L(G, F, D_X, D_Y)$$

Training tricks

1. use 0/1 regression loss instead of the Bernoulli GAN

$$L_Y(G, D_Y, X, Y) = E_{y \sim \hat{p}_Y(y)}[(D_Y(y) - 1)^2] + E_{x \sim \hat{p}_X(x)}[D_Y(G(x))^2]$$
$$L_X(F, D_X, X, Y) = E_{x \sim \hat{p}_X(x)}[(D_X(x) - 1)^2] + E_{y \sim \hat{p}_Y(y)}[D_X(F(y))^2]$$

2. Practical tricks
   - update discriminators D_X and D_Y using a history of 50 generated images
   - fix $\lambda = 10$ for all
   - ADAM with decaying learning rate

CycleGAN / Experiments I

Cityscapes

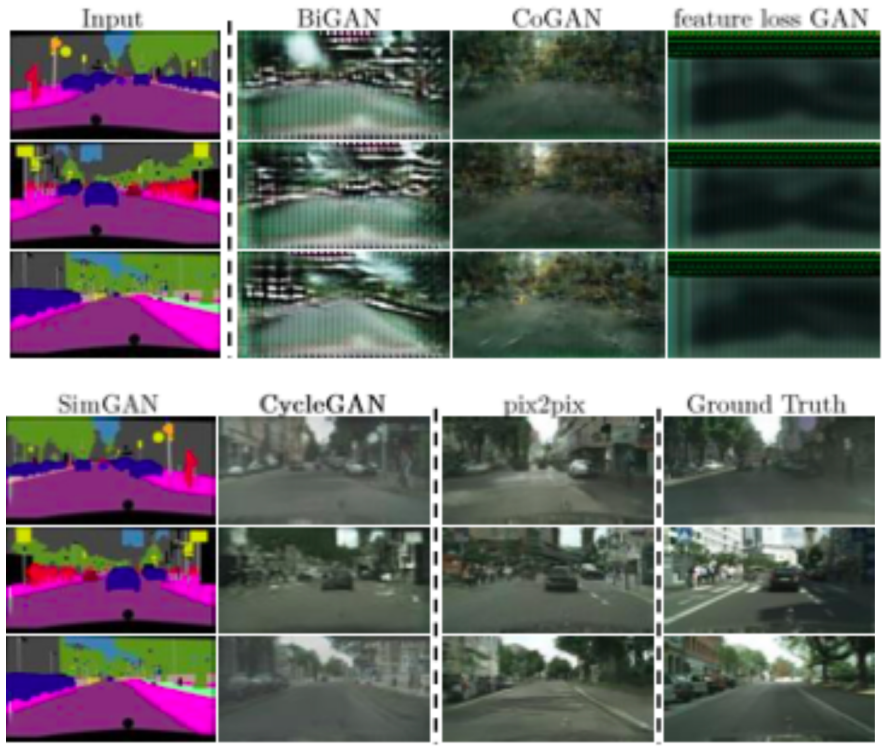
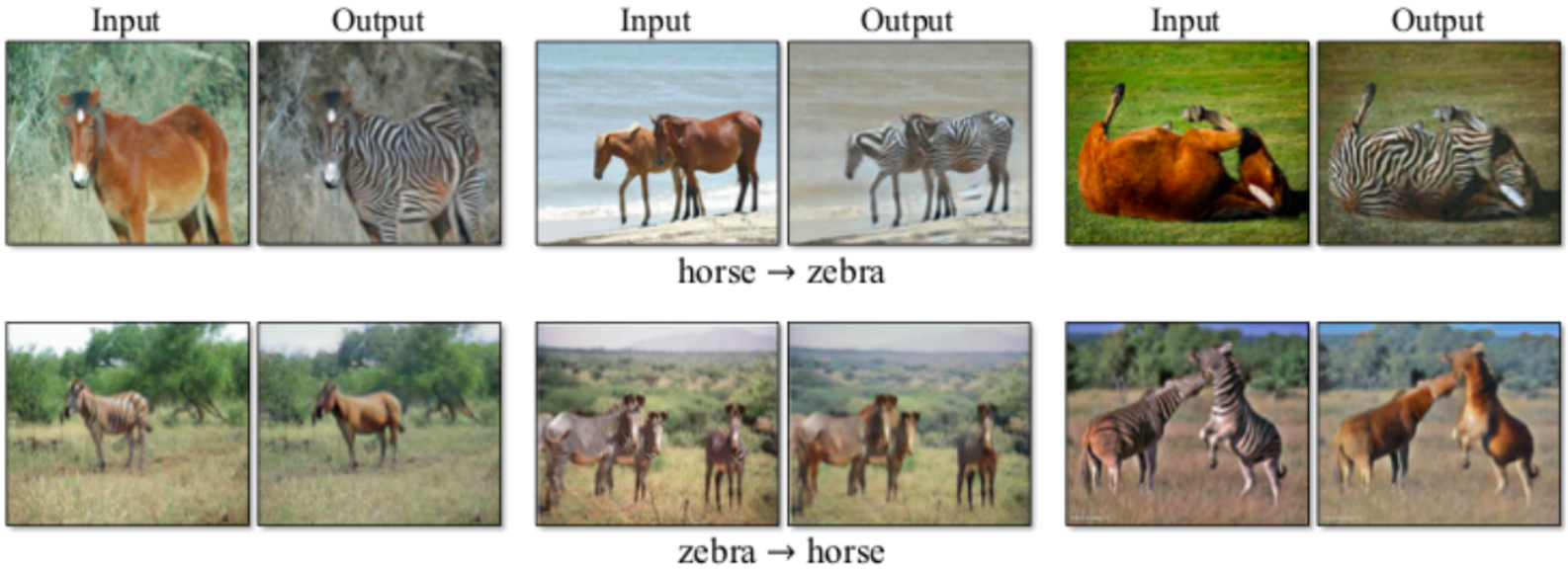


