SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient

Lantao Yu[†], Weinan Zhang[†], Jun Wang[‡], Yong Yu[†]

[†]Shanghai Jiao Tong University, [‡]University College London

Attribution

- Multiple slides taken from
  - Hung-yi Lee
  - Paarth Neekhara
  - Ruirui Li
  - Original authors at AAAI 2017

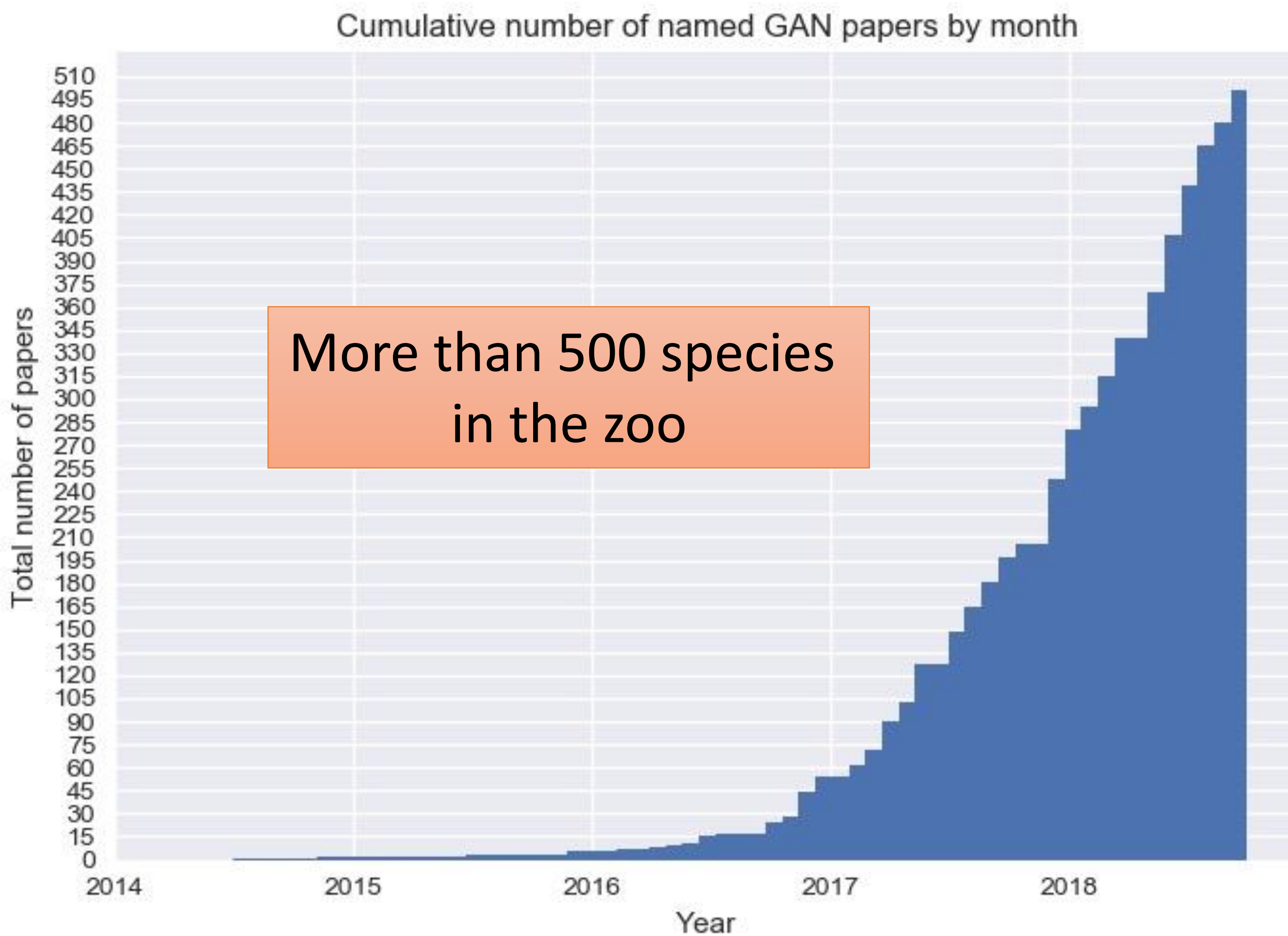
- Presented by: Pratyush Maini

Outline

1. **Introduction to GANs**
2. Brief theoretical overview of GANs
3. Overview of GANs in Sequence Generation
4. SeqGAN
5. Other recent work: Unsupervised Conditional Sequence Generation

All Kinds of GAN …

https://github.com/hindupuravinash/the-gan-zoo



(not updated since 2018.09)

All Kinds of GAN …

https://github.com/hindupuravinash/the-gan-zoo

GAN
ACGAN
BGAN
CGAN
DCGAN
EBGAN
fGAN
GoGAN
⋮

- SeUDA - Semantic-Aware Generative Adversarial Nets for Unsupervised Domain Adapt Segmentation
- SG-GAN - Semantic-aware Grad-GAN for Virtual-to-Real Urban Scene Adaption (githu
- SG-GAN - Sparsely Grouped Multi-task Generative Adversarial Networks for Facial Attr
- SGAN - Texture Synthesis with Spatial Generative Adversarial Networks
- SGAN - Stacked Generative Adversarial Networks (github)
- SGAN - Steganographic Generative Adversarial Networks
- SGAN - SGAN: An Alternative Training of Generative Adversarial Networks
- SGAN - CT Image Enhancement Using Stacked Generative Adversarial Networks and Tr Segmentation Improvement
- sGAN - Generative Adversarial Training for MRA Image Synthesis Using Multi-Contrast
- SiftingGAN - SiftingGAN: Generating and Sifting Labeled Samples to Improve the Rem Classification Baseline in vitro
- SiGAN - SiGAN: Siamese Generative Adversarial Network for Identity-Preserving Face H
- SimGAN - Learning from Simulated and Unsupervised Images through Adversarial Trai
- SisGAN - Semantic Image Synthesis via Adversarial Learning

Mihaela Rosca, Balaji Lakshminarayanan, David Warde-Farley, Shakir Mohamed, "Variational Approaches for Auto-Encoding Generative Adversarial Networks", arXiv, 2017

[2]We use the Greek α prefix for α-GAN, as AEGAN and most other Latin prefixes seem to have been taken https://deephunt.in/the-gan-zoo-79597dc8c347.

Three Categories of GAN

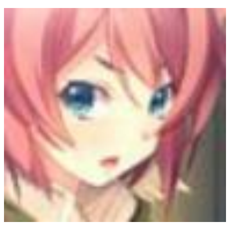
1. *Generation*

$$\begin{bmatrix} -0.3 \\ 0.1 \\ \vdots \\ 0.9 \end{bmatrix}$$

random vector → Generator →

image

2. *Conditional Generation*



blue eyes,
red hair,
short hair

paired data

"Girl with red hair"
text

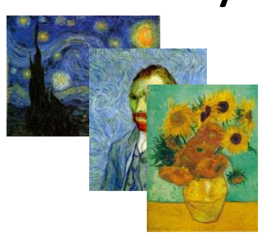
→ Generator →

image

3. *Unsupervised Conditional Generation*

domain x

domain y



unpaired data

x

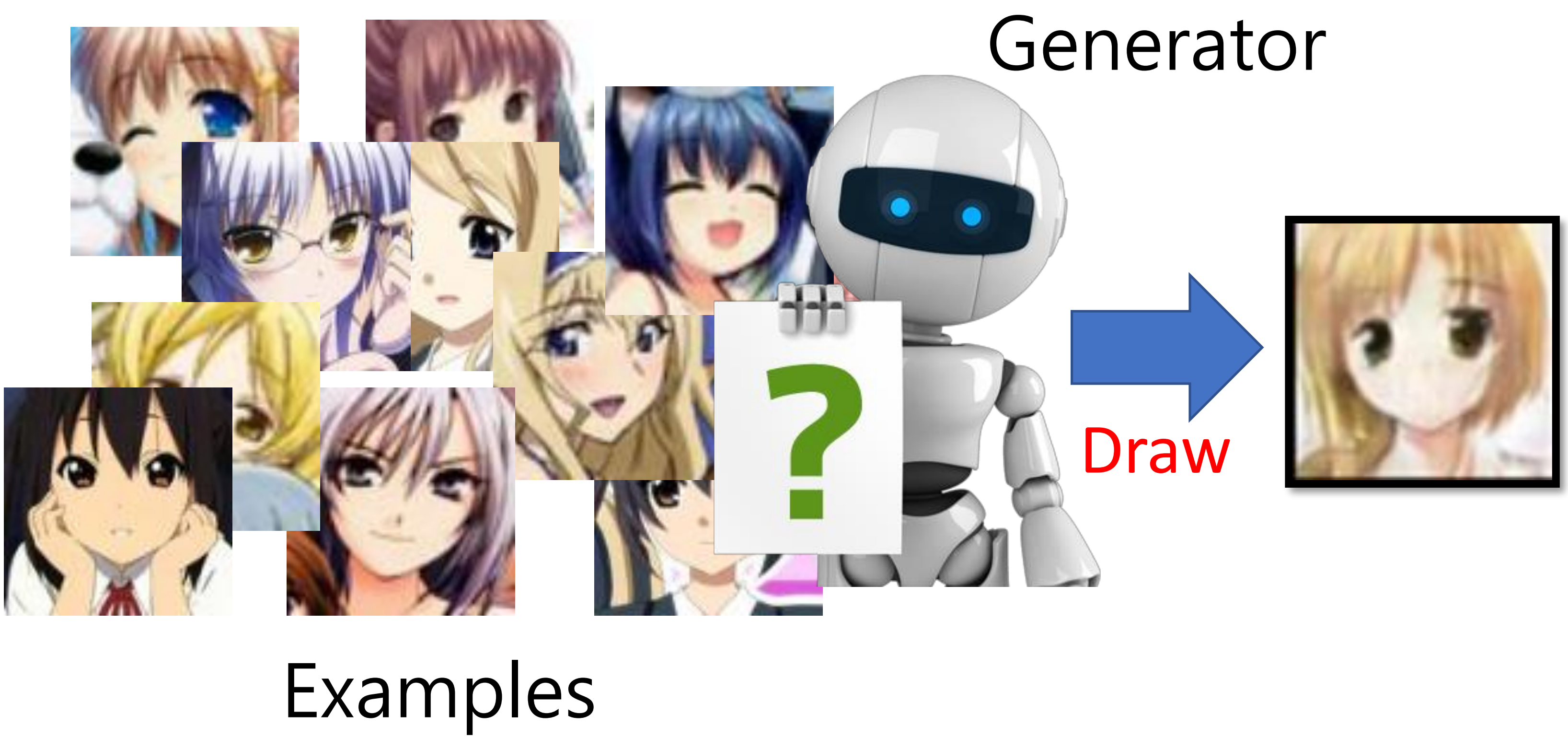
Photo

→ Generator →

y

Vincent van Gogh's style

Anime Face Generation



Powered by: http://mattya.github.io/chainer-DCGAN/

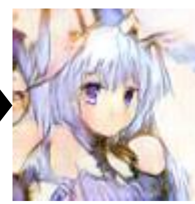
Basic Idea of GAN

It is a neural network (NN), or a function.

vector → Generator → image

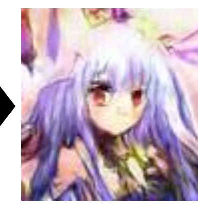
high dimensional vector

$\begin{bmatrix} 0.1 \\ -3 \\ \vdots \\ 2.4 \\ 0.9 \end{bmatrix}$ → Generator →



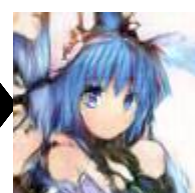
Each dimension of input vector represents some characteristics.

