Autoregressive Models

Hao Dong

Peking University

Autoregressive Models

- Definition of Autoregressive Models (Ⅰ)
- Challenge of Generative Models
- Definition of Autoregressive Models (Ⅱ)
- Learning and Inference of Autoregressive Models
- Examples of Autoregressive Models
  - Fully Visible Sigmoid Belief Network (FVSBN)
  - Neural Autoregressive Density Estimation (NADE)
  - Masked Autoencoder for Distribution Estimation (MADE)
  - PixelRNN, PixelCNN, WaveNet…. (Next Lecture)

- Definition of Autoregressive Models (Ⅰ)
- Challenge of Generative Models
- Definition of Autoregressive Models (Ⅱ)
- Learning and Inference of Autoregressive Models
- Examples of Autoregressive Models
  - Fully Visible Sigmoid Belief Network (FVSBN)
  - Neural Autoregressive Density Estimation (NADE)
  - Masked Autoencoder for Distribution Estimation (MADE)
  - PixelRNN, PixelCNN, WaveNet….

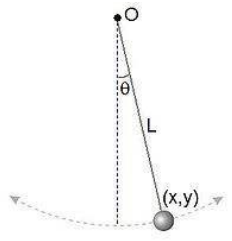
Definition of Autoregressive Models

The term ***autoregressive*** originates from the literature on time-series models where observations from the previous time-steps are used to predict the value at the current time step.

Put simply, an autoregressive model is merely a feed-forward model which predicts future values from past values:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \ldots + \phi_p y_{t-p} + \varepsilon_t \quad , \quad \varepsilon_t \sim \mathrm{N}(0, \sigma^2)$$

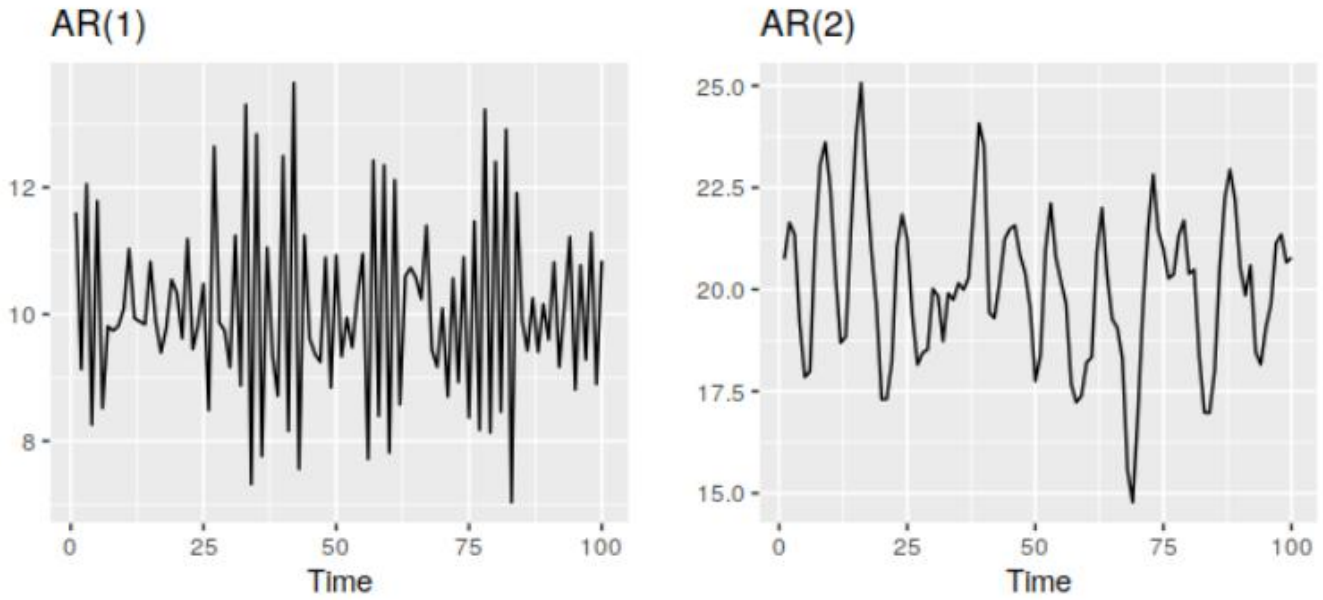
y_i could be:
The specific stock price of day i…
The amplitude of a simple pendulum at period i…
Or any variable that depends on its preceding values!



Definition of Autoregressive Models

Autoregressive Models have a strong ability in data representation.

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \ldots + \phi_p y_{t-p} + \varepsilon_t \,, \varepsilon_t \sim \mathrm{N}(0, \sigma^2)$$



Two examples of data from autoregressive models with a few different parameters.
Left: AR(1) with $y_t = 18 - 0.8 y_{t-1} + \varepsilon_t$. Right: AR(2) with $y_t = 8 + 1.3 y_{t-1} - 0.7 y_{t-2} + \varepsilon_t$.

Definition of Autoregressive Models

Autoregressive Models have a strong ability in data representation.

- Regression
- Generation
- Prediction

Recap: Statistical Generative Models

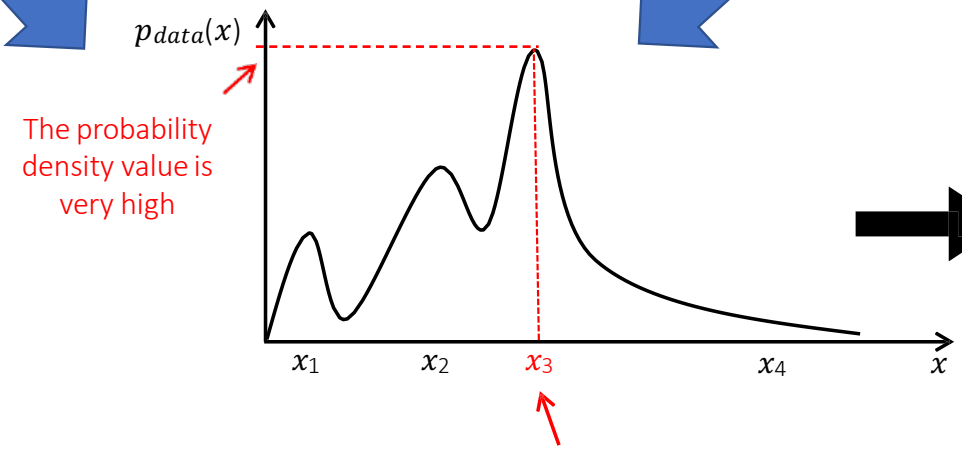
dataset $\mathcal{D}$

+

Model family, loss function, optimisation algorithm, etc.

