Generative model intro & Autoregressive model

Instructor: Seunghoon Hong

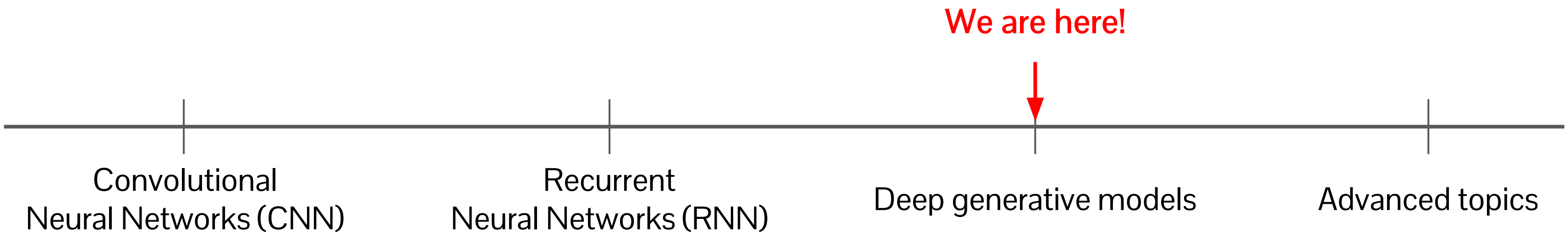
Course overview

- ❑ Image classification
- ❑ Object detection
- ❑ Semantic segmentation
- ❑ Visualization
- ❑ Style transfer
- ❑ Adversarial attacks

- ❑ Text modeling
- ❑ Machine translation
- ❑ Image captioning
- ❑ Visual question answering

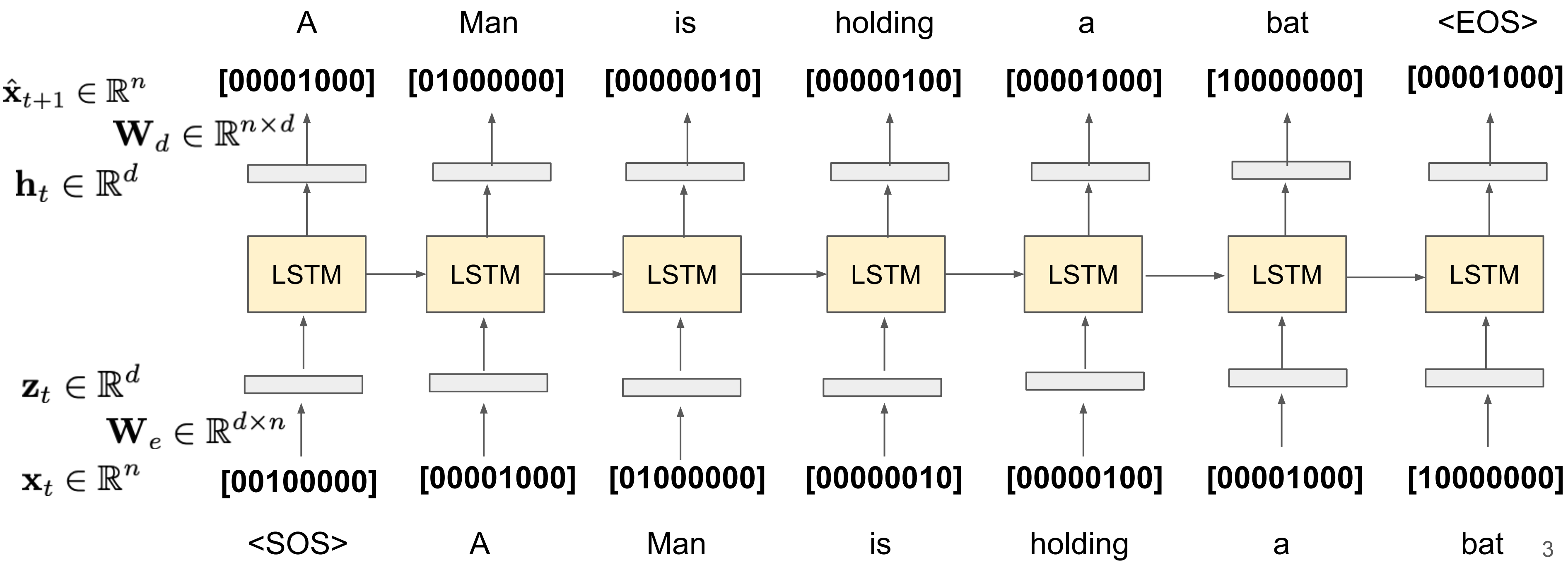
- ❑ Image generation
- ❑ Text generation
- ❑ Img-to-img translation

- ❑ Attention and versatile networks
- ❑ Self- and Semi-supervised learning
- ❑ Multi-modal learning
- ❑ Graph neural networks