$\begin{bmatrix} \boxed{3} \\ -3 \\ \vdots \\ 2.4 \\ 0.9 \end{bmatrix}$ → Generator →



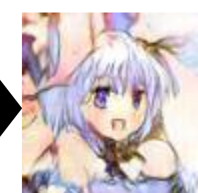
Longer hair

$\begin{bmatrix} 0.1 \\ 2.1 \\ \vdots \\ \boxed{5.4} \\ 0.9 \end{bmatrix}$ → Generator →



blue hair

$\begin{bmatrix} 0.1 \\ -3 \\ \vdots \\ 2.4 \\ \boxed{3.5} \end{bmatrix}$ → Generator →



Open mouth

Basic Idea of GAN

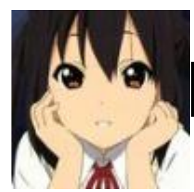
It is a neural network (NN), or a function.

image → Discri-minator → scalar

scalar

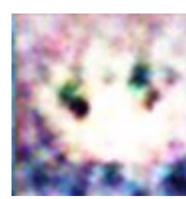
Larger value means real, smaller value means fake.

Discri-minator → 1.0



Discri-minator → 1.0

Discri-minator → 0.1



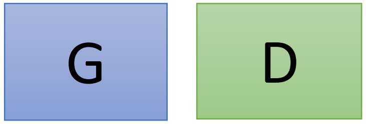
Discri-minator → 0.1

Outline

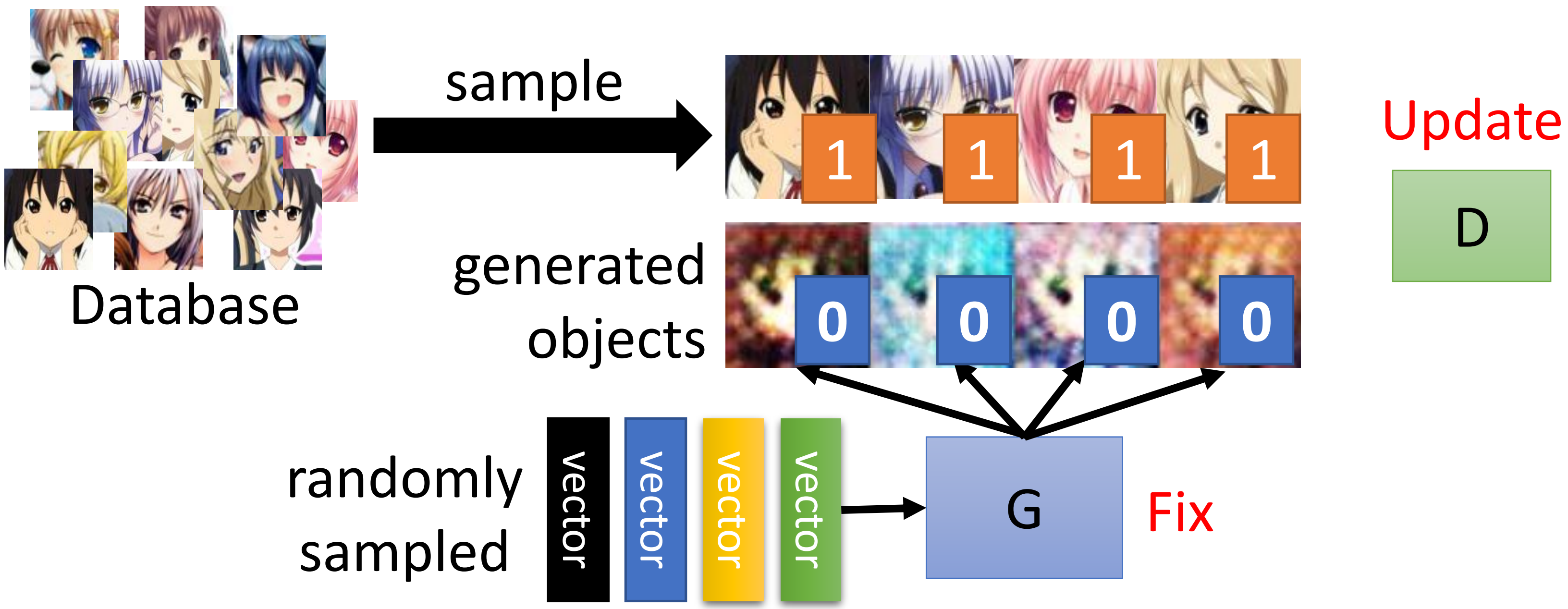
1. Introduction to GANs
2. **Brief theoretical overview of GANs**
3. Overview of GANs in Sequence Generation
4. SeqGAN
5. Other recent work: Unsupervised Conditional Sequence Generation

Algorithm

- Initialize generator and discriminator
- In each training iteration:



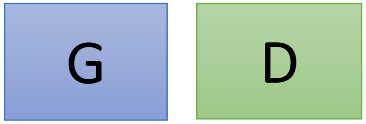
Step 1: Fix generator G, and update discriminator D



Discriminator learns to assign high scores to real objects and low scores to generated objects.

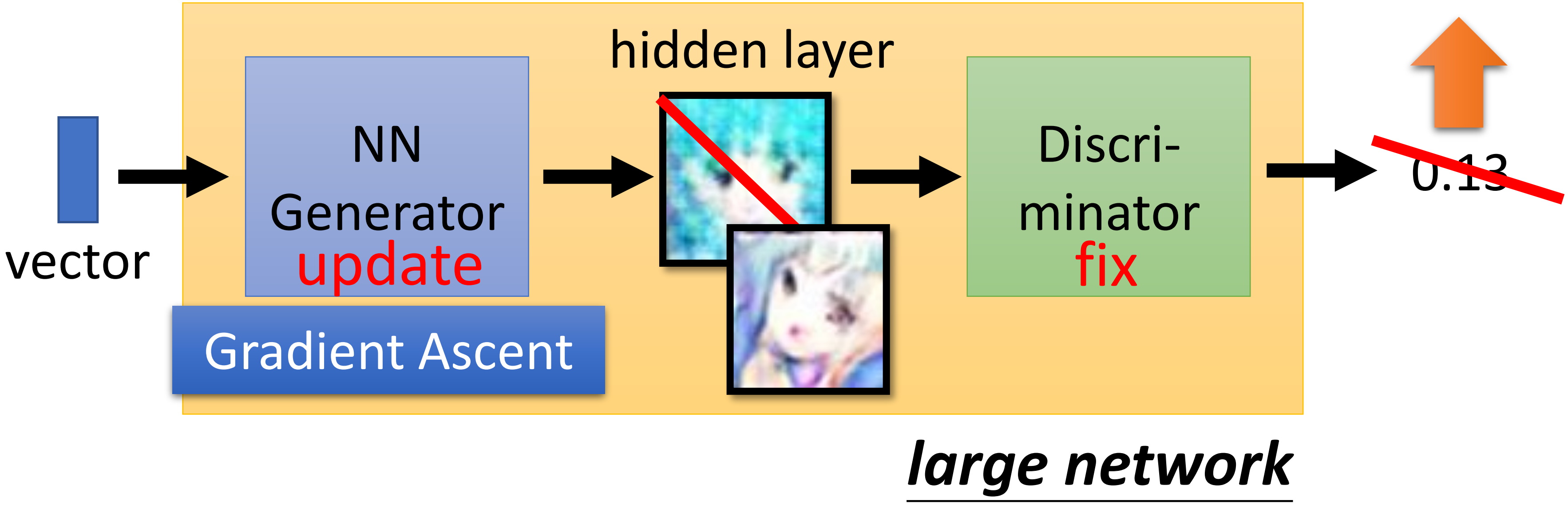
Algorithm

- Initialize generator and discriminator
- In each training iteration:



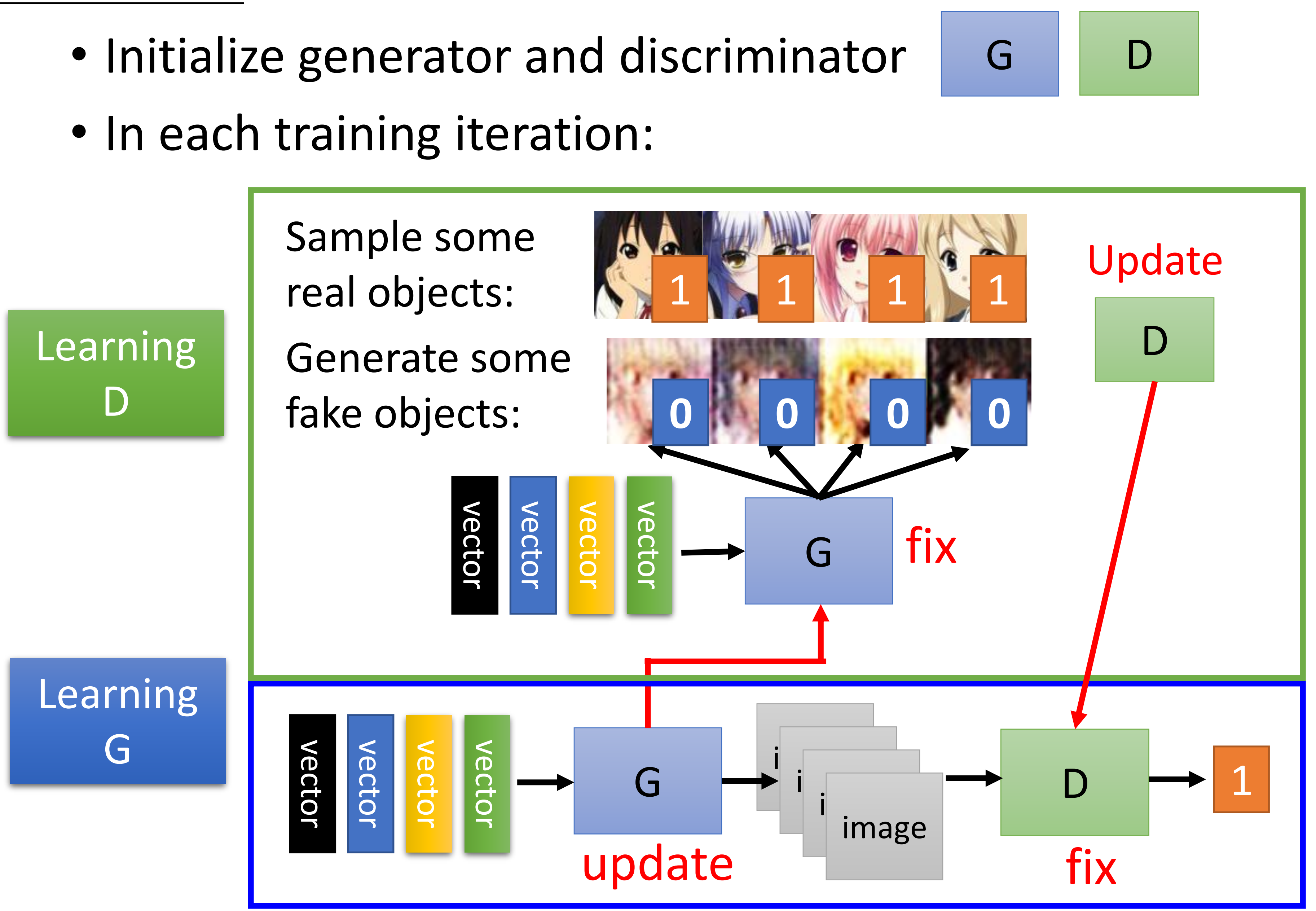
Step 2: Fix discriminator D, and update generator G