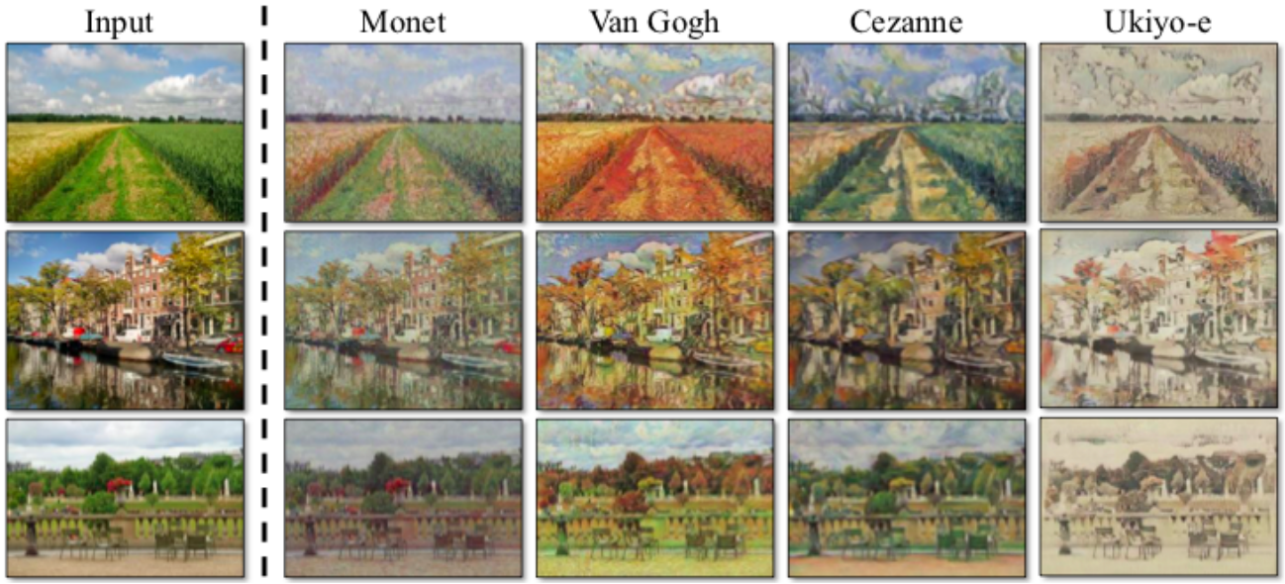
Figure: Different variants of the method for mapping labels $\rightarrow$ photos trained on cityscapes.

Evaluation metrics

- Amazon Mechanical Turk
- comparing overlaps to ground truth using standard semantic segmentation metrics
- inception score [explain idea only]

CycleGAN / Experiments II





Contents

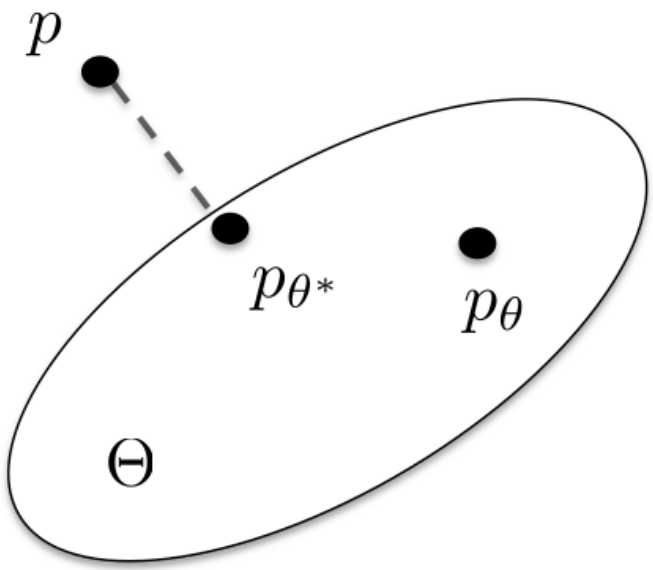
GAN models

- GAN: Generative Adversarial Nets
- InfoGAN: learning semantic features
- CycleGAN: unpaired image-to-image translation
- **f-GAN: learning neural samplers by div. min.**

GAN models for Bayesian inference

- VAE: Neural variational Bayes
- ALI: Adversarially Learned Inference
- AVB: Adversarial Variational Bayes

f-GAN / Approximating probability densities



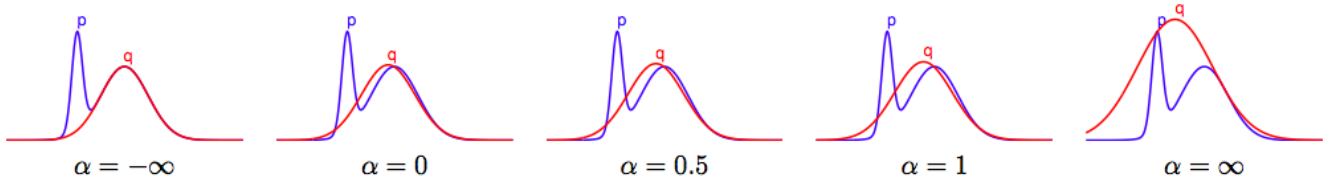
Parametric approximation of pdfs

- target distribution $p(x)$ given only through a finite set of samples

$$\hat{p}(x) = \frac{1}{n} \sum_i \delta(x - x_i), \quad x_i \sim p(x)$$

- p_θ limited to $\theta \in \Theta$
- what should the loss be?