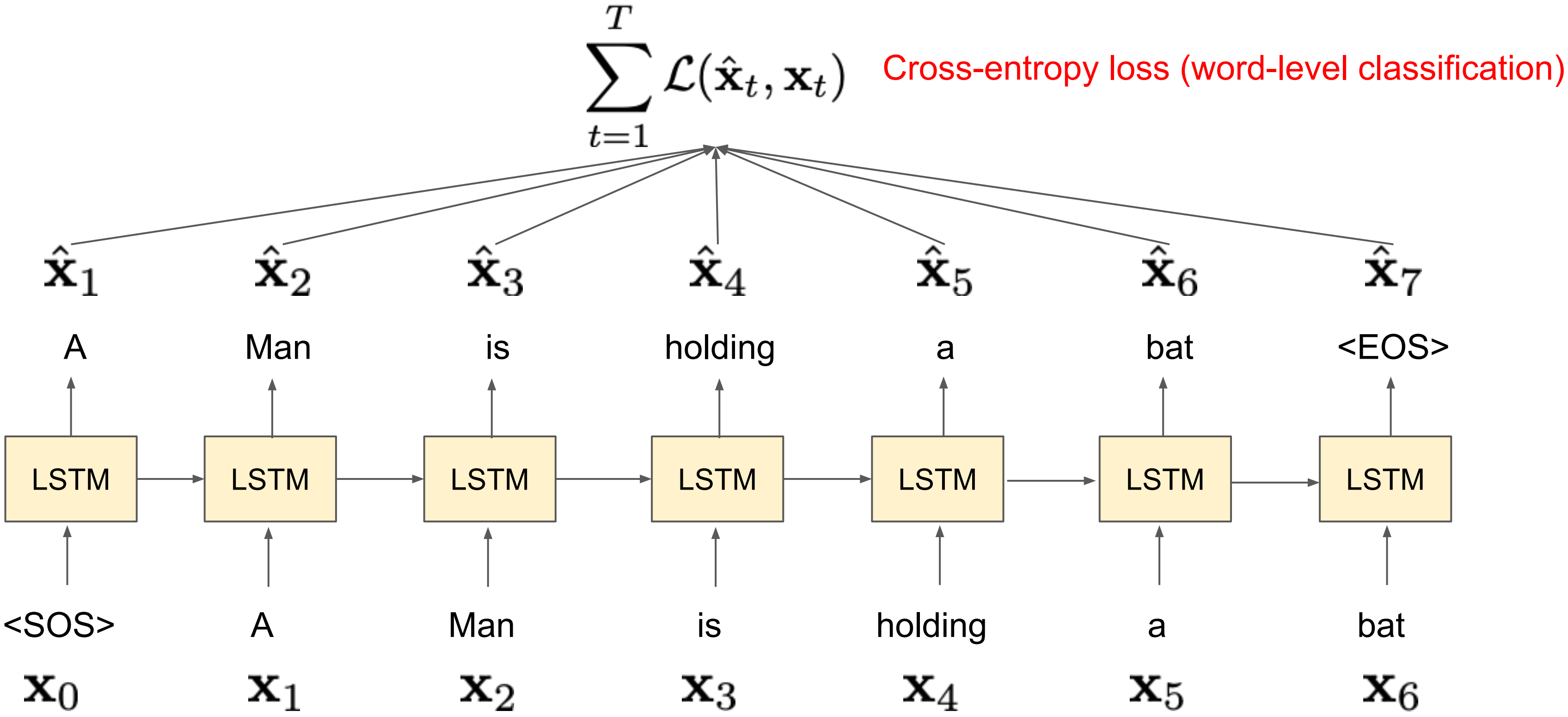


Recap: RNN as a language model

- Sentence generation = predicting a next token



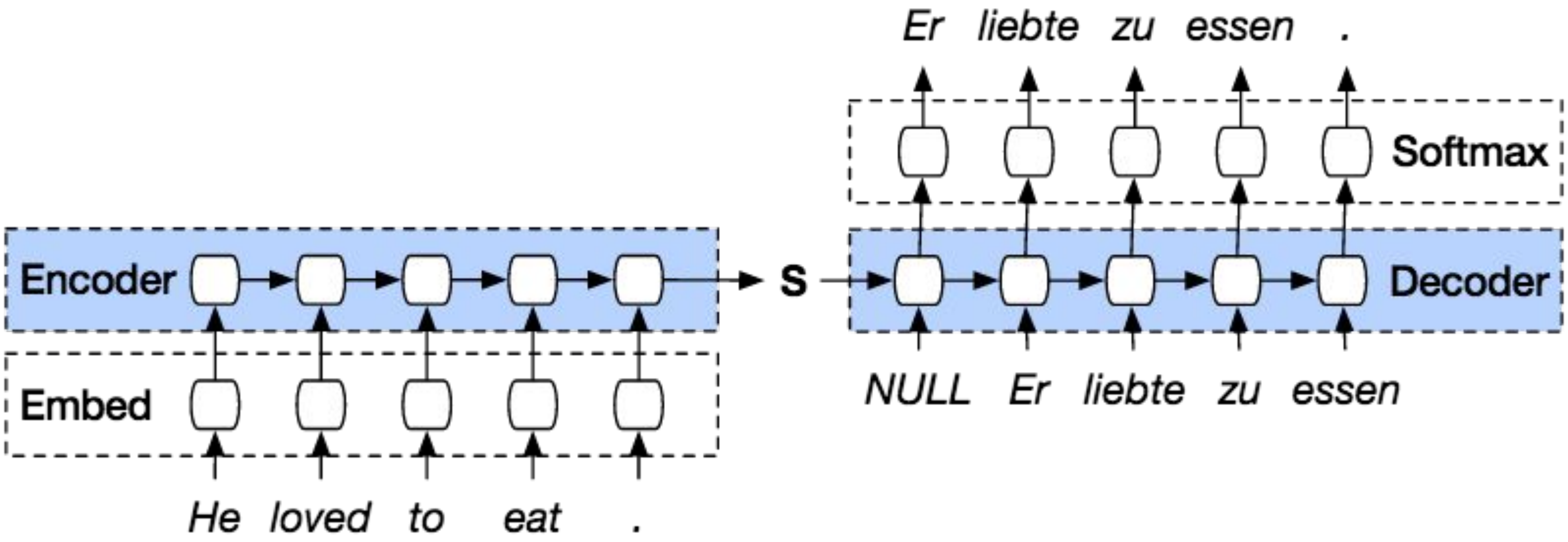
Recap: Training: RNN-based language model



We feed ground-truth words as inputs (also known as **teacher forcing**)

Recap: Machine translation

- Translate a sentence in one language to another



Recap: Bayes' Theorem

- Conditional probability

$$P(A|B) = \frac{P(A,B)}{P(B)}, \quad P(B|A) = \frac{P(A,B)}{P(A)}$$

$$P(A,B) = P(A|B)P(B) = P(B|A)P(A)$$

- Bayes' Theorem

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

$$posterior \propto likelihood \times prior$$

Today's agenda

- Introduction to generative models
- Autoregressive models

Introduction to generative models

Machine Learning for Understanding Data

- Learning to **perceive** and **reason** from a data

Concepts?

Person, elephant, field, sky, fence

Relationship between concepts?

One person is holding another
Two people are standing next to fence
An elephant is standing on a grass

Context?

A father went to zoo with his son watching an elephant

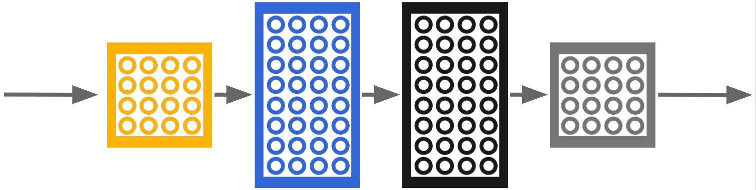
Understanding via Categorization

- Learning to associate input to pre-defined, task-specific labels
- Examples: **classification** (concept)

input x

model

$$f_\theta(x)$$



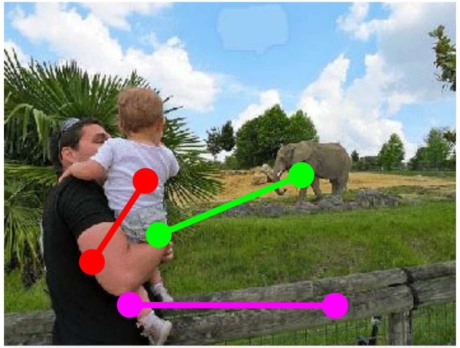
output y

- "dog"
- **"person"**
- "apple"
- **"elephant"**
- ⋮
- **"field"**

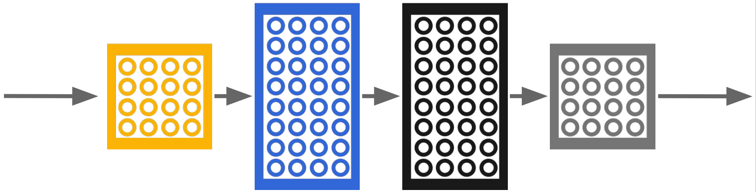
Understanding via Categorization

- Learning to associate input to pre-defined, task-specific labels
- Examples: **classification** (relationship)

input x



$$f_\theta(x)$$



output y

"sitting"

"holding"

"below"

"next to"

⋮

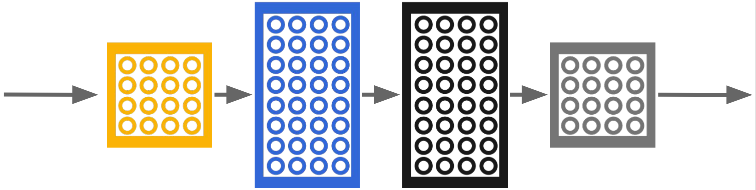
"watching"