Generator learns to "fool" the discriminator

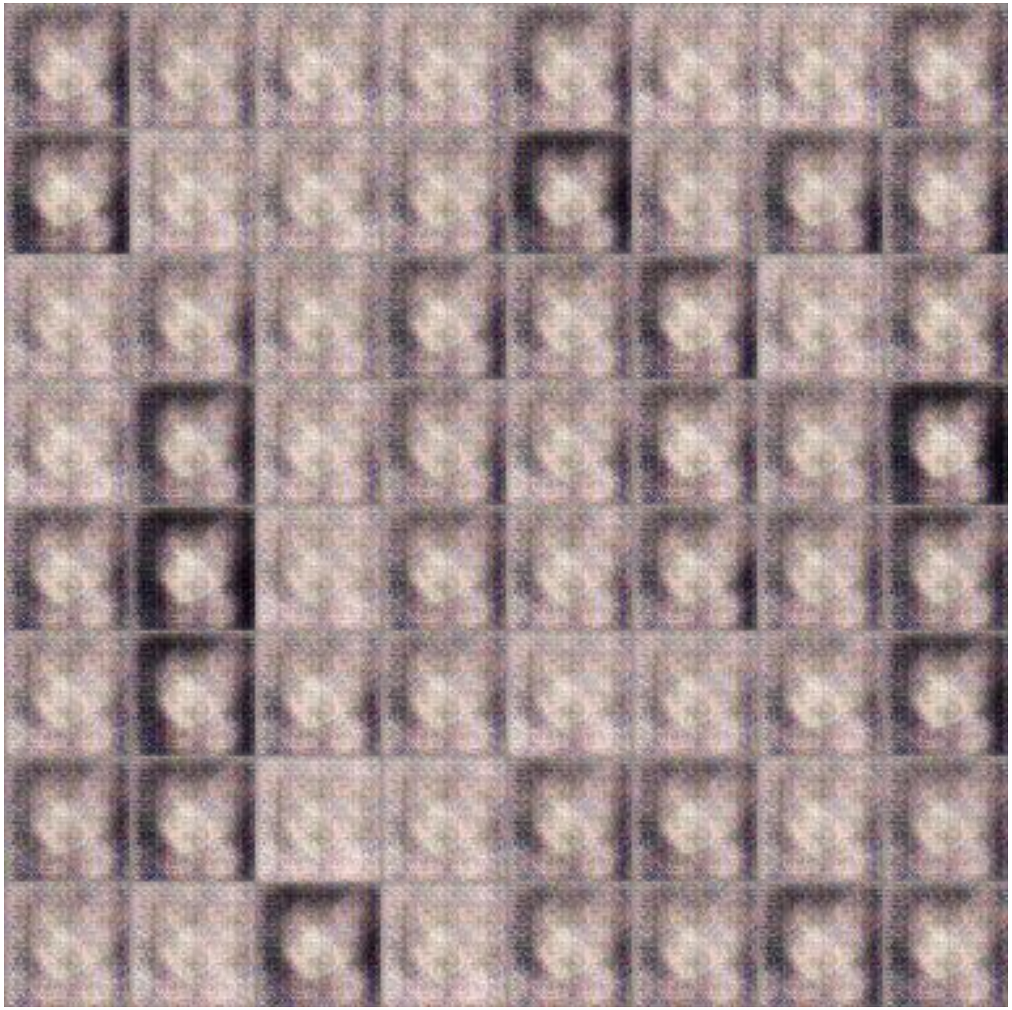


Algorithm

- Initialize generator and discriminator
- In each training iteration:



Anime Face Generation



100 updates

Source of training data: https://zhuanlan.zhihu.com/p/24767059

Anime Face Generation

1000 updates

Anime Face Generation

2000 updates

Anime Face Generation

5000 updates

Anime Face Generation

10,000 updates

Anime Face Generation

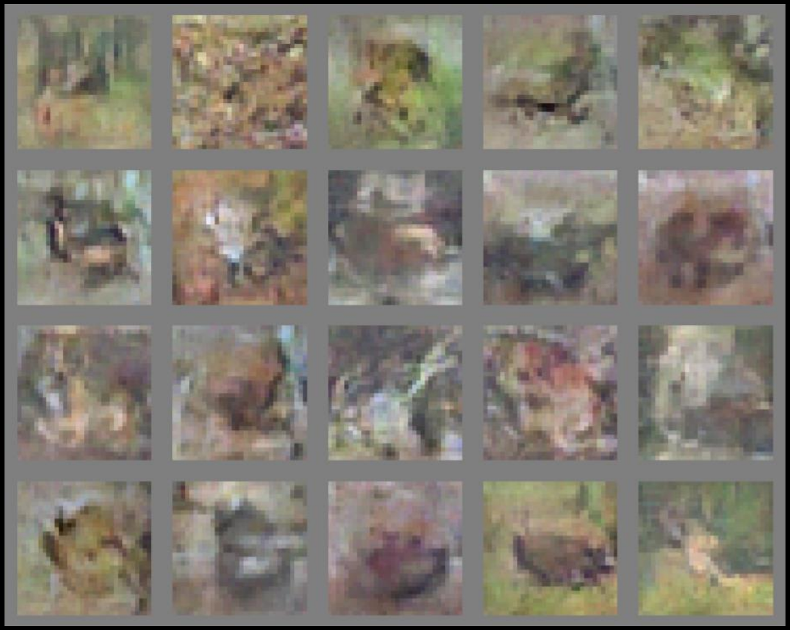
20,000 updates

Anime Face Generation

50,000 updates

In 2019, with StyleGAN ……

Source of video:
https://www.gwern.net/Faces

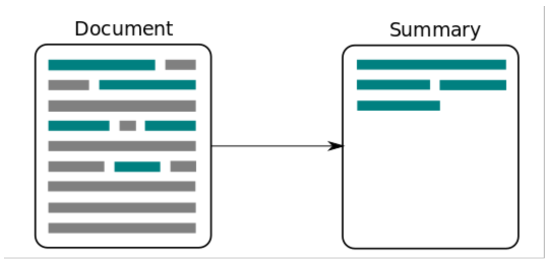


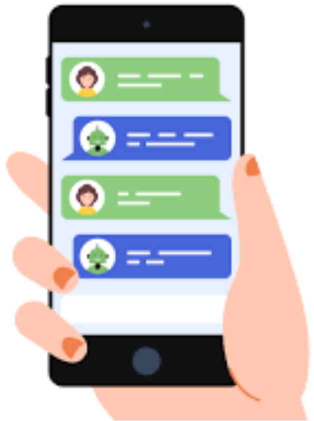
The first GAN

[Ian J. Goodfellow, et al., NIPS, 2014]

Outline

1. Introduction to GANs
2. Brief theoretical overview of GANs
3. **Overview of GANs in Sequence Generation**
   1. **Reinforcement Learning**
   2. GAN + RL
4. SeqGAN
5. Other recent work: Unsupervised Conditional Sequence Generation

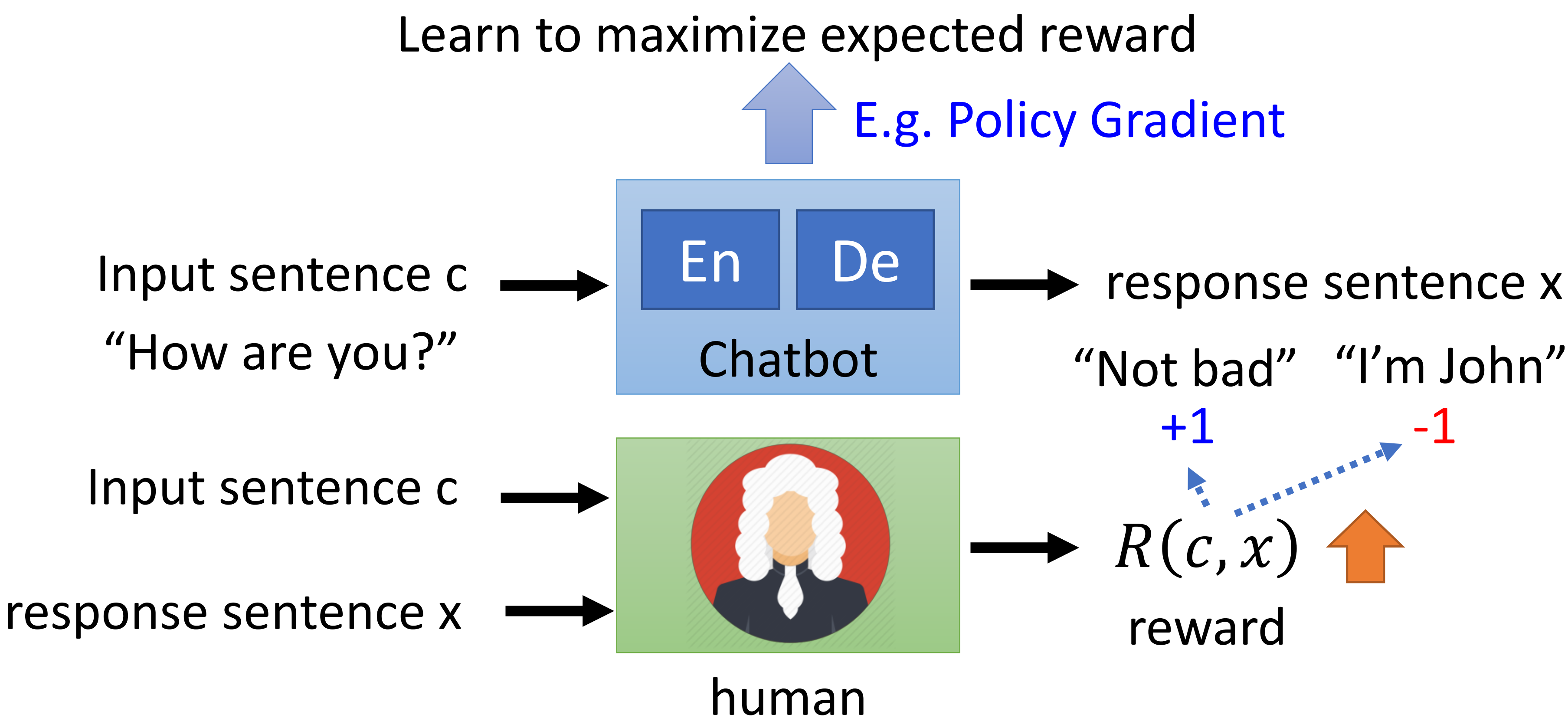




NLP tasks usually involve **Sequence Generation**

How to use GAN to improve sequence generation?

Reinforcement Learning



[Li, et al., EMNLP, 2016]

Policy Gradient

θ^t

(c^1, x^1) $R(c^1, x^1)$

(c^2, x^2) $R(c^2, x^2)$

⋮ ⋮

(c^N, x^N) $R(c^N, x^N)$

$$\theta^{t+1} \leftarrow \theta^t + \eta \boxed{\nabla \bar{R}_{\theta^t}}$$

$$\frac{1}{N} \sum_{i=1}^{N} R(c^i, x^i) \nabla log P_{\theta^t}(x^i | c^i)$$

$R(c^i, x^i)$ is positive

Updating θ to increase $P_\theta(x^i | c^i)$

$R(c^i, x^i)$ is negative

Updating θ to decrease $P_\theta(x^i | c^i)$

Policy Gradient

	Maximum Likelihood	Reinforcement Learning - Policy Gradient
Objective Function	$\frac{1}{N}\sum_{i=1}^{N} logP_\theta(\hat{x}^i\|c^i)$	$\frac{1}{N}\sum_{i=1}^{N} R(c^i,x^i)logP_\theta(x^i\|c^i)$
Gradient	$\frac{1}{N}\sum_{i=1}^{N} \nabla logP_\theta(\hat{x}^i\|c^i)$	$\frac{1}{N}\sum_{i=1}^{N} R(c^i,x^i)\nabla logP_\theta(x^i\|c^i)$
Training Data	$\{(c^1,\hat{x}^1),\dots,(c^N,\hat{x}^N)\}$ $R(c^i,\hat{x}^i)=1$	$\{(c^1,x^1),\dots,(c^N,x^N)\}$ obtained from interaction weighted by $R(c^i,x^i)$

Outline

1. Introduction to GANs
2. Brief theoretical overview of GANs
3. **Overview of GANs in Sequence Generation**
    1. Reinforcement Learning
    2. **GAN + RL**
4. SeqGAN
5. Other recent work: Unsupervised Conditional Sequence Generation

Why we need GAN?

