

Machine Learning for Graphs and Sequential Data

Deep Generative Models - Generative Adversarial Networks

lecturer: Prof. Dr. Stephan Günnemann
www.daml.in.tum.de

Summer Term 2020

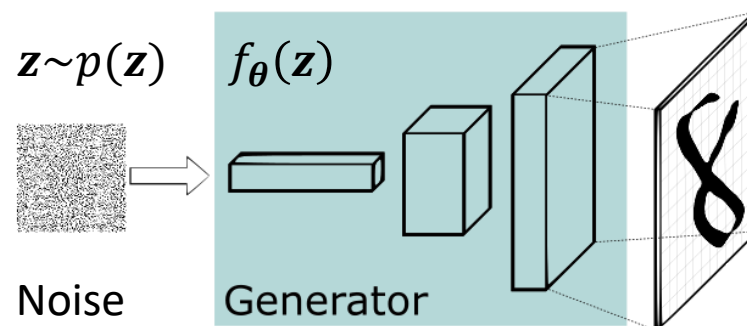
Data Analytics and
Machine Learning 

Roadmap

- Chapter: Deep Generative Models
 1. Introduction
 2. Normalizing Flows
 3. Variational Inference
 - 4. Generative Adversarial Networks**
 5. Summary

Learn a Generative Model with Fewer Assumptions

- The variational autoencoder assumes a latent variable structure and specific forms of the likelihood and the variational posterior
- Can we model the data better by having fewer assumptions?
- Idea: Transform given initial noise using a flexible function
 1. Draw an initial noise vector from the prior $\mathbf{z} \sim p(\mathbf{z})$
 2. Deterministically transform \mathbf{z} into the target data/output space $\mathbf{x} = f_{\theta}(\mathbf{z})$



- In contrast to Normalizing Flows, $f_{\theta}(\mathbf{z})$ can be any function

Resulting Distribution

1. Draw an initial noise vector from the prior $\mathbf{z} \sim p(\mathbf{z})$
2. Deterministically transform \mathbf{z} into the target data/output space $\mathbf{x} = f_{\theta}(\mathbf{z})$

- This process defines a valid density on the output space

$$p_{\theta}(\mathbf{x}) = \frac{d}{dx_1} \cdots \frac{d}{dx_d} \int_{\{f_{\theta}(\mathbf{z}) \leq \mathbf{x}\}} p(\mathbf{z}) d\mathbf{z}$$

- Unfortunately $p_{\theta}(\mathbf{x})$ is often intractable to compute
 - The integration region $\{f_{\theta}(\mathbf{z}) \leq \mathbf{x}\}$ itself may be very hard to determine
 - The integral itself may then be computationally demanding
 - Taking d partial derivatives in a high-dimensional space could be challenging

Learn a Generative Model with Fewer Assumptions

1. Draw an initial noise vector from the prior $\mathbf{z} \sim p(\mathbf{z})$
 2. Deterministically transform \mathbf{z} into the target data/output space $\mathbf{x} = f_{\theta}(\mathbf{z})$
- Sampling is easy; density evaluation is hard!
- What can we do if we can only sample but we cannot evaluate the density?
 - Without a (tractable) **likelihood function**, many widely-used tools for inference and parameter learning become **unavailable** (e.g. the previously seen variational inference)

Forward Pointer: Generative Adversarial Networks

- GANs have become very popular for learning deep generative models
- Informally, the main idea is:

- Two competing neural network models

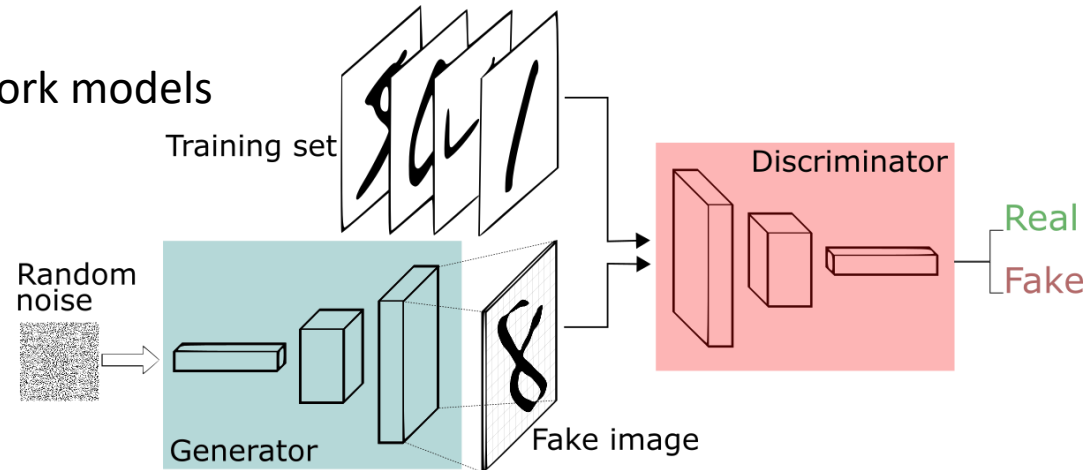
- Generator: takes noise as input and generates ("fake") samples