Understanding via Categorization

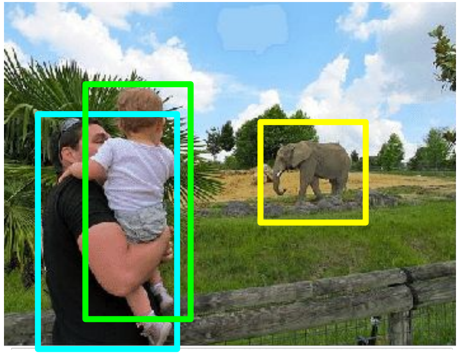
- Learning to associate input to pre-defined, task-specific labels
- Examples: classification, **detection**

input x

$$f_\theta(x)$$



output y

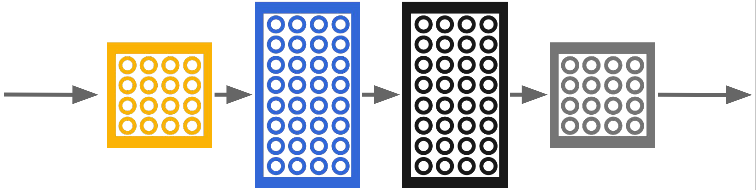


Understanding via Categorization

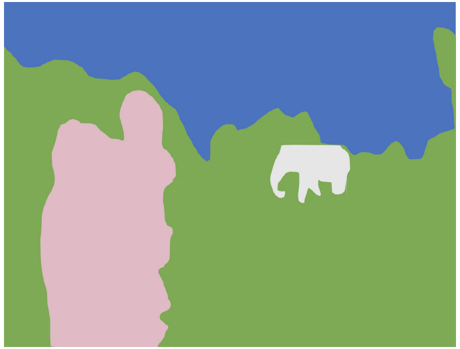
- Learning to associate input to pre-defined, task-specific labels
- Examples: classification, detection, **segmentation**, ...

input x

$$f_\theta(x)$$



output y

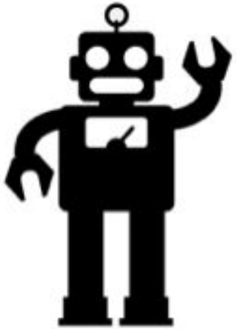


Understanding via Categorization

- Limitations
  - Requires labels (human annotations) for training
  - Learns a biased knowledge to solve the specific task

Understanding via Generation

- Learning to synthesize the data itself
- Why do we care about generation?
  - Generation requires implicit understanding of underlying structure of data
  - No need for labels → unsupervised learning
  - Can learn something useful for downstream tasks
  - Generated data itself can be useful



generate

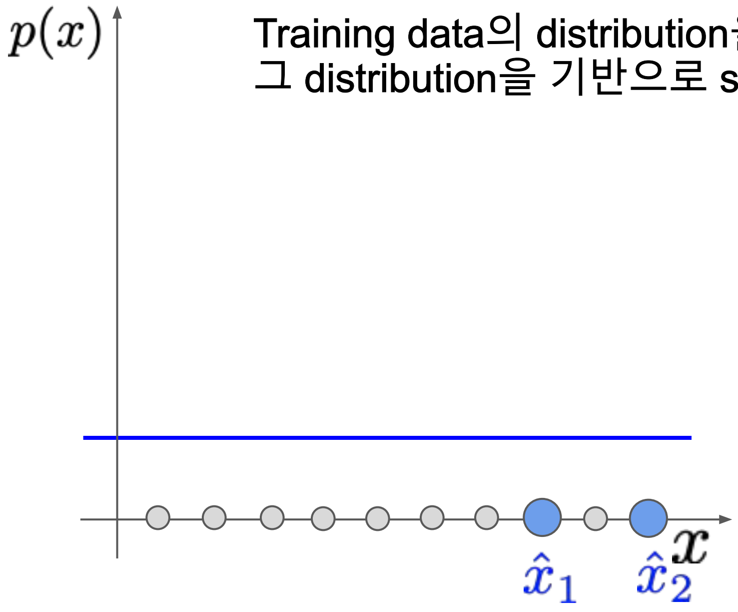
data x

How do we define the task of *generation*?

- What is the task of generative modeling?
- What is the objective function for learning generative models?

Q1: what is it like building a generative model?

- Consider that our data is composed of 1d points
- Example 1: 1d data is distributed according to Uniform distribution



Training data의 distribution을 알 수 있다면...
그 distribution을 기반으로 sampling 가능

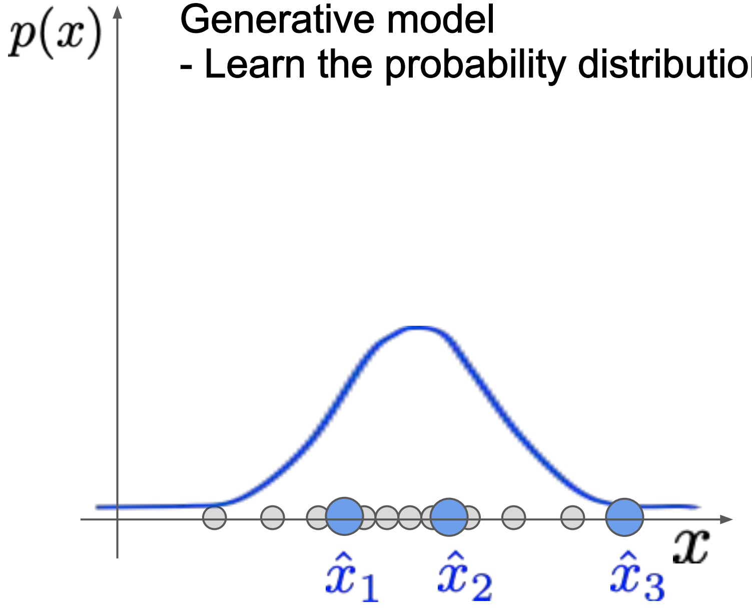
$$p(x) = \mathrm{Uniform}(0, a)$$

once we know the underlying function that produces data, we can create new samples by sampling from (or to follow) this function

Q1: what is it like building a generative model?

- Consider that our data is composed of 1d points
- Example 2: data is distributed according to 1d Gaussian distribution

Generative model
- Learn the probability distribution from training data!



$$p(x) = c \cdot \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Again, once we know the underlying function that produces data, we can create new samples by sampling from (or to follow) this function

Q1: what is it like building a generative model?

- If we know the ground-truth distribution of data, we can generate new one by sampling from the distribution (or to follow the distribution)

$$\hat{x} \sim P(X)$$

즉, we assume that training data follows some distribution

- Since the ground-truth distribution is unknown, we use a neural network to approximate the distribution

$$G_\theta \approx P(X)$$

Deep Generative Models

Assumption:
many many training data will follow some distribution

- Learning a model that its outputs follow the true data distribution

$$G_\theta \sim P(X)$$

Generated images from G

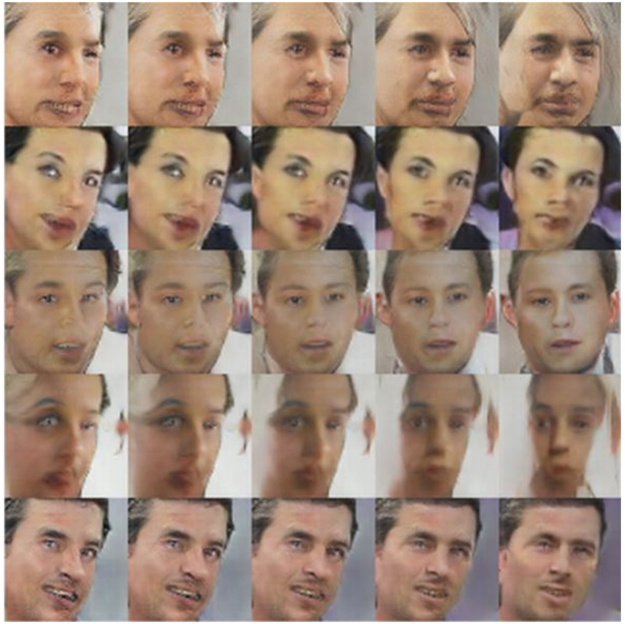
True Images X

Recent applications of generative models

Generative models have improved enormously

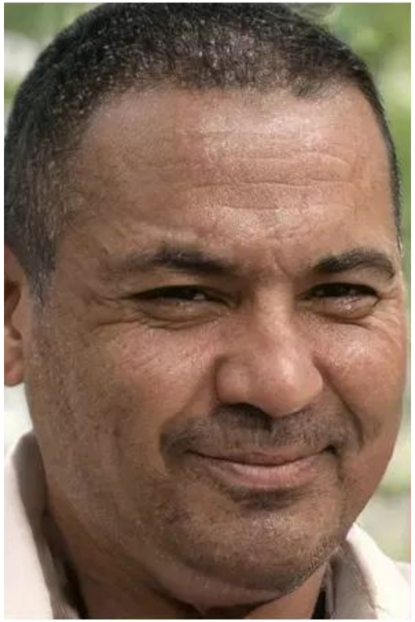
2014

Goodfellow et al.