Learning

Prior Knowledge



Sampling from p(x) **generates** new images

x_3 is a 64x64x3 high dimensional vector representing a woman with blonde hair.

北京大學
PEKING UNIVERSITY

- Definition of Autoregressive Models (Ⅰ)
- Challenge of Generative Models
- Definition of Autoregressive Models (Ⅱ)
- Learning and Inference of Autoregressive Models
- Examples of Autoregressive Models
  - Fully visible Sigmoid Belief Network (FVSBN)
  - Neural Autoregressive Density Estimation (NADE)
  - Masked Autoencoder for Distribution Estimation (MADE)
  - PixelRNN, PixelCNN, WaveNet….

Recap: Challenge of Generative Models

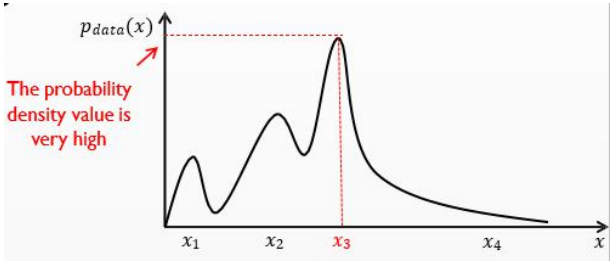
■ Compactness

Suppose x_1, x_2, x_3 are binary variables. $P(x_1, x_2, x_3)$ can be specified with $(2^3 - 1) = 7$ parameters

What about a 28×28 black/white digit image?

$2^{28 \times 28} - 1 = 2^{784} - 1 \approx 10^{236}$ parameters!

But with only 10 peaks of $0, 1, 2, \ldots 9$



$O(2^n)$

Main challenge: distributions over high dimensional objects is actually very sparse!!

Too many possibilities! ⟶ Main idea: **write as a product of simpler terms**

Recap: Challenge of Generative Models

- Solution #1: Factorisation

Definition of conditional probability:

$$P(x_1, x_2) = P(x_1)\, P(x_2|x_1)$$

Product rule:

$$P(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} p_\theta(x_i|x_{<i})$$

Divide and conquer ! We can solve the joint distribution $P(\boldsymbol{x})$ by solving simpler conditional distributions $p_\theta(x_i|x_{<i})$ one by one

Still complex!!

It's hard to exactly modelling every conditional distribution

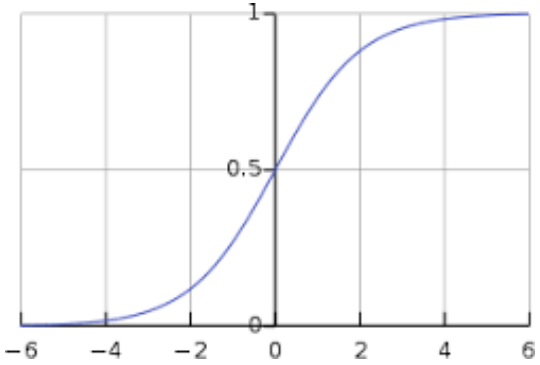
Can you tell the exact likelihood of the next pixel (noted as a red point) conditioned on the given pixels?

Recap: Challenge of Generative Models

Solution #2a: use simple **functions** to form the conditionals

$P(x_4|x_1, x_2, x_3) \approx sigmoid\ (W_1x_1 + W_2x_2 + W_3x_3)$

- Only requires storing 3 parameters
- Relationship between x_4 and (x_1, x_2, x_3) could be too simple



sigmoid function

Sigmoid function can be used to binarize the output

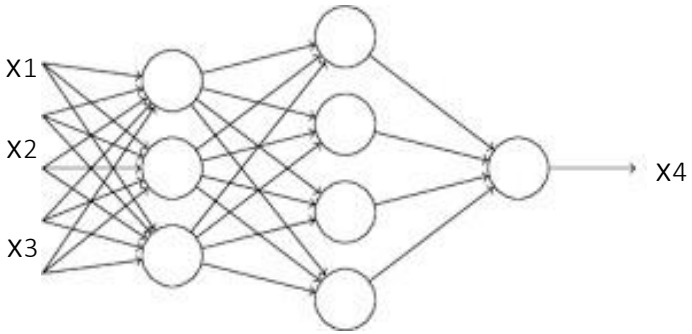
Solution #2b: use more complex functional form **Neural network**

$Z_1 = f_{11}(x_1, x_2, x_3), Z_2 = f_{12}(x_1, x_2, x_3), Z_3 = f_{13}(x_1, x_2, x_3)$

$Y_1 = f_{21}(Z_1, Z_2, Z_3), Y_2 = f_{22}(Z_1, Z_2, Z_3), Y_3 = f_{23}(Z_1, Z_2, Z_3)\ldots$

$P(x_4|x_1, x_2, x_3) \approx sigmoid\ (W_1Y_1 + W_2Y_2 + W_3Y_3 \ldots)$

- More flexible
- More parameters
- More powerful on fitting data



Neural network

finally, it's possible to model the data distributions!

- Definition of Autoregressive Models (Ⅰ)
- Challenge of Generative Models
- Definition of Autoregressive Models (Ⅱ)
- Learning and Inference of Autoregressive Models
- Examples of Autoregressive Models
  - Fully Visible Sigmoid Belief Network (FVSBN)
  - Neural Autoregressive Density Estimation (NADE)
  - Masked Autoencoder for Distribution Estimation (MADE)
  - PixelRNN, PixelCNN, WaveNet….

Definition of Autoregressive Models

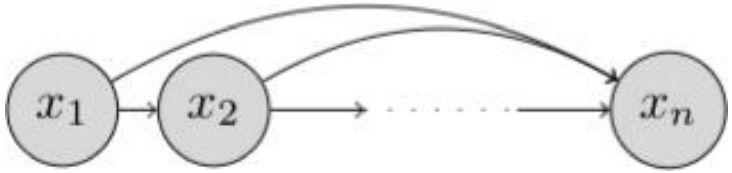
However, by defining $\widehat{x_i}$, the output of step i , as a random variable that follows the conditional distribution based on previous inputs $x_1, x_2 \ldots x_{i-1}$, we get the probability model, which can present the joint distribution of $p_\theta(x_1, x_2, \ldots x_n)$

- **Key idea:** Decompose the joint distribution as a product of tractable conditionals