- Discriminator: receives samples from both generator and training data

and has to distinguish between the two → classify input as "real" or "fake"

- Goal: Train the generator in such a way that the discriminator can **not** distinguish between real and "fake" samples

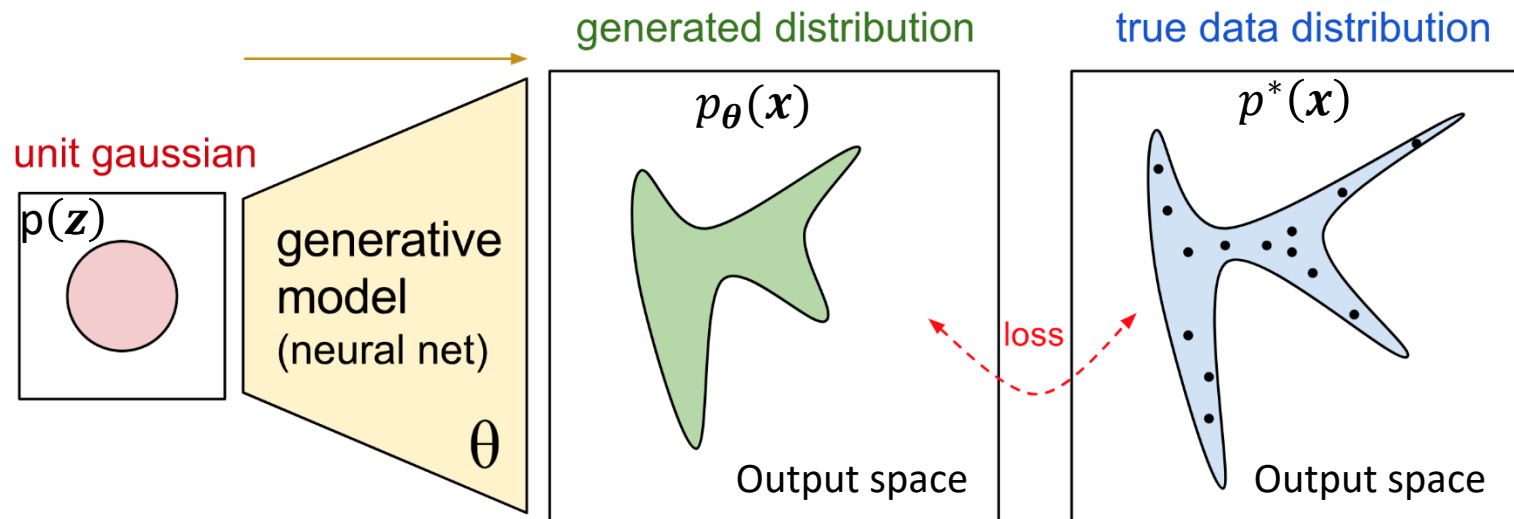
➤ In this case, the generator generates realistic examples