2016

Radford et al.



2018

Karras et al.

[slide credit: Tim Saliman]

Generative models have improved enormously

a pitcher is about to throw the ball to the batter.

a picture of a very clean living room.

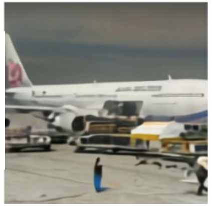
a sheep standing in a open grass field.

2016

Reed et al.

a very cute cat laying by a big bike.

china airlines plain on the ground at an airport with baggage cars nearby.



a table that has a train model on it with other cars and things

2021

Ramesh et al.

A cute corgi lives in a house made out of sushi.

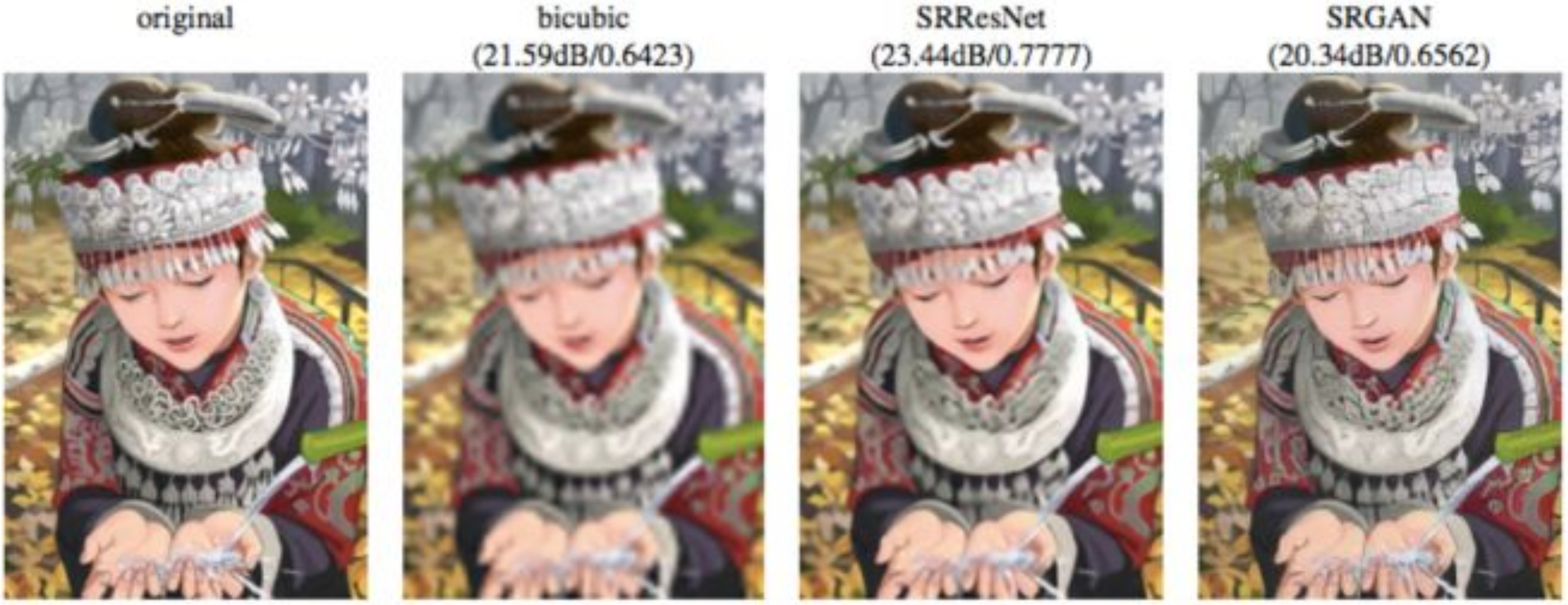
2022

Saharia et al.

Generative models are started to become useful

- Image upsampling / image compression

Ledig et al. (2016)



Generative models are started to become useful

- Video synthesis

https://tcwang0509.github.io/vid2vid/

Generative models are started to become useful

- Text-to-image synthesis

A wall in a royal castle. There are two paintings on the wall. The one on the left a detailed oil painting of the royal raccoon king. The one on the right a detailed oil painting of the royal raccoon queen.

A group of teddy bears in suit in a corporate office celebrating the birthday of their friend. There is a pizza cake on the desk.

A chrome-plated duck with a golden beak arguing with an angry turtle in a forest.

Saharia et al., Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding

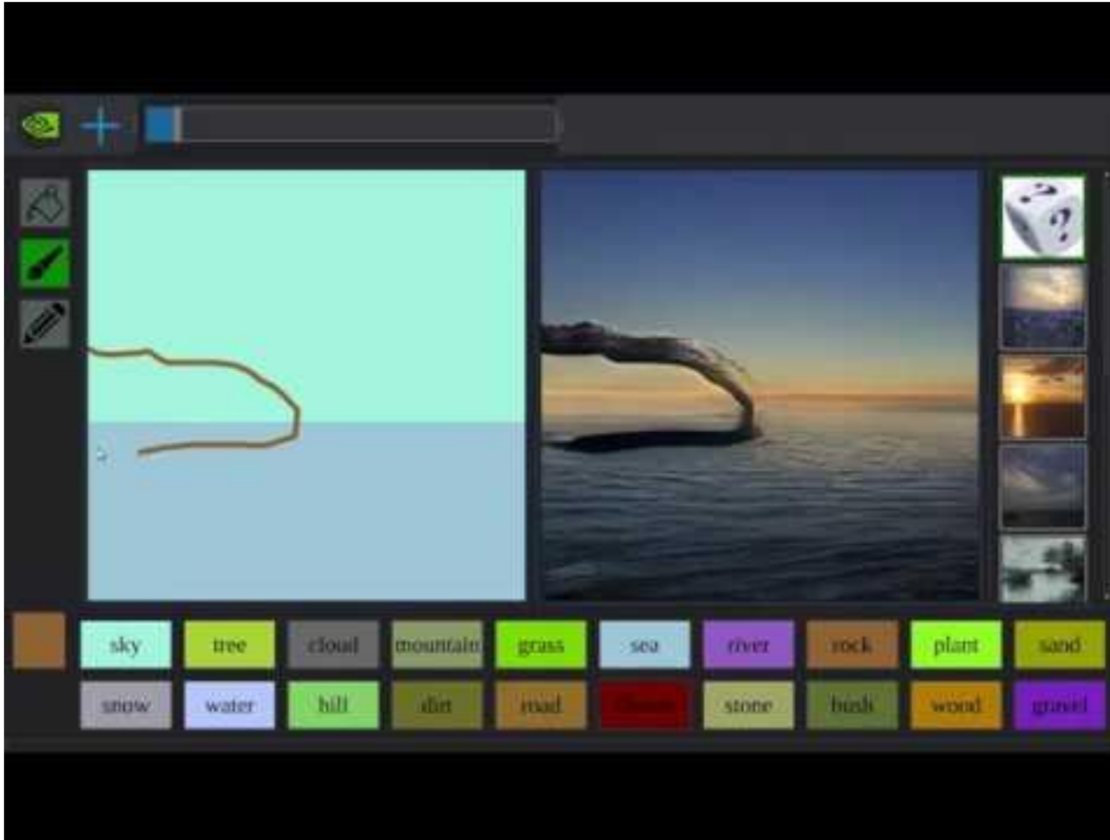
Generative models are started to become useful

