


# Generative Model

Lingqiao Liu
University of Adelaide

# Outlines

- What is generative model?
- Auto-regressive models
  - Introduction
  - Recurrent Neural Network and other form of networks
  - Applications
- Generative Adversarial Network
  - Basic GAN
  - Other development
  - Application of GAN

# What is generative model

- Generative model: Machine learning models that can model the generative process of data
- What can generative model do?
  - Sampling a data sample
  - [optional] Used as the probability density function: probability evaluation
- Generative model vs. Discriminative model
  - Model P(x) or P(x|y)
  - Model P(y|x)
  - Can be connected through Bayes theorem

# Generative model vs. Discriminative Model

- Connection: Bayes theorem

$$P(y|x) = \frac{P(x|y)P(y)}{P(x)} = \frac{P(x|y)P(y)}{\sum_y P(x|y)P(y)}$$

- Remark
  - If P(x|y) is perfectly modelled, P(y|x) derived from the Bayes theorem is the optimal classifier
  - In real world, discriminative model usually performs better than generative model

# How to perform sampling?

- Sampling: generate a random variable from a given distribution
  - Example, generate a number from distribution [0.1,0.6,0.3]
  - Modern computer allows us generate from a limited number of distributions

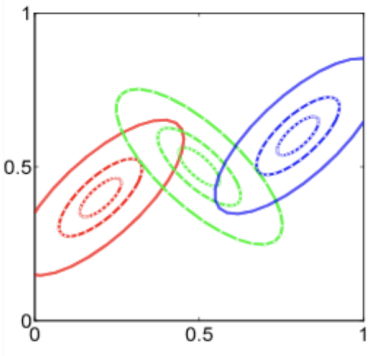
- Sampling from more complex distributions?
  - Decompose the generation into a sequence of simpler decision
  - Build a transforming function to convert an easy-to-sample random variable (vector) to a data sample

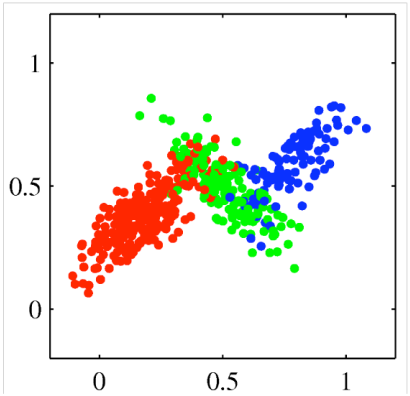
# Generative model we have learned

## Gaussian mixture model

- Likelihood $\Pr(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$ where

$$\sum_{k=1}^{K} \pi_k = 1, 0 \leq \pi_k \leq 1.$$



# Generative model we have learned

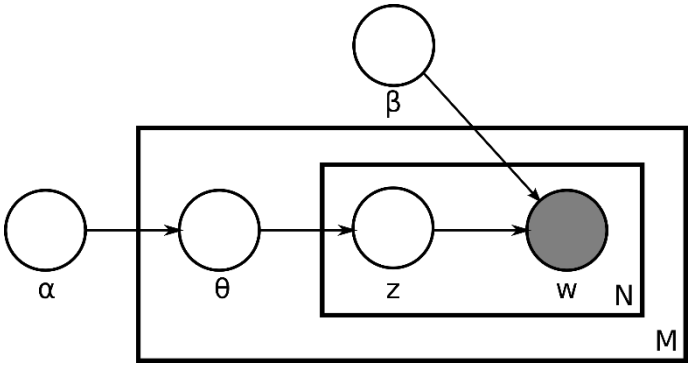
- Generative process:

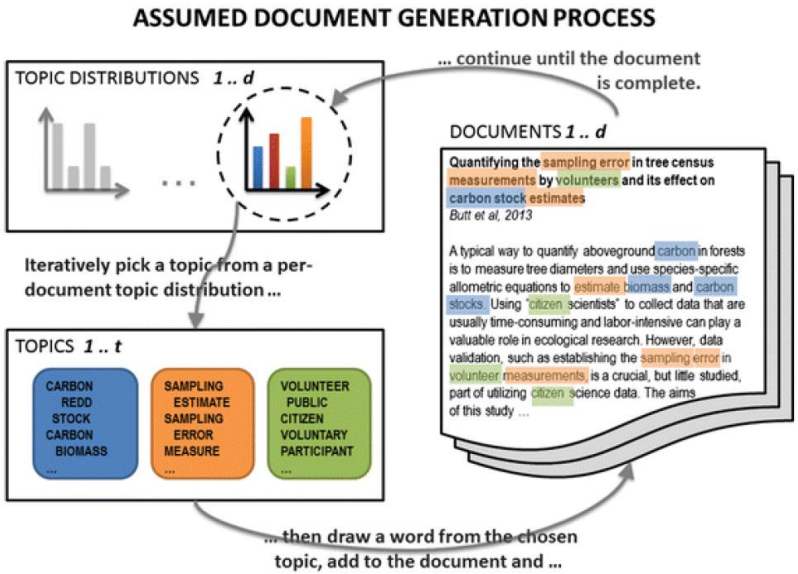
$\{\pi_k, \mu_k, \Sigma_k\}$

1. Randomly select the k-th Gaussian distribution according to
2. Randomly sample x from the selected Gaussian distribution

# Generative model we have learned

## Topic model




Generating a document by using LDA

- 1. Sample a topic distribution
- 2. From the distribution, sample a topic
- 3. From the topic, sample a word
- 4. repeat 2-3 to sample N words in a document

# Better model? Complex data?

- Challenges
  - More general-purpose models
  - Ability to model complex internal structures

