Unsupervised Anomaly Detection with Likelihood-based Generative Models

杨 帆

fanyang01@zju.edu.cn

2019年10月18日

OUTLINE

- Unsupervised Anomaly Detection
- Likelihood-based DGMs for Density Estimation
- Is Likelihood A Good OOD Measure?

OUTLINE

- **Unsupervised Anomaly Detection**
- Likelihood-based DGMs for Density Estimation
- Is Likelihood A Good OOD Measure?

Anomaly Detection: A Survey. Varun Chandola, et al. ACM Computer Surverys 2009
A Review of Novelty Detection. Marco A. F. Pimentel, et al. Signal Processing 2014
Neural Density Estimation and Likelihood-free Inference. George Papamakarios. PhD Thesis, 2019

Anomaly Detection

- "**anomaly/abnormality**" in different scenarios
  - **Medicine**: Unusual patterns in EEG/ECG/Ultrasound/CT/fMRI
  - **Geoscience**: Natural disaster, climate change, pollution
  - **Finance**: Transactional fraud, fake account, promotion abuse
  - **Cybersecurity**: Intrusion, vulnerability, malware, malicious behavior/input
  - **DevOps**: KPI change, QoS violation, system failure, software bug

- Definition of "anomaly detection" ?
  - AD is an application-specific technique without a universal formalization

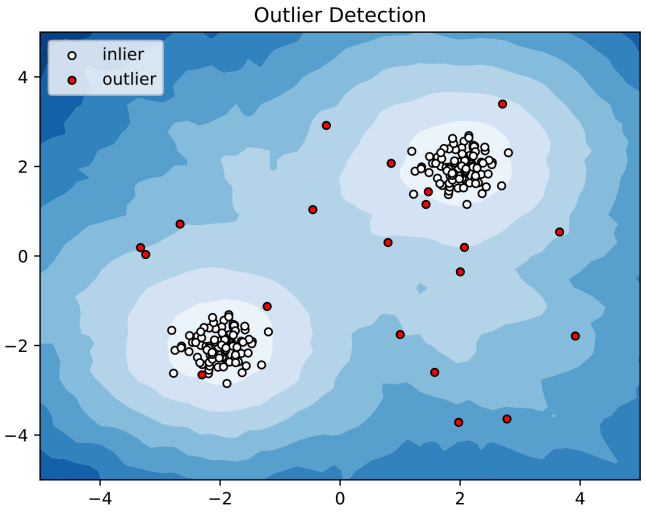
- Closely related terms
  - Outlier detection
  - Novelty detection
  - Out-Of-Distribution detection

Anomaly Detection: Terminology

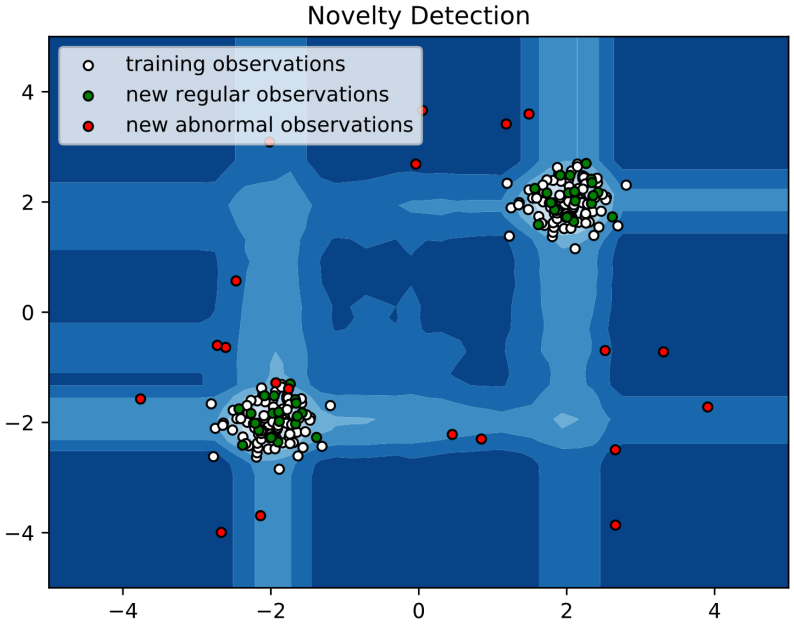
- **Outlier Detection**
  - Given a dataset $\mathcal{D} = \{x_i\}_{i=1}^N$, find $\mathcal{O} \subseteq \mathcal{D}$ such that $\forall\, x_i \in \mathcal{O}$ is an outlier in $\mathcal{D}$
    - An outlier is a data point that differs significantly from other observations
  - Formally, assume that for $\forall\, i = 1, \ldots, N$:

$$z_i \sim \text{Bernoulli}(\epsilon)\,, \qquad x_i \sim \begin{cases} p_*(x), & z_i = 1 \\ p_o(x), & z_i = 0 \end{cases}$$

  - Then infer $p(z_i = 0 \mid \mathcal{D})$
    - i.e., the probability that x_i came from the outlier distribution $p_o(x)$



Anomaly Detection: Terminology



- **Novelty Detection**
  - a.k.a. **O**ut-**o**f-**D**istribution detection
  - Given a **reference** dataset $\mathcal{D} = \{x_i\}_{i=1}^N$, test if a **new** data point x_+ is **similar** to some data points in $\mathcal{D}$
  - Formally, assume that $\forall\, i = 1, \ldots, N\colon\ x_i \sim p_*(x)$, then guess $p_*(x_+)$

Anomaly Detection: Terminology

- **Outlier Detection**
  - Given a dataset $\mathcal{D} = \{x_i\}_{i=1}^N$, find $\mathcal{O} \subseteq \mathcal{D}$ such that $\forall\, x_i \in \mathcal{O}$ is an outlier in $\mathcal{D}$
    - An outlier is a data point that differs significantly from other observations
  - Formally, assume that for $\forall\, i = 1, \dots, N$:

$$z_i \sim \text{Bernoulli}(\epsilon)\,, \qquad x_i \sim \begin{cases} p_*(x), & z_i = 1 \\ p_o(x), & z_i = 0 \end{cases}$$

  - Then infer $p(z_i = 0 \mid \mathcal{D})$ and $p(z_+ = 0 \mid x_+, \mathcal{D})$
    - i.e., the probability that x_i came from the outlier distribution $p_o(x)$

- **Novelty Detection**
  - a.k.a. **O**ut-**O**f-**D**istribution detection
  - Given a **reference** dataset $\mathcal{D} = \{x_i\}_{i=1}^N$, test if a **new** datapoint x_+ is **similar** to some datapoints in $\mathcal{D}$
  - Formally, assume that $\forall\, i = 1, \dots, N$: $x_i \sim p_*(x)$, then guess $p_*(x_+)$

Anomaly Detection: Terminology

- **Outlier Detection**
  - Given a dataset $\mathcal{D} = \{x_i\}_{i=1}^N$, find $\mathcal{O} \subseteq \mathcal{D}$ such that $\forall\, x_i \in \mathcal{O}$ is an outlier in $\mathcal{D}$
    - An outlier is a data point that differs significantly from other observations
  - Formally, assume that for $\forall\, i = 1, \ldots, N$:

$$z_i \sim \text{Bernoulli}(\epsilon)\,, \qquad x_i \sim \begin{cases} p_*(x), & z_i = 1 \\ p_o(x), & z_i = 0 \end{cases}$$

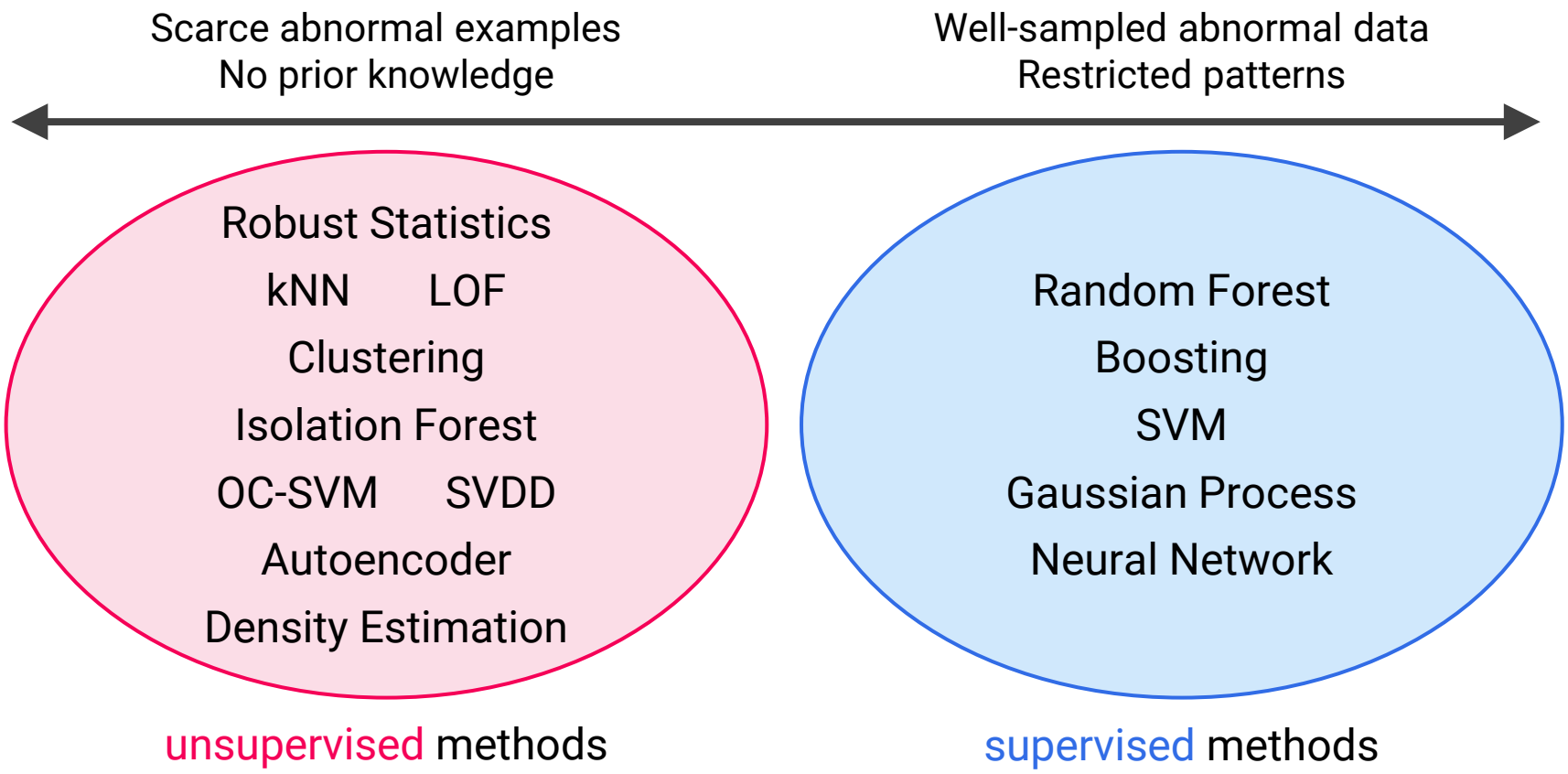
  - Then infer $p(z_i = 0 \mid \mathcal{D})$ and $p(z_+ = 0 \mid x_+, \mathcal{D})$
    - i.e., the probability that x_i came from the outlier distribution $p_o(x)$

Anomaly Detection?