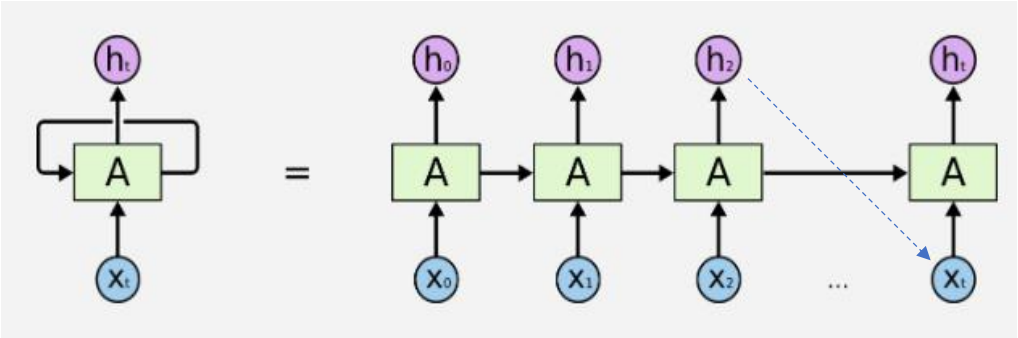
$$\widehat{x_i} = p_\theta(x_i | x_1, x_2, \ldots, x_{i-1})$$

$$p_\theta(\boldsymbol{x}) = \prod_{i=1}^{n} p_\theta(x_i | x_1, x_2, \ldots, x_{i-1}) = \prod_{i=1}^{n} p_\theta(x_i | x_{<i})$$

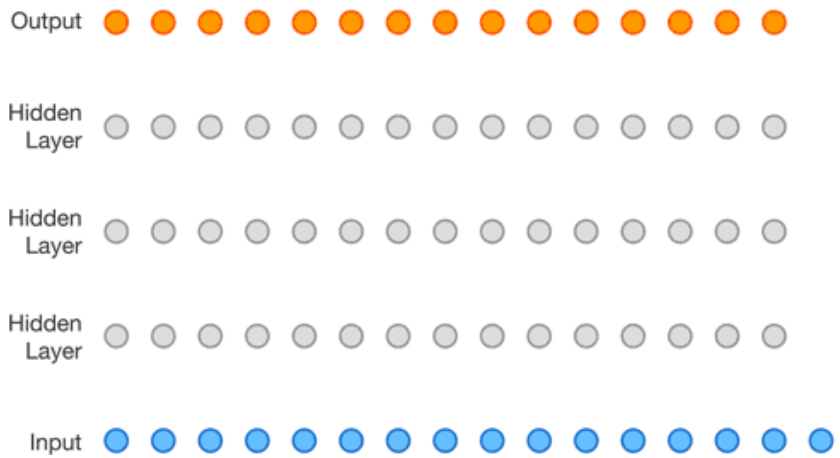
- **Graph model**: Directed, fully-observed Bayesian network



Definition of Autoregressive Models



Obligatory RNN diagram. Source: Chris Olah.



WaveNet animation. Source: Google DeepMind.

Relationship with RNN:

Like an RNN, an autoregressive model's output h_t,at time t depends on not just x_t, but also $x_1, x_2, \ldots, x_{i-1}$ from previous time steps.

However, unlike an RNN, the previous $x_1, x_2, \ldots, x_{i-1}$ are not provided via some hidden state: they are given just as an input to the model.

北京大學
PEKING UNIVERSITY

- Definition of Autoregressive Models (Ⅰ)
- Challenge of Generative Models
- Definition of Autoregressive Models (Ⅱ)
- Learning and Inference of Autoregressive Models
- Examples of Autoregressive Models
  - Fully Visible Sigmoid Belief Network (FVSBN)
  - Neural Autoregressive Density Estimation (NADE)
  - Masked Autoencoder for Distribution Estimation (MADE)
  - PixelRNN, PixelCNN, WaveNet….

Learning and Inference of Autoregressive Models

- **Learning** to maximise the model log-likelihood over the dataset !

$$\min_{\theta \in M} d_{KL}(p_{data}, p_\theta) = \min_{\theta \in M} \mathrm{E}_{\mathrm{x} \sim \mathrm{p_{data}}}[\log \mathrm{p_{data}}(\mathrm{x}) - \log \mathrm{p}_\theta(\mathrm{x})] \propto \max_{\theta \in M} E_{x \sim p_{data}} \log p_\theta(x)$$

$$\max_\theta \log p_\theta(D) = \sum_{x \in \mathrm{D}} \log p_\theta(\boldsymbol{x}) = \sum_{x \in D} \sum_{i=1}^{n} \log p_\theta(x_i | x_{<\mathrm{i}})$$

Tractable:
The distribution is simple enough to be modeled explicitly.

Tractable conditionals make conditional distribution learning meaningful, and thus allow for **exact likelihood evaluation.**

Learning and Inference of Autoregressive Models

- **Inference** samples each variable of one data from estimated conditional distributions step by step, until the whole data is generated.

Ancestral sampling:

A process of producing samples from a probabilistic model.

First sample variables which has **no conditional** constraints using their prior distribution. $x_1 \sim p_\theta(x_1)$

Then sample child variables using **conditional distribution** based on their parents and repeat so on. $x_2 \sim p_\theta(x_2|x_1)$.....

The attribute of Autoregressive Models that directly model and output distributions allows for ancestral sampling.

Learning and Inference of Autoregressive Models

Differences between **Autoregressive models (AR), VAE and GAN**:

GAN model doesn't define any distribution, it adapts discriminator to learn the data distribution implicitly. $P(X, Z) = P(X|Z)P(Z)$

VAE model believes the data distribution is too complex to model directly, thus it tries to learn the distribution by defining an intermediate distribution and learning the map between the defined simple distribution to the complex data distribution. $P(X, Z) = P(X|Z)P(Z)$

AR model on the one hand assumes that the data distribution can be learned directly (tractable), then it define its outputs as conditional distributions to solve the generation problem by directly modeling each conditional distribution.

Learning and Inference of Autoregressive Models

北京大學 PEKING UNIVERSITY

Conclusion:

1. Using complex networks, each step Autoregressive Models output an approximated complex conditional distribution $\widehat{x_i} = p_\theta(x_i|x_1, x_2, \dots, x_{i-1})$

2. Taking in the previous inputs $x_1, x_2, \dots, x_{i-1}$ and the next input x_i by sampling previous estimated conditional distribution $\widehat{x_i}$, Autoregressive Model is able to generate all conditional distributions iteratively

$x_1 \sim P_\theta(x_1)$, $x_2 \sim P_\theta(x_2|x_1)$, $x_3 \sim P_\theta(x_3|x_1, x_2)$, … $x_n \sim P_\theta(x_n|x_1, \dots, x_{n-1})$

3. Product rule makes sure the generated data that made up of sampled result x_i from each step follows the data distribution.

$$(x_1, x_2, \dots, x_n) \sim \prod_{i=1}^{n} p_\theta(x_i|x_{<i})$$

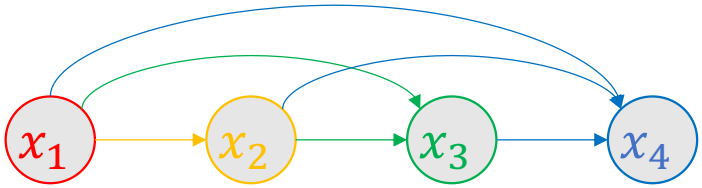
北京大學
PEKING UNIVERSITY

- Definition of Autoregressive Models (Ⅰ)
- Challenge of Generative Models
- Definition of Autoregressive Models (Ⅱ)
- Learning and Inference of Autoregressive Models
- Examples of Autoregressive Models
  - Fully Visible Sigmoid Belief Network (FVSBN)
  - Neural Autoregressive Density Estimation (NADE)
  - Masked Autoencoder for Distribution Estimation (MADE)
  - PixelRNN, PixelCNN, WaveNet….