- Deep generative models
  - Make use of the powerful modelling capability of deep neural network
  - Active research direction in Machine learning
    - Generative Adversarial Network (GAN)
    - Variational Autoencoder
    - Flow-based methods
    - Autoregressive model

# Autoregressive Model

- Theoretical foundation

$$P(x_1, x_2, x_3, \cdots, x_k) = P(x_1)P(x_2|x_1)P(x_3|x_1, x_2)\cdots P(x_k|x_1, \cdots, x_k - 1)$$

Chain rule to factorize the joint probability

The factorization order is not unique

$$\begin{aligned} P(x_1, x_2, x_3) &= P(x_1)P(x_2|x_1)P(x_3|x_1, x_2) \\ &= P(x_2)P(x_1|x_2)P(x_3|x_1, x_2) \end{aligned}$$

# Autoregressive Model

- Key idea: model $P(x_t|x_{1:t-1})$
- Especially natural for model a sequence
- But also apply to nonsequential data, e.g. images
- $P(x_t|x_{1:t-1})$ essentially build a predictive model to predict next time value based on the existing historical observations
- $P(x_t|x_{1:t-1})$ can be implemented in various ways

# Recurrent neural network

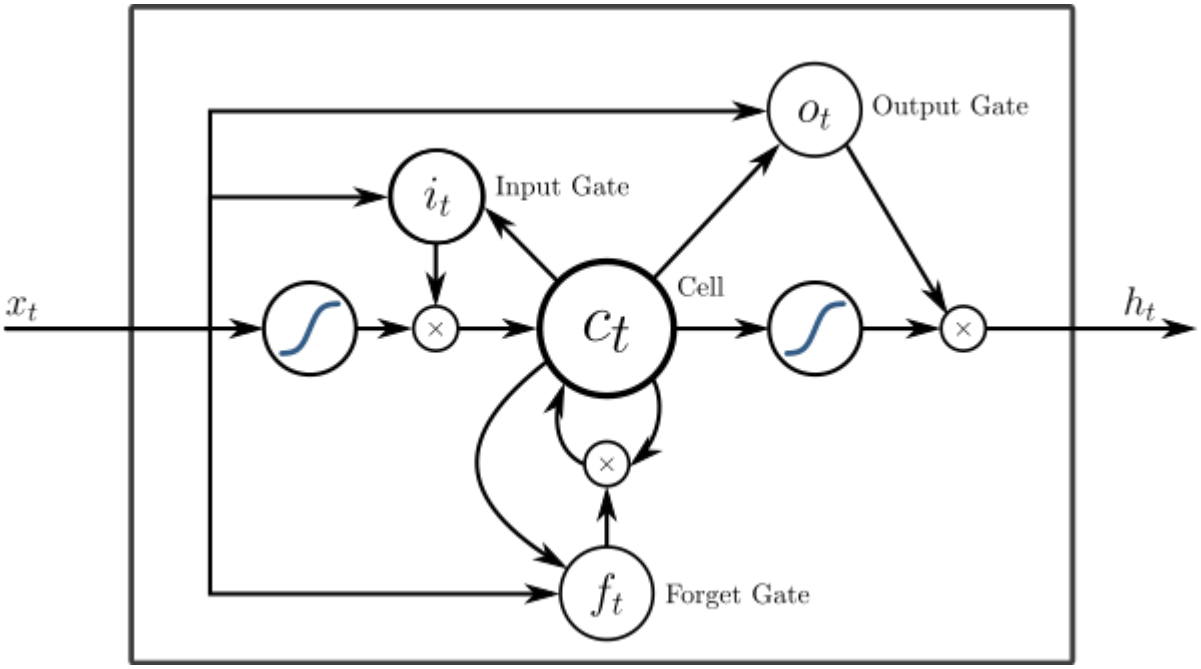
- Recurrent neural network

$$h_t = f(h_{t-1}, x_t)$$
$$P(x_t|x_{1:t-1}) = P(x_t|h_t, x_t)$$

  - Avoid the need of processing all the historical observations for each time step, but keep track of the state variables.
  - For f, it takes the previous output as the input to the network
  - Many possible implementations for f.
  - In the simplest case, f is a MLP (Vanilla RNN).
  - Model design of f involves mapping functions with gates

# Recurrent neural network

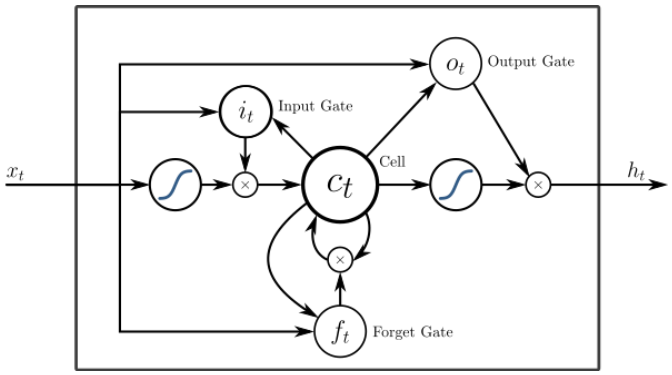
- Example: (Long short-term memory) LSTM



# Recurrent neural network

- Example: LSTM

- Gates enable more flexible state update in RNN
  - Input gate: ability to ignore the current input
  - Forget gate: ability to reset the state



# Training auto-regressive model

- Most straightforward approach:

Construct pairs $([x_0, x_1, \cdots, x_{t-1}], x_t)$

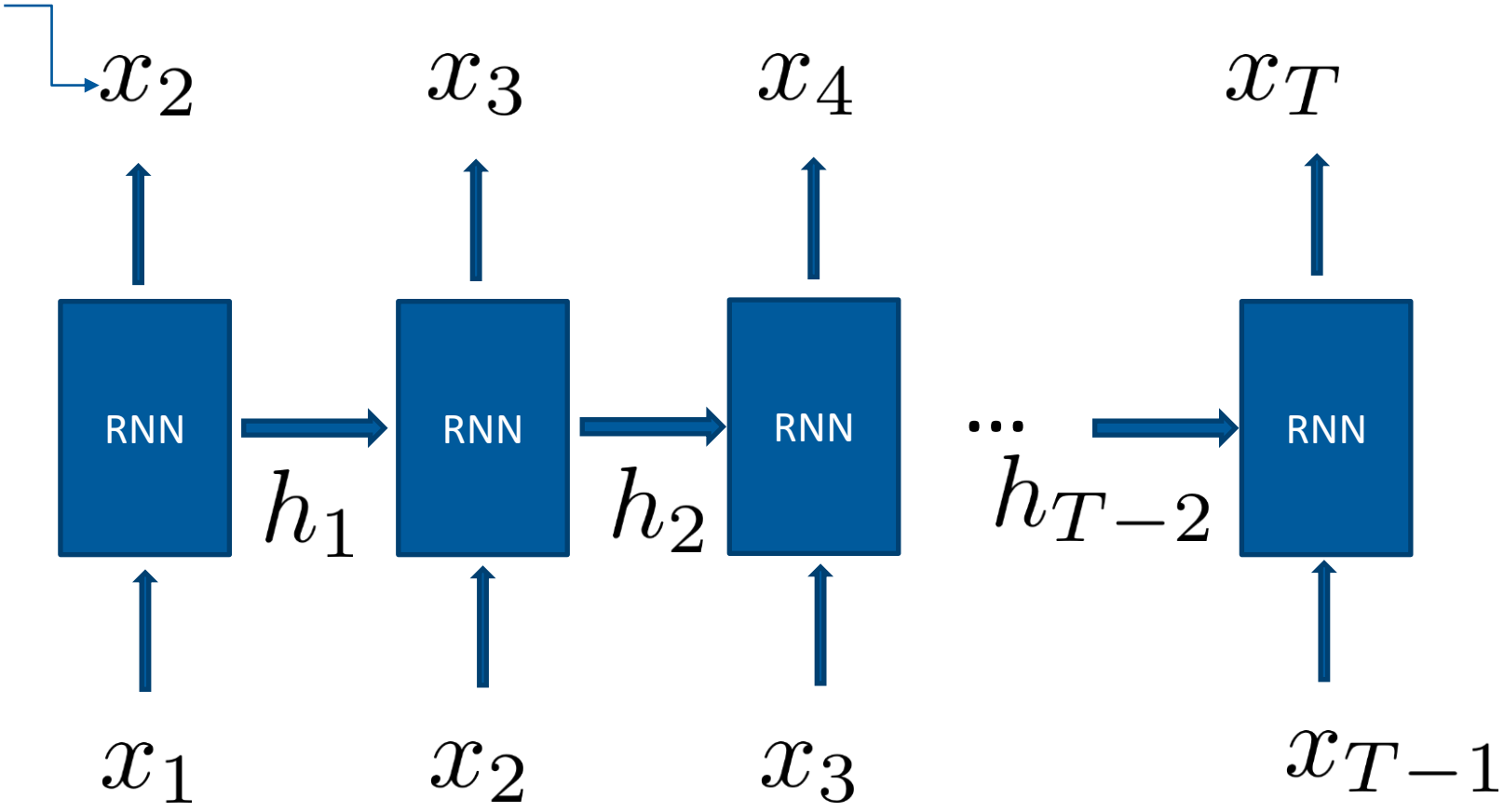
  - The input is with different length
  - The input needs to be sequentially processed by RNN

$$h_t = f(h_{t-1}, x_t)$$
$$P(x_t|x_{1:t-1}) = P(x_t|h_t, x_t)$$

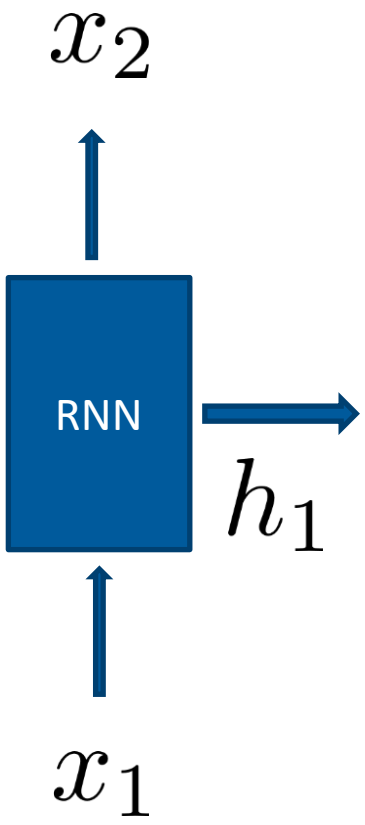
  - Need a lot of storages to store all the training samples
  - A lot of redundancy calculation