- **Novelty Detection**
  - a.k.a. **O**ut-**O**f-**D**istribution detection
  - Given a **reference** dataset $\mathcal{D} = \{x_i\}_{i=1}^N$, test if a **new** datapoint x_+ is **similar** to some datapoints in $\mathcal{D}$
  - Formally, assume that $\forall\, i = 1, \ldots, N$: $x_i \sim p_*(x)$, then guess $p_*(x_+)$

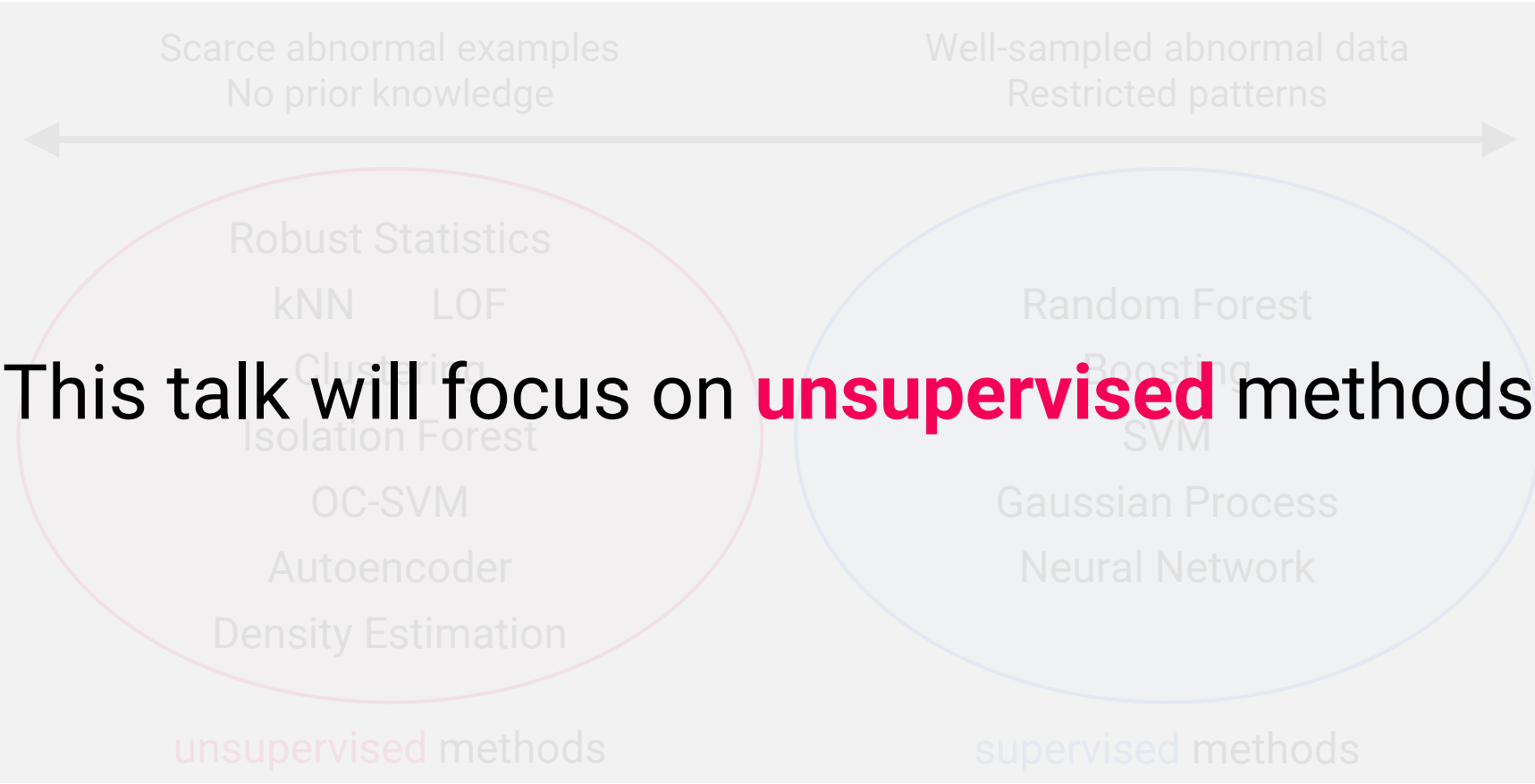
Machine Learning Approaches for AD

- Most AD tasks can be formalized as either outlier or OOD detection
  - Outlier detection: supervised learning or unsupervised learning
  - OOD detection: unsupervised learning by definition



Machine Learning Approaches for AD

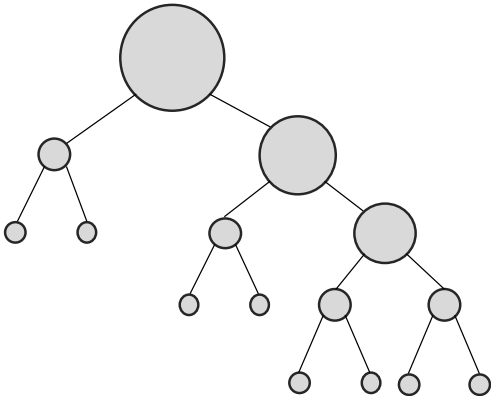
- Most AD tasks can be formalized as either outlier or OOD detection
  - Outlier detection: supervised learning or unsupervised learning
  - OOD detection: unsupervised learning by definition



This talk will focus on **unsupervised** methods.

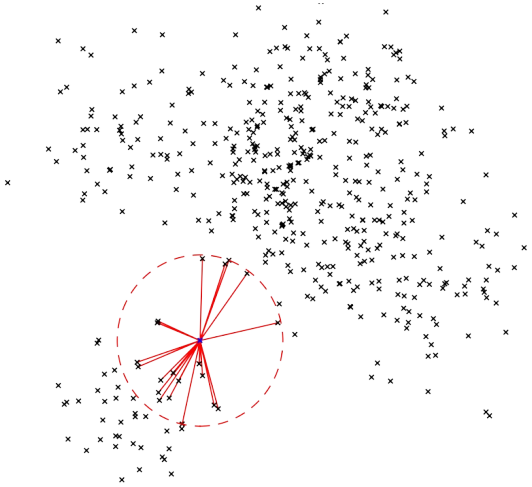
Unsupervised Anomaly Detection

- Most UAD methods do implicit/explicit **density estimation**
  - Implicit: Using a proxy measurement of density



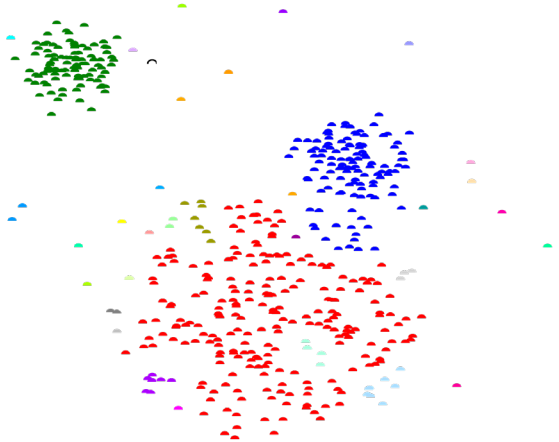
Isolation Forest

avg depth in random partition trees



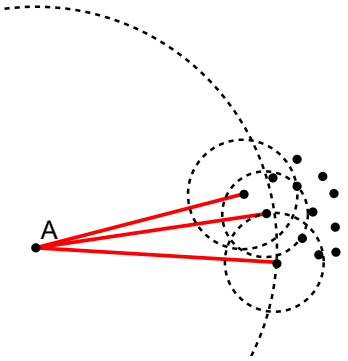
k Nearest Neighbors

avg distance to neighbors
or indegree in kNN graph



Clustering

cluster size
or distance to prototype

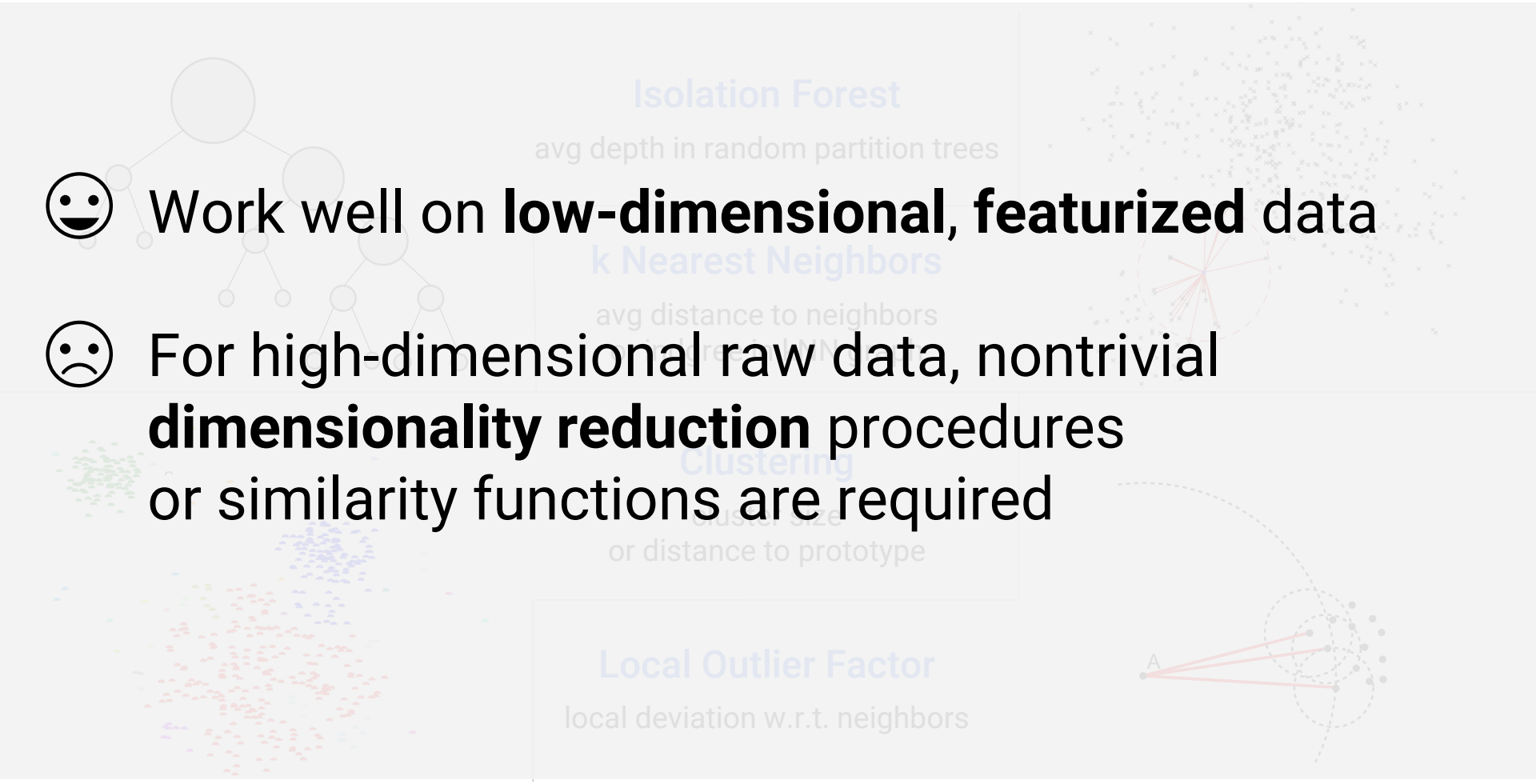


Local Outlier Factor

local deviation w.r.t. neighbors

Unsupervised Anomaly Detection

- Most UAD methods do implicit/explicit **density estimation**
  - Implicit: Using a proxy measurement of density



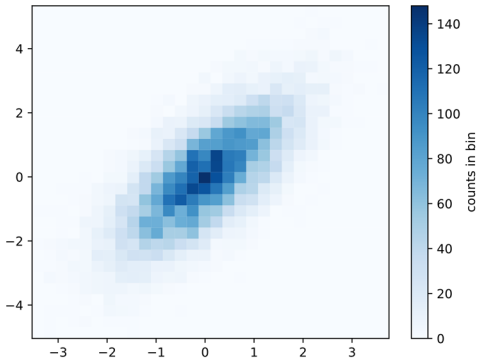
☺ Work well on **low-dimensional**, **featurized** data

☹ For high-dimensional raw data, nontrivial **dimensionality reduction** procedures or similarity functions are required

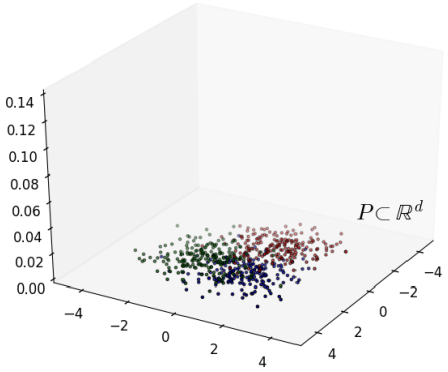
Unsupervised Anomaly Detection

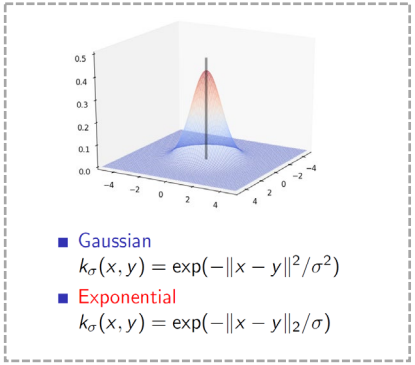
- Most UAD methods do implicit/explicit **density estimation**
  - Explicit: **nonparametric**/parametric density estimation