f-GAN / Divergence measures



(Minka, 2015)

Comparing distributions by using divergence measures

- α divergence

$$\mathbb{D}_\alpha[p \parallel q] = \frac{1}{(1-\alpha)\alpha}\left(1 - \int dx\; p(x)^\alpha q(x)^{1-\alpha}\right)$$

- important special cases

$$\mathbb{D}_0[p \parallel q] = \int dx\; q(x) \log \frac{q(x)}{p(x)} \quad \text{and} \quad \mathbb{D}_1[p \parallel q] = \int dx\; p(x) \log \frac{p(x)}{q(x)}$$

- Jensen-Shannon (max of the GAN objective w.r.t. D_ϕ)

$$\mathbb{D}_{JS}[p \parallel q] = \mathbb{D}_0\left[p \parallel \frac{p+q}{2}\right] + \mathbb{D}_0\left[q \parallel \frac{p+q}{2}\right]$$

- others: total variation, etc.

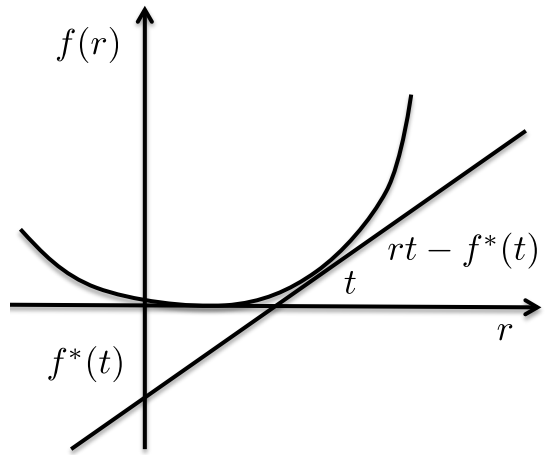
Sample based divergence minimisation (f-GAN)

Divergence measure (f convex, $f(1) = 0$)

$$\mathrm{D}_f\left[q_\phi||p\right] = \int d\boldsymbol{x}\, q_\phi(\boldsymbol{x}) f\Big(\frac{p(\boldsymbol{x})}{q_\phi(\boldsymbol{x})}\Big)$$

Lower bound [Nguyen et al.; 2010]

$$\begin{aligned}\mathrm{D}_f\left[q_\phi||p\right] &\geq \int d\boldsymbol{x}\, q_\phi(\boldsymbol{x})\Big[\frac{p(\boldsymbol{x})}{q_\phi(\boldsymbol{x})} t(\boldsymbol{x}) - f^*\big(t(\boldsymbol{x})\big)\Big] \\ &= \mathrm{E}_{\boldsymbol{x}\sim p}\left[t(\boldsymbol{x})\right] - \mathrm{E}_{\boldsymbol{x}'\sim q_\phi}\left[f^*\big(t(\boldsymbol{x}')\big)\right]\end{aligned}$$



$$f^*(t) = \sup_r\{rt - f(r)\}$$
$$f(r) = \sup_t\{rt - f^*(t)\}$$

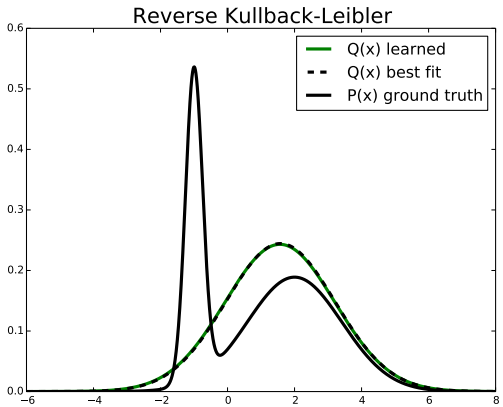
f-GAN objective

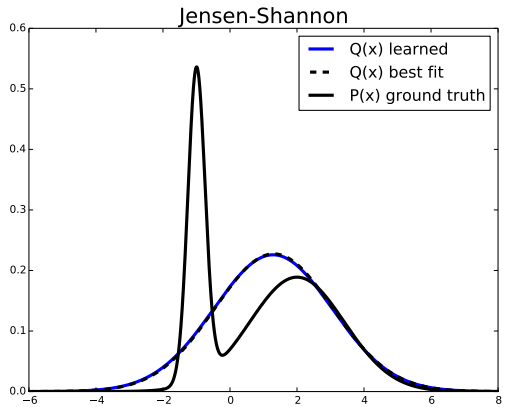
$$\min_\phi \max_\omega \; \mathrm{E}_{\boldsymbol{x}\sim p}\left[t_\omega(\boldsymbol{x})\right] - \mathrm{E}_{\boldsymbol{x}'\sim q_\phi}\left[f^*\big(t_\omega(\boldsymbol{x}')\big)\right]$$

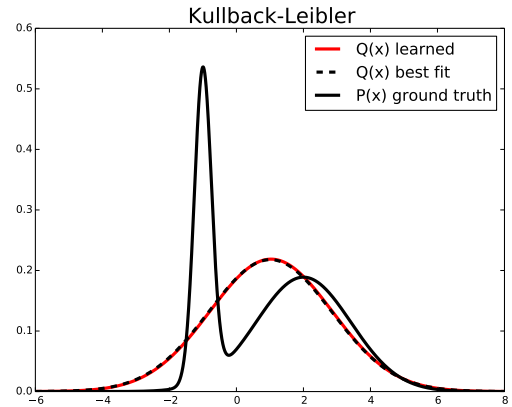
Training: stochastic gradient ascent-descent using mini-batches

f-divergences and conjugate duals

Name	$D_f(P\|Q)$	Generator $f(u)$	$t^*(x)$
Kullback-Leibler	$\int p(x) \log \frac{p(x)}{q(x)} \, \mathrm{d}x$	$r \log r$	$1 + \log \frac{p(x)}{q(x)}$
Reverse KL	$\int q(x) \log \frac{q(x)}{p(x)} \, \mathrm{d}x$	$-\log r$	$-\frac{q(x)}{p(x)}$
Jensen-Shannon	$\frac{1}{2} \int p(x) \log \frac{2p(x)}{p(x)+q(x)} + q(x) \log \frac{2q(x)}{p(x)+q(x)} \, \mathrm{d}x$	$-(r+1) \log \frac{1+r}{2} + r \log r$	$\log \frac{2p(x)}{p(x)+q(x)}$