- Chat-bot as example

Output:	Not bad	I'm John.
Human	better	
Training Criterion		better

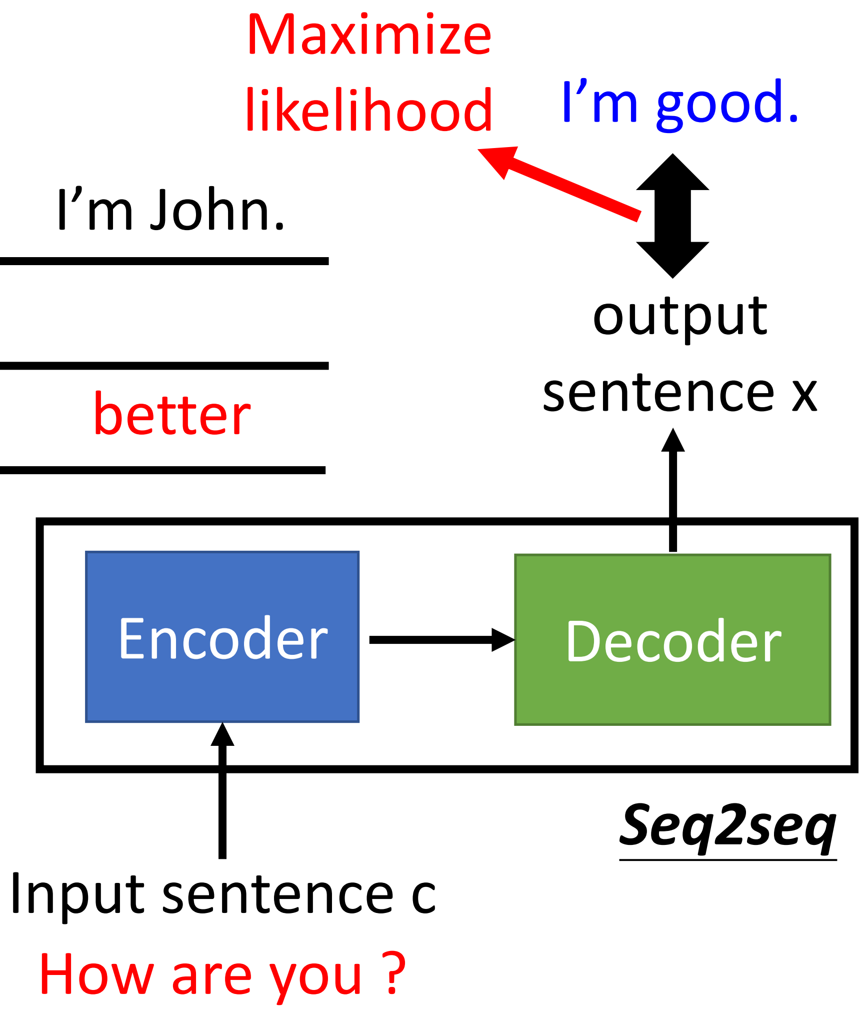
Training data:

⋮

A: How are you ?

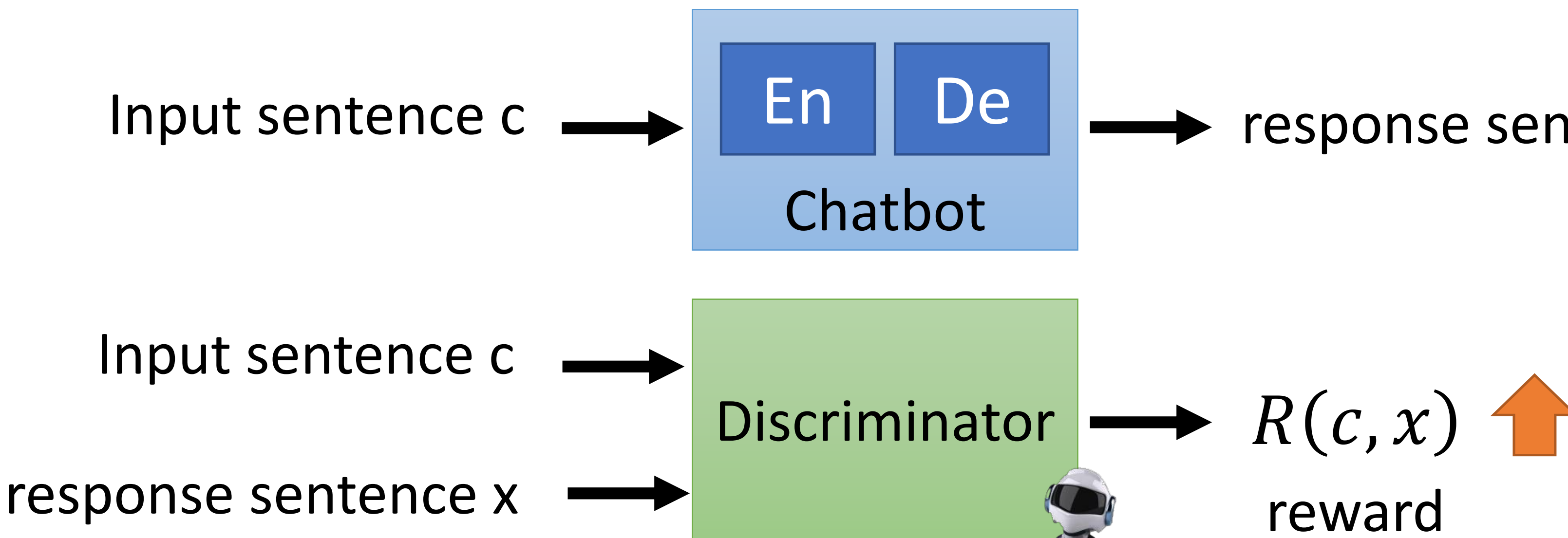
B: I'm good.

⋮



Conditional GAN

However, there is an issue when you train your generator.



Replace human evaluation with **machine evaluation**

[Li, et al., EMNLP, 2017]

Three Categories of Solutions

Gumbel-softmax

- [Matt J. Kusner, et al., arXiv, 2016][Weili Nie, et al. ICLR, 2019]

Continuous Input for Discriminator

- [Sai Rajeswar, et al., arXiv, 2017][Ofir Press, et al., ICML workshop, 2017][Zhen Xu, et al., EMNLP, 2017][Alex Lamb, et al., NIPS, 2016][Yizhe Zhang, et al., ICML, 2017]

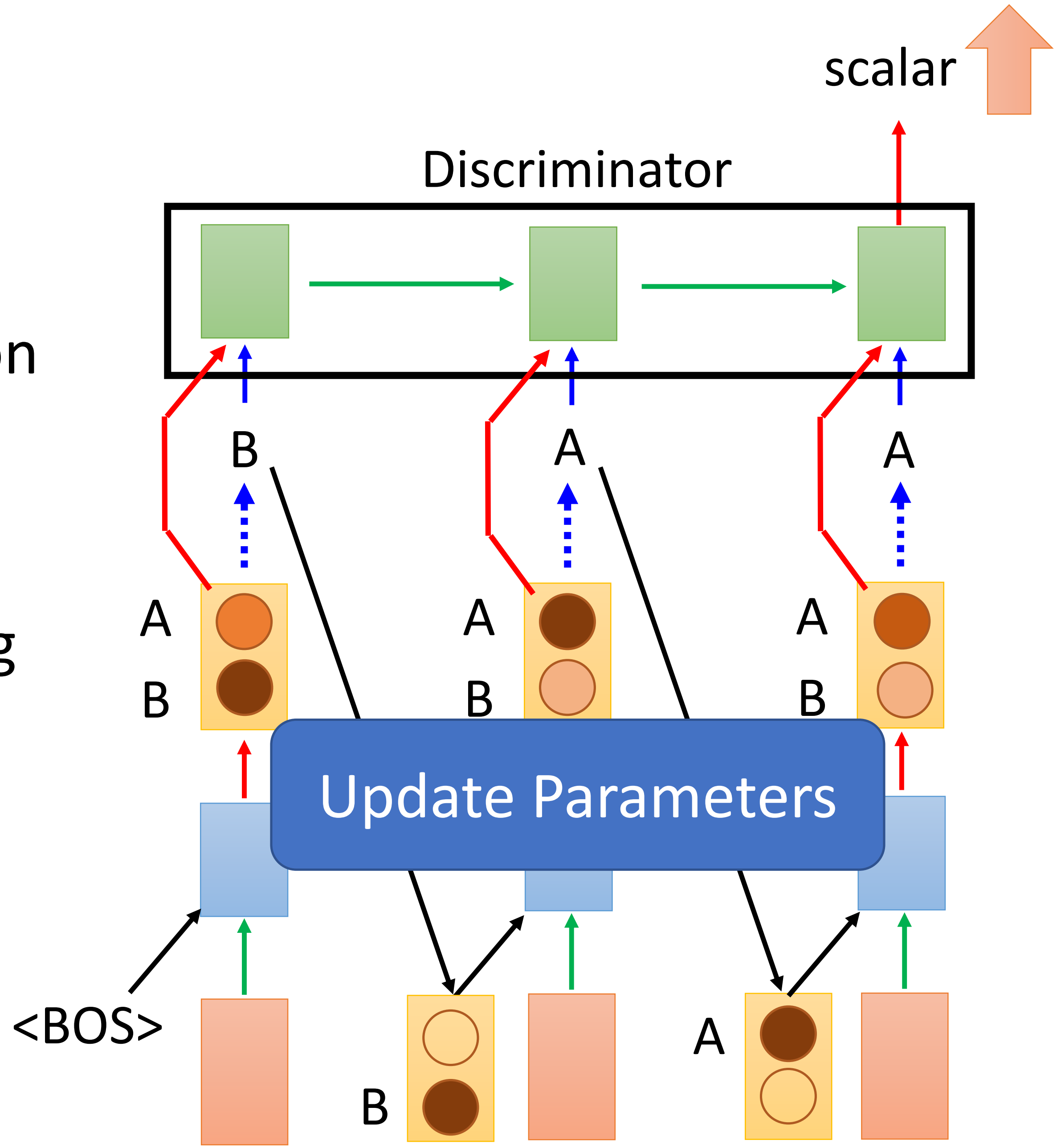
Reinforcement Learning

- [Yu, et al., AAAI, 2017][Li, et al., EMNLP, 2017][Tong Che, et al, arXiv, 2017][Jiaxian Guo, et al., AAAI, 2018][Kevin Lin, et al, NIPS, 2017][William Fedus, et al., ICLR, 2018]

Use the distribution as the input of discriminator

Avoid the sampling process

We can do backpropagation now.



What is the problem?

Discriminator with constraint (e.g. WGAN) can be helpful.