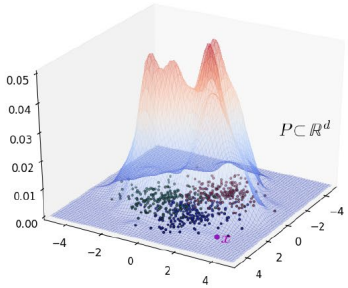
Histogram



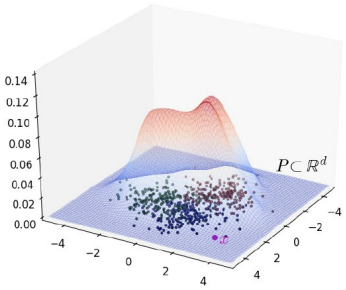
Kernel Density Estimation



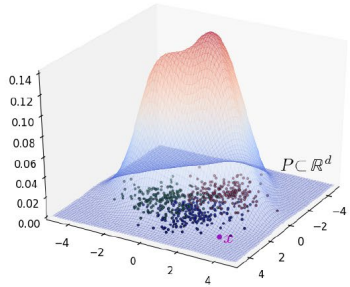




$\sigma_1 = \frac{1}{2} \cdot \text{std}$



$\sigma_2 = \frac{3}{4} \cdot \text{std}$

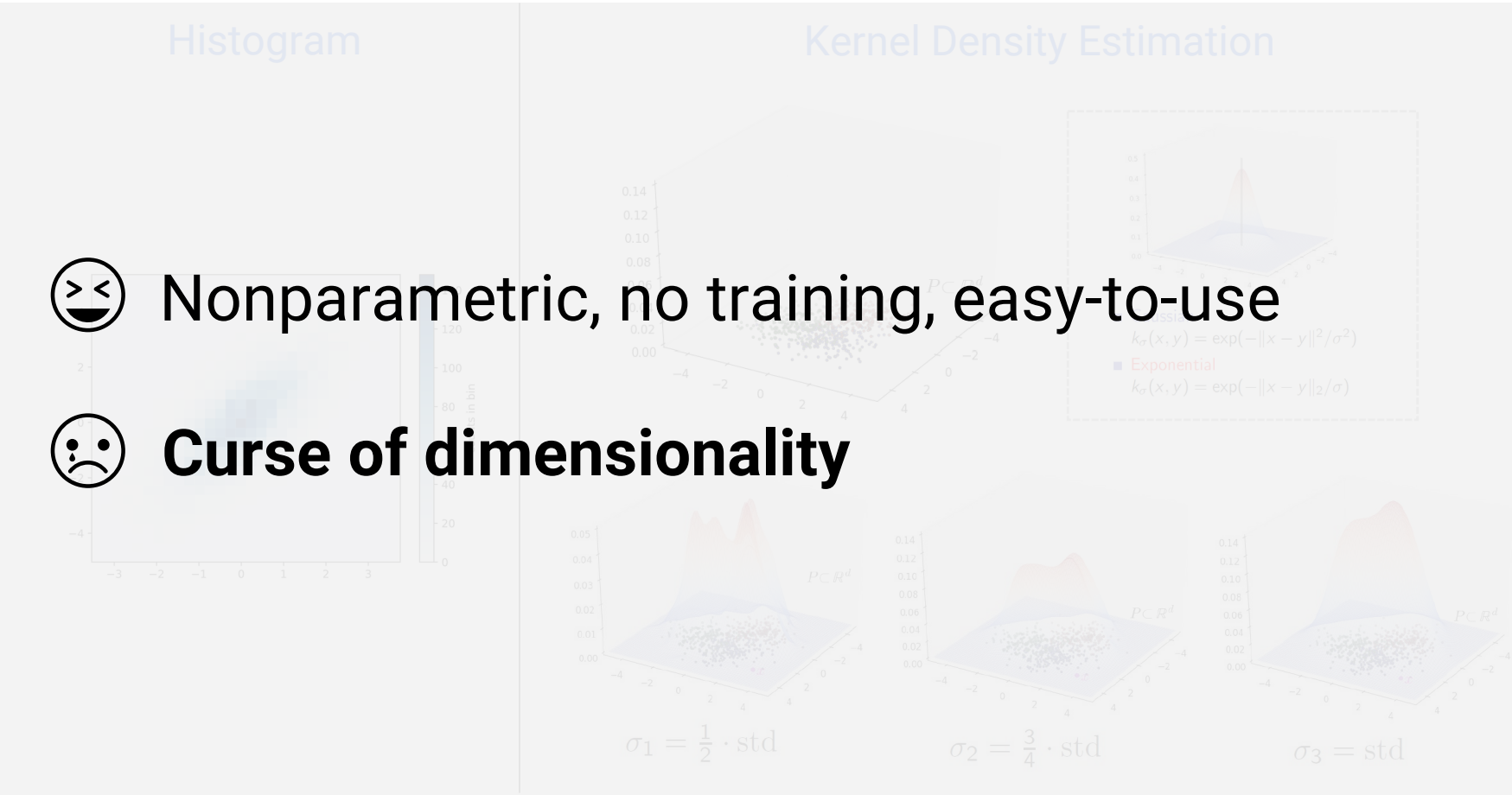


$\sigma_3 = \text{std}$

https://web.stanford.edu/~psimin/presentations/dawn17.pdf by Paris Siminelakis.

Unsupervised Anomaly Detection

- Most UAD methods do implicit/explicit **density estimation**
  - Explicit: **nonparametric**/parametric density estimation



😆 Nonparametric, no training, easy-to-use

☹ **Curse of dimensionality**

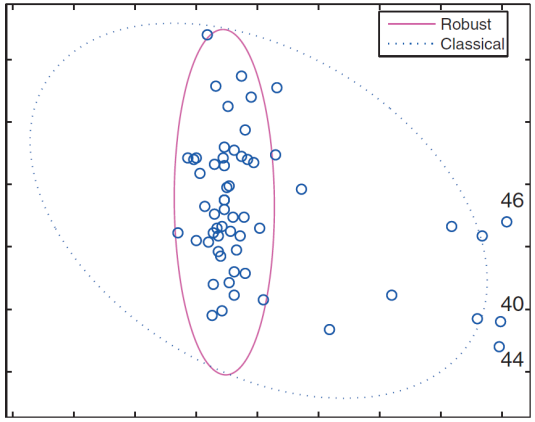
Unsupervised Anomaly Detection

- Most UAD methods do implicit/explicit **density estimation**
  - Explicit: nonparametric/**parametric** density estimation

Simple/Robust Statistics

3-sigma, covariance, MAD, MCD

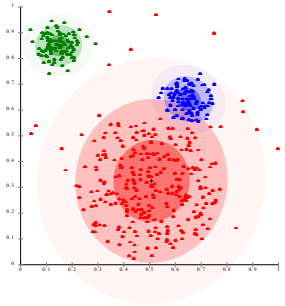
normally distributed low-dim data

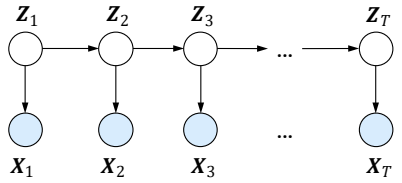


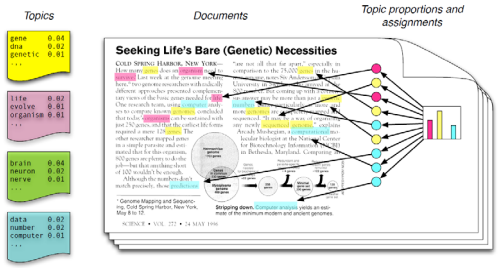
Generative Model

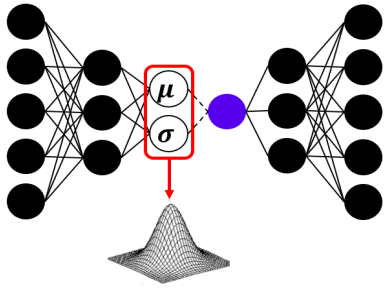
GMM, HMM, Topic Model, …, Deep Generative Model

can model highly complex high-dim data distribution









Unsupervised Anomaly Detection

- Most UAD methods do implicit/explicit **density estimation**
  - Explicit: nonparametric/**parametric** density estimation

Simple/Robust Statistics

3-sigma, covariance, MAD, MCD

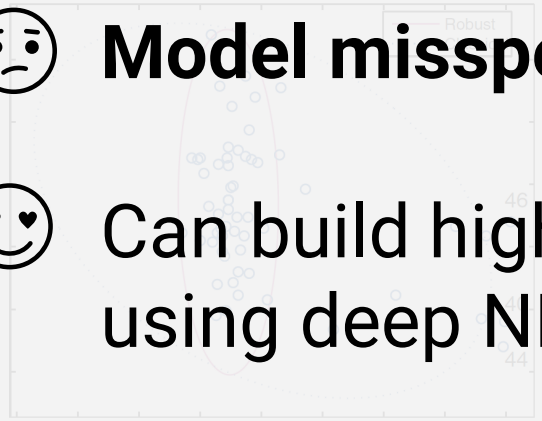
Generative Model

GMM, HMM, Topic Model, ..., Deep Generative Model

☺ Principled approach to deal with high-dim data

😟 **Model misspecification**, training cost

☺ Can build highly flexible Neural Density Estimators using deep NNs



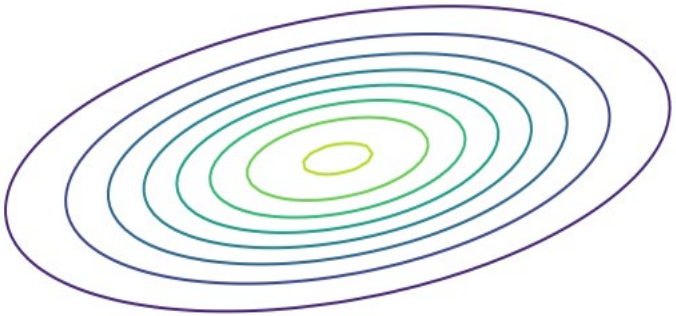
OUTLINE

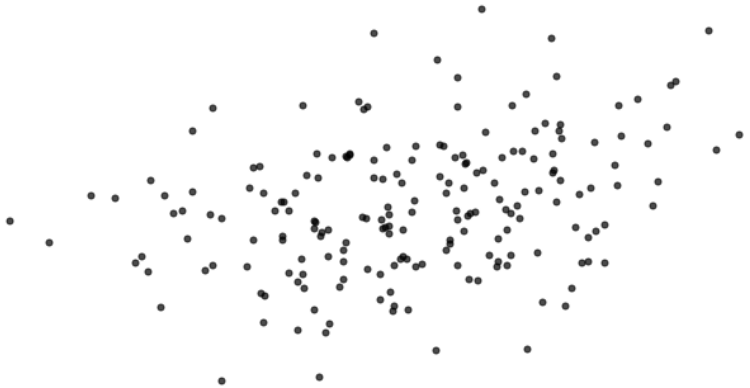
- Unsupervised Anomaly Detection
- **Likelihood-based DGMs for Density Estimation**
- Is Likelihood A Good OOD Measure?

An Introduction to Variational Autoencoders. Diederik P. Kingma, Max Welling. Arxiv 2019
Normalizing Flows: Introduction and Ideas. Ivan Kobyzev, et al. Arxiv 2019

Generative Modeling

- Treat the data points as **samples** from an unknown distribution





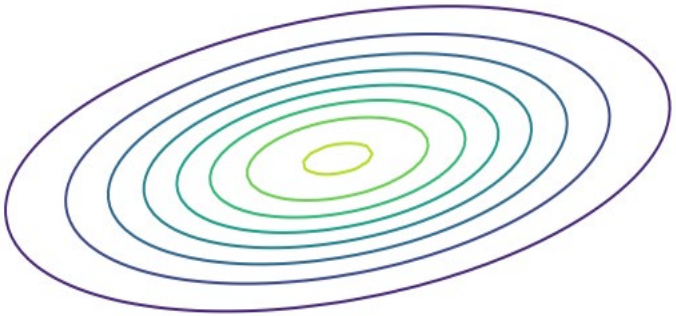
$p^*(\boldsymbol{x})$

$\{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(N)}\} \sim p^*(\boldsymbol{x})$

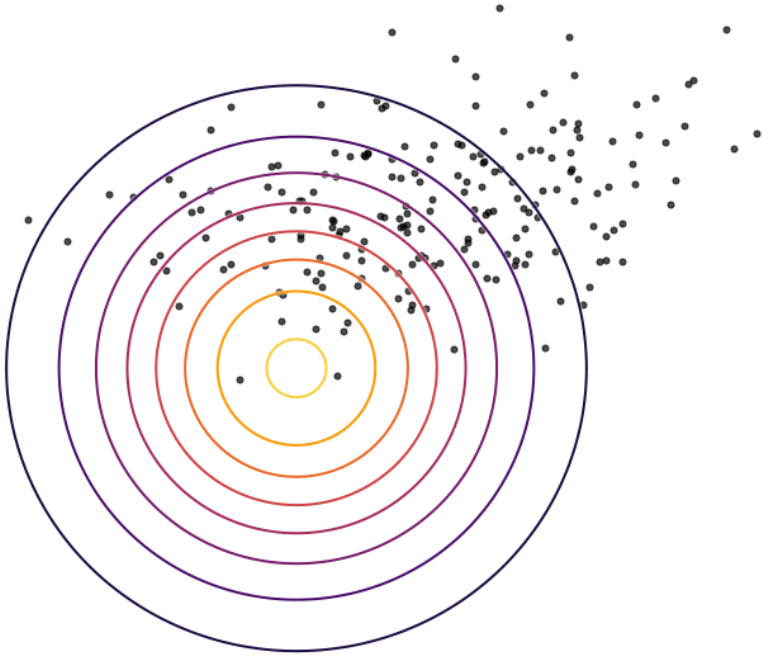
https://colinraffel.com/blog/gans-and-divergence-minimization.html

Generative Modeling

- Treat the data points as **samples** from an unknown distribution
- Design a **parameterized** distribution
- And **learn** its parameters to fit the samples