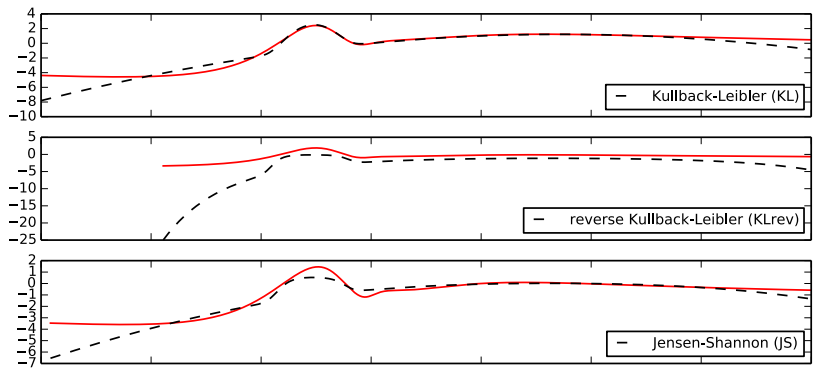






Optimal variational function

$$t^*(\boldsymbol{x}) = f'\left(\frac{p(\boldsymbol{x})}{q_\phi(\boldsymbol{x})}\right)$$



f-GAN practical considerations

Name	Output activation h_f	dom_{f^*}	Conjugate $f^*(t)$	$f'(1)$
Kullback-Leibler (KL)	v	$\mathbb{R}$	$\exp(t-1)$	1
Reverse KL	$-\exp(-v)$	$\mathbb{R}_-$	$-1-\log(-t)$	-1
Jensen-Shannon	$\log(2)-\log(1+\exp(-v))$	$t<\log(2)$	$-\log(2-\exp(t))$	0

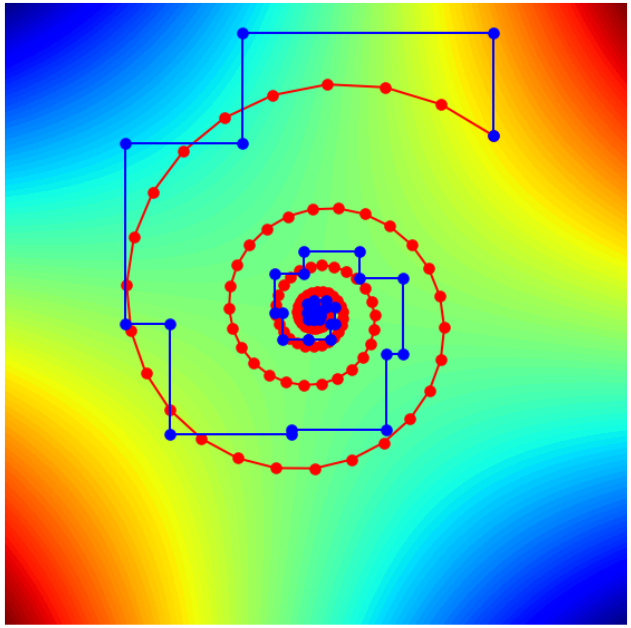
Use $t_\omega(\boldsymbol{x}') = h_f(w_\omega(\boldsymbol{x}'))$ to map into f^* input domain

$$L(\theta,\omega) = \mathrm{E}_{\boldsymbol{x}\sim p}\left[h_f(v_\omega(\boldsymbol{x}))\right] - \mathrm{E}_{\boldsymbol{z}\sim q_0}\left[f^*\left(h_f(v_\omega(g_\theta(\boldsymbol{z}))\right)\right]$$

Synchronous gradient ascent-descent

$$\theta_{t+1} = \theta_t - \boldsymbol{\Gamma}_{\theta,t}\,\partial_\theta \tilde{L}(\theta_t,\omega_t)$$

$$\omega_{t+1} = \omega_t + \boldsymbol{\Gamma}_{\omega,t}\,\partial_\omega \tilde{L}(\theta_t,\omega_t)$$

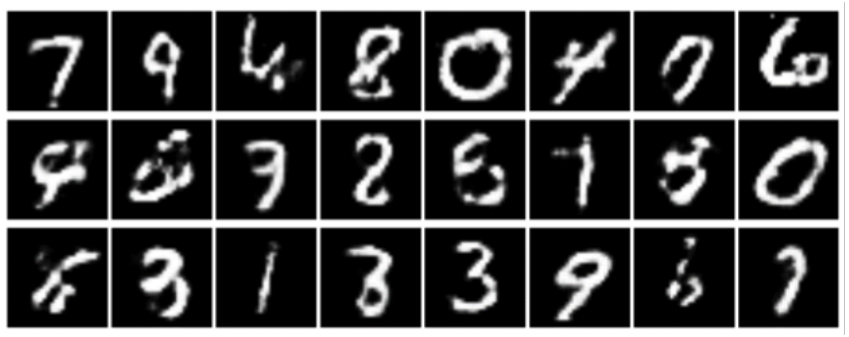


MNIST

- Model from the original GAN work [Goodfellow et al.; 2014]: simple 2/3 layer NN
- Evaluation using KDE log-likelihoods
  - build KDE on model samples, evaluate on real data
  - known shortcomings, but popular in other works (old approach)