- Real sentence

1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

- Generated

0.9	0.1	0.1	0	0
0.1	0.9	0.1	0	0
0	0	0.7	0.1	0
0	0	0.1	0.8	0.1
0	0	0	0.1	0.9

Can never be 1-hot

Discriminator can immediately find the difference.

Three Categories of Solutions

Gumbel-softmax

- [Matt J. Kusner, et al., arXiv, 2016][Weili Nie, et al. ICLR, 2019]

Continuous Input for Discriminator

- [Sai Rajeswar, et al., arXiv, 2017][Ofir Press, et al., ICML workshop, 2017][Zhen Xu, et al., EMNLP, 2017][Alex Lamb, et al., NIPS, 2016][Yizhe Zhang, et al., ICML, 2017]

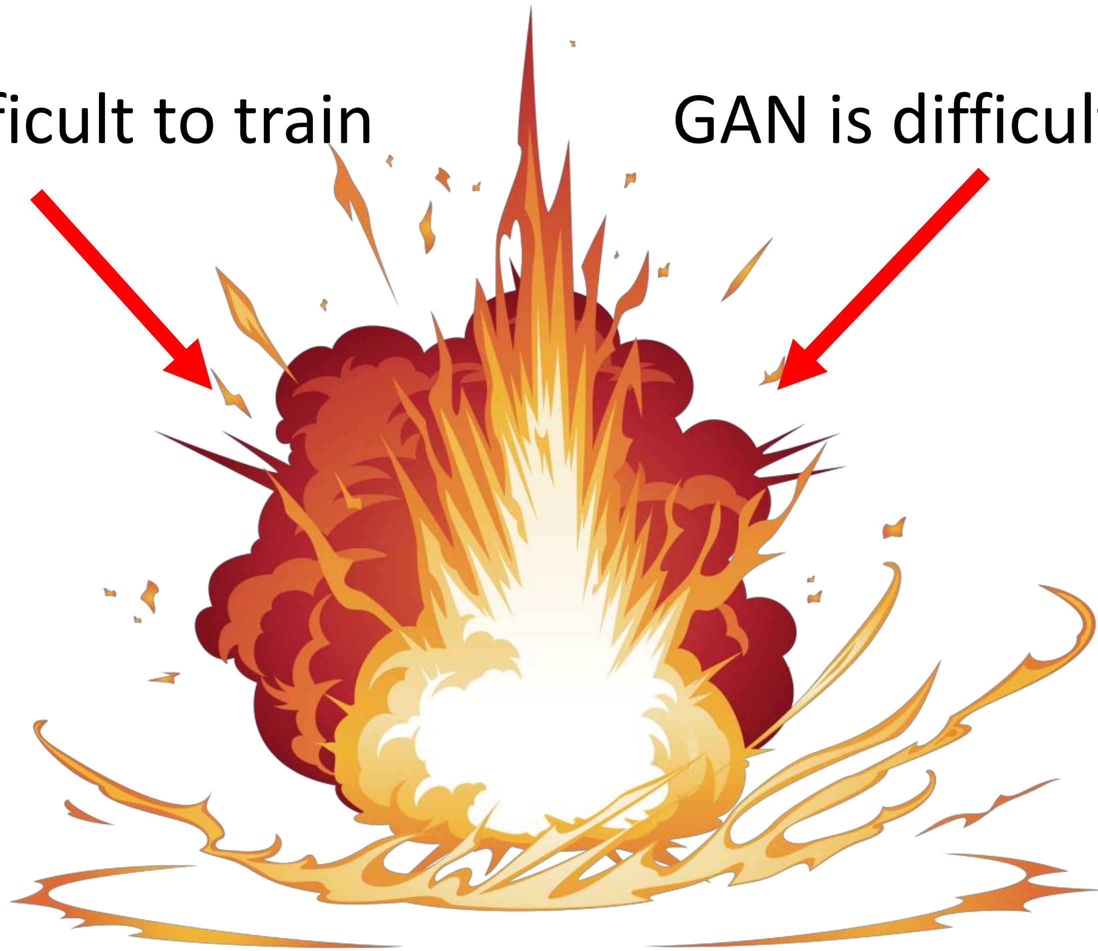
Reinforcement Learning

- [Yu, et al., AAAI, 2017][Li, et al., EMNLP, 2017][Tong Che, et al, arXiv, 2017][Jiaxian Guo, et al., AAAI, 2018][Kevin Lin, et al, NIPS, 2017][William Fedus, et al., ICLR, 2018]

Tips for Sequence Generation GAN

RL is difficult to train

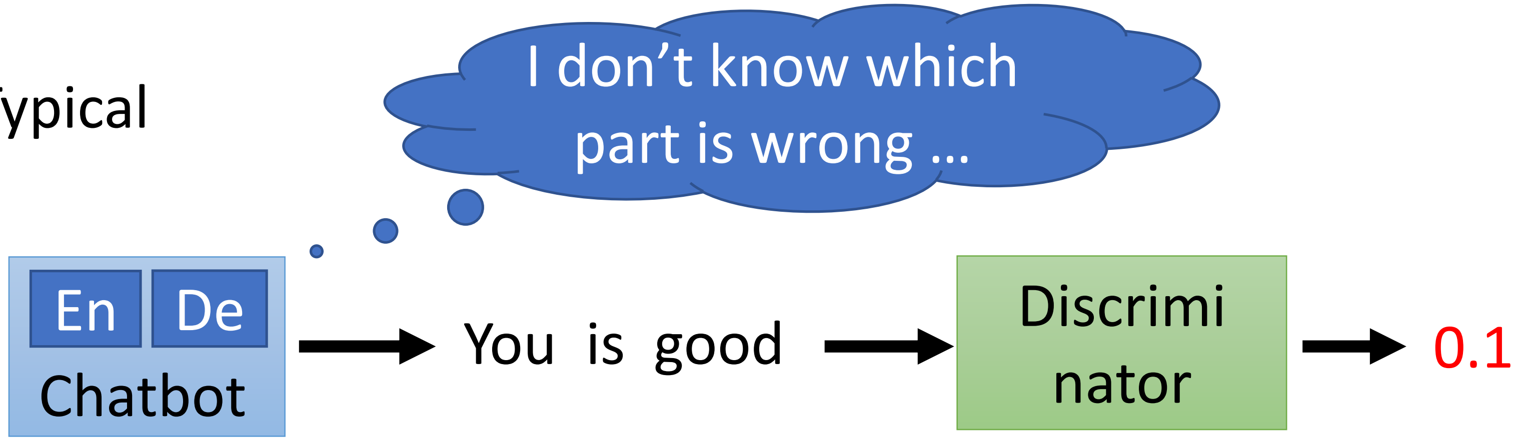
GAN is difficult to train



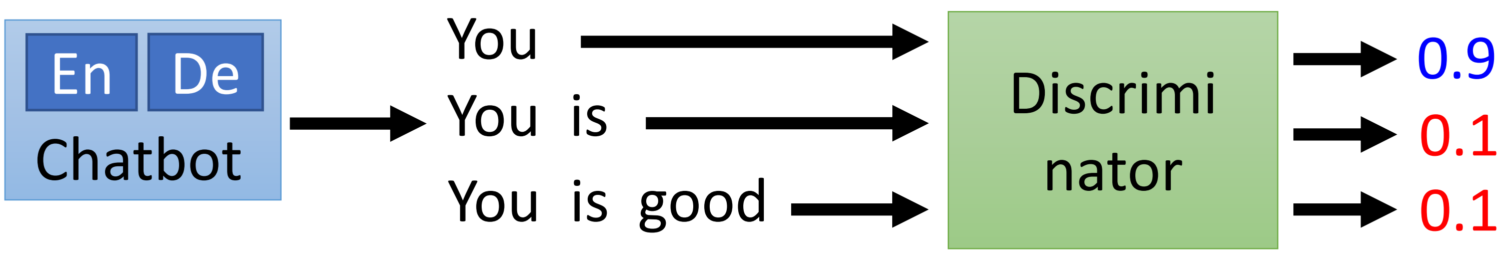
Sequence Generation GAN (RL+GAN)

Tips for Sequence Generation GAN

- Typical

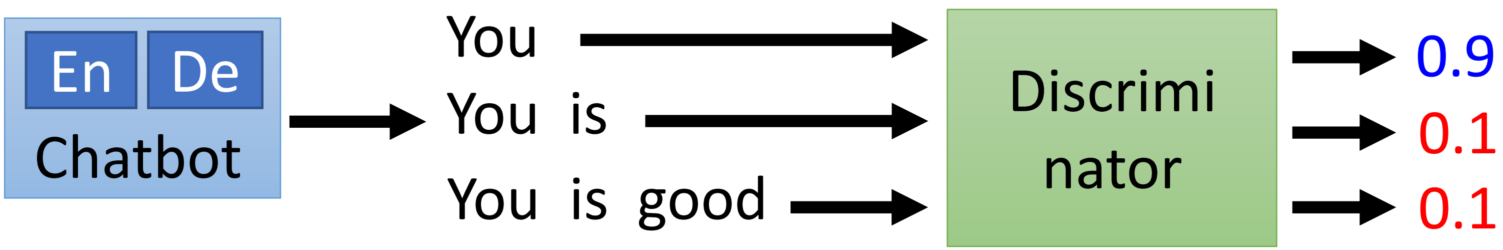


- Reward for Every Generation Step



Tips for Sequence Generation GAN

- Reward for Every Generation Step



Method 1. Monte Carlo (MC) Search [Yu, et al., AAAI, 2017]

Method 2. Discriminator For Partially Decoded Sequences [Li, et al., EMNLP, 2017]

Method 3. Step-wise evaluation [Tual, Lee, TASLP, 2019][Xu, et al., EMNLP, 2018][William Fedus, et al., ICLR, 2018]

Outline

1. Introduction to GANs
2. Brief theoretical overview of GANs
3. Overview of GANs in Sequence Generation
4. **SeqGAN**
5. Other recent work: Unsupervised Conditional Sequence Generation

Task

1. Given a dataset of real-world structured sequences, train a generative model G_θ to produce sequences that mimic the real ones.
2. We want G_θ to fit the unknown true data distribution $p_{\text{true}}(y_t / Y_{1:t-1})$, which is only revealed by the given dataset $D = \{Y_{1:T}\}$.

