Deep Learning (BEV033DLE)
Lecture 12 Generative models

Czech Technical University in Prague

- Generative models in machine learning
- Generative adversarial networks (GAN)
- Variational autoencoders (VAE)

Generative models in machine learning

Machine learning Observable variables $x \in \mathcal{X}$, object state $y \in \mathcal{Y}$.

- a generative model is a statistical model of the joint distribution $p(x, y)$,
- a discriminative model is a model of the conditional distribution $p(y|x)$ or, even a predictor $h\colon \mathcal{X} \to \mathcal{Y}$ only.
- generative learning: given a class of distributions $p_\theta(x, y) \in \mathcal{P}(\Theta)$ and training data $\mathcal{T}^m$, estimate the unknown parameters θ

  a) supervised learning: $\mathcal{T}^m = \{(x_j, y_j) \mid j = 1, \ldots, m\}$

  b) unsupervised learning: - $\mathcal{T}^m = \{x_j \mid j = 1, \ldots, m\}$

A generative model can be used to generate random instances of pairs (x, y) or an observation x conditioned on the state y.

It is usually assumed, that the model is given in a form that allows to compute $p(x, y)$ for a given pair (x, y) by an algorithm or a formula.

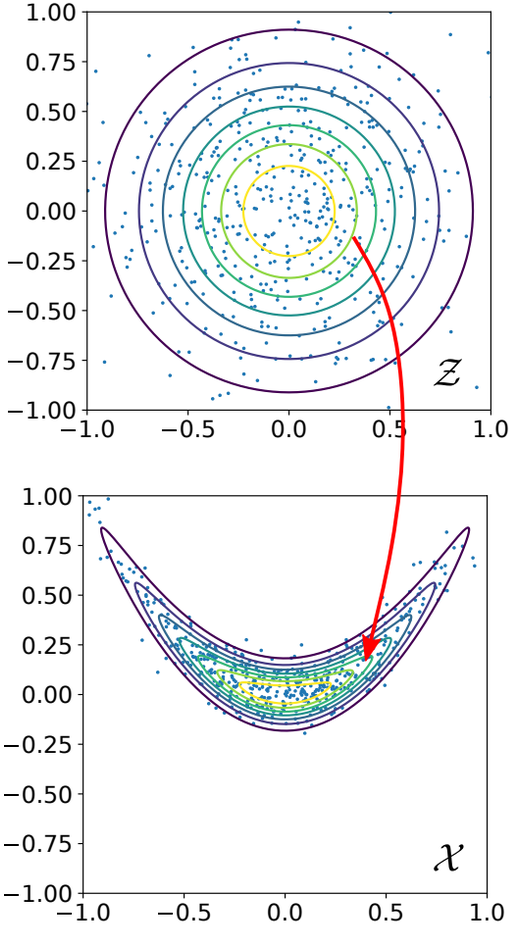
Generative models in machine learning

Deep generative models

Given training data $\mathcal{T}^m = \{x_j \mid j = 1, \ldots, m\}$ drawn i.i.d. from an unknown distribution $p_*(x)$, the goal is to learn a DNN model that allows to generate random instances of x similar to $x \sim p_*(x)$.

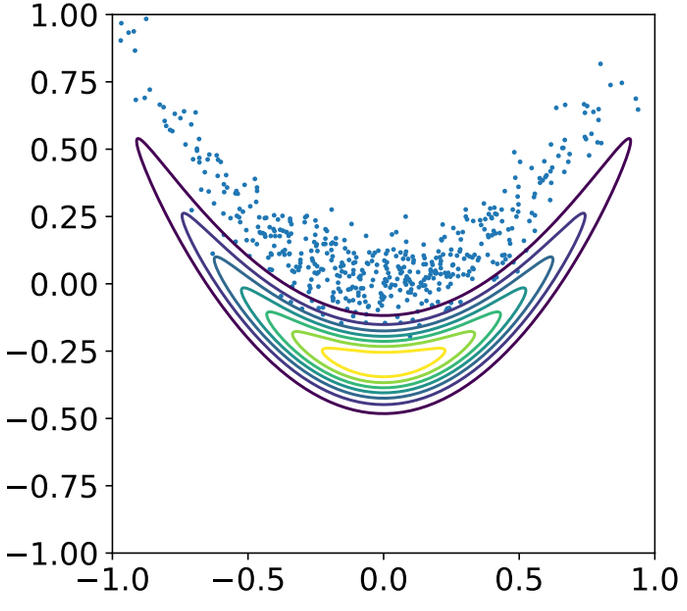
Approach:

- fix a latent noise space $\mathcal{Z}$ and a distribution $p_0(z)$ on it,
- design a parametrised mapping (neural network) g_w from the noise space $\mathcal{Z}$ to the feature space $\mathcal{X}$,
- given a training set $\mathcal{T}^m = \{x_j \mid j = 1, \ldots, m\}$ learn the parameters w so that the distribution $p_w(x)$ defined by $x = f_w(z)$ and $z \sim p_0(z)$ "reproduces" the data distribution.



Generative models in machine learning

Q: How to measure the distance (divergence) between a discrete distribution $x \sim \mathcal{T}^m$ and a continuous or discrete distribution $p_w(x)$ defined by $x = f_w(z)$ and $z \sim p_0(z)$?



- data log-likelihood requires to compute $\log p_w(x_j)$, $x_j \in \mathcal{T}^m$, but we can not compute $p_w(x)$ since we have no inverse mapping from $\mathcal{X}$ to $\mathcal{Z}$.
- other measures?

Many distances for distributions are problematic when

- one distribution is discrete and the other is absolutely continuous (has a density),
- the distributions have disjoint support.

Generative adversarial networks

(Goodfellow et al., 2014):

Measure the distance implicitly in terms of an *adversarial* (network) that aims at learning to distinguish patterns from the training set $\mathcal{T}^m$ and patterns sampled from the distribution $p_w(x)$. The model consists of a pair of networks

- a generator network $g_w(z)$ with parameters w and a distribution $p_0(z)$, which is easy to sample from.
- a binary adversarial classifier network $d_v(x)$ with parameters v that discriminates natural patterns $x \sim \mathcal{T}^m$ from patterns generated by $p_w(x)$. Its output encodes $p(y = 1 \mid x) = d_v(x)$, i.e. the probability of the class "natural".

Define the loss for the the pair of networks as

$$L(w, v) = \mathbb{E}_{x \sim \mathcal{T}^m}\Big[\log d_v(x)\Big] + \mathbb{E}_{z \sim p(z)}\Big[\log\big(1 - d_v \circ g_w(z)\big)\Big]$$

and solve the minimax optimisation task

$$\min_w \max_v L(v, w).$$

Generative adversarial networks

Let us analyse this saddle point task under the assumptions

- infinite training set $\mathcal{T}^m$, $m \to \infty$, denote by $p_*(x)$
- the generator model has infinite(!) capacity

The maximum w.r.t. the discriminator is achieved at $d^*(x) = \frac{p_*(x)}{p_*(x)+p_g(x)}$ because

$$L(g,d) = \int p_*(x) \log d(x)\, dx + \int p(z) \log\bigl(1 - d \circ g(z)\bigr)\, dz =$$

