# Deep Generative Modelling

Presenter:

Kurra Bavanya Choudhry

18075034
BTech CSE Part-IV

# Core goal of Generative modelling

- Let's say we have an observed data $D$, for eg: images of cats.
- $D$ can be thought of as a sample from probability distribution $p_{\text{data}}$ (true distribution).
- The goal of Generative modelling is to find an approximate probability distribution $p_\theta$ which is as close as possible to $p_{\text{data}}$ for an observed data $D$.
- Now we can generate unseen data by sampling from $p_\theta$.

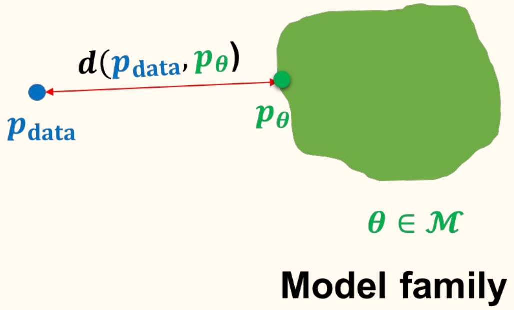
source: Karras et al. (2018)

**Surprisingly, this is not an image of a real child!**

This image was produced by the style based generator proposed by T Karras in 2018.

# 2 Tasks: Learning and Inference (1/2)

- **Learning (unsupervised and parametric):**
  - $p_\theta$ is picked from a family of model distributions $M$.
  - **<u>Optimization problem:</u>** $\min_{\theta \in M} d(p_{\text{data}}, p_\theta)$
  - where: d( · ) is a notion of distance between probability distributions.
  - **Three questions arise here:**
    - How to represent model family $M$?
    - What is the objective function $d(.)$?
    - How do we minimize $d(.)$?



# 2 Tasks: Learning and Inference (2/2)

- **Inference**
  - Generative models are used for 3 fundamental inference tasks:
    - **Density estimation:**
      - Obtaining $p_\theta(\mathbf{x})$ for a given datapoint $\mathbf{x}$.
    - **Sampling:**
      - Generating unseen data $\mathbf{x}_{\mathbf{new}} \sim p_\theta(\mathbf{x})$.
    - **Unsupervised representation extraction:**
      - Latent space feature representation vector $\boldsymbol{f}_{\mathbf{vector}}$ for a given datapoint $\mathbf{x}$.

There are many deep generative models (DGMs) present in the market today and we choose a model suitable for our inference requirements.

# Generative approach to classification task

- **<u>Problem:</u>** Detect an image as cat or dog):

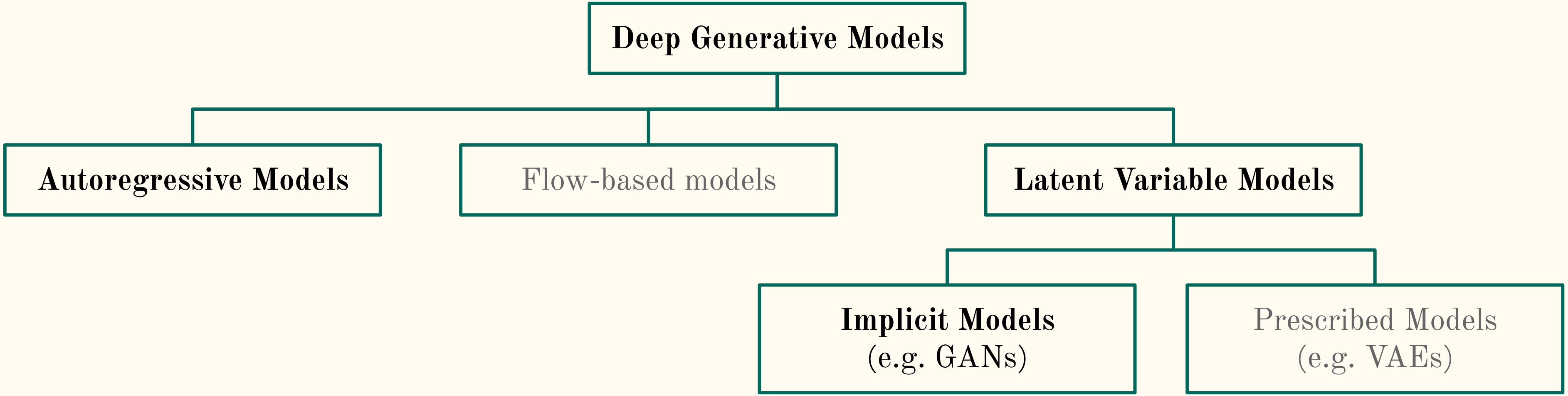
- **<u>Solution:</u>**
  - Obtain $p_{\theta}$ for $D_{cat}$ and $p_{\theta 1}$ for $D_{dog}$.
  - where: $D_{cat} \cup D_{dog} = D$ & $D_{cat} \cap D_{dog} = \phi$.
  - For test datapoint $\mathbf{x}_{\mathbf{test}}$ , find $p_{\theta}(\mathbf{x}_{\mathbf{test}})$ and $p_{\theta 1}(\mathbf{x}_{\mathbf{test}})$
    (to find out which distribution is it most probably came from.)