- Text-to-image synthesis

generated by novel.ai

Generative models are started to become useful

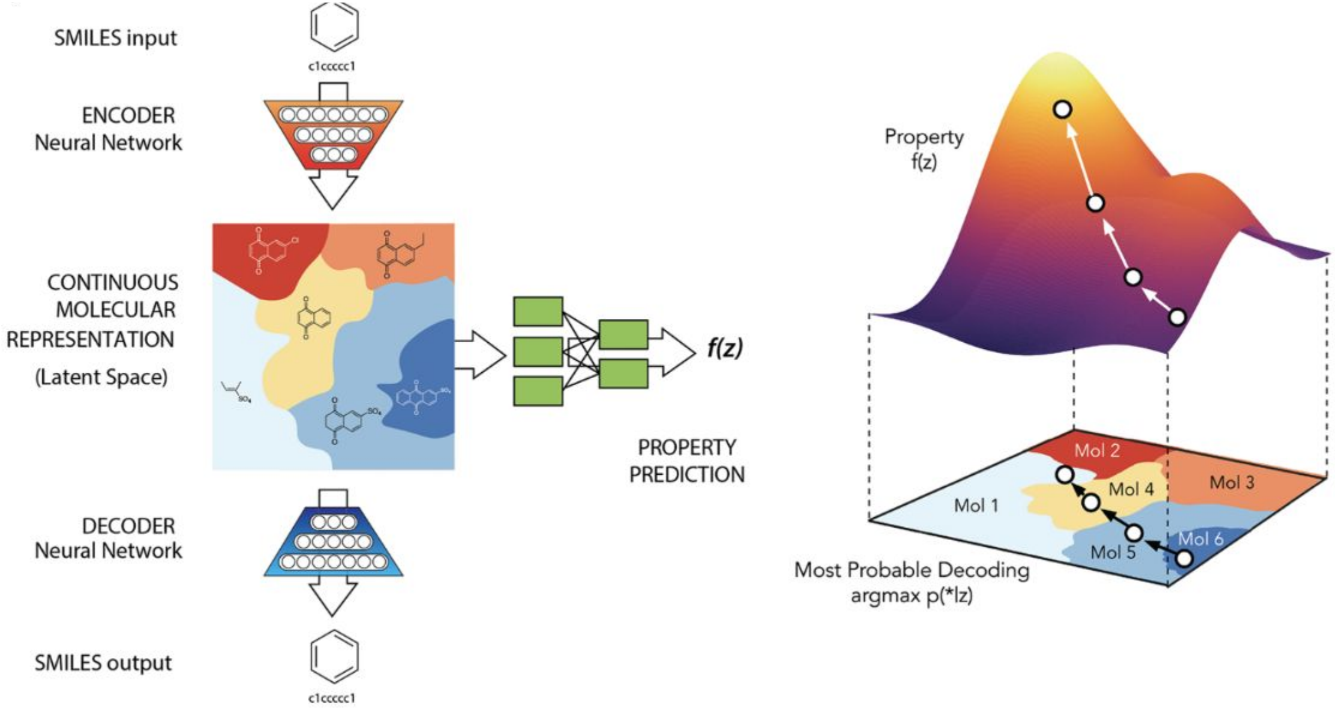
- Interactive drawing



https://blogs.nvidia.com/blog/2019/03/18/gaugan-photorealistic-landscapes-nvidia-research/

Generative models are started to become useful

- Drug discovery



Generative models are started to become useful

- Code generation

fetch_tweets.js fetch_tweets.py fetch_tweets.rb fetch_tweets.ts fetch_tweets.go

GitHub Copilot

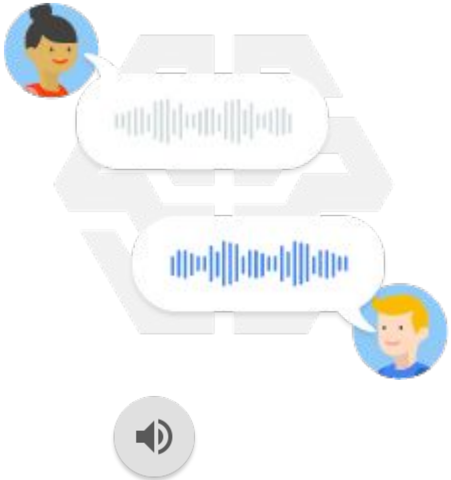
```
const token = process.env["TWITTER_BEARER_TOKEN"]

const fetchTweetsFromUser = async (screenName, count) => {
  const response = await fetch(
    `https://api.twitter.com/1.1/statuses/user_timeline.json?screen_name=${screenName}&count=${cou
    {
      headers: {
        Authorization: `Bearer ${token}`,
      },
    }
  )
  const json = await response.json()
  return json
}
```

Copilot

Generative models are started to become useful

- Text-to-speech or speech-to-text synthesis



hand-coded

Wavenet generative model

https://deepmind.com/blog/wavenet-generative-model-raw-audio/

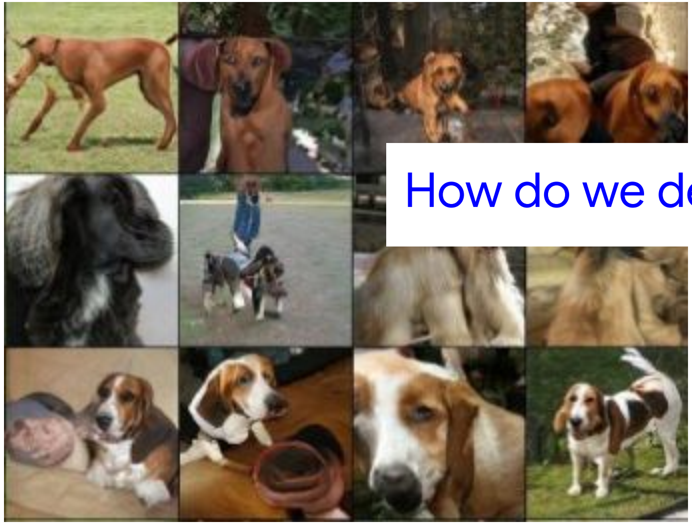
Autoregressive models

What is the loss for generative model?