- Definition of Autoregressive Models (Ⅰ)
- Challenge of Generative Models
- Definition of Autoregressive Models (Ⅱ)
- Learning and Inference of Autoregressive Models
- Examples of Autoregressive Models
  - Fully Visible Sigmoid Belief Network (FVSBN)
  - Neural Autoregressive Density Estimation (NADE)
  - Masked Autoencoder for Distribution Estimation (MADE)
  - PixelRNN, PixelCNN, WaveNet….

Fully Visible Sigmoid Belief Network (FVSBN)

- the fully visible sigmoid belief network **without any hidden units** is denoted FVSBN.



FVSBN

The conditional variables $x_i \mid x_1, \ldots, x_{i-1}$ in FVSBN are Bernoulli with parameters.

Some conditionals are too complex. So FVSBN assume:

$$\hat{x_i} = p(x_i = 1 | x_1, x_2, \ldots x_{i-1}) = f_i(x_1, x_2, \ldots, x_{i-1}; \alpha^i)$$

$$= \sigma(\alpha_0^{(i)} + \alpha_1^{(i)} x_1 + \cdots + \alpha_{i-1}^{(i)} x_{i-1})$$

- σ denotes the **sigmoid function**
  $\alpha^i = \{\alpha_0^{(i)}, \alpha_1^{(i)}, \ldots, \alpha_{i-1}^{(i)}\}$ denotes the parameters
- The conditional for variable xi requires i parameters, and hence the total number of parameters in the model is given by $\sum_{i=1}^{n} i = O(n^2) \ll O(2^n)$

Gan Z , Henao R , Carlson D , et al. Learning **Deep Sigmoid Belief Networks** with Data Augmentation[C]// Artificial Intelligence and Statistics (AISTATS). 2015.

FVSBN Example

- Suppose we have a dataset D of handwritten digits (binarised MNIST)

- Each image has n = 28×28x1 = pixels. Each pixel can either be black (0) or white (1).
- We want to learn a probability distribution $p(x) = p(x_1, \dots, x_{784})$ over $x \in \{0,1\}^{784}$ such that when $x \sim p(x)$, x looks like a digit.
- Idea: define a FVSBN model , then pick a good one based on training data D. (more on that later)

FVSBN Example

- We can pick an ordering, i.e., order variables (pixels) from top-left (x_1) to bottom-right (x_{784}).
- Use product rule factorisation without loss of generality:

$$p(x_1, \cdots, x_{784}) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2) \cdots p(x_{784} \mid x_1, \cdots, x_{783})$$

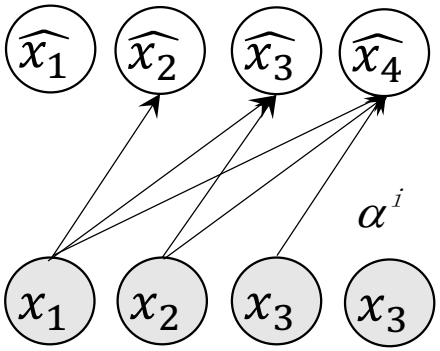
- FVSBN model assume: (less parameters)

$$\widehat{x_i} = p(x_i = 1 | x_1, x_2, \ldots x_{i-1}) = f_i(x_1, x_2, \ldots, x_{i-1}; \alpha^i)$$

$$= \sigma(\alpha_0^{(i)} + \alpha_1^{(i)} x_1 + \cdots + \alpha_{i-1}^{(i)} x_{i-1})$$

- Note: This is a **modelling assumption**. We are using a logistic regression to predict next pixel distribution based on the previous ones. Called autoregressive.

FVSBN Example



binary hidden variables

- How to **evaluate** $p(x_1,\cdots,x_{784})$ i.e. **density estimation**? Multiply all the conditionals (factors)

  In the above example:

$$p(x_1=0,x_2=1,x_3=1,x_4=0)$$
$$=p(x_1=0)p(x_2=1|x_1=0)p(x_3=1|x_1=0,x_2=1)p(x_4=0|x_1=0,x_2=1,x_3=1)$$
$$=(1-\widehat{x_1})\times\widehat{x_2}\times\widehat{x_3}\times(1-\widehat{x_4})$$

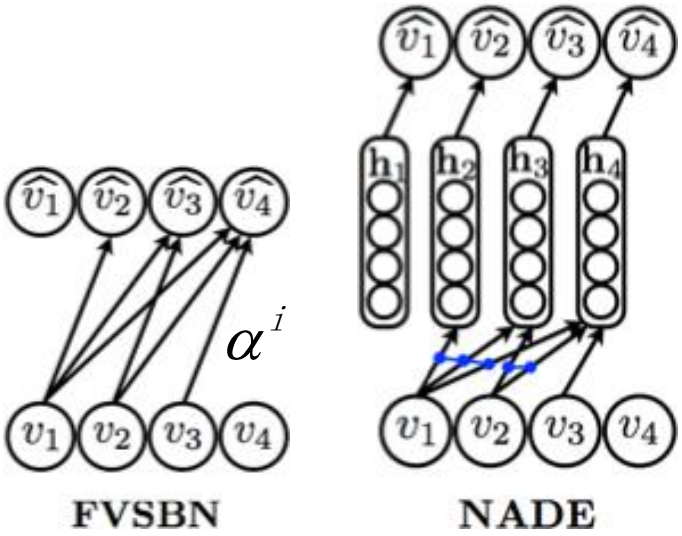
- How to **sample** from $p(x_1,\cdots,x_{784})$?

1. Sample $\overline{x_1}\sim p(x_1)$ $(np.random.choice([1,0],p=[\widehat{x_1},1-\widehat{x_1}]))$
2. Sample $\overline{x_2}\sim p(x_2|x_1=\overline{x_1})$
3. Sample $\overline{x_3}\sim p(x_3|x_1=\overline{x_1},x_2=\overline{x_2})$

...