# A more efficient formulation: Sequence-to-Sequence mapping

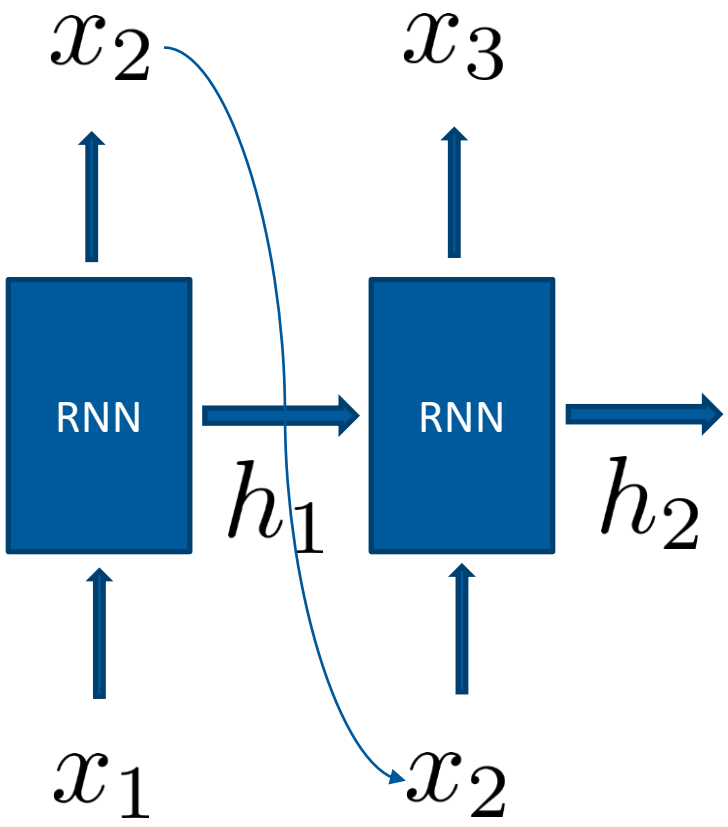
Prediction targets:



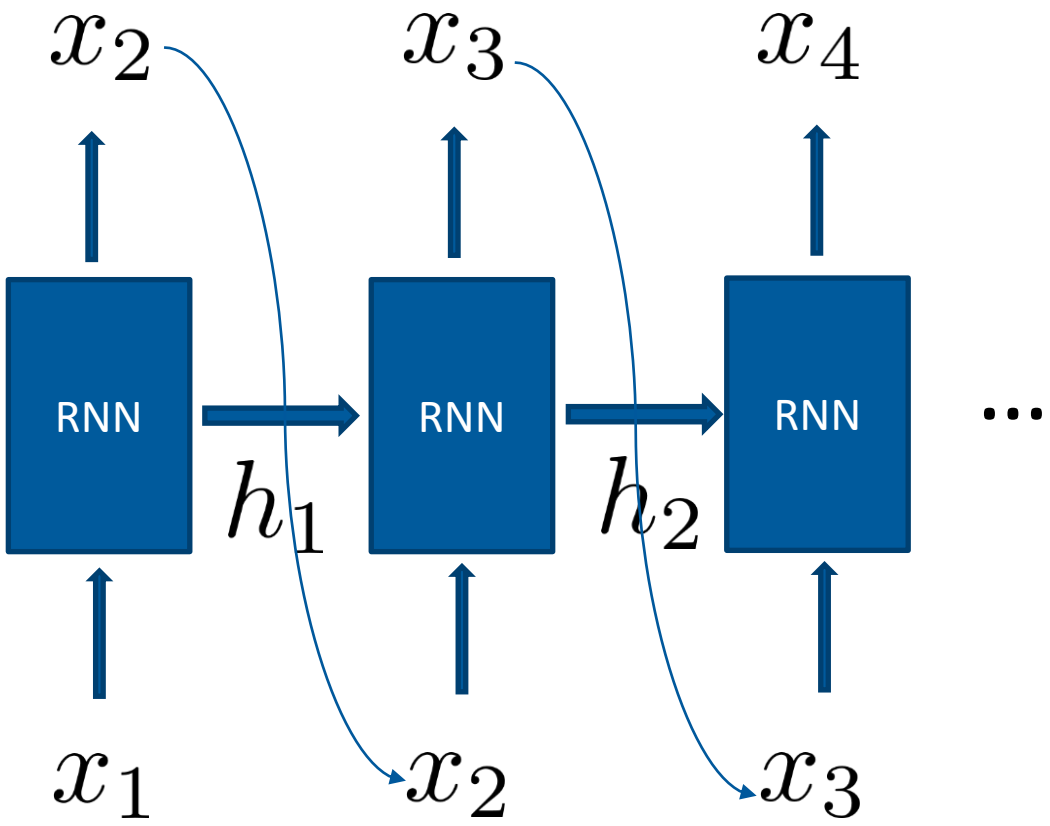
# Generation by auto-regressive models



# Generation by auto-regressive models



# Generation by auto-regressive models



# Example: generating music via RNN

- Monophonic music generation
  - Use RNN to predict the next note based on previous notes
  - To generate, start with a small amount of initial note, the model will generate the next note which will be used as the input again

# Other ways to implement Autoregressive Model

- Wavenet, Pixel CNN:
  - Use convolutional neural network with special design
  - Demo