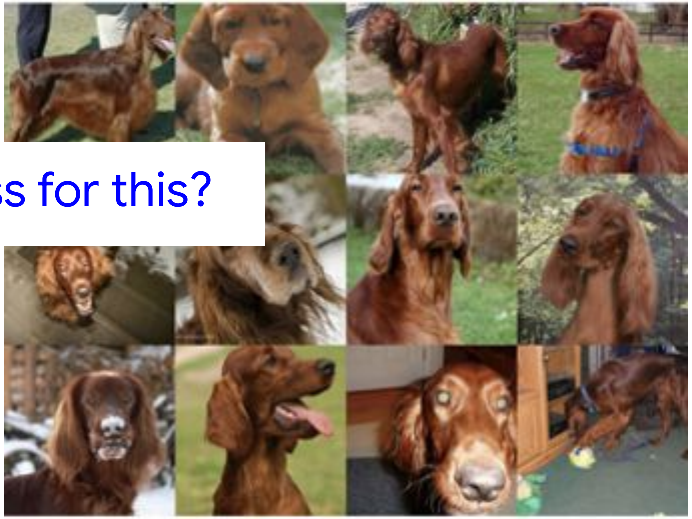
- Learning objective:
  train a generator such that its outputs are distributed according to the target distribution

$$G_\theta \approx P(X)$$

Generated images from G



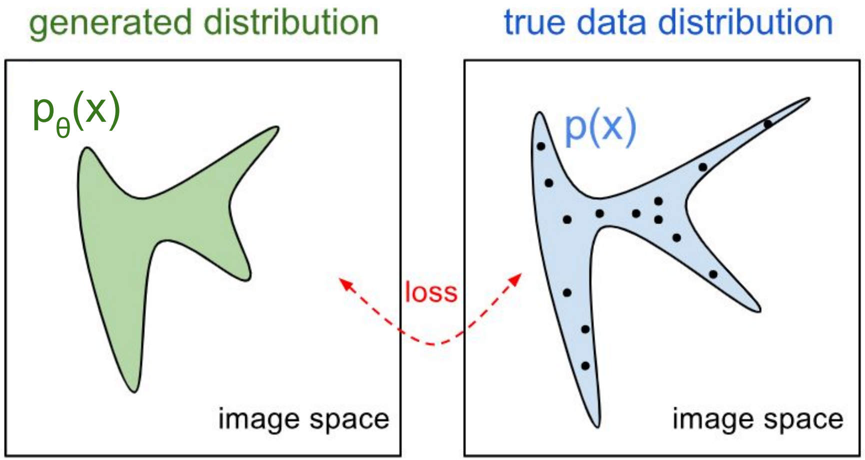
True Images X



How do we define a loss for this?

What is the loss for generative model?

- Let’s consider that our generator (e.g. neural network) produces a probability measure for any input data x
- Generator output for input x: $p_\theta(\mathbf{x})$
- **Assume** that we know the true probability: $p^*(\mathbf{x})$
- Then, we will define a loss as a discrepancy b/w two probability distributions



What is the loss for generative model?

- There are various distance measures for two probability distributions
- Here, we consider **KL divergence**

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

$$= \int p^*(x) \log \frac{p^*(x)}{p_\theta(x)} dx$$

$$= \underbrace{\int p^*(x) \log p^*(x) dx}_{\text{This term is irrelevant to parameters}} \boxed{\overbrace{- \int p^*(x) \log p_\theta(x) dx}^{\text{Cross-entropy loss!}}}$$

$$\leftrightarrow \min_\theta \boxed{\mathbb{E}_{p^*(x)}[-\log p_\theta(x)]}$$

In the sense of minimizing cross-entropy loss, it is similar to supervised learning

L But we don't know actually the true dist. $p^*(x)$.

What is the loss for generative model?

$$\min_{\theta} \mathbb{E}_{\boxed{p^*(x)}}[-\log p_\theta(x)]$$

- Remaining issues:
  - We do not know the true distribution
  - We do not have an access to infinite amount of data for expectation! Even if we have one, it is computationally intractable.

(위의 두 문제를 동시에 해결!)

- Solution using samples: ∵ sample이 true dist.로부터 나오므로 OK
  - We assume that the training data approximate the true distribution
  - Then we can optimize the following

$x_i \sim p^*(x)$.

Note : weight는 sample에 의해 자연스럽게 반영됨. (Think tossing coin)

$$\min_{\theta} \boxed{-\frac{1}{N}\sum_{I=1}^{N} \log p_\theta(x_i)} = \mathbb{E}_{p^*(x)}[-\log p_\theta(x)] \quad \text{if } N \to \infty$$

What is the loss for generative model?

- Maximum Likelihood Estimation (MLE)

Intuition!
increasing prob of dog images
<-> decreasing prob. of other images
즉, 다른 경우를 surpress!

$$\hat{\theta} = \arg\max_{\theta \in \Theta} \sum_{x_i \in \mathcal{X}} \boxed{\log p_\theta(x_i)}$$

Log Likelihood

$$\leftrightarrow \arg\max_{\theta \in \Theta} \prod_{x_i \in \mathcal{X}} \boxed{p_\theta(x_i)}$$

Likelihood

Find model parameters that maximize the probability of sampling training data (likelihood)

$$\leftrightarrow \arg\min_{\theta \in \Theta} \sum_{x_i \in \mathcal{X}} -\log p_\theta(x_i)$$

In practice, we minimize the negative log likelihood for gradient descent

Challenges in evaluating likelihood

$$\hat{\theta} = \arg\min_{\theta \in \Theta} \sum_{x_i \in \mathcal{X}} -\log \boxed{p_\theta(x_i)}$$

- For high-dimensional data, it is difficult to optimize the joint distribution at once

$$x_i = [x_i^1, x_i^2, x_i^3, \ldots, x_1^d] \in \mathbb{R}^d$$
$$p(x_i) = p(x_i^1, x_i^2, x_i^3, \ldots, x_1^d)$$

- Examples of high-dimensional data
  - Image (d = number of pixels)
  - Sentence (d = length of sentence)

Auto-Regressive Model (AR)

- Factorizing the likelihood via **chain rule**

$$p(a, b) = p(a|b)p(b)$$

Auto-Regressive Model (AR)

- Factorizing the likelihood via chain rule

(즉, x는 T-dim vector 가정)

$$
\begin{aligned}
p_\theta(x) &= p_\theta(x_1, x_2, x_3, \dots, x_T) \\
&= p_\theta(x_T | x_1, x_2, \dots, x_{T-1}) \boxed{p_\theta(x_1, x_2, \dots, x_{T-1})} \\
&= \prod_{t=1}^{T} p_\theta(x_t | x_1, \dots, x_{t-1})
\end{aligned}
$$

apply recursively

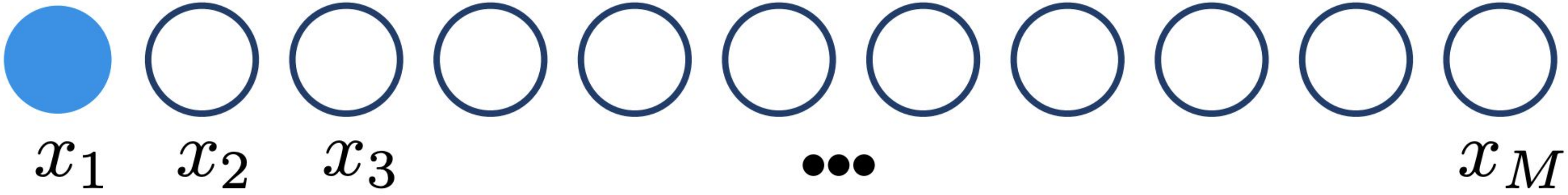
each conditional has lower dimension 임.

– predicting one value at a time.