$p^*(\boldsymbol{x})$

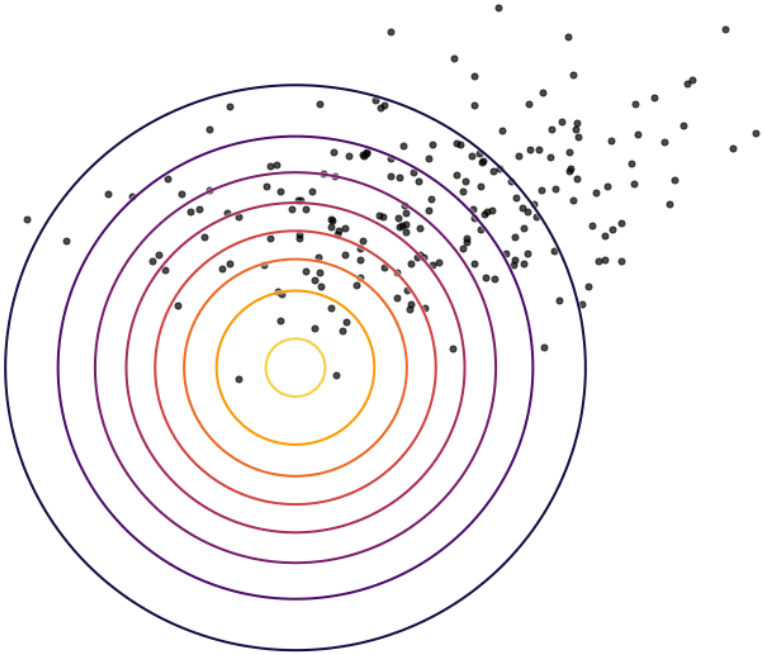


$p_\theta(\boldsymbol{x})$

https://colinraffel.com/blog/gans-and-divergence-minimization.html

Generative Modeling

$\boldsymbol{x}^{(i)} \in \mathbb{R}^2$



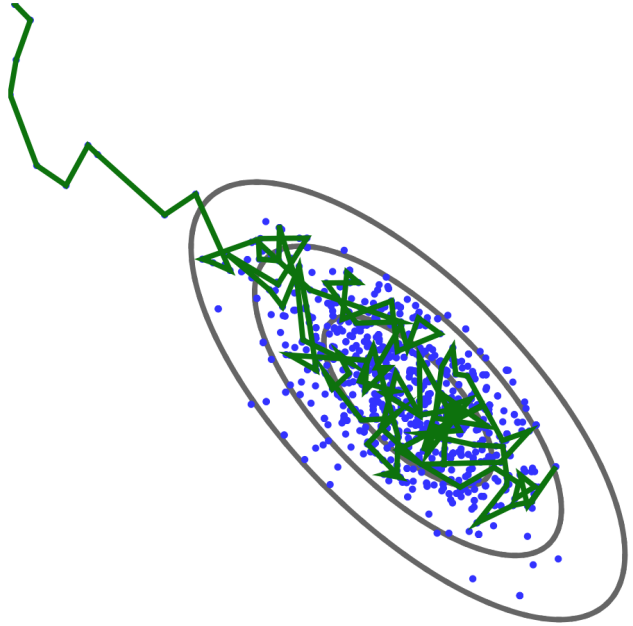
$p_\theta(\boldsymbol{x})$

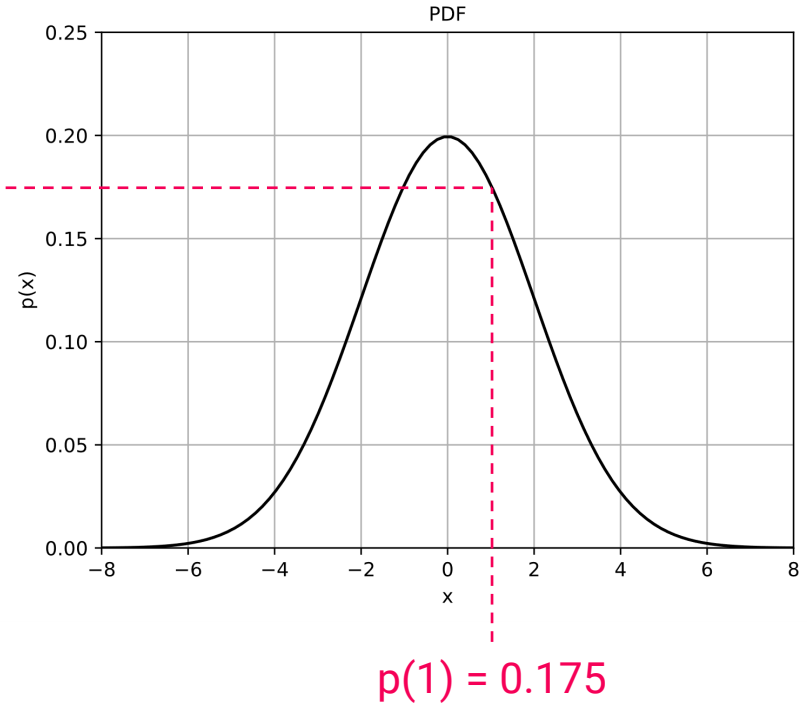
But $\boldsymbol{x}^{(i)}$ can also be:

- An **image** $\in \{0, \dots, 255\}^{W \times H \times 3}$
- A **sequence**
  - sentence, music notes, code
  - time series, audio, video
- A **graph**
  - molecular graph, AST, NN (!!!)
- A **set** (e.g., point cloud)
- A record in DB
- A distribution / function …

Applications of Generative Models

- Main uses of a **distribution** $p(x)$
  - **Sampling**: Generate novel samples $x_+ \sim p(x)$
  - **Density estimation**: Evaluate the prob. density $p(x_*)$ of a given sample x_*





Crafting Parameterized Distributions

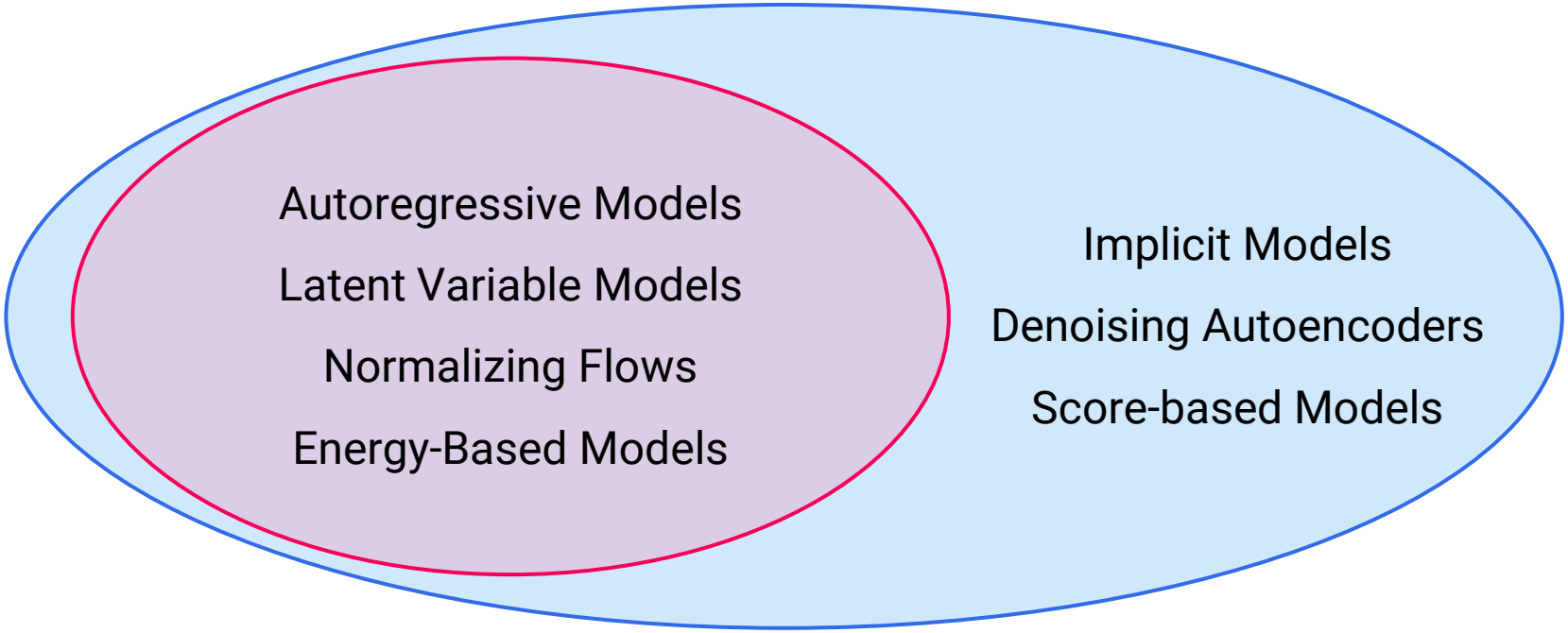
- Classical approach: Writing down the PDF

$$f_{\mathbf{x}}(x_1, \ldots, x_k) = \frac{\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)}{\sqrt{(2\pi)^k|\boldsymbol{\Sigma}|}}$$

- Nowadays

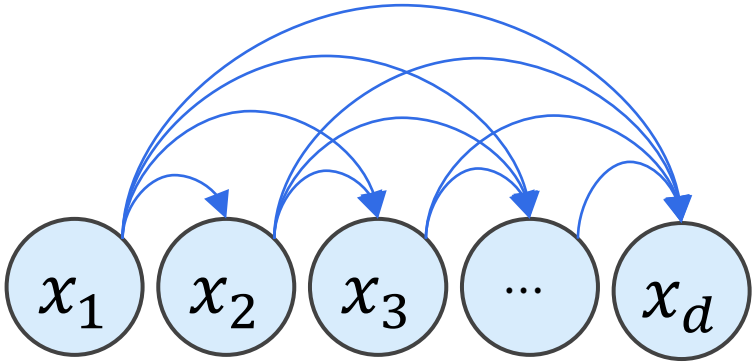
parameterize a density function

parameterize a sampler



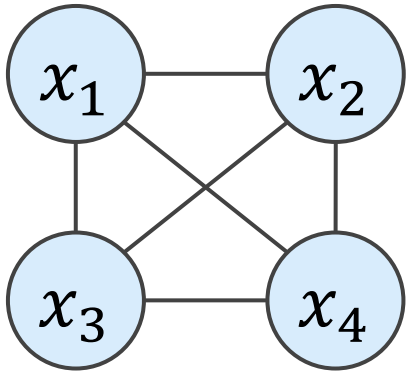
Likelihood-Based Generative Models

Autoregressive Models



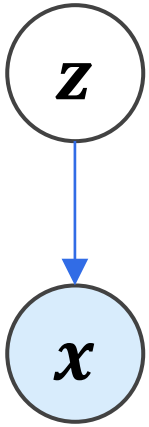
$$p_\theta(\boldsymbol{x}) = \prod_{i=1}^{d} p_\theta(x_i | x_{<i})$$

Energy-Based Models



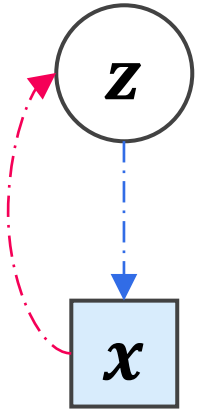
$$p_\theta(\boldsymbol{x}) = \frac{e^{-E_\theta(\boldsymbol{x})}}{\int e^{-E_\theta(\boldsymbol{x})}\, \mathrm{d}\boldsymbol{x}}$$

Latent Variable Models



$$p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x}|\boldsymbol{z}) p_\theta(\boldsymbol{z})\, \mathrm{d}\boldsymbol{z}$$

Normalizing Flows

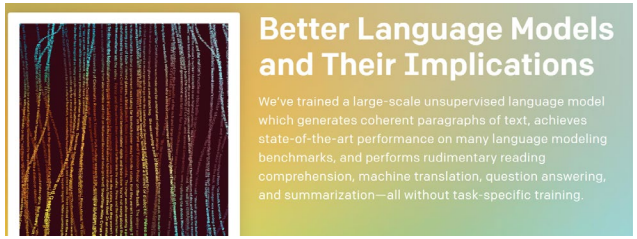


$$\boldsymbol{x} = f_\theta(\boldsymbol{z})$$

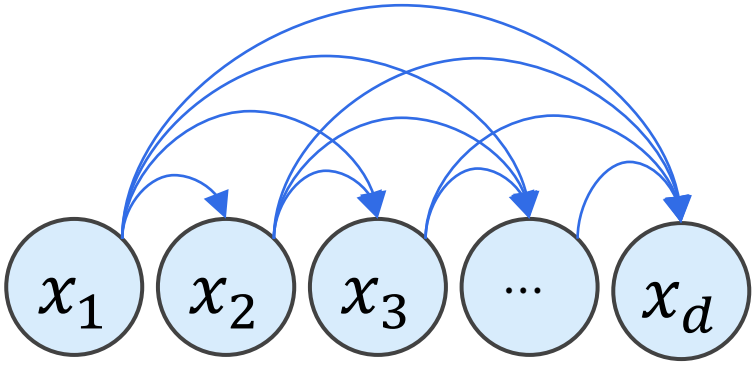
$$p_\theta(\boldsymbol{x}) = p_\theta(\boldsymbol{z}) \left| \det \frac{\partial f_\theta(\boldsymbol{z})}{\partial \boldsymbol{z}} \right|^{-1}$$

Autoregressive Models

- Any distribution can be factorized into an AR form (*chain rule*)
- Exact likelihood, easy to train
- Very generic and powerful
- Modern NNs make the training parallelizable
  - WaveNet, Causal CNN, Transformer