- Definition of Autoregressive Models (Ⅰ)
- Challenge of Generative Models
- Definition of Autoregressive Models (Ⅱ)
- Learning and Inference of Autoregressive Models
- Examples of Autoregressive Models
  - Fully Visible Sigmoid Belief Network (FVSBN)
  - Neural Autoregressive Density Estimation (NADE)
  - Masked Autoencoder for Distribution Estimation (MADE)
  - PixelRNN, PixelCNN, WaveNet….

NADE: Neural Autoregressive Density Estimation



Improve FVSBN: use one hidden layer neural network instead of logistic regression

$$\boldsymbol{h_i} = \sigma(A_i \boldsymbol{x_{<i}} + \boldsymbol{c_i})$$

$$\hat{\boldsymbol{x}}_{\boldsymbol{i}} = p(x_i | x_1, x_2 \dots x_{i-1}; \underbrace{A_i, \boldsymbol{c_i}, \boldsymbol{\alpha_i}, b_i}_{\text{parameters}}) = \sigma(\boldsymbol{\alpha_i h_i} + b_i)$$

$x_i = v_i$

Tie weights are shared to *reduce the number of parameters* and *speed up computation*
(see blue dots in the figure)

$$\mathbf{h}_2 = \sigma\left(\underbrace{\begin{pmatrix} \vdots \\ w_1 \\ \vdots \end{pmatrix}}_{A_2} x_1\right) \quad \mathbf{h}_3 = \sigma\left(\underbrace{\begin{pmatrix} \vdots & \vdots \\ w_1 & w_2 \\ \vdots & \vdots \end{pmatrix}}_{A_3} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}\right) \quad \mathbf{h}_4 = \sigma\left(\underbrace{\begin{pmatrix} \vdots & \vdots & \vdots \\ w_1 & w_2 & w_3 \\ \vdots & \vdots & \vdots \end{pmatrix}}_{A_3} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}\right)$$

Uria B, Côté M A, Gregor K, et al. Neural autoregressive distribution estimation[J]. The Journal of Machine Learning Research, 2016, 17(1): 7184-722

NADE: Neural Autoregressive Density Estimation

$$\boldsymbol{h_i} = \sigma(A_i \boldsymbol{x_{<i}} + \boldsymbol{c_i})$$

$$\widehat{\boldsymbol{x_i}} = p(x_i | x_1, x_2 \dots x_{i-1}; A_i, \boldsymbol{c_i}, \boldsymbol{\alpha_i}, b_i)$$

$$= f_i(x_1, x_2, \dots, x_{i-1}) = \sigma(\boldsymbol{\alpha_i h_i} + b_i)$$

$\boldsymbol{x_{<i}} \in R^{i-1}$, denotes the vector made of preceding $\boldsymbol{x}$s

$\boldsymbol{h_i} \in R^d$, denotes the hidden layer activations of the MLP

$$\theta_i = \{A_i \in R^{d \times (i-1)}, \boldsymbol{c_i} \in R^d, \boldsymbol{\alpha_i} \in R^d, bi \in R\}$$

are the set of parameters.

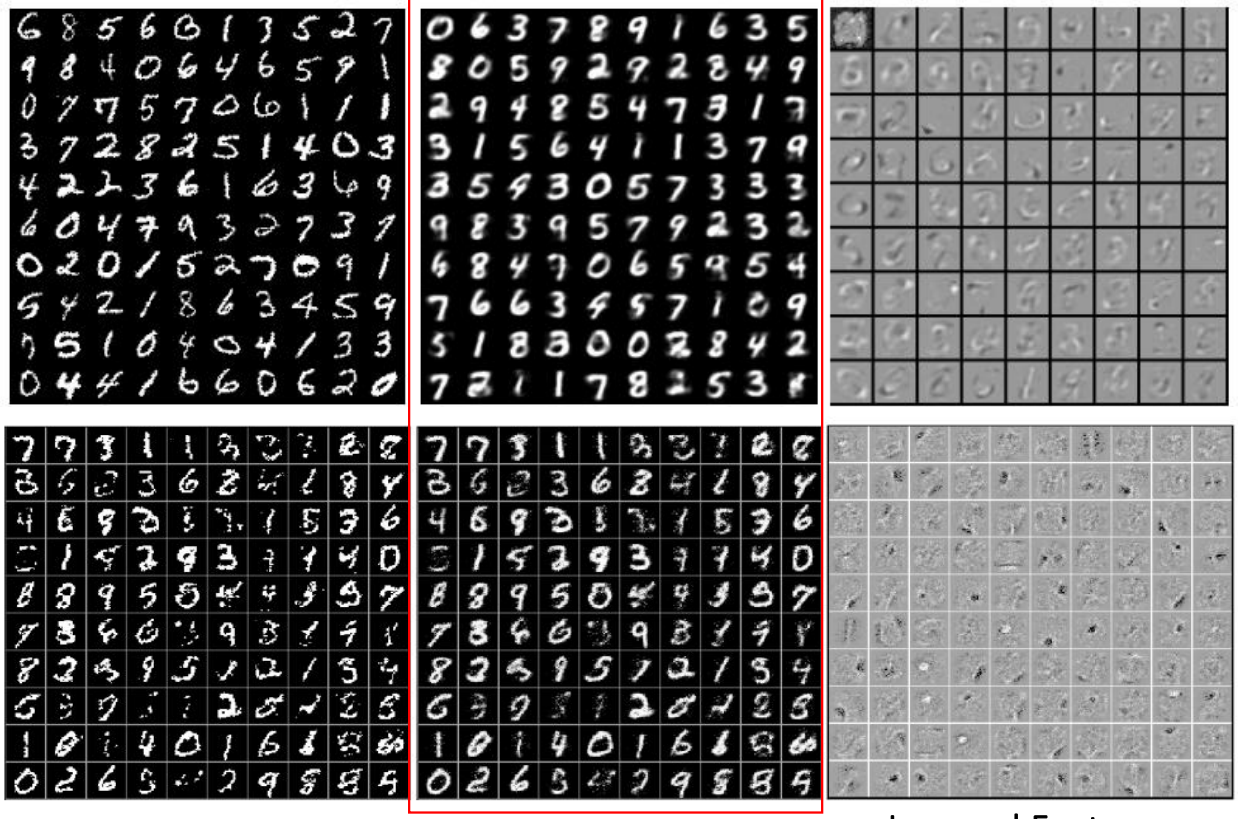
The total number of parameters in this model is dominated by the matrices $\{\boldsymbol{A_1}, \boldsymbol{A_2}, \dots, \boldsymbol{A_n}\}$ given by $O(n^2 d)$.

Sharing parameters: Tie weights are shared to reduce the number of parameters and speed up computation. —> $O(nd)$.