Training divergence	KDE $\langle LL \rangle$ (nats)	$\pm$ SEM
Kullback-Leibler	416	5.62
Reverse Kullback-Leibler	319	8.36
Pearson χ^2	429	5.53
Neyman χ^2	300	8.33
Squared Hellinger	-708	18.1
Jeffrey	-2101	29.9
Jensen-Shannon	367	8.19
GAN	305	8.97
Variational Autoencoder	445	5.36
KDE MNIST train (60k)	502	5.99

Kullback-Leibler



Kullback-Leibler (reverse)

Hellinger

Contents

GAN models

- GAN: Generative Adversarial Nets
- InfoGAN: learning semantic features
- CycleGAN: unpaired image-to-image translation
- f-GAN: learning neural samplers by div. min.

GAN models for Bayesian inference

- **VAE: Neural variational Bayes**
- ALI: Adversarially Learned Inference
- AVB: Adversarial Variational Bayes

VAE / Unsupervised Bayesian learning through amortisation

Auto-Encoding Variational Bayes

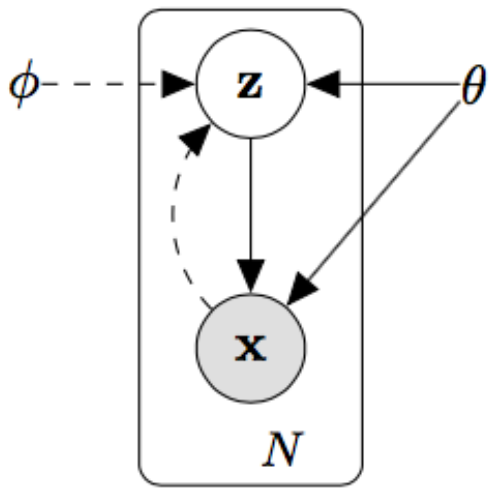
Diederik P. Kingma
Machine Learning Group
Universiteit van Amsterdam
dpkingma@gmail.com

Max Welling
Machine Learning Group
Universiteit van Amsterdam
welling.max@gmail.com

- Bayesian unsupervised learning

$$p_\theta(x) = \int dz\, p_\theta(x|z)\, p_0(z)$$

- variational Bayes approximations for likelihood optimisation and posterior inference
- unifies VB with auto-encoders to simplify inference $\rightarrow$ scaling it to large data sets



VAE / bounding the likelihood function

Variational bound

$$\begin{aligned}
\log p_\theta(x_i) &= \log \int dz_i \; p(x_i|z_i)\, p_0(z_i) \\
&= \log \int dz_i \; p(x_i|z_i) \; \frac{p_0(z_i)}{q_i(z_i)} q_i(z_i) \\
&\geq \int dz_i \; q_i(z_i) \; \log \frac{p(x_i|z_i) p_0(z_i)}{q_i(z_i)} \\
&= \mathbb{E}_{q_i}[\log p_\theta(x_i|z_i)] - \mathbb{D}[q_i(z_i) \parallel p_0(z_i)] \qquad \equiv L(\theta, q_i : x_i)
\end{aligned}$$

Properties

- for each x_i we have

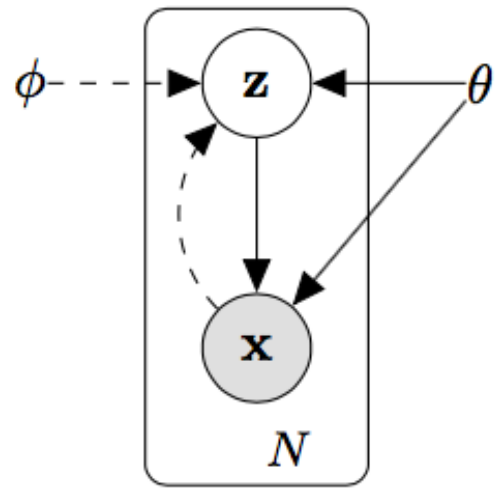
$$q_i^*(z_i) = \frac{p(x_i|z_i)\, p(z_i)}{p_\theta(x_i)} \quad \rightarrow \quad q_i^*(z_i) = p(z_i|x_i)$$

- tight bound

$$L(\theta, q_i^*; x_i) = \log p_\theta(x_i)$$

- optimisation gap

$$\log p_\theta(x_i) - L(\theta, q_i; x_i) = D[q_i(z_i) \parallel p(z_i|x_i)]$$



VAE / large scale data

Divergence bound

$$\log p_\theta(x_i) \geq L(\theta, q_i; x_i) \quad \left(= \mathbb{E}_{q_i}[\log p_\theta(x_i|z_i)] - \mathbb{D}[q_i(z_i) \parallel p_0(z_i)] \right)$$

VAE approach

- overall likelihood bound

$$\sum_i \log p_\theta(x_i) \geq \sum_i L(\theta, q_i; x_i)$$

- tight bound (many independent optimisation problems)

$$\max_\theta \sum_i \log p_\theta(x_i) \geq \max_\theta \sum_i \max_{q_i} L(\theta, q_i; x_i)$$

- amortisation and parameter sharing

$$q_i(z_i) \equiv q_\phi(z_i|x_i) \quad \text{e.g.} \quad q_\phi(z_i|x_i) = \mathcal{N}(z_i; \mu_\phi(x_i), \sigma_\phi(x_i)^2)$$