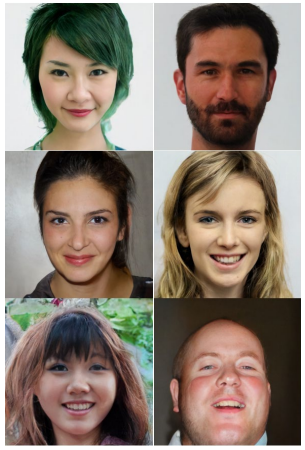
https://openai.com/blog/better-language-models/



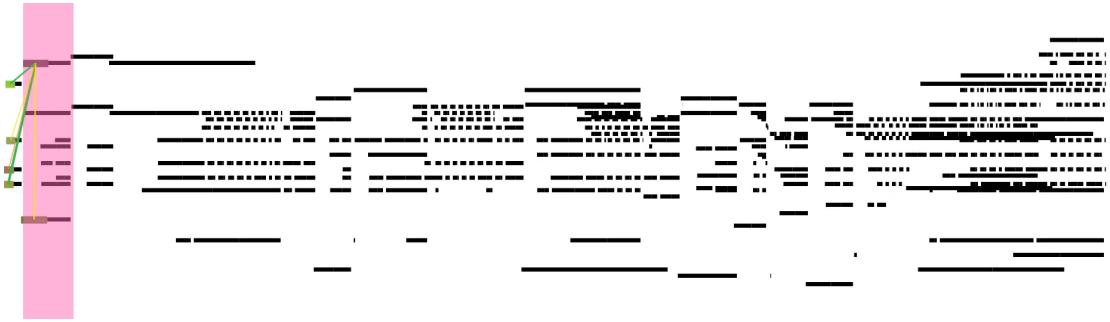
$$p_\theta(\boldsymbol{x}) = \prod_{i=1}^{d} p_\theta(x_i | x_{<i})$$

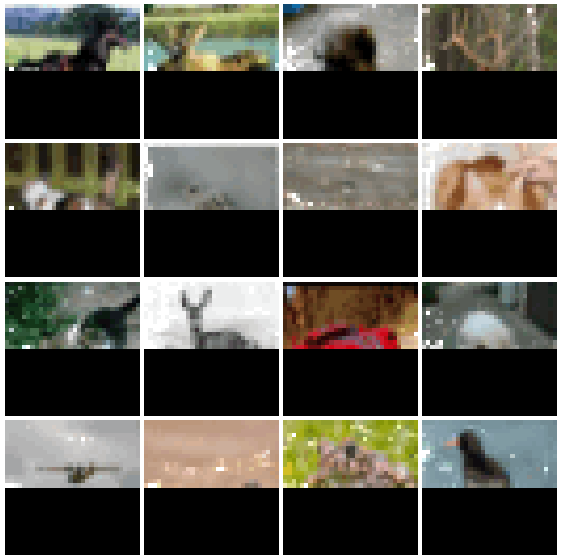
$$p_\theta(\boldsymbol{x}|\boldsymbol{c}) = \prod_{i=1}^{d} p_\theta(x_i | x_{<i}, \boldsymbol{c})$$

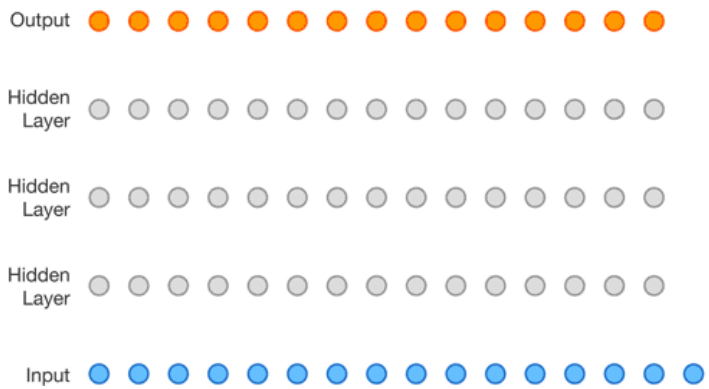
(conditional, e.g., seq2seq)



Generating Diverse High-Fidelity Images with VQ-VAE-2.
Ali Razavi, et al. NeurIPS 2019
Hierarchical Autoregressive Image Models with Auxiliary Decoders.
Jeffrey De Fauw, et al. Arxiv 2019

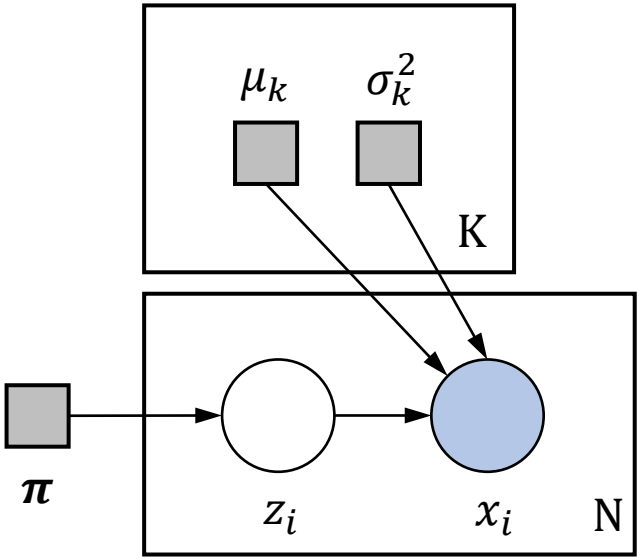





Output
Hidden Layer
Hidden Layer
Hidden Layer
Input

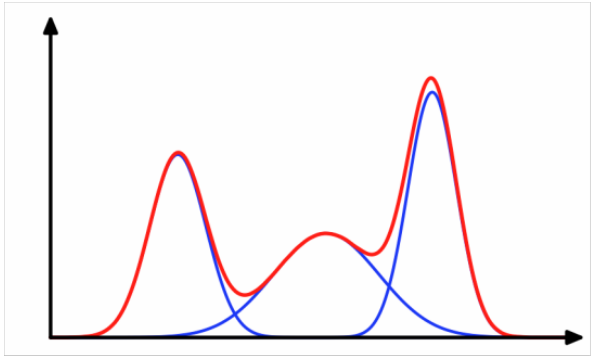
Latent Variable Models

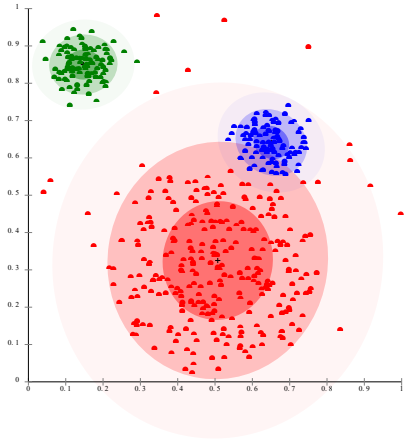
- Example: Gaussian Mixture Models



$$z_i \sim \text{categorical}(\pi_1, \ldots, \pi_K), \quad i = 1, \ldots, N,$$

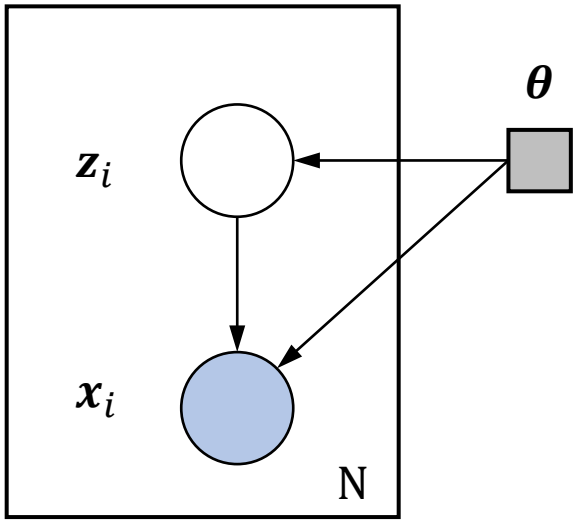
$$x_i | z_i \sim \mathcal{N}(\mu_{z_i}, \sigma_{z_i}^2) \qquad i = 1, \ldots, N.$$





Latent Variable Models

- Example: Deep Latent Gaussian Models



$$\boldsymbol{z}_i \sim \mathcal{N}(\mathbf{0}, \mathbb{I}), \qquad i = 1, \ldots, N,$$

$$\boldsymbol{x}_i | \boldsymbol{z}_i \sim \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{z}_i), \boldsymbol{\sigma}^2(\boldsymbol{z}_i)\mathbb{I}), \quad i = 1, \ldots, N.$$

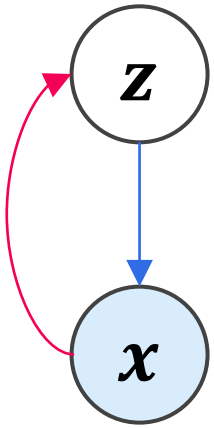
where $\boldsymbol{\mu}(\cdot)$ and $\boldsymbol{\sigma}(\cdot)$ are neural networks,
$\boldsymbol{\theta} = \{\text{parameters of } \boldsymbol{\mu} \text{ and } \boldsymbol{\sigma}\}$

BIVA: A Very Deep Hierarchy of Latent Variables for Generative Modeling. Lars Maaløe, et al. NeurIPS 2019

LVM: Variational Autoencoders

- VAE learns LVM by introducing a parameterized ***inference network*** (a.k.a. encoder)
  - Parameters of the encoder and the decoder are jointly optimized
    - By SGD on the Evidence Lower Bound Objective (ELBO)
  - Make **P**robabilistic **G**raphical **M**odels great again

- Density estimation: Using the encoder as a **proposal** to perform **importance sampling**



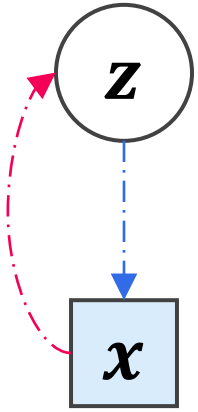
$$\log p_\theta(\boldsymbol{x}) = \log \int p_\theta(\boldsymbol{x}, \boldsymbol{z})\, \mathrm{d}\boldsymbol{z} = \log \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\frac{p_{\boldsymbol{\theta}}(\boldsymbol{z}, \boldsymbol{x})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right]$$

$$\geq \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log \frac{p_{\boldsymbol{\theta}}(\boldsymbol{z}, \boldsymbol{x})}{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\right]$$

An Introduction to Variational Autoencoders. Diederik P. Kingma, Max Welling. Arxiv 2019

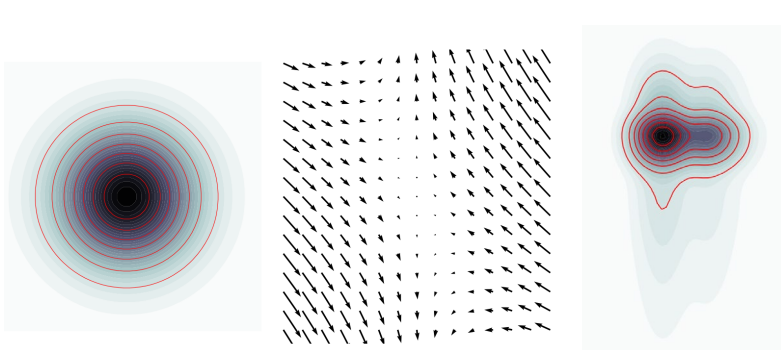
Normalizing Flows

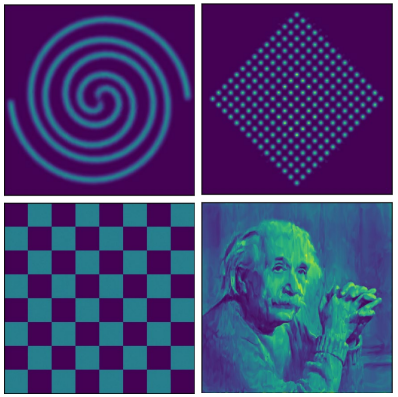
- Transform a simple density to a complex one
  - Using multilayer ***invertible*** mappings (or ODEs)
- Exact likelihood by the ***change of variable*** formula
- Usually fast sampling
- Exciting new area, many interesting ideas
  - Bonus: can help reduce memory footprint

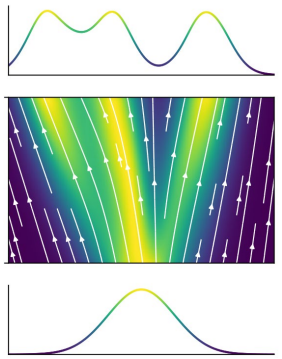


$$\boldsymbol{x} = f_\theta(\boldsymbol{z})$$

$$p_\theta(\boldsymbol{x}) = p_\theta(\boldsymbol{z}) \left| \det \frac{\partial f_\theta(\boldsymbol{z})}{\partial \boldsymbol{z}} \right|^{-1}$$







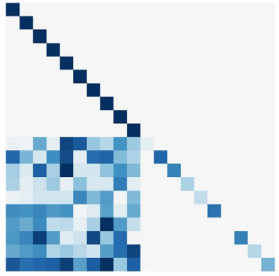
Normalizing Flows: Introduction and Ideas. Ivan Kobyzev, et al. Arxiv 2019
https://en.wikipedia.org/wiki/Jacobian_matrix_and_determinant