Generate samples

FVSBN

NADE



Learned Features

Performance on the MNIST dataset. (Left) Training data. **(Middle) Averaged synthesised samples**. (Right) Learned features at the bottom layer.

Generate other distributions

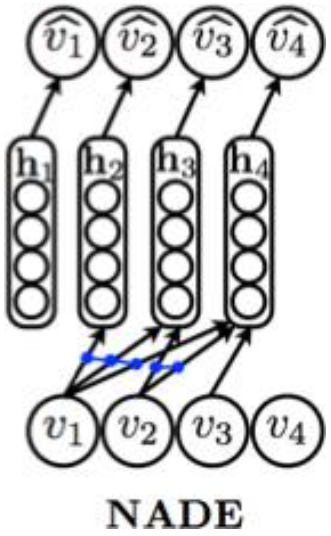
- How to model non-binary discrete random variables $V_i \in \{1, \ldots K\}$? E.g., pixel intensities varying from 0 to 255?

- One solution: Let $\hat{\mathbf{v}}_\mathbf{i}$ parameterise a categorical distribution

$$\boldsymbol{h_i} = \sigma(A_i \boldsymbol{v_{<i}} + \boldsymbol{c_i})$$

$$\hat{\mathbf{v}}_\boldsymbol{i} = p(v_i | v_1, \ldots, v_{i-1}) = (p_i^1, p_i^2 \ldots, p_i^k)$$

$$= softmax(U_i \boldsymbol{h_i} + \boldsymbol{b_i})$$



NADE

- **Softmax** generalises the sigmoid/logistic function $\sigma(\cdot)$ and transforms a vector of K numbers into a vector of K *probabilities* (non-negative, sum to 1).

$$softmax(\boldsymbol{a}) = softmax(a^1, \ldots, a^K) = (\frac{\exp(a^1)}{\sum_i \exp(a^i)}, \ldots, \frac{\exp(a^K)}{\sum_i \exp(a^i)})$$

Generate other distributions

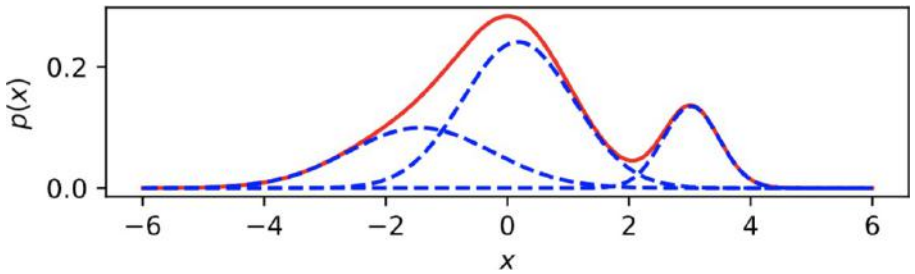
- How to model continuous random variables $V_i \in \mathbb{R}$? E.g., speech signals ?

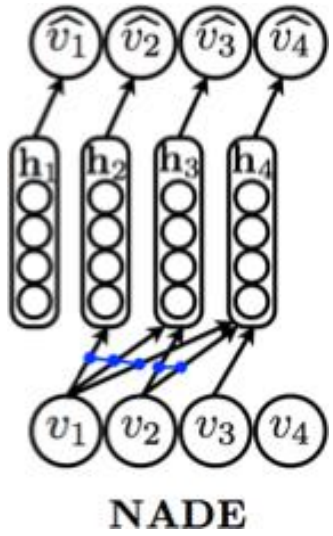
- One solution: Let $\hat{\mathbf{v}}_i$ parameterise a continuous distribution
  E.g., uniform mixture of K Gaussians

$$\boldsymbol{h_i} = \sigma(A_i \boldsymbol{v_{<i}} + \boldsymbol{c_i})$$

$$\hat{\mathbf{v}}_i = f(\boldsymbol{h_i}) = \left(\mu_i^1, \ldots, \mu_i^K, \sigma_i^1, \ldots \sigma_i^K\right)$$

$$p(v_i | v_1, \ldots, v_{i-1}) = \sum_{j=1}^{K} \frac{1}{K} \left(\mathcal{N}(v_i; \mu_i^j, \sigma_i^j)\right)$$





- Definition of Autoregressive Models (Ⅰ)
- Challenge of Generative Models
- Definition of Autoregressive Models (Ⅱ)
- Learning and Inference of Autoregressive Models
- Examples of Autoregressive Models
  - Fully Visible Sigmoid Belief Network (FVSBN)
  - Neural Autoregressive Density Estimation (NADE)
  - Masked Autoencoder for Distribution Estimation (MADE)
  - PixelRNN, PixelCNN, WaveNet….

Autoregressive model vs. Autoencoders

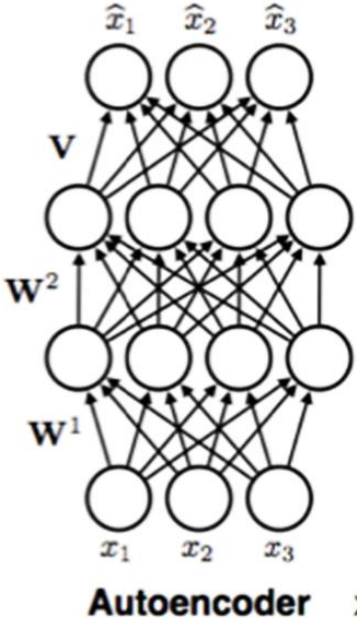
- FVSBN and NADE look similar to an **Autoencoder.**
- An **encoder** e(·), E.g., $e(x) = \sigma(W^2(W^1 x + b^1) + b^2)$
- A **decoder** such that $d\big(e(x)\big) \approx x$

  Binary:

$$\min_{W^1,W^2,b^1,b^2,V,c} \sum_{x \in D} \sum_{i} (-x_i \log \hat{x}_i - (1 - x_i) \log(1 - \hat{x}_i))$$

  Continuous:

$$\min_{W^1,W^2,b^1,b^2,V,c} \sum_{x \in D} \sum_{i} (x_i - \hat{x}_i)^2$$



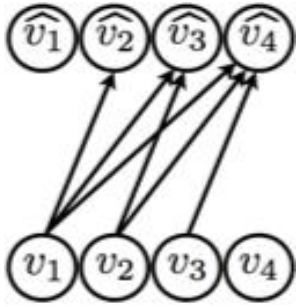
Autoencoder

- Encoder: feature learning
- A vanilla autoencoder is not a generative model: it does not define a distribution over x we can sample from to generate new data points.