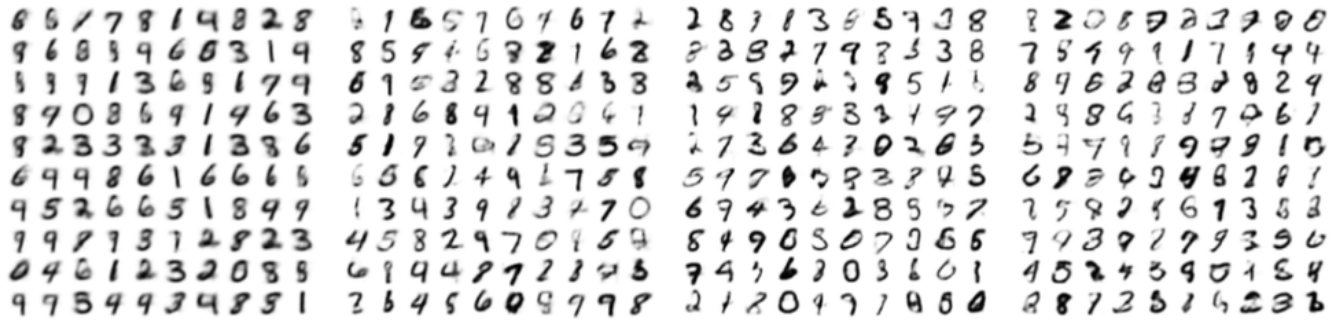
- SGD for

$$\max_\theta \max_\phi L(\theta, q_\phi(z_i; x_i); x_i)$$

- neural network parameterisation of Gaussians / reparameterisation

$$[\, \mu_\phi(x_i), \log \sigma_\phi(x_i)) \,] = \mathsf{NN}_\phi(x_i) \quad \text{and} \quad z_i^{(s)} = \mu_\phi(x_i) + \sigma_\phi(x_i)\, \epsilon_i^{(s)}$$

VAE / Experiments I



(a) 2-D latent space (b) 5-D latent space (c) 10-D latent space (d) 20-D latent space

Unsupervised learning of digits

- details of the model

$$p_0(z) = \mathcal{N}(z; 0, I_d), \quad \text{and} \quad p_\theta(x|z) = \mathcal{N}(x|\mu_\theta(z), \sigma_\phi(z)^2)$$

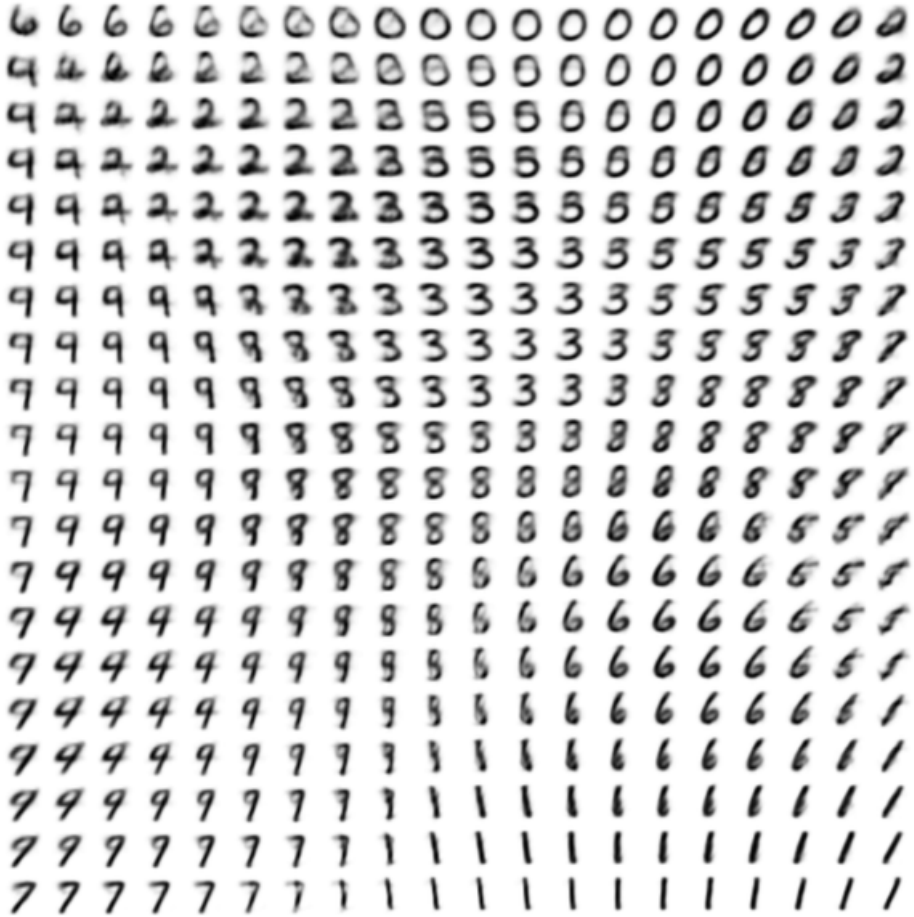
- details of the approximation

$$q_\phi(z_i|x_i) = \mathcal{N}(z_i; \mu_\phi(x_i), \sigma_\phi(x_i)^2) \quad \text{s.t.} \quad [\, \mu_\phi(x_i), \log \sigma_\phi(x_i)) \,] = \mathsf{NN}_\phi(x_i)$$