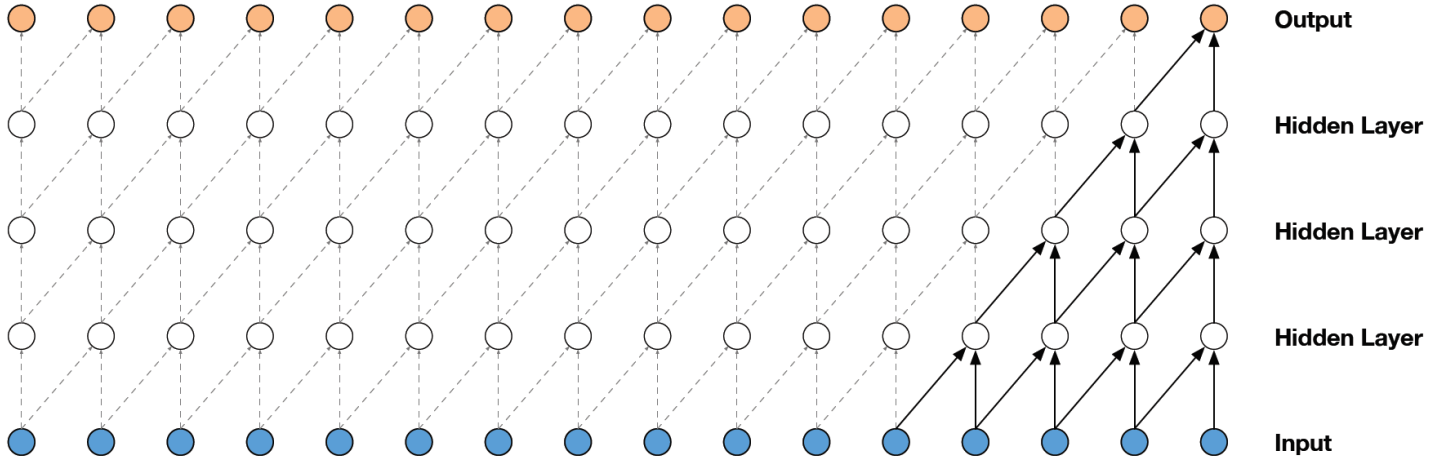
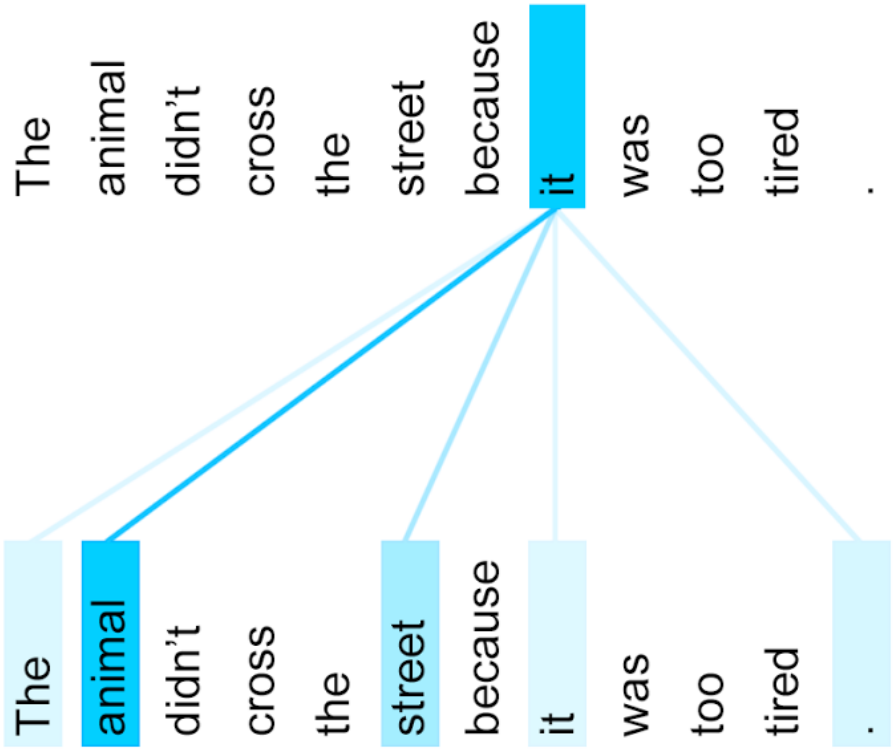



Figure 2: Visualization of a stack of causal convolutional layers.

# Other ways to implement Autoregressive Model

- Transformer
  - Use attention-based predictive function



# Demo

- Input sentence:

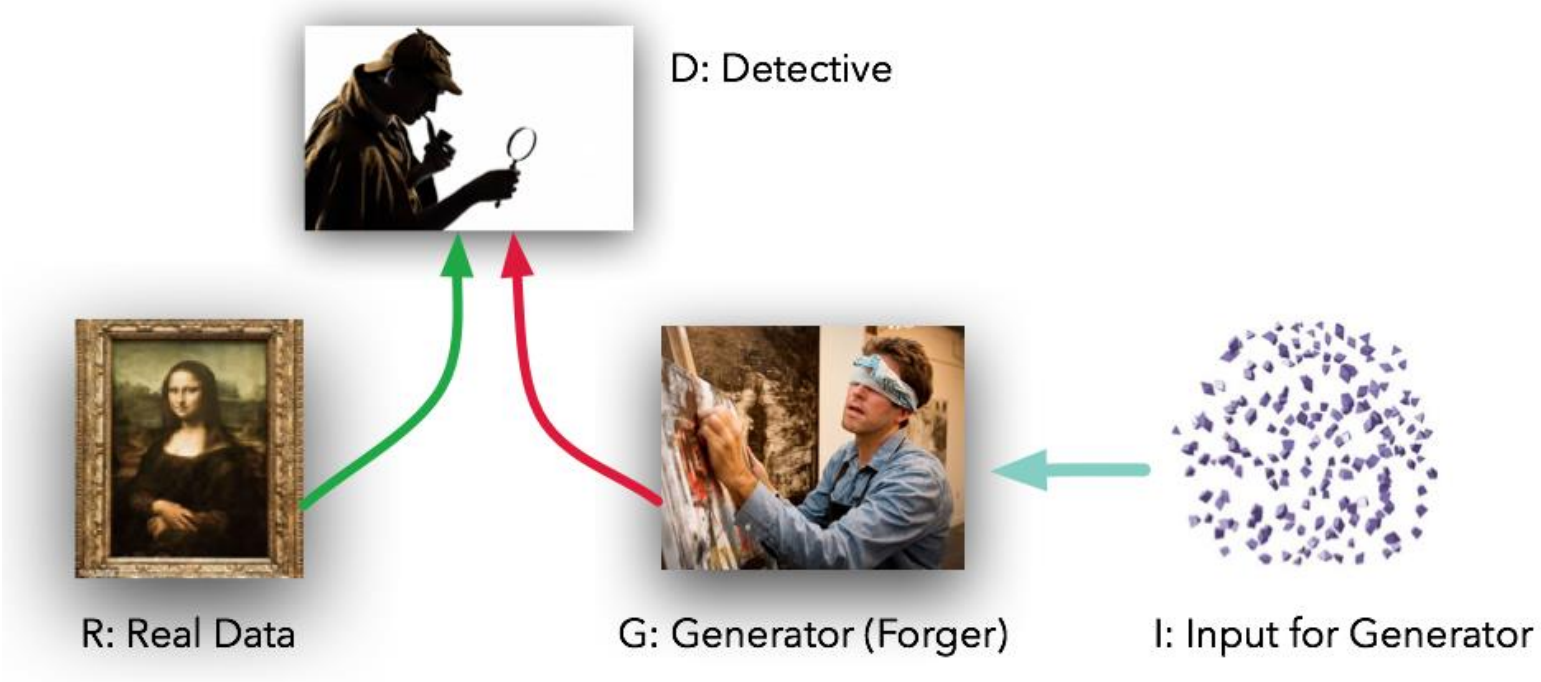
  *I went to a lounge to celebrate my birthday and*
- Generated sentences:

*I called Donna and told her I had just adopted her. She thought my disclosure was a donation, but I'm not sure if Donna met the criteria. Donna was a genuinely sweet, talented woman who put her life here as a love story. I know she thanked me because I saw her from the photo gallery and she appreciated my outrage. It was most definitely not a gift. I appreciate that I was letting her care about kids, and that she saw something in me. I also didn't have much choice but to let her know about her new engagement, although this doesn't mean I wasn't concerned, I am extremely thankful for all that she's done to this country. When I saw it, I said, "Why haven't you become like Betty or Linda?" "It's our country's baby and I can't take this decision lightly." "But don't tell me you're too impatient." Donna wept and hugged me. She never expresses milk, otherwise I'd think sorry for her but sometimes they immediately see how much it's meant to her. She apologized publicly and raised flagrant error of judgment in front of the society of hard choices to act which is appalling and didn't grant my request for a birth certificate. Donna was highly emotional. ...*

# Generative Adversarial Networks (GANs)

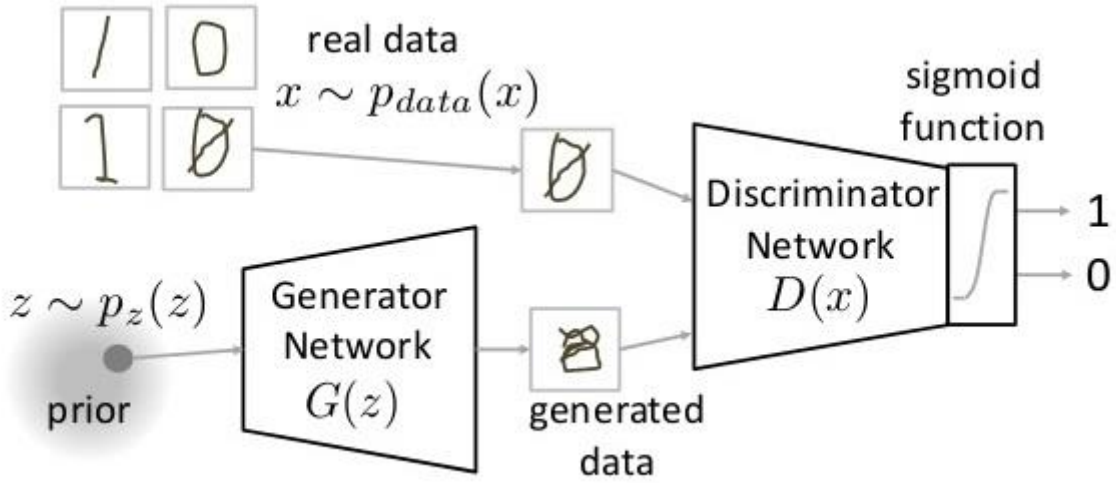
- Referring to GANs, Facebook's AI research director Yann LeCun called adversarial training "the most interesting idea in the last 10 years in ML."
- Do not directly modelling P(x) (but implicitly). It is based on a simple intuition: if the generation is perfectly done, you can not tell the difference between the real and the generated samples
- Based on two modules: a generator and a discriminator
  - Generator tries to fool the discriminator
  - Discriminator tries to identify the difference