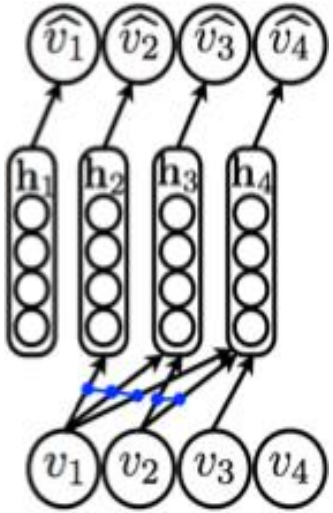
Autoregressive model vs. Autoencoders

- FVSBN and NADE look similar to an autoencoder.
- Can we get a generative model from an Autoencoder?

A dependency order constraint is required for Autoencoder to make it a Bayesian Network.

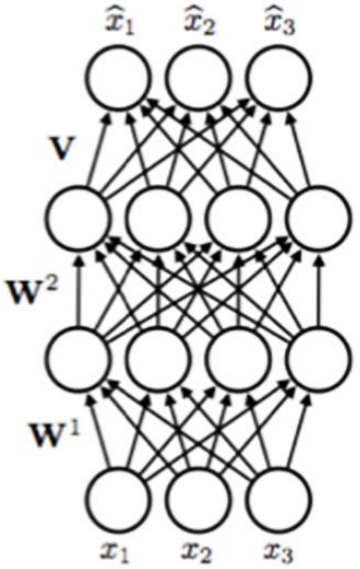


FVSBN



NADE

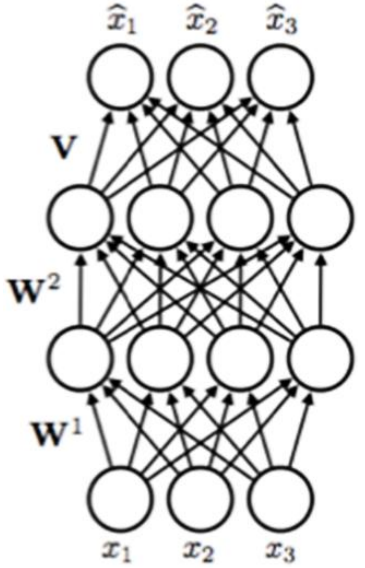
VS



Autoencoder

Autoregressive model vs. Autoencoders

- To get an autoregressive model from an Autoencoder,
- we need to make sure it corresponds to a valid Bayesian Network, so we need an ordering. If the ordering is 1,2,3, then:
  - $\widehat{x_1}$ cannot depend on any input x.
  - $\widehat{x_2}$ can only depend on x_1.
  - $\widehat{x_3}$ can only depend on x_1, x_2.
- **Bonus:** we can use a single neural network (with n outputs) to produce all the parameters. In contrast, NADE requires n passes. Much more efficient on modern hardware.



Autoencoder

MADE: Masked Autoencoder for Distribution Estimation

Use Masks to constraint dependency paths!

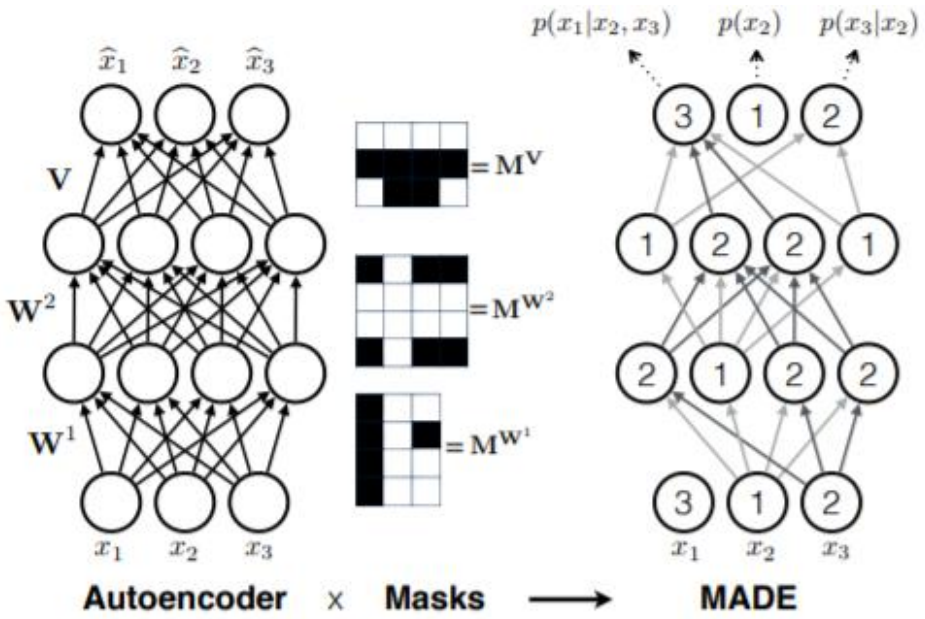
Each output unit is an **estimated distribution**, it only depends on the inputs with orderings that before its chosen ordering

With the order x_2,x_3,x_1:

1. $p(x_2)$ doesn't depends on any input

2. $p(x_3|x_2)$ depends on input x_2

3. $p(x_1|x_2, x_3)$ depends on input x_2 , x_3



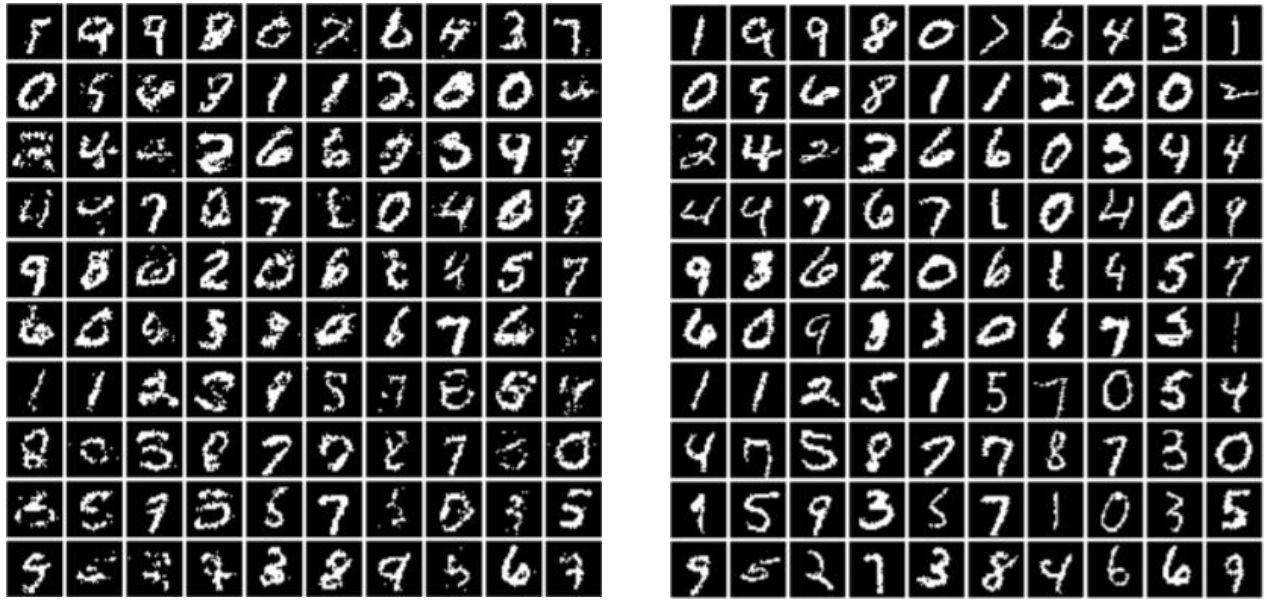
Masked Autoencoder

Mathieu M. Masked Autoencoder for Distribution Estimation[J]. 2015.