- training: SDG steps
  - sample $x_i \sim \hat{p}(x)$, sample $\epsilon_i^{(k)} \sim \mathcal{N}(0, I_d)$, $z_i^{(k)} = \mu_\phi(x_i) + \sigma_\phi(x_i)\, \epsilon_i^{(k)}$
  - approx of MC loss

$$\tilde{L}(\theta, \phi) = \frac{1}{N_x} \sum_i \Big\{ \frac{1}{N_z} \sum_j \log p_\theta(x_i|z_i) - \underbrace{\mathbb{D}[q_\phi(z_i|x_i) \,\|\, \mathcal{N}(z_i; 0, I_d)]}_{\text{analytic}} \Big\}$$

VAE / Experiments II



Structure of the latent space z

- fit a 2d latent space for z
- map 2d uniform grid $z_{i,j} = (i.j)/N, \quad i, j = 1, \ldots, N$ through normal inverse cdf
- sample / compute mean of $p_\theta(x|z_{i,j})$

Contents

GAN models

- GAN: Generative Adversarial Nets
- InfoGAN: learning semantic features
- CycleGAN: unpaired image-to-image translation
- f-GAN: learning neural samplers by div. min.

GAN models for Bayesian inference

- VAE: Neural variational Bayes
- **ALI: Adversarially Learned Inference**
- AVB: Adversarial Variational Bayes

Adversarially Learned Inference

ADVERSARIALLY LEARNED INFERENCE

Vincent Dumoulin[1], **Ishmael Belghazi**[1], **Ben Poole**[2]
Olivier Mastropietro[1], **Alex Lamb**[1], **Martin Arjovsky**[3]
Aaron Courville[1†]
[1] MILA, Université de Montréal, firstname.lastname@umontreal.ca.
[2] Neural Dynamics and Computation Lab, Stanford, poole@cs.stanford.edu.
[3] New York University, martinarjovsky@gmail.com.
[†]CIFAR Fellow.

ADVERSARIAL FEATURE LEARNING

Jeff Donahue
jdonahue@cs.berkeley.edu
Computer Science Division
University of California, Berkeley

Philipp Krähenbühl
philkr@utexas.edu
Department of Computer Science
University of Texas, Austin

Trevor Darrell
trevor@eecs.berkeley.edu
Computer Science Division
University of California, Berkeley

A version of GAN that

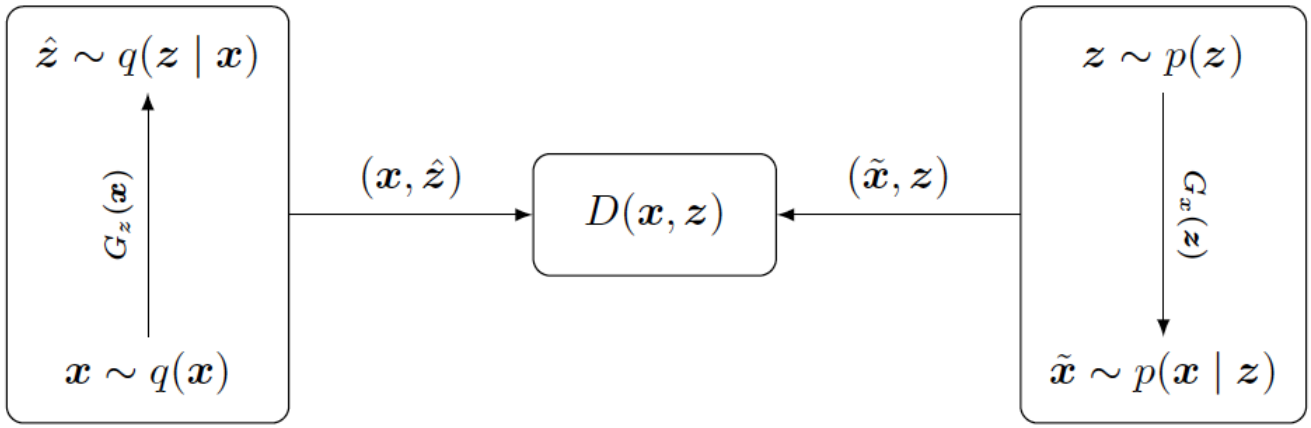
1. does Bayesian inference by directly matching densities
2. main idea: write the joint $p(x, z)$ as

$$p(x, z) = p_\theta(x|z) \times p_0(z) = q_\phi(z|x) \times \hat{p}(x)$$

2. optimise θ and ϕ such that

$$p_\theta(x|z) \times p_0(z) \approx q_\phi(z|x) \times \hat{p}(x)$$

Adversarially Learned Inference



Loss function and algorithm

1. discriminator $D_\omega(x, z)$ and loss

$$L(\{\theta, \phi\}, \omega) = \mathbb{E}_{(z', x) \sim q_\phi(z|x)\hat{p}(x)} \left[\log D_\omega(z', x) \right] + \mathbb{E}_{(x', z) \sim p_\theta(z|x)p_0(z)} \left[\log(1 - D_\omega(z, x')) \right]$$

2. Algorithm
    1. sample $z_i \sim p_0(z)$ and $x_i' \sim p_\theta(x \mid z_i)$ for $s \in \{1, \ldots, N\}$
    2. sample $x_i \sim \hat{p}(x)$ and $z_i' \sim q_\phi(z \mid x_i)$ for $s \in \{1, \ldots, N\}$
    3. do gradient ascent-descent on the MC estimate of $L(\{\theta, \phi\}, \omega)$

Adversarially Learned Inference / Results

SVHN dataset

original

learned

Interpolation in latent space of CelebA

;

Contents

GAN models

- GAN: Generative Adversarial Nets
- InfoGAN: learning semantic features
- CycleGAN: unpaired image-to-image translation
- f-GAN: learning neural samplers by div. min.