# Generative Adversarial Network

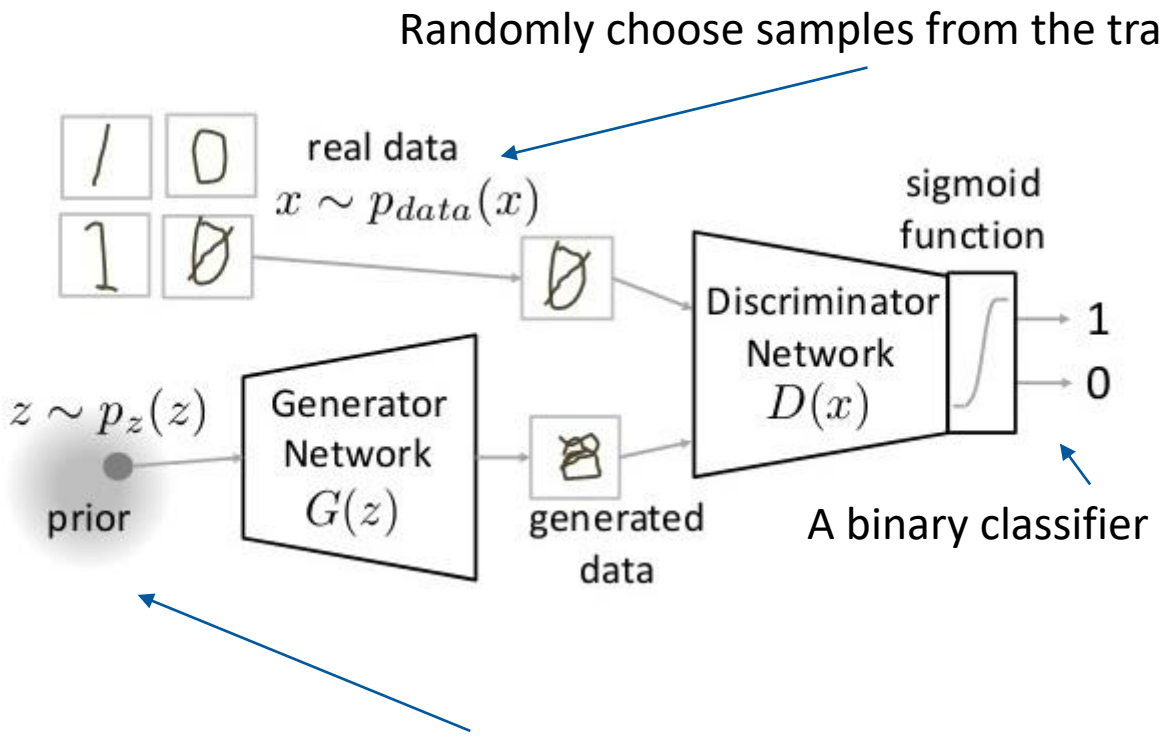



Updating both the Generator and discriminator until reach to an equilibrium

# Generative Adversarial Network



# Generative Adversarial Network



# Generative Adversarial Network

- Loss function

Ideally, D(z) gives a close-to-one value if z is a true data sample, otherwise, D(z) gives a close-to-zero value

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

True data

Noise provided for generating data

# Generative Adversarial Network

- Loss function

Ideally, D(z) gives a close-to-one value if z is a true data sample, otherwise, D(z) gives a close-to-zero value

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

True data

Noise provided for generating data

For a given G, find the best D to tell the difference.
The quality of the discrimination is measured by max V(D,G)

# Generative Adversarial Network

- Loss function

Ideally, D(z) gives a close-to-one value if z is a true data sample, otherwise, D(z) gives a close-to-zero value

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

True data

Noise provided for generating data

Find best G that can fool the best discriminator

# Generative Adversarial Network

**Algorithm 1** Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, $k$, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

**for** number of training iterations **do**

  **for** $k$ steps **do**

  - Sample minibatch of $m$ noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of $m$ examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

V(D,G): fix G, optimize D

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[ \log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  **end for**

  - Sample minibatch of $m$ noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

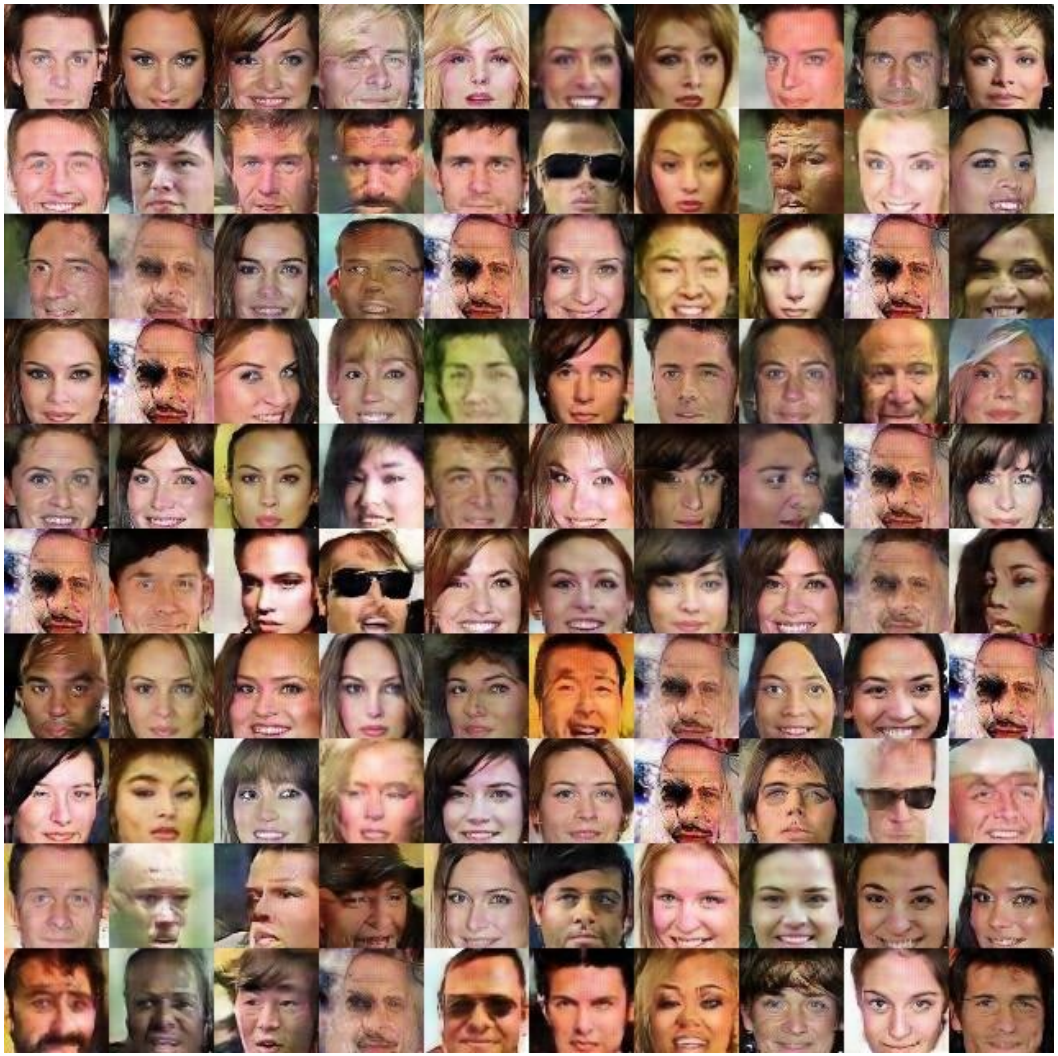
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