$$\int p_*(x) \log\bigl(d(x)\bigr) + p_g(x) \log\bigl(1 - d(x)\bigr)\, dx$$

Then the minimum w.r.t. the generator is achieved at $p_g = p_*$ because

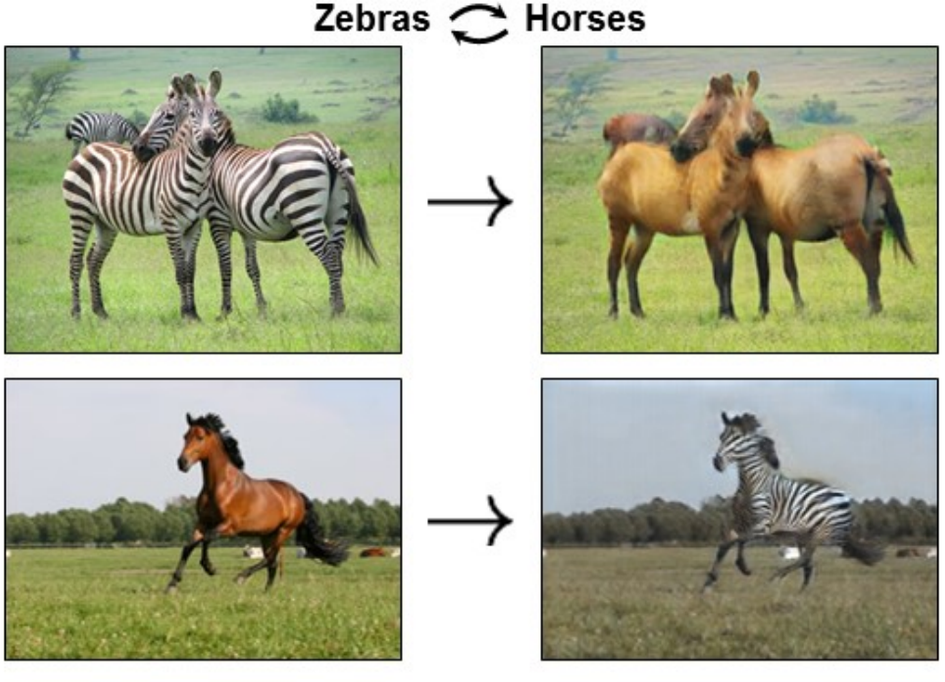
$$C(g) = \max_d L(g,d) = \mathbb{E}_{x\sim p_*}\left[\log \frac{p_*(x)}{p_*(x)+p_g(x)}\right] + \mathbb{E}_{x\sim p_g}\left[\log \frac{p_g(x)}{p_*(x)+p_g(x)}\right]$$

Generative adversarial networks

Quite impressive results achieved by deep convolutional GANs, but there are technical problems

- what happens with $\log\big(1 - d(g(z))\big)$ early in learning, when g is poor?
- mode collapse: what happens if g maps all $z \in \mathcal{Z}$ on one x?
- searching saddle points of functions is complicated

Learning GANs is difficult and sensitive to hyper-parameter and architecture setup.



(Zhu et al., 2017) Cycle GANs

Wasserstein GANs (for additional reading)

Wasserstein GANs: (Arjovsky et al., 2017)

Recall: $\mathbb{P}_*$ is the fixed data distribution on $\mathcal{X}$ given by $\mathcal{T}^m$, Z is a random variable on $\mathcal{Z}$ and $g_w \colon \mathcal{Z} \to \mathcal{X}$ is a parametrised family of mappings.

Denoting the distribution of $g_w(Z)$ by $\mathbb{P}_w$, we want a distribution distance $D(\mathbb{P}_*, \mathbb{P}_w)$ that behaves nicely as a function of w.

Definition 1. The Wasserstein distance of distributions $\mathbb{P}_1$ and $\mathbb{P}_2$ is defined by

$$DW(\mathbb{P}_1, \mathbb{P}_2) = \inf_{\gamma \in \Pi(\mathbb{P}_1, \mathbb{P}_2)} \mathbb{E}_{(x,y)\sim\gamma} \|x - y\|,$$

where $\Pi(\mathbb{P}_1, \mathbb{P}_2)$ is the set of all joint distributions $\gamma(x, y)$ with marginals $\mathbb{P}_1$ and $\mathbb{P}_2$

Remarks:

- This distance is sometimes called earth-mover distance.
- $\gamma(x, y)$ is interpreted as a transport plan and W is the cost of the optimal plan.
- the dual task reads

$$D_W(\mathbb{P}_*, \mathbb{P}_w) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x\sim\mathbb{P}_*}\big[f(x)\big] - \mathbb{E}_{x\sim\mathbb{P}_w}\big[f(x)\big],$$

  where the supremum is over all 1-Lipschitz functions $f \colon \mathcal{X} \to \mathbb{R}$ (critic network).