Normalizing Flows

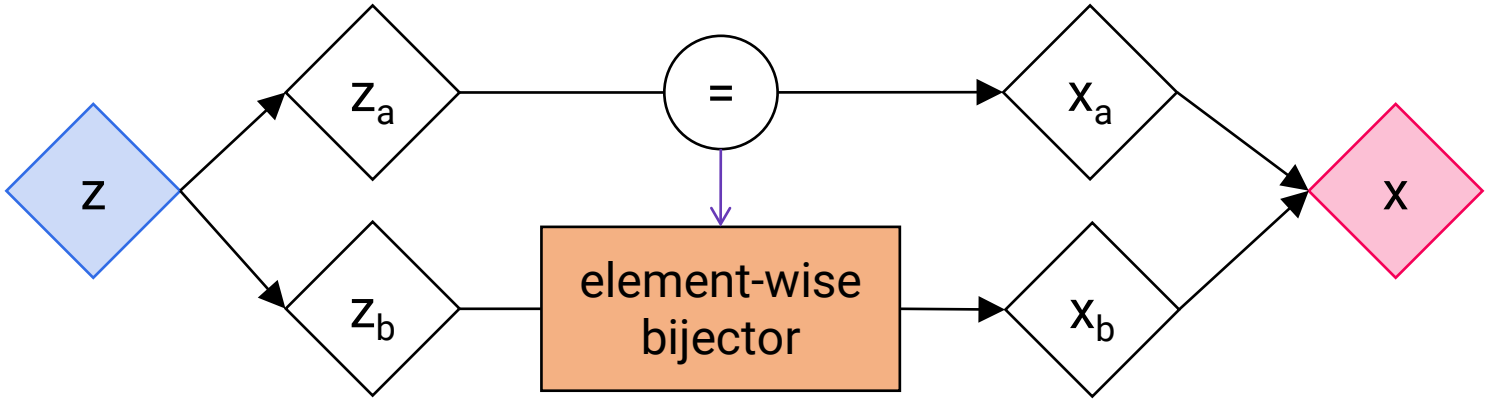
- Example: Design an invertible mapping $f_\theta: \mathbb{R}^D \to \mathbb{R}^D$ **[*coupling layer*]**

$$x_{1:d} = z_{1:d}$$

$$x_{d+1:D} = g\left(z_{d+1:D};\ s_\theta\left(z_{1:d}\right)\right)$$



$$\frac{\partial x}{\partial z}$$



Density Estimation using Real NVP. L Dinh, et al. ICLR 2017

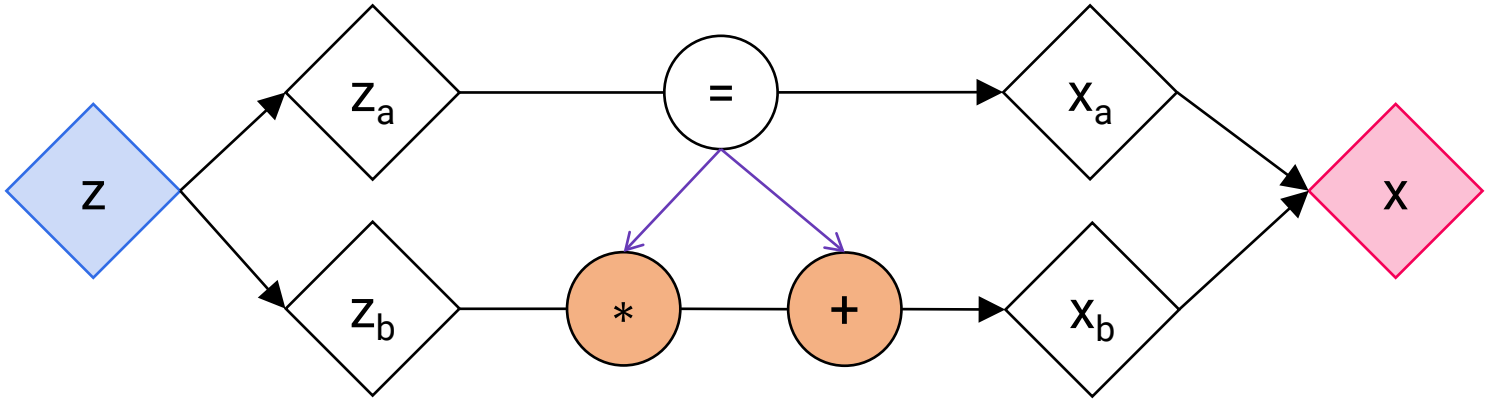
Normalizing Flows

- Example: Design an invertible mapping $f_\theta: \mathbb{R}^D \to \mathbb{R}^D$ **[*coupling layer*]**

$$x_{1:d} = z_{1:d}$$

Example: *affine coupling*

$$x_{d+1:D} = z_{d+1:D} \odot \exp\left(s_\theta\left(z_{1:d}\right)\right) + t_\theta\left(z_{1:d}\right)$$

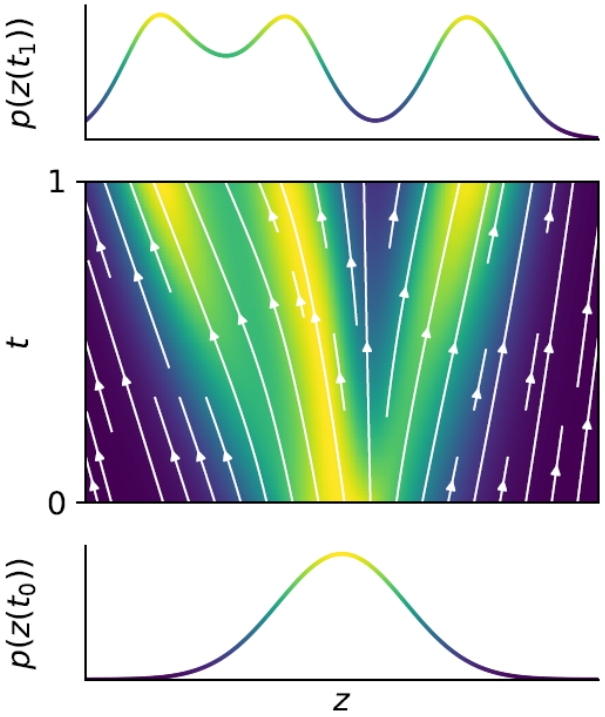


$$\log\left|\det \frac{\partial f_\theta^{-1}(\boldsymbol{x})}{\partial \boldsymbol{x}}\right| = -\sum_{i=1}^{D-d} s_\theta(\boldsymbol{z}_a)_i$$

Density Estimation using Real NVP. L Dinh, et al. ICLR 2017

Continuous Normalizing Flows

- Parameterize the mapping as **continuous dynamics**
  - Using neural ordinary differential equations (ODE)
- Can be interpreted as **infinite-depth** flows



Model the generative process with continuous dynamics:

$$z_0 \sim p(z_0)$$

$$\frac{\partial z_t}{\partial t} = f_\theta(z_t, t)$$

$$x = z_1 = z_0 + \int_{t_0}^{t_1} f_\theta(z_t, t)dt$$

To obtain the density we solve the initial value problem (IVP):

$$\log p(x) = \log p(z_0) - \int_{t_0}^{t_1} \mathrm{Tr}\left(\frac{\partial f_\theta}{\partial z_t}\right)dt \qquad (2)$$

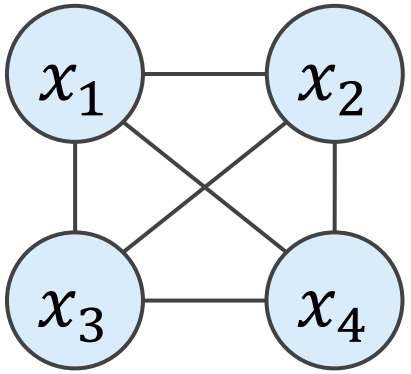
FFJORD: Free-form Continuous Dynamics for Scalable Reversible Generative Models. W Grathwohl, et al. ICLR 2019

Energy-Based Models

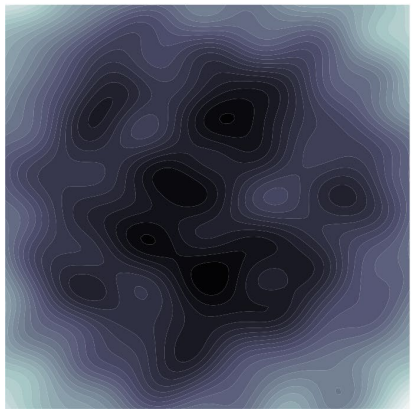
- Parametrize any scalar ***energy*** function $E_\theta(\boldsymbol{x})$
  - $-E_\theta(\boldsymbol{x})$ can be interpreted as a ***score*** function
    - i.e., it gives the "goodness of configurations"
- Intractable likelihood, hard to train
  - Because the partition function Z_θ is intractable
- Sampling is usually hard and slow (need MCMC)
- Highly flexible and expressive
  - Also some nice properties, e.g., compositionality

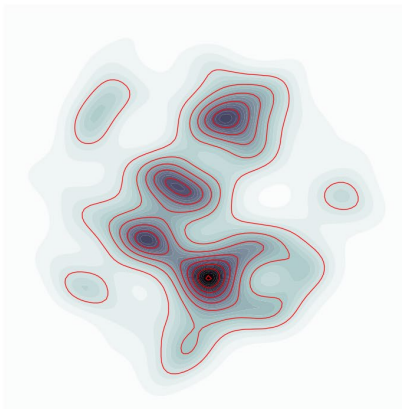
$$p_\theta(\boldsymbol{x}) = \frac{e^{-E_\theta(\boldsymbol{x})}}{Z_\theta}$$

$$Z_\theta = \int e^{-E_\theta(\boldsymbol{x})}\,\mathrm{d}\boldsymbol{x}$$

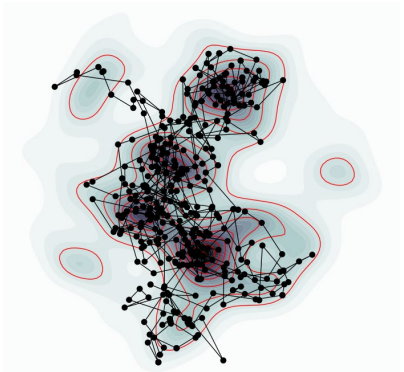


$-E_\theta(\boldsymbol{x})$





$p_\theta(\boldsymbol{x})$



A Tutorial on Deep Probabilistic Generative Models. Ryan P. Adams. Machine Learning Summer School 2018

Energy-Based Models

- There is renewed interest in them; but still have a long way to go

https://openai.com/blog/energy-based-models/

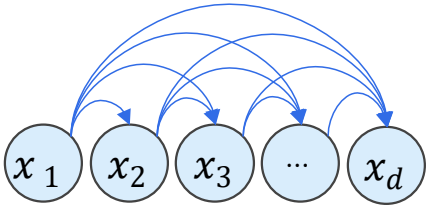
Implicit Generation and Modeling with Energy Based Models. Yilun Du, Igor Mordatch. NeurIPS 2019

Maximum Entropy Generators for Energy-Based Models. Rithesh Kumar, et al. Arxiv 2019

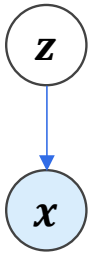
On the Anatomy of MCMC-based Maximum Likelihood Learning of Energy-Based Models. Erik Nijkamp, et al. CVPR 2019

Summary

Autoregressive Models



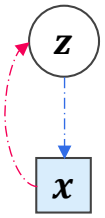
$p_\theta(\boldsymbol{x}) = \prod_{i=1}^{d} p_\theta(x_i | x_{<i})$



$p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x}|\boldsymbol{z}) p_\theta(\boldsymbol{z})\, \mathrm{d}\boldsymbol{z}$

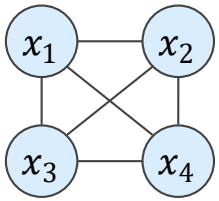
Latent Variable Models

Normalizing Flows



$\boldsymbol{x} = f_\theta(\boldsymbol{z})$

$p_\theta(\boldsymbol{x}) = p_\theta(\boldsymbol{z}) \left| \det \frac{\partial f_\theta^{-1}(\boldsymbol{x})}{\partial \boldsymbol{x}} \right|$



Energy-Based Models

$p_\theta(\boldsymbol{x}) = \frac{e^{-E_\theta(\boldsymbol{x})}}{\int e^{-E_\theta(\boldsymbol{x})}\, \mathrm{d}\boldsymbol{x}}$

	Sampling	**Likelihood**	**Training**	**Density Estimation**
AR	Slow	Exact	MLE	Straightforward
Flow	Fast*	Exact	MLE	Straightforward
LVM	Fast	Approx.	Approx. MLE	Importance Sampling
EBM	Very slow	Intractable	Approx. MLE, NCE, ...	Straightforward, Unnormalized

OUTLINE

- Unsupervised Anomaly Detection
- Likelihood-based DGMs for Density Estimation
- **Is Likelihood A Good OOD Measure?**

Do Deep Generative Models Know What They Don't Know? Eric Nalisnick, et al. ICLR 2019

Likelihood-based Models for OOD detection

- Collect a training dataset $\mathcal{D} = \{x_i\}_{i=1}^N$
- Train an AR/VAE/NF model: $\theta^* \approx \underset{\theta}{\operatorname{argmax}} \sum_i \log p_\theta(x_i)$
- Select a threshold δ
- Report x_+ as an anomaly if $\log p_{\theta^*}(x_+) < \delta$
- Problem solved???