- Traditional objective: maximum likelihood estimation (MLE)

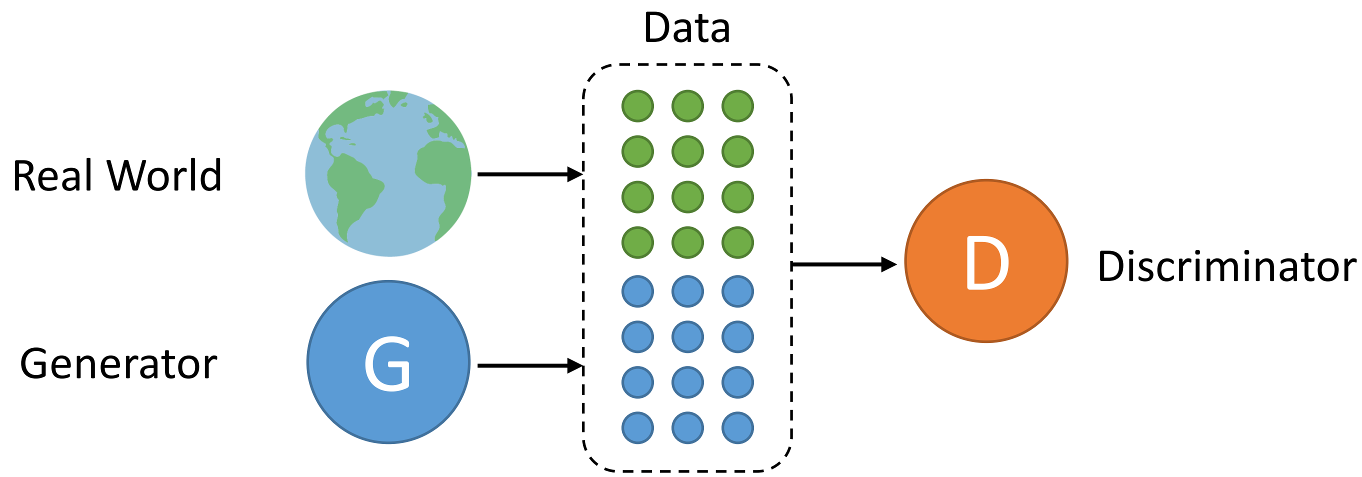
$$\max_{\theta} \frac{1}{|D|} \sum_{Y_{1:T} \in D} \sum_{t} \log[G_\theta(y_t | Y_{1:t-1})]$$

  - Check whether a true data is with a high mass density of the learned model

- Suffer from so-called $exposure\ \ bias$ in the inference stag:

Training	Inference
Update the model as follows:	When generating the next token y_t, sample from:
$\max_{\theta} \mathbb{E}_{Y \sim p_{\text{true}}} \sum_t \log G_\theta(y_t \vert \boxed{Y_{1:t-1}})$	$G_\theta(\hat{y}_t \vert \boxed{\hat{Y}_{1:t-1}})$
The **real** prefix	The **guessed** prefix

A promising method: Generative Adversarial Nets (GANs)

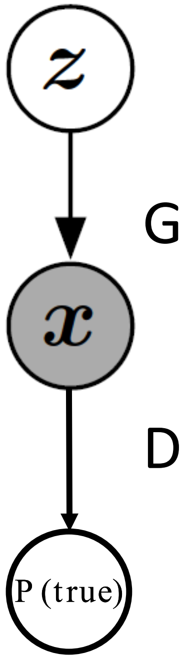


- Discriminator tries to correctly distinguish the true data and the fake model-generated data
- Generator tries to generate high-quality data to fool discriminator
- Ideally, when D cannot distinguish the true and generated data, G nicely fits the true underlying data distribution

[Goodfellow I, Pouget-Abadie J, Mirza M, et al. 2014. Generative adversarial nets. In NIPS 2014.]

Generator Network in GANs

$$x = G(z; \theta^{(G)})$$



- Must be differentiable
- Popular implementation: multi-layer perceptron
- Linked with the discriminator and get guidance from it

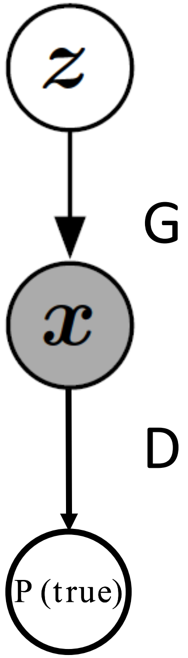
$$\min_G \max_D \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(x)] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z}))]$$

Problem for Discrete Data

- On continuous data, there is direct gradient

$$\nabla_{\theta^{(G)}} \frac{1}{m} \sum_{i=1}^{m} \log(1 - D(G(z^{(i)})))$$

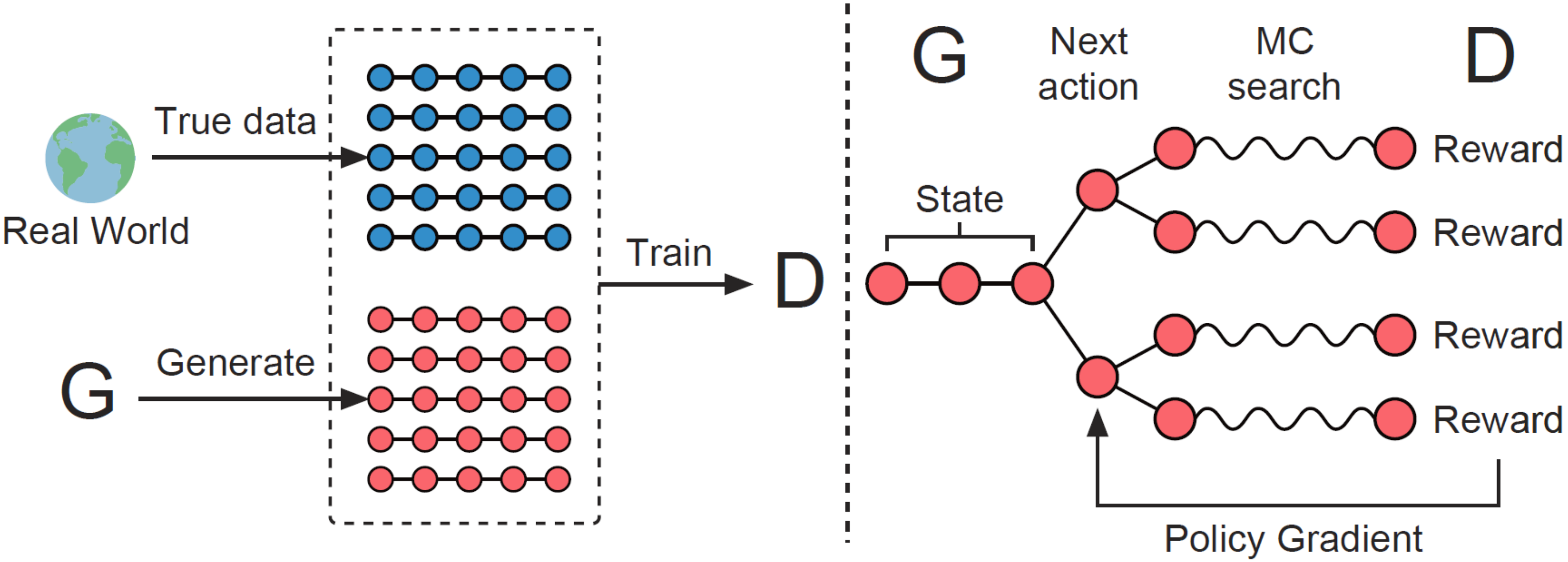
  - Guide the generator to (slightly) modify the output



- No direct gradient on discrete data
  - Text generation example
    - “I caught a penguin in the park”
    - From Ian Goodfellow: “If you output the word ‘penguin’, you can't change that to "penguin + .001" on the next step, because there is no such word as "penguin + .001". You have to go all the way from "penguin" to "ostrich".”

[https://www.reddit.com/r/MachineLearning/comments/40ldq6/generative_adversarial_networks_for_text/]

SeqGAN



- Generator is a reinforcement learning policy $G_\theta(y_t|Y_{1:t-1})$ of generating a sequence
  - decide the next word to generate (action) given the previous ones as the state
- Discriminator provides the reward (i.e. the probability of being true data) $D_\phi(Y^n_{1:T})$ for the sequence

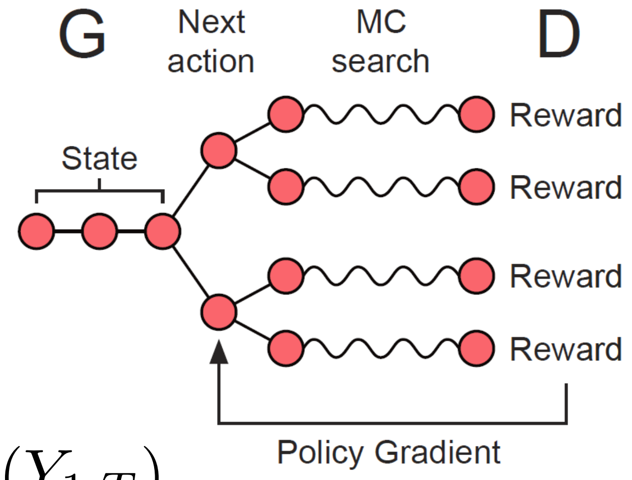
Sequence Generator

- Objective: to maximize the expected reward

$$J(\theta) = \mathbb{E}[R_T | s_0, \theta] = \sum_{y_1 \in \mathcal{Y}} G_\theta(y_1 | s_0) \cdot Q_{D_\phi}^{G_\theta}(s_0, y_1)$$

  - State-action value function $Q_{D_\phi}^{G_\theta}(s, a)$ is the expected accumulative reward that
    - Start from state *s*
    - Taking action *a*
    - And following policy *G* until the end
  - Reward is only on completed sequence (no immediate reward)

$$Q_{D_\phi}^{G_\theta}(s = Y_{1:T-1}, a = y_T) = D_\phi(Y_{1:T})$$



State-Action Value Setting

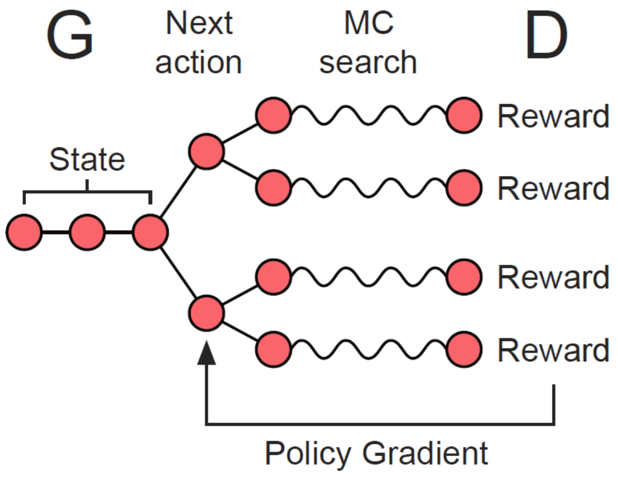
- Reward is only on completed sequence
  - No immediate reward
  - Then the last-step state-action value

$$Q_{D_\phi}^{G_\theta}(s = Y_{1:T-1}, a = y_T) = D_\phi(Y_{1:T})$$

- For intermediate state-action value
  - Use Monte Carlo search to estimate

$$\{Y_{1:T}^1, \ldots, Y_{1:T}^N\} = \mathrm{MC}^{G_\beta}(Y_{1:t}; N)$$

  - Following a roll-out policy G



$$Q_{D_\phi}^{G_\theta}(s = Y_{1:t-1}, a = y_t) =$$

$$\begin{cases} \frac{1}{N}\sum_{n=1}^{N} D_\phi(Y_{1:T}^n),\ Y_{1:T}^n \in \mathrm{MC}^{G_\beta}(Y_{1:t}; N) & \text{for} \quad t < T \\ D_\phi(Y_{1:t}) & \text{for} \quad t = T, \end{cases}$$

Training Sequence Discriminator

- Objective: standard bi-classification

$$\min_{\phi} -\mathbb{E}_{Y\sim p_{\mathrm{data}}}[\log D_\phi(Y)] - \mathbb{E}_{Y\sim G_\theta}[\log(1 - D_\phi(Y))]$$

Training Sequence Generator

- Policy gradient (REINFORCE)

$$\nabla_\theta J(\theta) = \mathbb{E}_{Y_{1:t-1}\sim G_\theta}\Big[\sum_{y_t\in\mathcal{Y}} \nabla_\theta G_\theta(y_t|Y_{1:t-1}) \cdot Q_{D_\phi}^{G_\theta}(Y_{1:t-1}, y_t)\Big]$$

$$\simeq \frac{1}{T}\sum_{t=1}^{T}\sum_{y_t\in\mathcal{Y}} \nabla_\theta G_\theta(y_t|Y_{1:t-1}) \cdot Q_{D_\phi}^{G_\theta}(Y_{1:t-1}, y_t)$$

$$= \frac{1}{T}\sum_{t=1}^{T}\sum_{y_t\in\mathcal{Y}} G_\theta(y_t|Y_{1:t-1})\nabla_\theta \log G_\theta(y_t|Y_{1:t-1}) \cdot Q_{D_\phi}^{G_\theta}(Y_{1:t-1}, y_t)$$

$$= \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_{y_t\sim G_\theta(y_t|Y_{1:t-1})}[\nabla_\theta \log G_\theta(y_t|Y_{1:t-1}) \cdot Q_{D_\phi}^{G_\theta}(Y_{1:t-1}, y_t)],$$

$$\theta \leftarrow \theta + \alpha_h \nabla_\theta J(\theta)$$

[Richard Sutton et al. Policy Gradient Methods for Reinforcement Learning with Function Approximation. NIPS 1999.]