Variational autoencoders

As before $\mathcal{X}$ denotes the feature space and $\mathcal{Z}$ denotes a noise space. We may assume e.g. $\mathcal{Z} = \mathbb{R}^n$ and fix a simple distribution $Z \sim \mathcal{N}(0, \mathbb{I})$ on it.

- Consider a deterministic (convolutional) neural network, whose outputs are interpreted as parameters of a distribution on $\mathcal{X}$. E.g. $X \mid Z \sim \mathcal{N}(\mu_w(z), \operatorname{diag}(\sigma_w(z)))$.
- This defines a parametrised family of conditional distributions $p_w(x \mid z)$.
- Given a sample $\mathcal{T}^m = \{x_j \in \mathcal{X} \mid j = 1, \ldots, m\}$, we want to learn the parametrised conditional distribution $p_w(x \mid z)$ such that the likelihood of $\mathcal{T}^m$ w.r.t. to the joint model $p_w(x, z)$ is maximised.

The task reads as

$$\frac{1}{m} \sum_{j=1}^{m} \log p_w(x_j) = \frac{1}{m} \sum_{j=1}^{m} \log \int p_w(x_j, z)\, dz \to \max_w \tag{1}$$

Variational autoencoders

Let us now consider a summand in this objective

$$\log p_w(x) = \log \int p_w(x, z)\, dz$$

We can not compute this integral $\Rightarrow$ lower bound it in the same way as in the **expectation maximisation algorithm**

$$\log \int p_w(x, z)\, dz = \log \int \frac{q_v(z \mid x)}{q_v(z \mid x)} p_w(x, z)\, dz \geqslant \int q_v(z \mid x) \log \frac{p_w(x, z)}{q_v(z \mid x)}\, dz$$

This bound is tight if $q_v(z \mid x) = p_w(z \mid x)$, which is however hard to compute.

Main assumption of VAEs: the distributions $q_v(z \mid x)$ are modelled by a single neural network in the way discussed above for $p(x|z)$.

Notice that the lower bound may be not tight, if the requirement

$$\forall w\, \exists v(w) \text{ s.t. } q_{v(w)}(z \mid x) = p_w(z \mid x)$$

is not met.

Variational Autoencoders

With the assumptions made, the lower bound of the learning objective (1) reads

$$\frac{1}{m}\sum_{j=1}^{m}\Big[\int q_v(z \mid x^j)\log p_w(x^j \mid z) - D_{KL}(q_v(z \mid x^j) \parallel p(z))\Big] \to \max_{w,v}$$

- The KL-divergence can be computed in closed form if $q_v(z \mid x)$ is modelled as multivariate Gaussian distribution,
- the expectation $\mathbb{E}_{z\sim q_v}\log p_w(x \mid z)$ is approximated by a sample,
- notice however that the first term must be differentiated w.r.t. w and v,

How to differentiate a $z \sim q_v(z \mid x)$ w.r.t. v? By the reparametrisation trick

$$z \sim \mathcal{N}(\mu, \sigma) \;\Leftrightarrow\; z = \sigma\epsilon + \mu \text{ with } \epsilon \sim \mathcal{N}(0,1)$$

This results in

$$\mathbb{E}_{z\sim\mathcal{N}(\mu,\sigma)}\log p_w(x \mid z) = \mathbb{E}_{\epsilon\sim\mathcal{N}(0,1)}\log p_w(x \mid \sigma\epsilon + \mu)$$

Variational Autoencoders

Sophisticated versions of deep convolutional variational autoencoders achieve quite impressive results

Open problems:

When using deep belief networks both for the "decoder" and the "encoder", we may observe "latent variable collapse". This means that the last layers of the encoder model have the tendency to collapse into the decoder's prior, i.e.

$$q_v(z_t \mid z_{t-1}) \sim \mathcal{N}(0, \mathbb{I}).$$