Published as a conference paper at ICLR 2019

DO DEEP GENERATIVE MODELS KNOW WHAT THEY DON'T KNOW?

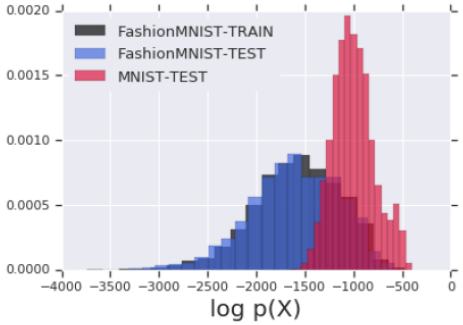
Eric Nalisnick[*†], **Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, Balaji Lakshminarayanan**[*]
DeepMind

ABSTRACT

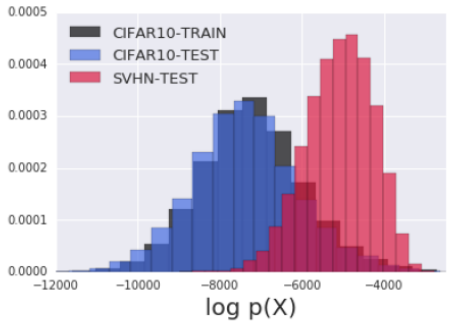
A neural network deployed in the wild may be asked to make predictions for inputs that were drawn from a different distribution than that of the training data. A plethora of work has demonstrated that it is easy to find or synthesize inputs for which a neural network is highly confident yet wrong. Generative models are widely viewed to be robust to such mistaken confidence as modeling the density of the input features can be used to detect novel, out-of-distribution inputs. In this paper we challenge this assumption. We find that the density learned by flow-based

Do DGMs Know What They Don't Know?

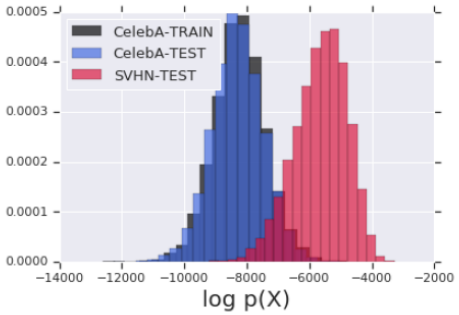
- After trained on dataset A, the model may report **higher** likelihoods for samples from dataset B ☹



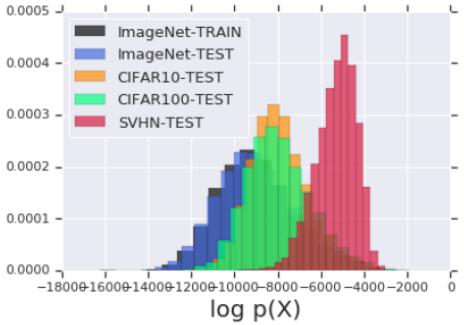
(a) Train on FashionMNIST, Test on MNIST



(b) Train on CIFAR-10, Test on SVHN



(c) Train on CelebA, Test on SVHN



(d) Train on ImageNet,
Test on CIFAR-10 / CIFAR-100 / SVHN

Figure 2: Histogram of Glow log-likelihoods for FashionMNIST vs MNIST (a), CIFAR-10 vs SVHN (b), CelebA vs SVHN (c), and ImageNet vs CIFAR-10 / CIFAR-100 / SVHN (d).

Do Deep Generative Models Know What They Don't Know?. Eric Nalisnick, et al. ICLR 2019

Conjecture (1): Not Bayesian Enough

- Models are learned by finding MLE $\theta^* \approx \underset{\theta}{\operatorname{argmax}} \sum_i \log p_\theta(x_i)$
  - This is frequentist's approach
  - Often result in over-confident likelihood estimates

- Bayesians say that
  we should treat θ as latent variable and perform **Bayesian inference**

$$p(x_+|\mathcal{D}) = \mathbb{E}_{p(\theta|\mathcal{D})}[p(x_+|\theta)]$$

  - Can be approximated by ensembles (multiply randomly initialized models)

- Variants:

$$\mathrm{WAIC}(x_+) \coloneqq \mathbb{E}_{p(\theta|\mathcal{D})}[\log p(x_+|\theta)] - \mathrm{Var}_{p(\theta|\mathcal{D})}[\log p(x_+|\theta)]$$

$$p(\theta|\mathcal{D}) \leftrightarrow p(\theta|\mathcal{D} \cup \{x_+\})$$

WAIC, but Why? Generative Ensembles for Robust Anomaly Detection. Hyunsun Choi, et al. Arxiv 2019
Bayesian Variational Autoencoders for Unsupervised Out-of-Distribution Detection. Anonymous, OpenReview 2019

Conjecture (2): Background Statistics

- Assume that an input is composed of two components
  - **Background**: characterized by population level background statistics
  - **Semantic**: characterized by patterns specific to the in-distribution data

- Fit a model $p_\theta(x)$ using in-distribution dataset $\mathcal{D}$
- Fit a background model $p_\phi(x)$ using perturbed data $\widetilde{\mathcal{D}}$
- Compute the likelihood ratio statistic

$$\mathrm{LLR}(x_+) = \log \frac{p_\theta(x_+)}{p_\phi(x_+)}$$

  - Which cancels out the likelihood for the background component
- Predict OOD if $\mathrm{LLR}(x_+)$ is small

Likelihood Ratios for Out-of-Distribution Detection. Jie Ren, et al. Arxiv 2019

Conjecture (3): Input Complexity

- Empirical observations: simple image → higher likelihood

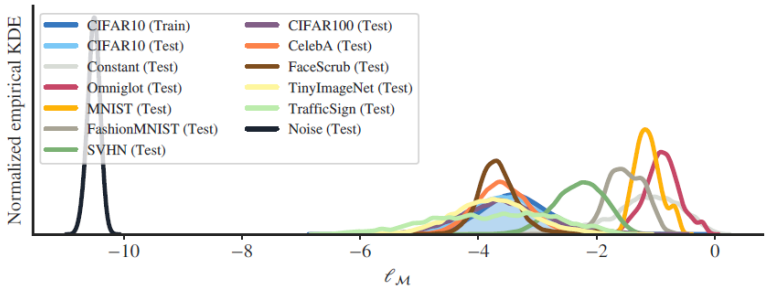


Figure 2: Log-likelihoods from a Glow model trained on CIFAR10. Qualitatively similar results are obtained for a PixelCNN++ model and when training with FashionMNIST.

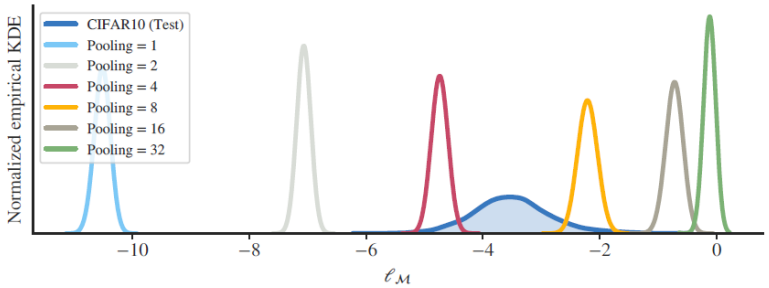


Figure 3: Pooled-image log-likelihoods obtained from a Glow model trained on CIFAR10. Qualitatively similar results are obtained for a PixelCNN++ model.

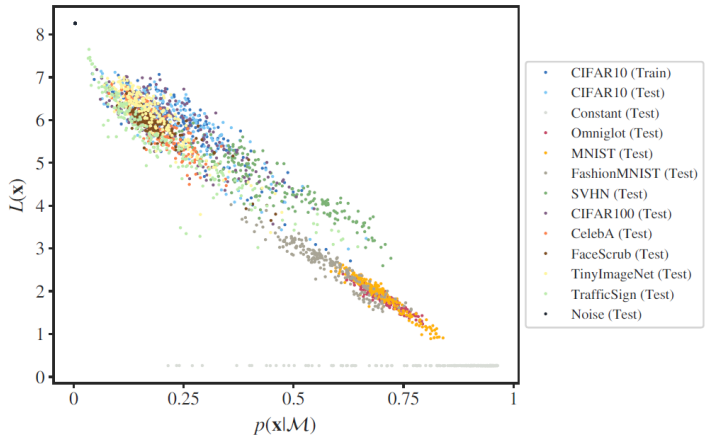


Figure 4: Normalized compressed lengths using a PNG compressor with respect to likelihoods of a PixelCNN++ model trained on CIFAR10 (for visualization purposes we here employ a sample of 200 images per data set). Qualitatively similar results are obtained for a Glow model and other compressors.

- Complexity-penalized likelihood

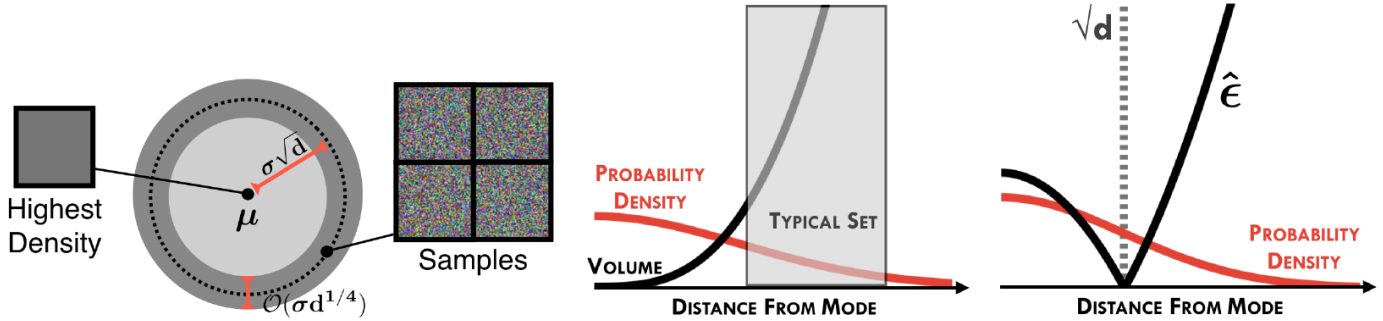
universal compressor

$$\mathcal{L}(x_+) := \log_2 p(x) - \log_2 p(x|\mathcal{M}_0)$$

Input Complexity and Out-of-distribution Detection with Likelihood-based Generative Models. Joan Serra, et al. Arxiv 2019

Conjecture (4): Likelihood ≠ Typicality

- A generative model will draw samples from its typical set
- Typical set may not necessarily intersect with regions of high probability density



(a) Gaussian Example (b) Illustration (c) Simulation

Figure 1: *Typical Sets*. Subfigure (a) shows the example of a Gaussian with its mean located at the high-dimensional all-gray image. Subfigure (b) shows how the typical set arises due to the nature of high-dimensional integration. The figure is inspired by Betancourt (2017)'s similar illustration.