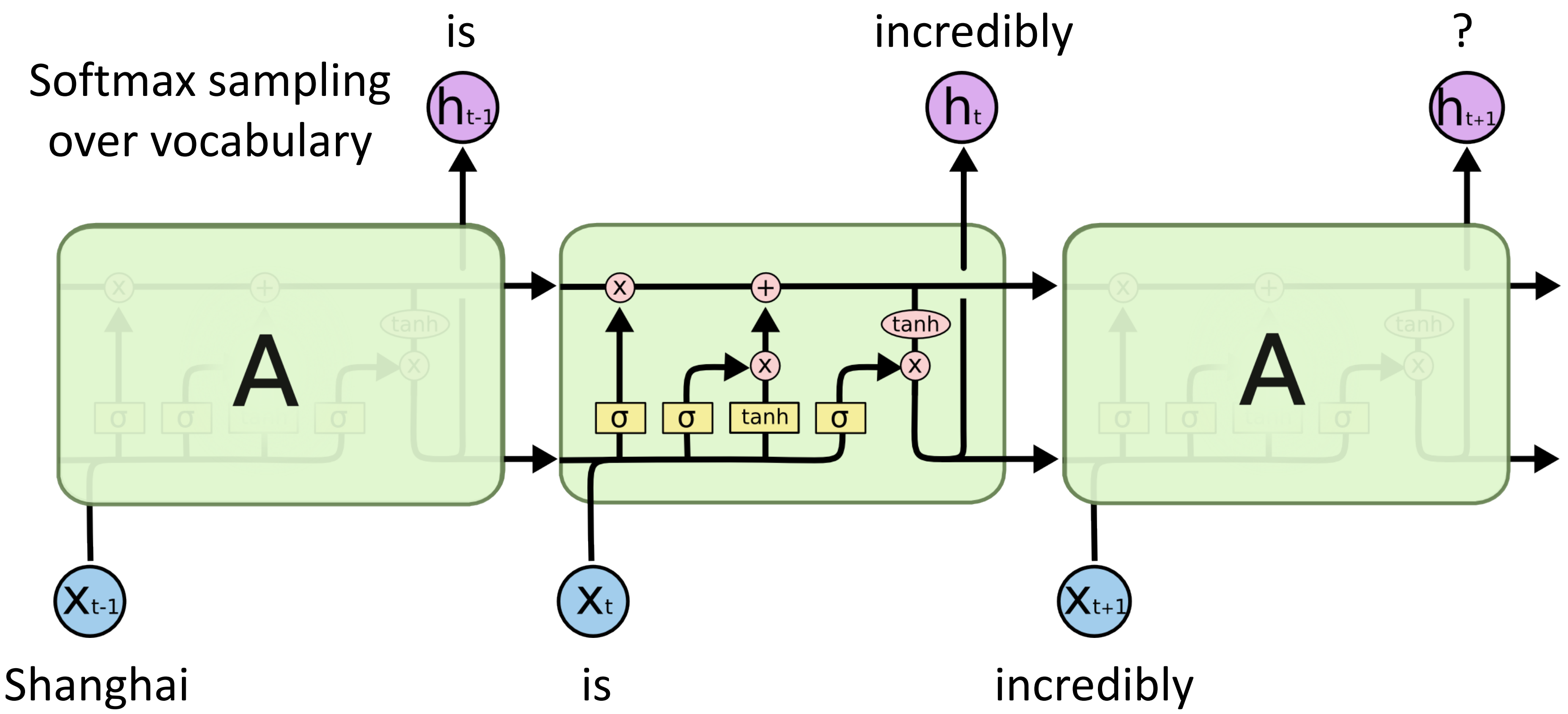
Overall Algorithm

Algorithm 1 Sequence Generative Adversarial Nets

Require: generator policy G_θ; roll-out policy G_β; discriminator D_ϕ; a sequence dataset $\mathcal{S} = \{X_{1:T}\}$

Initialize G_θ, D_ϕ with random weights θ, ϕ.
Pre-train G_θ using MLE on $\mathcal{S}$
$\beta \leftarrow \theta$
Generate negative samples using G_θ for training D_ϕ
Pre-train D_ϕ via minimizing the cross entropy
repeat
  for g-steps **do**
    Generate a sequence $Y_{1:T} = (y_1, \ldots, y_T) \sim G_\theta$
    for t in $1 : T$ **do**
      Compute $Q(a = y_t; s = Y_{1:t-1})$ by Eq. (4)
    end for
    Update generator parameters via policy gradient Eq. (8)
  end for
  for d-steps **do**
    Use current G_θ to generate negative examples and combine with given positive examples $\mathcal{S}$
    Train discriminator D_ϕ for k epochs by Eq. (5)
  end for
  $\beta \leftarrow \theta$
until SeqGAN converges

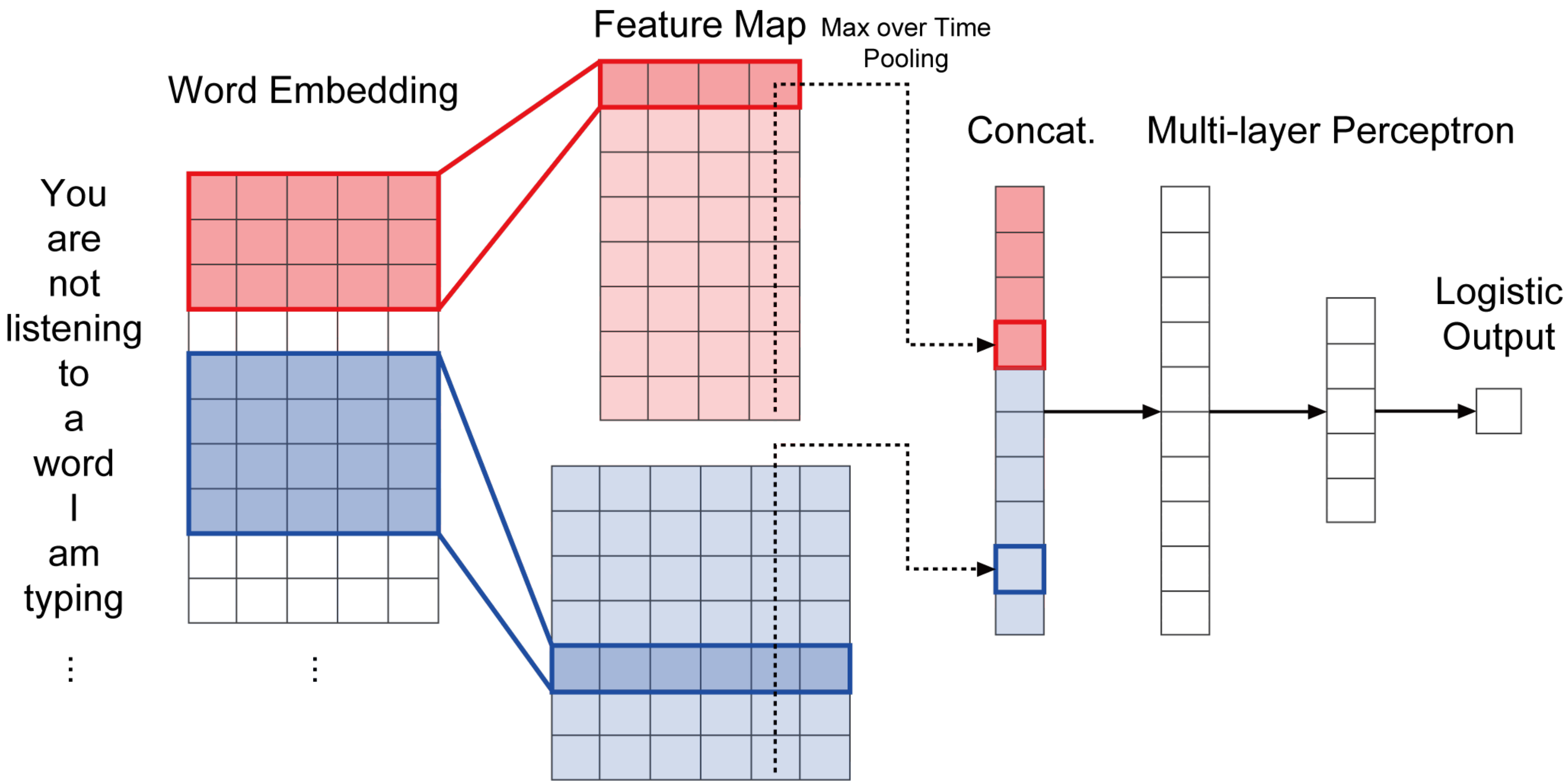
Sequence Generator Model



- RNN with LSTM cells

[Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. Neural computation 9(8):1735–1780.]

Sequence Discriminator Model



[Kim, Y. 2014. Convolutional neural networks for sentence classification. EMNLP 2014.]

Inconsistency of Evaluation and Use

- Given a generator G_θ with a certain generalization ability

$$\mathbb{E}_{x\sim p_{\text{true}}(x)}[\log G_\theta(x)]$$

Evaluation

- Check whether a true data is with a high mass density of the learned model
- Approximated by

$$\max_\theta \frac{1}{|D|}\sum_{x\in D}[\log G_\theta(x)]$$

$$\mathbb{E}_{x\sim G_\theta(x)}[\log p_{\text{true}}(x)]$$

Use

- Check whether a model-generated data is considered as real as possible
- More straightforward but it is hard or impossible to directly calculate $p_{\text{true}}(x)$

Experiments on Synthetic Data

- Evaluation measure with Oracle

$$\mathrm{NLL}_{\mathrm{oracle}} = -\mathbb{E}_{Y_{1:T} \sim G_\theta}\Big[\sum_{t=1}^{T} \log G_{\mathrm{oracle}}(y_t|Y_{1:t-1})\Big]$$