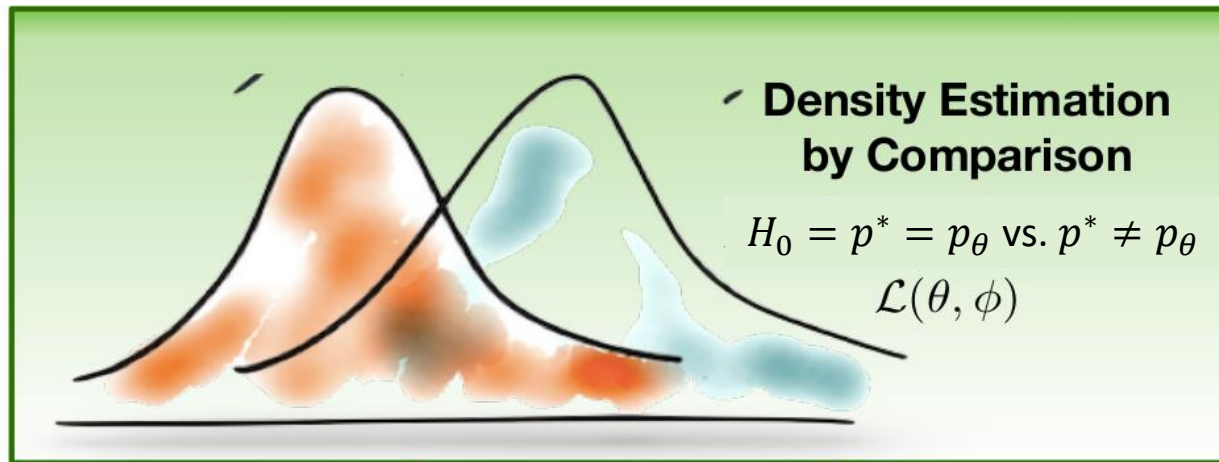
In the following, you learn the actual (and more general) concept behind it

Likelihood-free inference

- What can we do if we can **easily draw samples** from the model but we cannot evaluate the density?
- Idea: We can use **any** method that compares two **sets of samples**.
 - This process is called ***density estimation by comparison***.
- We **test** the **hypothesis** that the true data distribution $p^*(x)$ and the model distribution $p_\theta(x)$ are **equal**

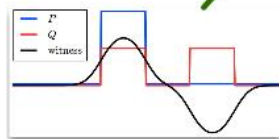


Density Estimation by Comparison

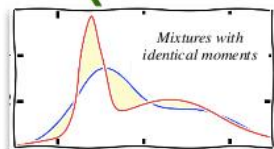


Density Difference

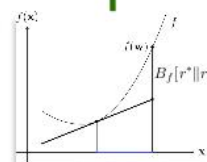
$$r_\phi = p^* - p_\theta$$



*Max Mean
Discrepancy*



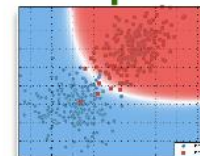
*Moment
Matching*



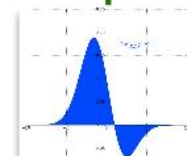
*Bregman
Divergence*

Density Ratio

$$r_\phi = \frac{p^*}{p_\theta}$$



*Class Probability
Estimation*



f-Divergence

Figure: [Mohamed and Lakshminarayanan, 2016]

Density Difference Approaches (I)

- Moment Matching
 - Use the fact that p^* and p_θ are identical iff the expectations of any test statistic $s(\mathbf{x})$ are identical
 - Thus, minimize the gap $\min_{\theta} (\mathbb{E}_{p^*(\mathbf{x})}[s(\mathbf{x})] - \mathbb{E}_{p(\mathbf{z})}[s(f_\theta(\mathbf{z}))])^2$
 - Approximate expectation by Monte-Carlo samples

Density Difference Approaches (II)

- Max Mean Discrepancy
 - When the statistics $s(\mathbf{x})$ are defined within a reproducing kernel Hilbert space, we obtain kernel-based forms of these objectives
 - These kernels are highly flexible and allow easy handling of data such as images, strings and graphs

Ratio-Based Approaches (I)

- Density ratio $r^*(\mathbf{x}) = p^*(\mathbf{x}) / p_\theta(\mathbf{x})$
 - In the best case always 1, i.e. the two distributions are indistinguishable
 - However, we cannot compute ratio in closed form/easily
- Idea: Approximate the true density ratio $r^*(\mathbf{x})$ by $r_\phi(\mathbf{x})$
 - Finding the approximation $r_\phi(\mathbf{x})$ often means solving again a learning problem
- Thus, we get the following general principle for learning
 - **Optimize Ratio loss:**
approximate the true density ratio $r^*(\mathbf{x})$ (i.e. learning ϕ)
 - **Optimize Generative loss:**
drive the density ratio towards 1 (i.e. learning θ)
 - Essentially a bi-level optimization problem, which is usually just solved alternatingly

Ratio-Based Approaches (II)