Auto-Regressive Model (AR)

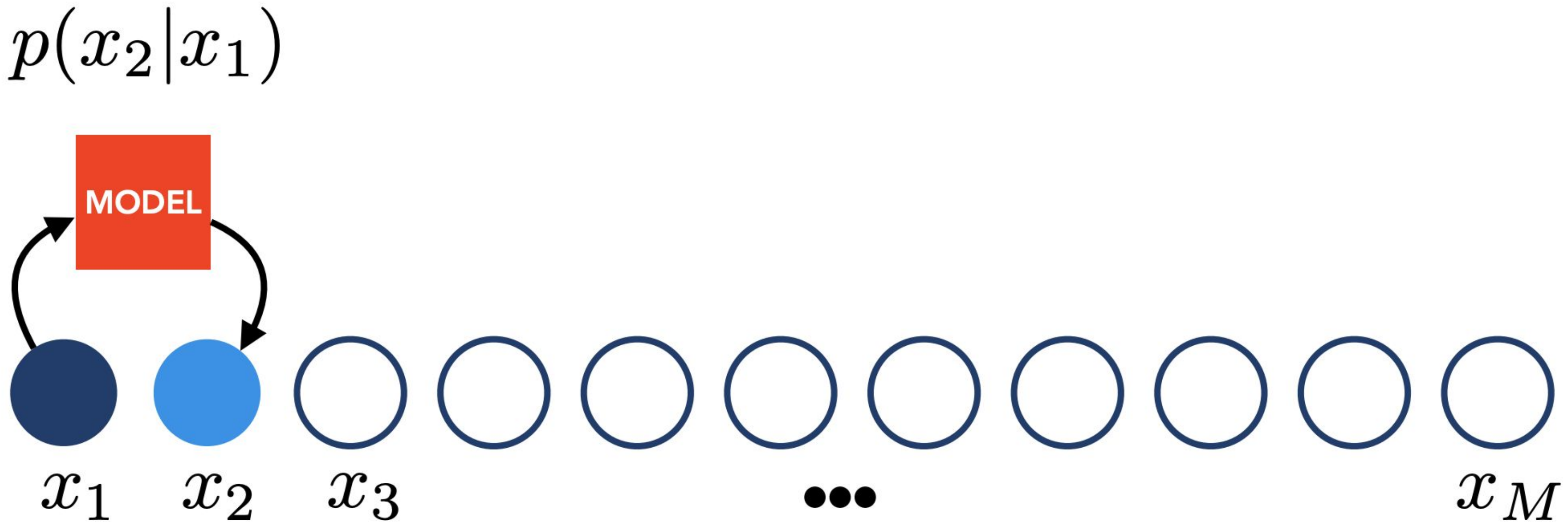
$$p_\theta(x) = \prod_{t=1}^{T} p_\theta(x_t | x_1, ..., x_{t-1})$$

$$p(x_1)$$



Auto-Regressive Model (AR)

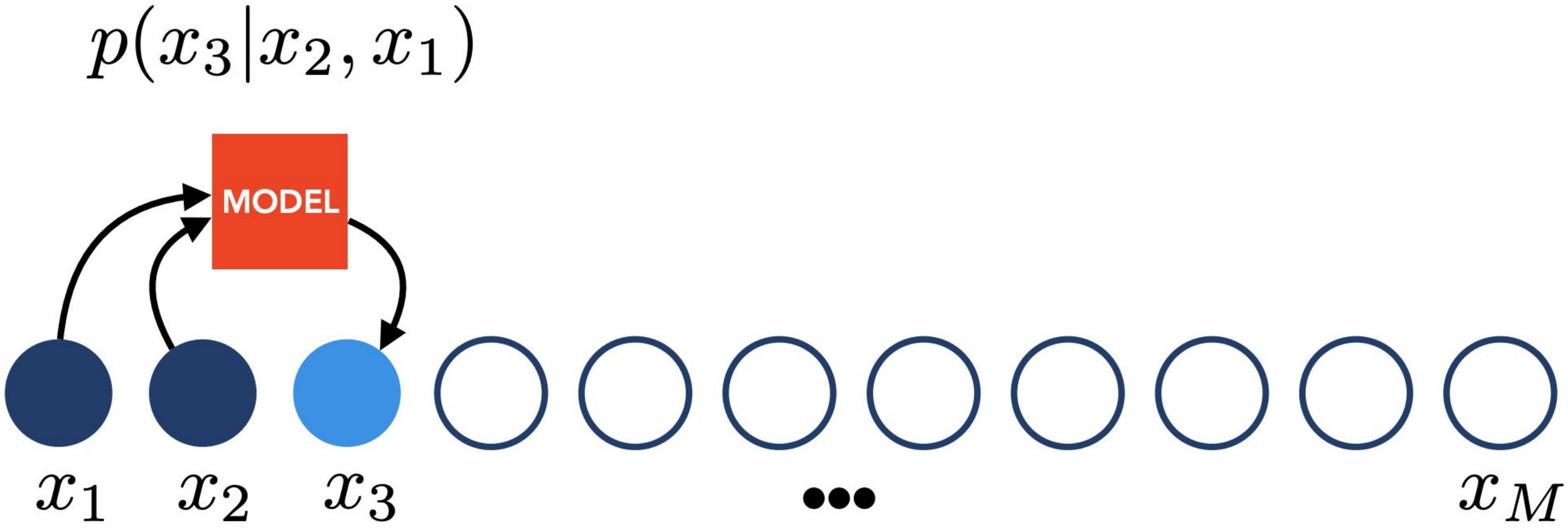
$$p_\theta(x) = \prod_{t=1}^{T} p_\theta(x_t | x_1, ..., x_{t-1})$$



Slide credit: http://www.yisongyue.com/courses/cs155/2018_winter/lectures/Lecture_17.pdf

Auto-Regressive Model (AR)

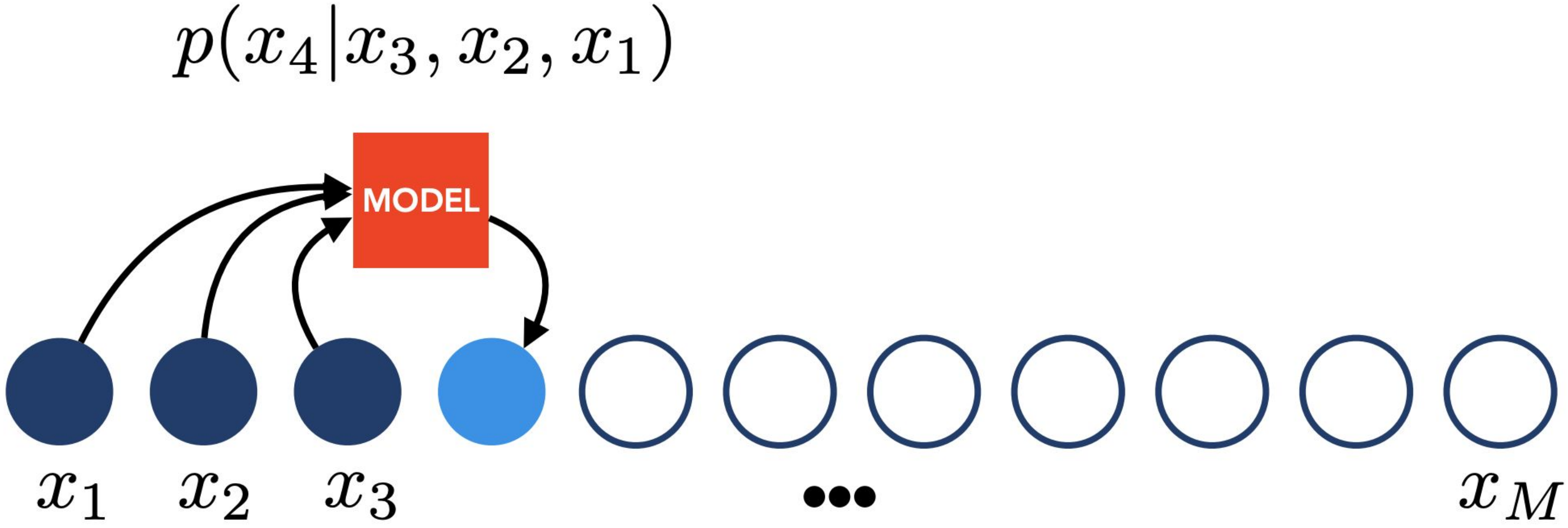
$$p_\theta(x) = \prod_{t=1}^{T} p_\theta(x_t | x_1, \ldots, x_{t-1})$$