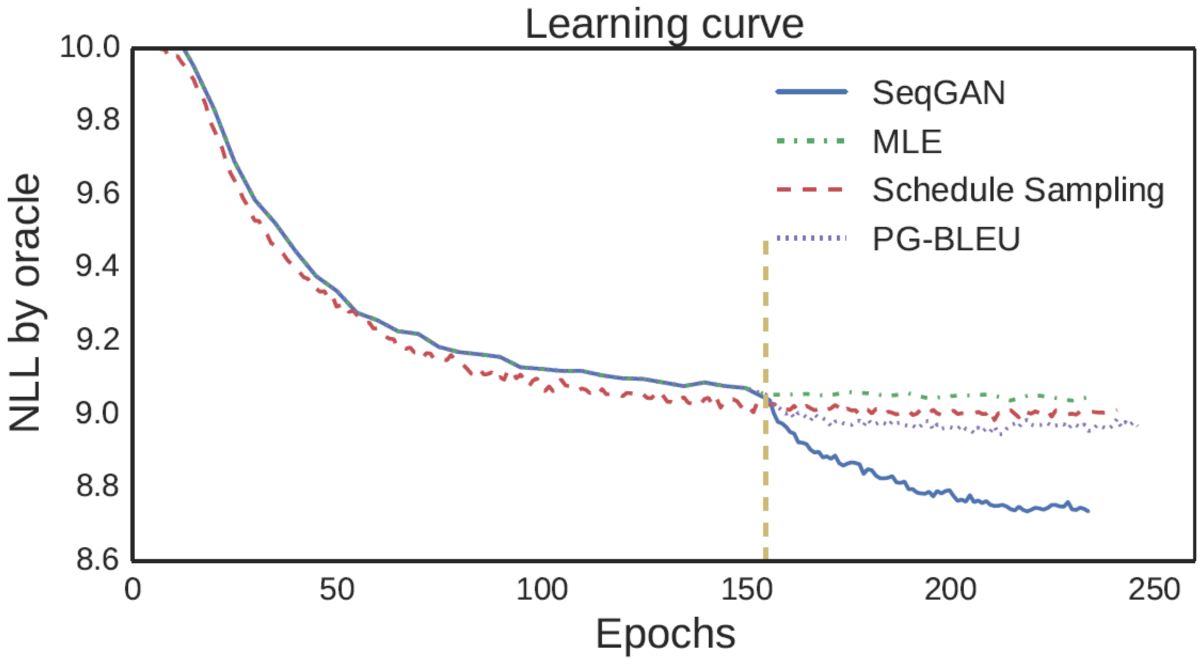
- An oracle model (e.g. the randomly initialized LSTM)
  - Firstly, the oracle model produces some sequences as training data for the generative model
  - Secondly the oracle model can be considered as the human observer to accurately evaluate the perceptual quality of the generative model

Experiments on Synthetic Data

- Evaluation measure with Oracle

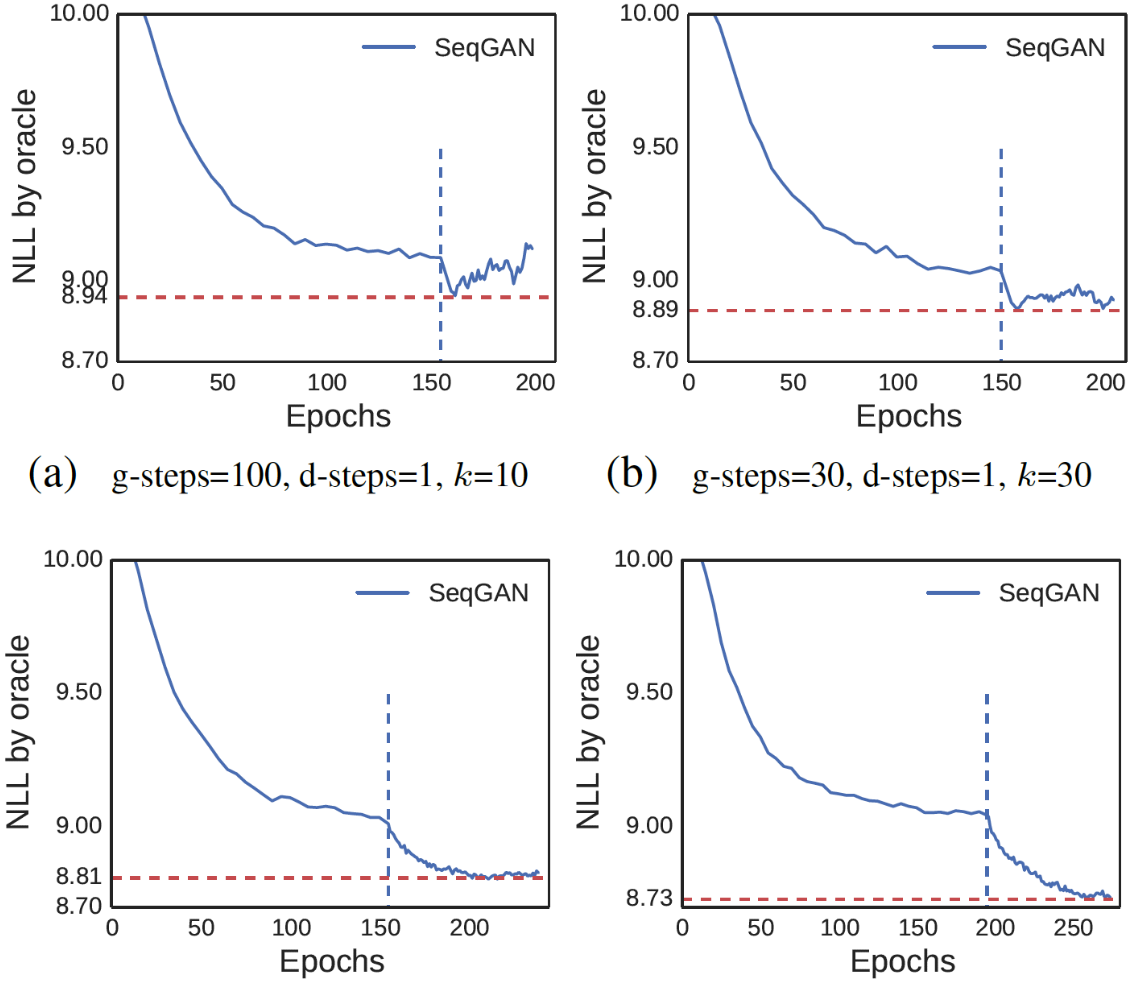
$$\mathrm{NLL}_{\mathrm{oracle}} = -\mathbb{E}_{Y_{1:T} \sim G_\theta}\left[\sum_{t=1}^{T} \log G_{\mathrm{oracle}}(y_t | Y_{1:t-1})\right]$$

Algorithm	Random	MLE	SS	PG-BLEU	SeqGAN
NLL	10.310	9.038	8.985	8.946	**8.736**
p-value	$< 10^{-6}$	$< 10^{-6}$	$< 10^{-6}$	$< 10^{-6}$	



Experiments on Synthetic Data

- The training strategy really matters.



(a) g-steps=100, d-steps=1, k=10

(b) g-steps=30, d-steps=1, k=30

(c) g-steps=1, d-steps=1, k=10

(d) g-steps=1, d-steps=5, k=3

Experiments on Real-World Data

- Chinese poem generation

Algorithm	Human score	p-value	BLEU-2	p-value
MLE	0.4165	0.0034	0.6670	$< 10^{-6}$
SeqGAN	**0.5356**		**0.7389**	
Real data	0.6011		0.746	

- Obama political speech text generation

Algorithm	BLEU-3	p-value	BLEU-4	p-value
MLE	0.519	$< 10^{-6}$	0.416	0.00014
SeqGAN	**0.556**		**0.427**	

- Midi music generation

Algorithm	BLEU-4	p-value	MSE	p-value
MLE	0.9210	$< 10^{-6}$	22.38	0.00034
SeqGAN	**0.9406**		**20.62**	

Experiments on Real-World Data

- Chinese poem generation

Human	Machine
南陌春风早，东邻去日斜。	山夜有雪寒，桂里逢客时。
紫陌追随日，青门相见时。	此时人且饮，酒愁一节梦。
胡风不开花，四气多作雪。	四面客归路，桂花开青竹。

Obama Speech Text Generation

- when he was told of this extraordinary honor that he was the most trusted man in america
- but we also remember and celebrate the journalism that walter practiced a standard of honesty and integrity and responsibility to which so many of you have committed your careers. it's a standard that's a little bit harder to find today
- i am honored to be here to pay tribute to the life and times of the man who chronicled our time.

Human

- i stood here today i have one and most important thing that not on violence throughout the horizon is OTHERS american fire and OTHERS but we need you are a strong source
- for this business leadership will remember now i can't afford to start with just the way our european support for the right thing to protect those american story from the world and
- i want to acknowledge you were going to be an outstanding job times for student medical education and warm the republicans who like my times if he said is that brought the

Machine

Issues

- Gradient vanishing problem:
  - Discriminator is trained to be much stronger than the generator
  - Extremely hard for the generator to have any actual updates
  - Any output instances of the generator will be scored as almost 0.

- Mode Collapse:
  - Due to REINFORCE algorithm
  - Probability of sampling particular tokens earning high evaluation from D.
  - G only manages to mimic a limited part of the target distribution

Summary

- We proposed a sequence generation method, called SeqGAN, to effectively train Generative Adversarial Nets for discrete structured sequences generation via policy gradient.

- Design an experiment framework with oracle evaluation metric to accurately evaluate the “perceptual quality” of model-generated sequences.

Review

- First solid and well-motivated study on using GANs for Discrete Sequences.
- Extensive experimentation on both synthetic and real-world data with convincing results.
- Requires a lot of engineering and hyper-parameter tuning: Pre-training, GAN parameters, g-steps, d-steps MC tree depth etc.

Pros

1. Succeed with RL+GAN / interesting idea [Everyone]
2. Well written [Keshav, Rajas]
3. Mathematical detail [Atishya, Jigyasa]
4. Multiple domains explored [Shubham]
5. Ablation study of train time [Pawan]
6. Pretraining generator with MLE can help reduce high variance in gradient estimate as very less samples are used in each episode. [Jigyasa]
7. The evaluation approach, of using a randomly initialized LSTM as an oracle is a very creative idea that provides a nice way to automatically compare how close the generator distribution is to the actual model of the world. [Rajas]
8. Using CNN for discriminator and getting good results is really noteworthy. [Vipul]

Cons

1. Real World Experiments should include all baselines not just MLE [Keshav, Siddhant, Saransh, Rajas]
2. Difficult to convince the community, given the added complications [Keshav, Atishya, Vipul, Saransh]
3. Needs more examples rather than just loss metrics. What about diversity? [Atishya, Jigyasa, Siddhant, Rajas]
4. Limitation of poems? [Soumya]
5. When to stop training? [Shubham]
6. Using language model as source for synthetic data [Pawan]
7. Doesn't offer any strong paradigm for intermediate reward calculation [Vipul]
8. MCTS not feasible on large datasets. [Lovish]
9. BLEU [Lovish]
10. The generator might start learning sentences in the gold set. [Rajas]

Extensions/Discussion

1. Intermediate Rewards:
    1. K-discriminators trained with partial/complete sequences [Keshav]
    2. K distinct Ds are expensive. Weight sharing [Atishya]
    3. Use LM for intermediate rewards. ”Surprise” value [Rajas/Soumya/Saransh]
2. Pre-trained discriminators (low/med/high):
3. Transformer models/LMs [Shubham]
4. Optimization in continuous space with periodic discrete updates [Pawan]
5. WGAN [Vipul]
6. Information Retrieval [Siddhant]
    1. Won’t work [Saransh]

Outline

1. Introduction to GANs
2. Brief theoretical overview of GANs
3. Overview of GANs in Sequence Generation
4. SeqGAN
5. **Other recent work: Unsupervised Conditional Sequence Generation**

Unsupervised Conditional Sequence Generation

- Text Style Transfer
- Unsupervised Abstractive Summarization
- Unsupervised Translation
- Unsupervised Speech Recognition

Three Categories of Solutions

Gumbel-softmax

- [Matt J. Kusner, et al., arXiv, 2016][Weili Nie, et al. ICLR, 2019]

Continuous Input for Discriminator

- [Sai Rajeswar, et al., arXiv, 2017][Ofir Press, et al., ICML workshop, 2017][Zhen Xu, et al., EMNLP, 2017][Alex Lamb, et al., NIPS, 2016][Yizhe Zhang, et al., ICML, 2017]

Reinforcement Learning

- [Yu, et al., AAAI, 2017][Li, et al., EMNLP, 2017][Tong Che, et al, arXiv, 2017][Jiaxian Guo, et al., AAAI, 2018][Kevin Lin, et al, NIPS, 2017][William Fedus, et al., ICLR, 2018]