- Test whether a batch of examples are in the typical set

$$\texttt{if } \left| \frac{1}{M} \sum_{m=1}^{M} -\log p(\tilde{\boldsymbol{x}}_m; \boldsymbol{\theta}) - \mathbb{H}[p(\mathbf{x}; \boldsymbol{\theta})] \right| = \hat{\epsilon} \leq \epsilon \texttt{ then } \widetilde{\boldsymbol{X}} \sim p(\mathbf{x}; \boldsymbol{\theta})$$

Detecting Out-of-Distribution Inputs to Deep Generative Models Using a Test for Typicality. Eric Nalisnick, et al. Arxiv 2019
Testing For Typicality with Respect to an Ensemble of Learned Distributions. Anonymous. OpenReview 2019

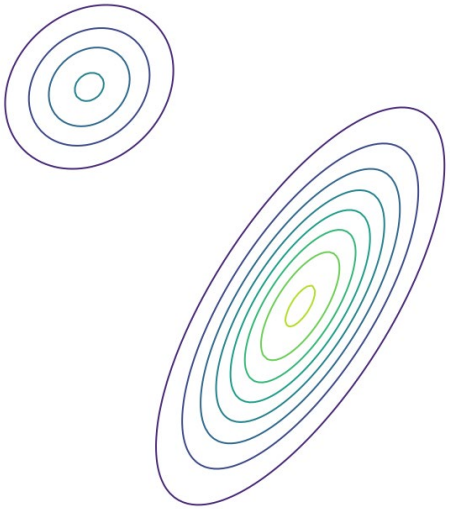
Conjecture (5): Model Misspecification & MLE

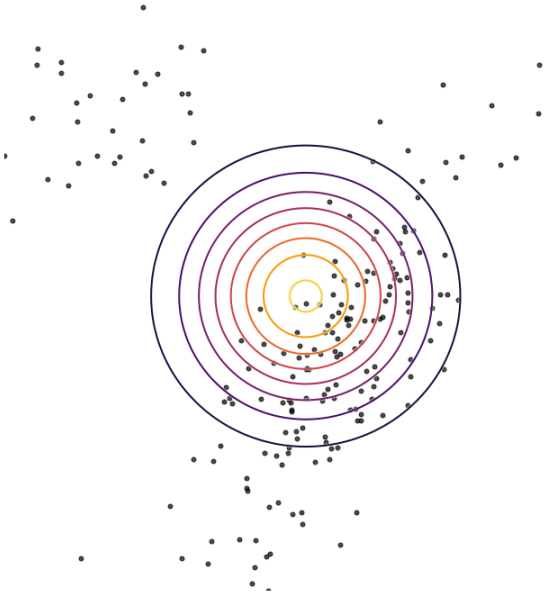
- Some generated images are not so realistic… Why?

$$\theta^* = \arg\min_{\theta} \mathbf{KL}\left(p^* \| p_\theta\right) \approx \arg\max_{\theta} \sum_{i=1}^{N} \log p_\theta\left(\boldsymbol{x}^{(i)}\right)$$

- For real data, models may be mis-specified: $p^* \notin \{p_\theta | \theta \in \Theta\}$

$p^*(\boldsymbol{x})$





$p_\theta(\boldsymbol{x})$

https://colinraffel.com/blog/gans-and-divergence-minimization.html

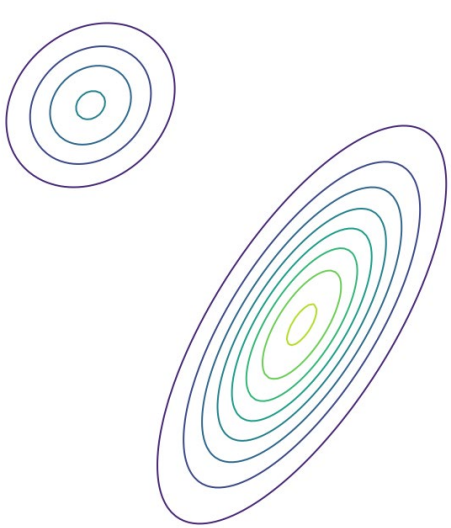
Unsupervised Out-of-Distribution Detection with Batch Normalization. Jiaming Song, et al. Arxiv 2019

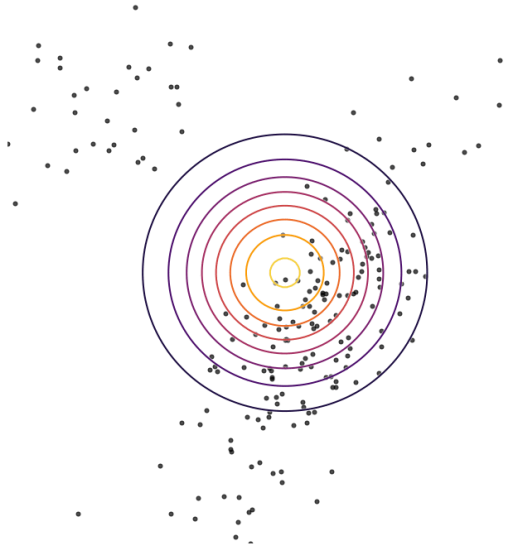
Conjecture (5): Model Misspecification & MLE

- Maybe other divergences are more appropriate for OOD detection?
  - **Reverse K**ullback–**L**eibler, **J**ensen–**S**hannon, Wasserstein, …
  - Adversarial training?

$$\min_{\theta} \mathbf{KL}\left(p_\theta \middle\| p^*\right) = \min_{\theta} \mathbb{E}_{\boldsymbol{x} \sim p_\theta}\left[\log \frac{p_\theta(\boldsymbol{x})}{p^*(\boldsymbol{x})}\right]$$

$p^*(\boldsymbol{x})$





$p_\theta(\boldsymbol{x})$

https://colinraffel.com/blog/gans-and-divergence-minimization.html

Summary

- DGMs are promising tools for OOD detection 
- Not as appealing as they looked at first 
  - May assign high likelihoods to OOD samples
- Several fixes have been suggested 
  - And they seem to work well on standard benchmarks
- Still an open problem 

Courses & Tutorials & Monographs

Courses

- Berkeley CS294-158: Deep Unsupervised Learning, Spring 2019. [link]
- Stanford CS 236: Deep Generative Models, Fall 2018-2019. [link]
- Edinburgh: Probabilistic Modelling and Reasoning, 2018/19. [link]

Tutorials

- Tutorial on Deep Generative Models. Shakir Mohamed and Danilo Rezende. UAI 2017 & CCN 2018. [slides 1][slides 2][video]
- A Tutorial on Deep Probabilistic Generative Models. Ryan P. Adams. MLSS 2018. [slides]
- Deep Generative Models. Aditya Grover and Stefano Ermon. IJCAI 2018. [slides]
- Planting the Seeds of Probabilistic Thinking: Foundations, Tricks and Algorithms. Shakir Mohamed. MLSS 2018 [slides][video]
- Variational Inference: Foundations and Innovations. David M. Blei. [slides]
- Deep Latent-Variable Models for Natural Language. Yoon Kim, et al. EMNLP 2018. [link]
- Variational Inference and Deep Generative Models. Wilker Aziz and Philip Schulz. ACL 2018. [link]

Monographs

- An Introduction to Variational Autoencoders. DP Kingma, Max Welling. Arxiv 2019. [link]
- Deep Latent Variable Models for Sequential Data. Marco Fraccaro. PhD Thesis 2018. [link]

References: AR

Image

- Generating Diverse High-Fidelity Images with VQ-VAE-2. Ali Razavi, et al. Arxiv 2019
- Generating High Fidelity Images with Subscale Pixel Networks and Multidimensional Upscaling. Jacob Menick, Nal Kalchbrenner. ICLR 2019
- Hierarchical Autoregressive Image Models with Auxiliary Decoders. JD Fauw, et al. Arxiv 2019
- PixelSNAIL: An Improved Autoregressive Generative Model. XI Chen, et al. ICML 2018
- Image Transformer. Niki Parmar, et al. ICML 2018

Sequence

- XLNet: Generalized Autoregressive Pretraining for Language Understanding. Z Yang, et al. Arxiv 2019
- MelNet: A Generative Model for Audio in the Frequency Domain. S Vasquez, M Lewis. Arxiv 2019
- Generating Long Sequences with Sparse Transformers. Rewon Child, et al. Arxiv 2019
- Scaling Autoregressive Video Models. Dirk Weissenborn, et al. Arxiv 2019
- Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. Z Dai, et al. ACL 2019
- Music Transformer: Generating Music with Long-Term Structure. CA Huang, et al. ICLR 2019
- The challenge of realistic music generation: modelling raw audio at scale. Sander Dieleman, et al. NeurIPS 2018

Graph

- GRAM: Scalable Generative Models for Graphs with Graph Attention Mechanism. W Kawai, et al. Arxiv 2019
- Generative Code Modeling with Graphs. M Brockschmidt, et al. ICLR 2019
- GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models. J You, et al. ICML 2018

References: LVM (1)

General

- Reweighted Expectation Maximization. Adji B. Dieng, John Paisley. Arxiv 2019
- Doubly Reparameterized Gradient Estimators for Monte Carlo Objectives. George Tucker, et al. ICLR 2019
- A Contrastive Divergence for Combining Variational Inference and MCMC. F Ruiz and MK Titsias. ICML 2019
- MIWAE: Deep Generative Modelling and Imputation of Incomplete Data Sets. PA Mattei, et al. ICML 2019
- Variational Autoencoder with Arbitrary Conditioning. Oleg Ivanov, et al. ICLR 2019
- DIVA: Domain Invariant Variational Autoencoders. Maximilian Ilse, et al. Arxiv 2019

Image

- Generating Diverse High-Fidelity Images with VQ-VAE-2. Ali Razavi, et al. Arxiv 2019
- BIVA: A Very Deep Hierarchy of Latent Variables for Generative Modeling. Lars Maaløe, et al. Arxiv 2019
- Diagnosing and Enhancing VAE Models. Bin Dai and David Wipf. ICLR 2019
- Multi-Object Representation Learning with Iterative Variational Inference. Klaus Greff, et al. ICML 2019
- A Hierarchical Probabilistic U-Net for Modeling Multi-Scale Ambiguities. Simon Kohl, et al. Arxiv 2019
- Spatial Broadcast Decoder: A Simple Architecture for Learning Disentangled Representations in VAEs. Nicholas Watters, et al. Arxiv 2019

References: LVM (2)

Sequence

- Learning Latent Dynamics for Planning from Pixels. Danijar Hafner, et al. ICML 2019
- Temporal Difference Variational Auto-Encoder. Karol Gregor, et al. ICLR 2019
- Stochastic Latent Actor-Critic: Deep Reinforcement Learning with a Latent Variable Model. Alex X. Lee, et al. Arxiv 2019
- Latent ODEs for Irregularly-Sampled Time Series. Yulia Rubanova, et al. Arxiv 2019
- ODE2VAE: Deep generative second order ODEs with Bayesian neural networks. Çağatay Yıldız, et al. Arxiv 2019
- Neural Ordinary Differential Equations. Ricky T. Q. Chen, et al. NeurIPS 2018
- Effective Estimation of Deep Generative Language Models. Tom Pelsmaeker, Wilker Aziz. Arxiv 2019
- A Hierarchical Latent Vector Model for Learning Long-Term Structure in Music. Adam Roberts, et al. ICML 2018
- STCN: Stochastic Temporal Convolutional Networks. Emre Aksan, Otmar Hilliges. ICLR 2019
- Z-Forcing: Training Stochastic Recurrent Networks. Anirudh Goyal, et al. NIPS 2017
- Sequential Neural Models with Stochastic Layers. Marco Fraccaro, et al. NIPS 2016

Graph / Set

- Graphite: Iterative Generative Modeling of Graphs. Aditya Grover, et al. ICML 2019
- Stochastic Blockmodels meet Graph Neural Networks. Nikhil Mehta, et al. ICML 2019
- PointFlow: 3D Point Cloud Generation with Continuous Normalizing Flows. Guandao Yang, et al. CVPR 2019
- Junction Tree Variational Autoencoder for Molecular Graph Generation. Wengong Jin, et al. ICML 2018

References: Flow

General

- Neural Spline Flows. Conor Durkan, et al. NeurIPS 2019
- Residual Flows for Invertible Generative Modeling. Ricky T. Q. Chen, et al. NeurIPS 2019
- FFJORD: Free-form Continuous Dynamics for Scalable Reversible Generative Models. Will Grathwohl, et al. ICLR 2019
- Integer Discrete Flows and Lossless Compression. Emiel Hoogeboom, et al. NeurIPS 2019
- Discrete Flows: Invertible Generative Models of Discrete Data. Dustin Tran, et al. NeurIPS 2019
- AlignFlow: Cycle Consistent Learning from Multiple Domains via Normalizing Flows. Aditya Grover, et al. Arxiv 2019

Image

- Flow++: Improving Flow-Based Generative Models with Variational Dequantization and Architecture Design. Jonathan Ho, et al. ICML 2019
- Glow: Generative Flow with Invertible 1x1 Convolutions. Diederik P. Kingma, Prafulla Dhariwal. NeurIPS 2018

Sequence

- Blow: a single-scale hyperconditioned flow for non-parallel raw-audio voice conversion. Joan Serrà, et al. Arxiv 2019
- Latent Normalizing Flows for Discrete Sequences. Zachary M. Ziegler and Alexander M. Rush. ICML 2019
- FloWaveNet: A Generative Flow for Raw Audio. Sungwon Kim, et al. ICML 2019
- VideoFlow: A Flow-Based Generative Model for Video. Manoj Kumar, et al. Arxiv 2019

Graph

- Graph Normalizing Flows. Jenny Liu, et al. Arxiv 2019
- GraphNVP: An Invertible Flow Model for Generating Molecular Graphs. Kaushalya Madhawa, et al. Arxiv 2019

References

OOD Detection

- WAIC, but Why? Generative Ensembles for Robust Anomaly Detection. Hyunsun Choi, et al. Arxiv 2019
- Likelihood Ratios for Out-of-Distribution Detection. Jie Ren, et al. Arxiv 2019
- Detecting Out-of-Distribution Inputs to Deep Generative Models Using a Test for Typicality. Eric Nalisnick, et al. Arxiv 2019
- Conditional Generative Models are not Robust. Ethan Fetaya, et al. Arxiv 2019
- Normalizing flows for novelty detection in industrial time series data. M Schmidt and M Simic. ICML 2019 INNF Workshop
- AdaFlow: Domain-Adaptive Density Estimator with Application to Anomaly Detection and Unpaired Cross-Domain Translation. Masataka Yamaguchi, et al. ICASSP 2019
- Anomaly Detection in Raw Audio Using Deep Autoregressive Networks. E Rushe, et al. ICASSP 2019

Applications in DB

- Selectivity Estimation with Deep Likelihood Models. Zongheng Yang, et al. Arxiv 2019
- Approximate Query Processing for Data Exploration using Deep Generative Models. S Thirumuruganathan, et al. Arxiv 2019

Thanks!

Questions?