Generate samples

北京大學
PEKING UNIVERSITY

MADE

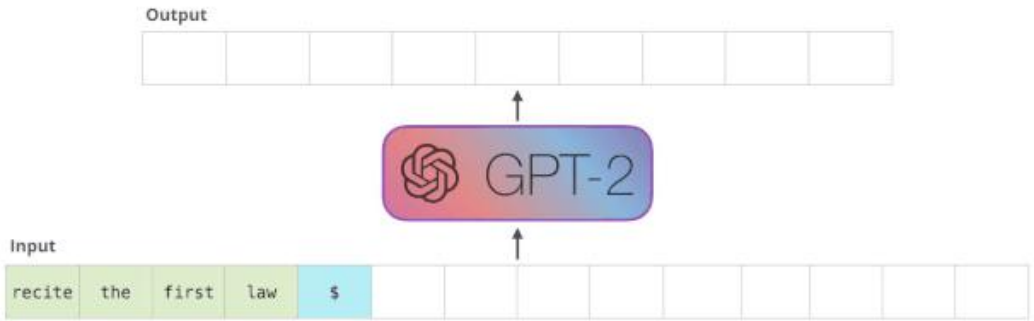


Performance on the MNIST dataset.
(Left) : Samples from a 2 hidden layer MADE
(Right): Nearest neighbor in binarized MNIST

Autoregressive Models in NLP

Natural language generation (NLG) is one of the important research fields of Artificial Intelligence, including text-to-text generation, meaning-to-text generation and image-to-test generation etc.

However, every time generating a word, it's always helpful to consider text that already generated! Thus Autoregressive Model is widely adopted in NLP.



Examples of powerful GPT-2 model generating "First Law Of Robotics"

Appendix A —Taxonomy of Generative Models

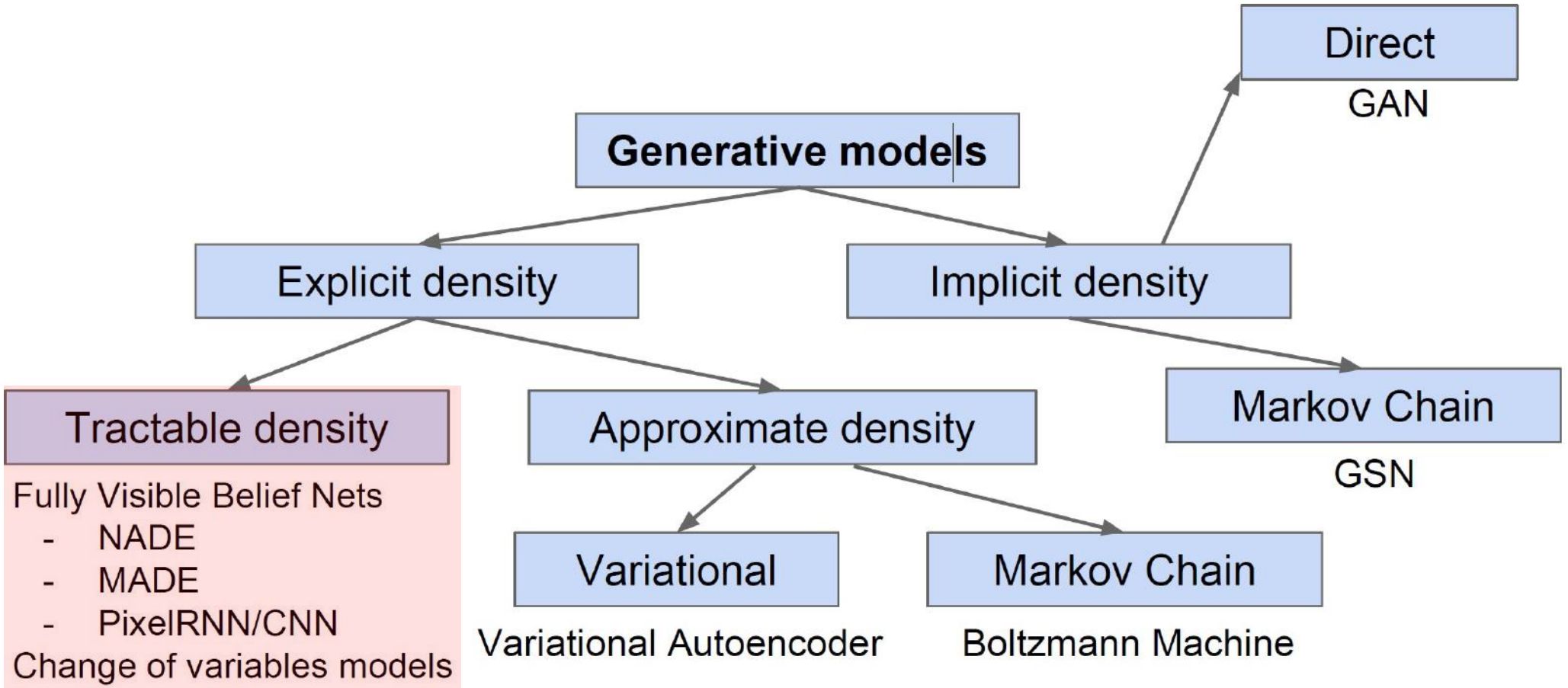


Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017

北京大學
PEKING UNIVERSITY

- Definition of Autoregressive Models (Ⅰ)
- Challenge of Generative Models
- Definition of Autoregressive Models (Ⅱ)
- Learning and Inference of Autoregressive Models
- Examples of Autoregressive Models
  - Fully Visible Sigmoid Belief Network (FVSBN)
  - Neural Autoregressive Density Estimation (NADE)
  - Masked Autoencoder for Distribution Estimation (MADE)
  - PixelRNN, PixelCNN, WaveNet…. (Next Lecture)

北京大學
PEKING UNIVERSITY

Thanks