Slide credit: http://www.yisongyue.com/courses/cs155/2018_winter/lectures/Lecture_17.pdf

Auto-Regressive Model (AR)

$$p_\theta(x) = \prod_{t=1}^{T} p_\theta(x_t | x_1, ..., x_{t-1})$$

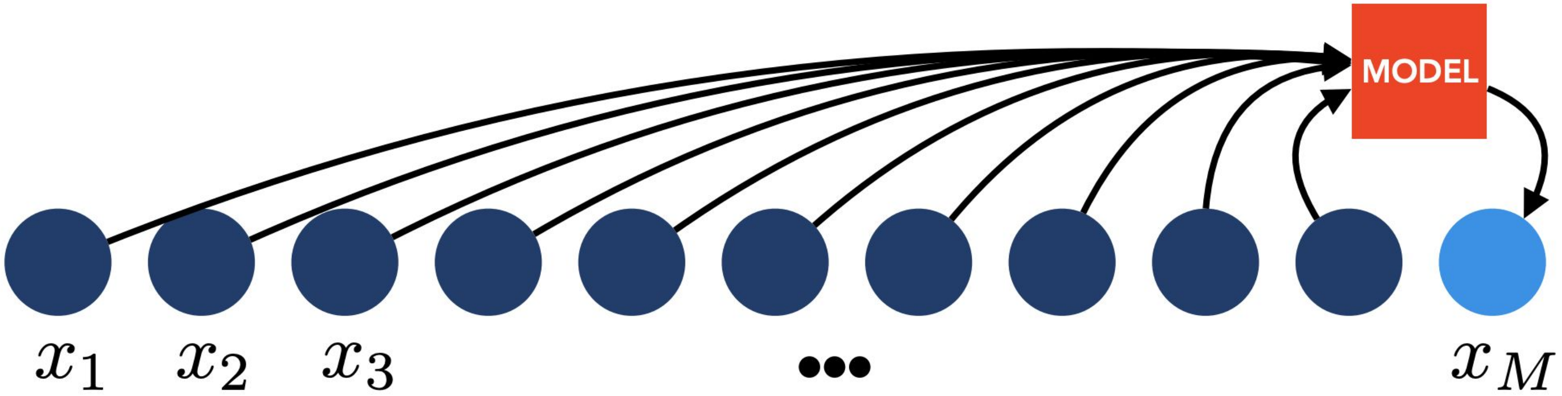


Slide credit: http://www.yisongyue.com/courses/cs155/2018_winter/lectures/Lecture_17.pdf

Auto-Regressive Model (AR)

$$p_\theta(x) = \prod_{t=1}^{T} p_\theta(x_t | x_1, ..., x_{t-1})$$

$$p(x_M | x_{M-1}, \ldots, x_1)$$



Slide credit: http://www.yisongyue.com/courses/cs155/2018_winter/lectures/Lecture_17.pdf

Autoregressive models

- Factorized objective function

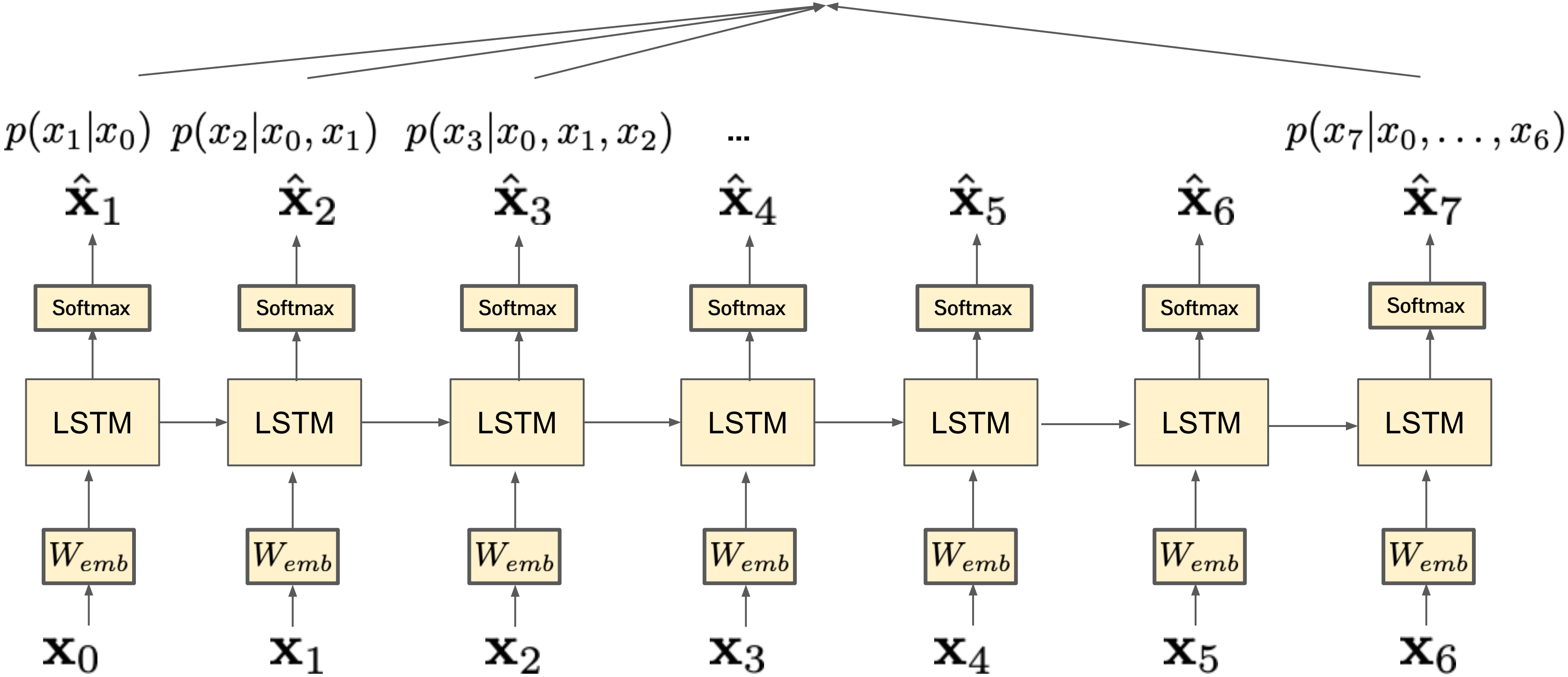
$$\hat{\theta} = \arg\min_{\theta \in \Theta} \sum_{x_i \in \mathcal{X}} - \log p_\theta(x_i)$$

$$= \arg\min_{\theta \in \Theta} \sum_{x_i \in \mathcal{X}} \sum_{1 \leq t \leq d} - \log p_\theta(x_i^t | x_i^1, \ldots, x_i^{t-1})$$

즉, RNN은 text generalization에서 generative model!

RNN revisited

$$\sum_{x_i \in \mathcal{X}} \sum_{1 \le t \le d} -\log p_\theta(\hat{x}_i^t = x_i^t | x_i^1, \ldots, x_i^{t-1})$$



Auto-Regressive Model: A Summary

- Maximizing factorized likelihood
  - Generate data one-by-one conditioned on previous outputs
- Appropriate to handle sequential data
  - Text, audio, video
- Fully-observable model
  - No latent representation of data

Challenges

- Modeling long-term dependency
- Serial processing → difficult for parallelization

Next

- Case study: autoregressive models
  - AR with attention for modeling long-term dependency
  - Task: machine translation