- The following three methods are an instantiation of this principle:
- Class Probability Estimation
 - Estimate the density ratio via a classifier which can distinguish between the observed data and data generated from the model (principle used in, e.g., **Generative Adversarial Networks (GANs)**)
- Divergence Minimization
 - Minimize an f -divergence between the true density and the generator density
 - For specific choices of the divergence function this can be equivalent to the class probability estimation case or KL divergence
- Ratio matching
 - Directly minimize the error between the true density ratio and an estimate of it:
$$\mathcal{L} = \mathbb{E}_{p_\theta} \left[\left(r_\phi(\mathbf{x}) - r^*(\mathbf{x}) \right)^2 \right]$$
 - Can be generalized beyond the squared error using the Bregman divergence

Learning via Class Probability Estimation (I)

- Let Y denote a random variable which assigns label $Y = 1$ to samples from the true data distribution; $Y = 0$ to those from the generator distribution
- Then $p(\mathbf{x} \mid Y = 1) = p^*(\mathbf{x})$ and $p(\mathbf{x} \mid Y = 0) = p_\theta(\mathbf{x})$

- Denote $P(Y = 1) = \pi$. From Bayes we have:

$$r^*(\mathbf{x}) = \frac{p^*(\mathbf{x})}{p_\theta(\mathbf{x})} = \frac{p(Y = 1 \mid \mathbf{x})}{p(Y = 0 \mid \mathbf{x})} \frac{1 - \pi}{\pi}$$

- Apparently, density ratio estimation is equal to **class-probability estimation**
 - Simply speaking: we can consider a classifier for \mathbf{x} (predicting labels $Y=0$ or 1)
- Specify a scoring function or a **discriminator** $D_\phi(\mathbf{x}) = p(Y = 1 \mid \mathbf{x})$
- e.g. logistic regression or a neural network

Learning via Class Probability Estimation (II)

- Specify a scoring function or a **discriminator** $D_\phi(\mathbf{x}) = p(Y = 1 | \mathbf{x})$
 - that tells you whether a sample \mathbf{x} is real or from the generator (“fake”)
 - e.g. logistic regression or a neural network
- For learning $D_\phi(\mathbf{x})$, we need a loss function, e.g., the cross-entropy loss:

$$\begin{aligned}\mathcal{L}_{\theta, \phi} &= \mathbb{E}_{(x, y) \sim p(x, y)} \left[-y \log[D_\phi(\mathbf{x})] - (1 - y) \log[1 - D_\phi(\mathbf{x})] \right] \\ &= \pi \mathbb{E}_{p^*(\mathbf{x})} \left[-\log D_\phi(\mathbf{x}) \right] + (1 - \pi) \mathbb{E}_{p(\mathbf{z})} \left[-\log[1 - D_\phi(f_\theta(\mathbf{z}))] \right]\end{aligned}$$

Learning via Class Probability Estimation (III)

1. Solving
$$\phi^*(\theta) = \underset{\phi}{\operatorname{argmin}} \mathcal{L}_{\theta, \phi}$$
 leads to the "best" discriminator for a given generative model (θ)
 - That is, we will approximate $r^*(\mathbf{x}) = \frac{p^*(\mathbf{x})}{p_{\theta}(\mathbf{x})} = \frac{p(Y=1 | \mathbf{x})}{p(Y=0 | \mathbf{x})} \approx \frac{D_{\phi^*(\theta)}(\mathbf{x})}{1 - D_{\phi^*(\theta)}(\mathbf{x})}$
 - here w.l.o.g. we set $\pi = 0.5$
2. We aim to drive the density ratio $r^*(\mathbf{x})$ towards 1
 - aim: $p(Y = 1 | \mathbf{x}) = p(Y = 0 | \mathbf{x})$
 - That is, find generative model (θ) such that (even) the "best" discriminator cannot distinguish the classes
 - Solving
$$\theta^* = \underset{\theta}{\operatorname{argmax}} \mathcal{L}_{\theta, \phi^*(\theta)}$$

Learning via Class Probability Estimation (IV)

Attention: Note
the flipped sign
in the loss
formulation!

$$-\mathcal{L}_{\theta,\phi}$$

- Generator and discriminator play a **minimax game**:

$$\min_{\theta} \max_{\phi} \pi \mathbb{E}_{p^*(x)} [\log D_{\phi}(x)] + (1 - \pi) \mathbb{E}_{p(z)} [\log [1 - D_{\phi}(f_{\theta}(z))]]$$

- Discriminator: aims to distinguish between (samples from) $p^*(x)$ and $p_{\theta}(x)$