- **How is it different from discriminative approach?**
  - Doesn't find a decision boundary that separates data points of different classes.
  - Estimates P(Y|X) using Bayes Rule on P(X|Y) i.e $p_{\theta}(\mathbf{x})$ and P(Y), whereas discriminative models finds it directly from training data.

# Family of Deep Generative Models (DGMs)

- "**<u>Deep</u>**" in DGM refer to multiple layers of neurons with hidden variables.
- Generative modelling can be done without using neural networks (NNs).
- But still NNs are widely used to parameterize generative models because they are flexible and powerful.

**Deep Generative models can be expressed into the following three main groups:**



# Autoregressive Models (1/2)

- We fix an ordering of the variables $x_1, x_2, \dots, x_n$.
- The distribution for the $i$-th random variable depends on the values of all the preceding random variables in the chosen ordering $x_1, x_2, \dots, x_{i-1}$.
- Expressing the above assumption in graphical form:


(source: website)

- $p_\theta(\mathbf{x}) = \prod_{i=1}^{n} p_\theta(\mathbf{x}_i | \mathbf{x}_{<i})$ where $\mathbf{x}_{<i} = [x_1, x_2, \dots, x_{i-1}]$

# Autoregressive Models (2/2)

- KL divergence is taken as the objective function $d(.)$.

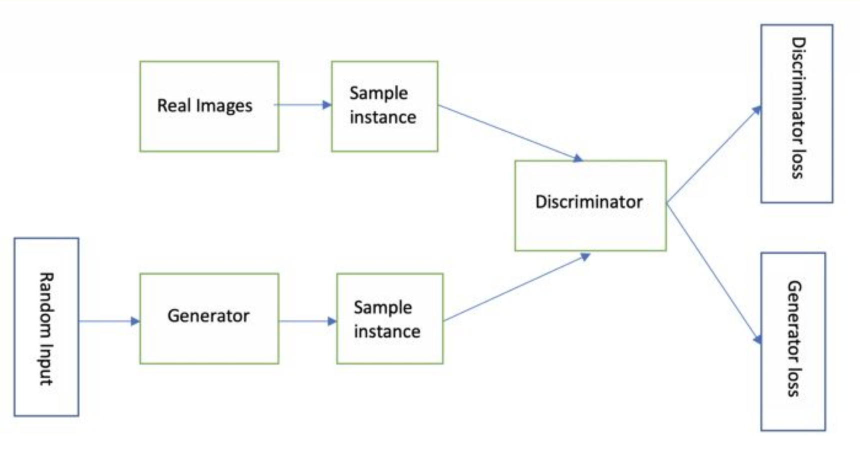
  $$\min_{\theta \in M} d_{\mathrm{KL}}(p_{\mathrm{data}}, p_\theta) = \max_{\theta \in M} \left(\Sigma_{\mathrm{x} \in D} \log(p_\theta(\mathbf{x}))\right)$$

  **i.e we pick the model parameters $\theta \in M$ that maximize the log-probability of the observed data points in $D$.**
- The model parameters $\theta \in M$ are optimized using mini-batch gradient descent.
- For density estimation and unseen data generation, sequential sampling procedure is followed.
- Hence both the inference tasks are expensive.
- These models do not learn unsupervised representation of data.
- Hence, they can't be used to get $\boldsymbol{f}_{\textbf{vector}}$ for a given datapoint.

# Generative Adversarial Networks (GANs)

- It has two parts:
  - **Generator**
    - Learns to generate plausible data.
    - It's output become negative training examples for the discriminator.
  - **Discriminator**
    - Learns to distinguish the generator's fake data from real data.
    - Penalizes the generator for producing implausible results.

**GAN Architecture**




source: medium post

# Training in GANs

- **at Discriminator:**
    - Connects to two loss functions: Generator and Discriminator loss.
    - Ignores the generator loss and just uses the discriminator loss.
    - The discriminator loss penalizes the discriminator for misclassifying a real instance as fake or a fake instance as real.
    - Network's weights are updated through backpropagation.

- **at Generator:**
    - Takes random noise sampled from any probability distribution as input.
    - This noise is transformed into meaningful output.
    - The gradient of the Generator loss is used to change the weights of only the generator.

# Real life applications

- **Data Generation**
  - Art creation
  - Code generation
  - Text generation
  - Audio synthesis
  - Text to image generation
  - etc.
- **Measuring the plausibility of data**
  - Anomaly detection
    - If $p_\theta(\mathbf{x}^1) <<< p_\theta(\mathbf{x})$ for $\mathbf{x} \in D$, then $\mathbf{x}^1$ is an outlier.
  - Classification *(it was discussed in previous slides)*
  - etc.


source: post link, generated using cc12m_1_cfg model developed by @RiversHaveWings

# THANK YOU