GAN models for Bayesian inference

- VAE: Neural variational Bayes
- ALI: Adversarially Learned Inference
- **AVB: Adversarial Variational Bayes**

Adversarial Variational Bayes

Adversarial Variational Bayes:
Unifying Variational Autoencoders and Generative Adversarial Networks

Lars Mescheder [1] **Sebastian Nowozin** [2] **Andreas Geiger** [1 3]

Bayesian inference by density ratio matching

1. approximate the KL divergence through density ratio matching

$$\log q_\phi(z|x) - \log p_0(z) = \operatorname*{argmax}_{r_\omega} \Big\{ \mathbb{E}_{(z',x)\sim q(z'|x)\hat{p}(x)} \big[\log \sigma(r_\omega(z',x)) \big] + \mathbb{E}_{(z,x')\sim p_0(z)\hat{p}(x)} \big[\log(1-\sigma(r_\omega(z,x'))) \big] \Big\}$$

2. do max. likelihood by

$$\max_\theta \max_\phi \mathbb{E}_{(z',x)\sim q(z|x)\hat{p}(x)} \big[\log p_\theta(x|z') - r^*_\omega(z',x) \big]$$

AVB / details

L.d.r. loss

$$L(r_\omega) = \mathbb{E}_{(z,x)\sim q(z|x)\hat{p}(x)}\left[\log \sigma(r_\omega(z,x))\right] + \mathbb{E}_{(z,x)\sim p_0(z)\hat{p}(x)}\left[\log(1-\sigma(r_\omega(z,x)))\right]$$

Variation w.r.t. r_ω

$$\delta_{r_\omega} L(r_\omega) = q(z|x)\,\hat{p}(x)\,(1-\sigma(r_\omega(z,x))) - p_0(z)\,\hat{p}(x)\,\sigma(r_\omega(z,x))$$

Optimaliity condition

$$\delta_{r_\omega(z,x_i)} L(r^*_\omega) = 0 \quad \rightarrow \quad r^*_\omega(z,x_i) = \log q_\phi(z|x_i) - \log p_0(z)$$

KL divergence estimate

$$\begin{aligned}\sum_i \mathrm{D}[q_\phi(z|x) \parallel p_0(z)] &= \mathbb{E}_{(z,x)\sim q(z|x)\hat{p}(x)}[\log q_\phi(z|x) - \log p_0(z)] \\ &= \mathbb{E}_{(z,x)\sim q(z|x)\hat{p}(x)}[r^*_\omega(z,x)]\end{aligned}$$

AVB / step by step

Define posterior approximation $q_\phi(z \mid x)$ as the neural sampler

$$z = z_\phi(x, \epsilon) \quad \text{were} \quad \epsilon \sim \mathcal{N}(0, I_d)$$

Algorithmic details

1. sample $\epsilon_i \sim \mathcal{N}(0, I_d)$, $x_i \sim \hat{p}(x)$, $z_i \sim p_0(z)$
2. do gradient update on the losses
   - 2.1 update θ using

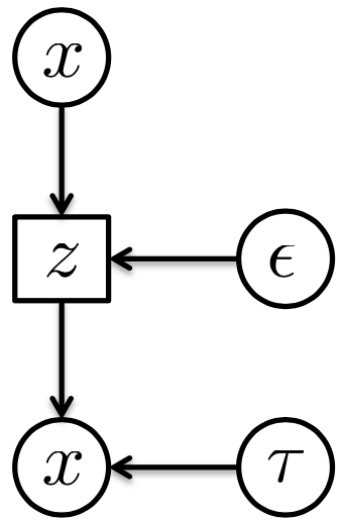
$$L(\theta) = \sum_i \log p(x_i \mid z_\phi(x_i, \epsilon_i))$$

   - 2.2 update ϕ using

$$L(\phi) = \sum_i \log p(x_i \mid z_\phi(x_i, \epsilon_i)) - r_\omega(z_\phi(x_i, \epsilon_i), x_i)$$

   - 2.3 update ω using

$$L(\omega) = \sum_i \log \sigma(r_\omega(z_\phi(x_i, \epsilon_i), x_i)) + \log\left(1 - \sigma(r_\omega(x_i, z_i))\right)$$



AVB / Experiments / MNIST

	$\log p(x) \geq$	$\log p(x) \approx$	
AVB (8-dim)	$(\approx -83.6 \pm 0.4)$	-91.2 ± 0.6	
AVB + AC (8-dim)	$\approx -96.3 \pm 0.4$	-89.6 ± 0.6	
AVB + AC (32-dim)	$\approx -79.5 \pm 0.3$	-80.2 ± 0.4	
VAE (8-dim)	-98.1 ± 0.5	-90.9 ± 0.6	
VAE (32-dim)	-87.2 ± 0.3	-81.9 ± 0.4	
VAE + NF (T=80)	-85.1	$-$	(Rezende & Mohamed, 2015)
VAE + HVI (T=16)	-88.3	-85.5	(Salimans et al., 2015)
convVAE + HVI (T=16)	-84.1	-81.9	(Salimans et al., 2015)
VAE + VGP (2hl)	-81.3	$-$	(Tran et al., 2015)
DRAW + VGP	-79.9	$-$	(Tran et al., 2015)
VAE + IAF	-80.8	-79.1	(Kingma et al., 2016)
Auxiliary VAE (L=2)	-83.0	$-$	(Maaløe et al., 2016)

AVB / Experiments / Faces

original (right) and generated (left)

interpolation results

What have we learned?

GAN models

- GAN: Generative Adversarial Nets
- InfoGAN: learning semantic features
- CycleGAN: unpaired image-to-image translation
- f-GAN: learning neural samplers by div. min.

GAN models for Bayesian inference

- VAE: Neural variational Bayes
- ALI: Adversarially Learned Inference
- AVB: Adversarial Variational Bayes