➤ Maximization

// minimization of cross-entropy

- Generator: aims to generate samples that are indistinguishable

➤ Minimization

// maximization of (lowest) cross-entropy

- This is a bilevel optimization problem

Learning via Class Probability Estimation (V)

- This bilevel problem is typically tackled via alternating optimization
 - usually does not lead to the optimal solution
 - depending on the setting, difficult to train (instable)

- **Ratio loss** (discriminator loss) optimization:

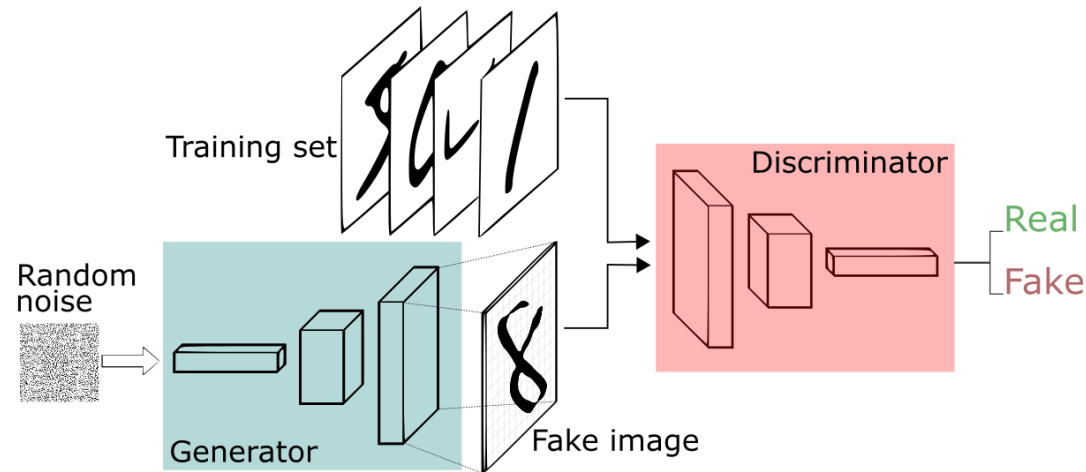
$$\min_{\phi} \pi \mathbb{E}_{p^*(x)} [-\log D_{\phi}(x)] + (1 - \pi) \mathbb{E}_{p(z)} [-\log [1 - D_{\phi}(f_{\theta}(z))]]$$

- **Generative loss** optimization:

$$\min_{\theta} \mathbb{E}_{p(z)} [\log [1 - D_{\phi}(f_{\theta}(z))]]$$

GANs – Generative Adversarial Networks

- Main idea
 - Two competing neural network models
 - Generator: takes noise as input and generates ("fake") samples
 - Discriminator: receives samples from both generator and training data and has to distinguish between the two
 - They play a minimax game against each other
 - Competition drives the generated samples to be indistinguishable from real



- In short: GAN approach =
Learning of a generator via class probability estimation where the generator and the discriminator are neural networks

GANs – Examples

- Synthetically generated images



Figure 5: 1024×1024 images generated using the CELEBA-HQ dataset. See Appendix F for a larger set of results, and the accompanying video for latent space interpolations.

From "Progressive Growing of GANs for Improved Quality, Stability, and Variation", Tero Karras et.al.

Questions – GANs

1. What can we say about the ratio $r^*(x) = p^*(x) / p_\theta(x)$ when:
 - a) The generator and the discriminator are optimal
 - b) The generator only is optimal
 - c) The discriminator only is optimal

2. For a given data x , what is the probability for the discriminator to be correct when:
 - a) The generator and the discriminator are optimal
 - b) The generator only is optimal
 - c) The discriminator only is optimal

References

- [1] Mohamed, Shakir, and Balaji Lakshminarayanan. “Learning in Implicit Generative Models.”: <https://arxiv.org/pdf/1610.03483.pdf>
- [2] Andrew Davison blog :
<https://casmls.github.io/general/2017/05/24/ligm.html>

External Sources

Web tutorial:

- Andrew Davison blog : <https://casmls.github.io/general/2017/05/24/ligm.html>
Eric Jang blog : <https://blog.evjang.com/2018/01/nf1.html>

Survey papers:

- Mohamed, Shakir, and Balaji Lakshminarayanan. “Learning in Implicit Generative Models.”: <https://arxiv.org/pdf/1610.03483.pdf>