V(D,G): fix D, optimize G

**end for**

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

# Generative Adversarial Network



# Generative Adversarial Network

- Very active research field
  - Many improved GANs are proposed

# GAN: Application

- Generator can be modified as a translator
- For example, image super-resolution


Traditional loss: $\|f(I_l;\lambda) - I_h\|$

Problem: images achieves low loss values may not necessarily have authentic look

# GAN: Application

- Using GAN
  - Build a discriminator D to discriminate recovered high-resolution images and real high-resolution images
  - Train the super-resolution model to fool the discriminator

$$\max_{\lambda} \min_{D} E(\| f(I_l; \lambda) - I_h \|) + \eta \left( E(\log(D(I_h))) + E(\log(1 - D(f(I_l; \lambda)))) \right)$$

# GAN: applications

## Result

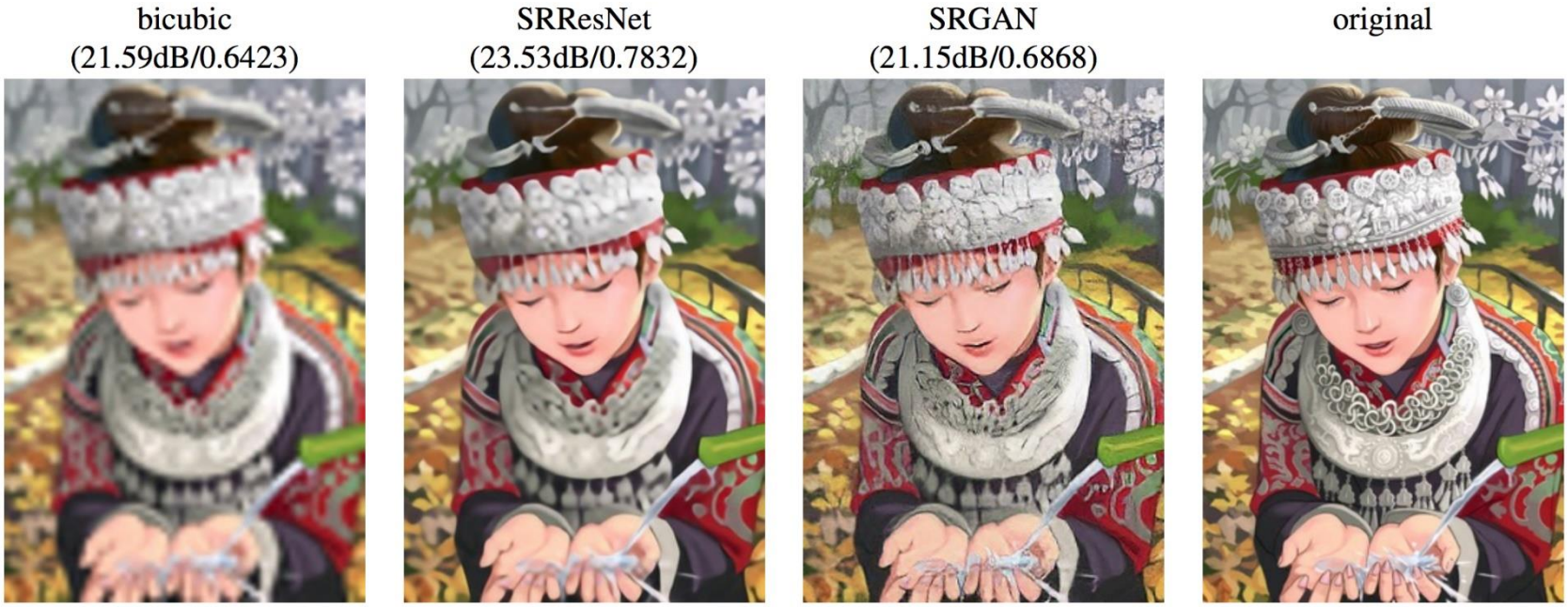



Figure 2: From left to right: bicubic interpolation, deep residual network optimized for MSE, deep residual generative adversarial network optimized for a loss more sensitive to human perception, original HR image. Corresponding PSNR and SSIM are shown in brackets. [4× upscaling]