SeqGAN

Sequence Generative Adversarial Nets with Policy Gradient

Lantao Yu[†], Weinan Zhang[†], Jun Wang[‡], Yong Yu[†]
[†]Shanghai Jiao Tong University, [‡]University College London

Email: yulantao@apex.sjtu.edu.cn

Feb. 7 2017 at AAAI

Problem Definition

- Given a dataset of real-world structured sequences, train a θ-parameterized generative model G_θ to produce sequences that mimic the real ones.
- In other words, we want G_θ to fit the unknown true data distribution $p_{\mathrm{true}}(y_t|Y_{1:t-1})$, 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

Update the model as follows:

$$\max_{\theta} \mathbb{E}_{Y \sim p_{\text{true}}} \sum_{t} \log G_\theta(y_t | \boxed{Y_{1:t-1}})$$

The **real** prefix

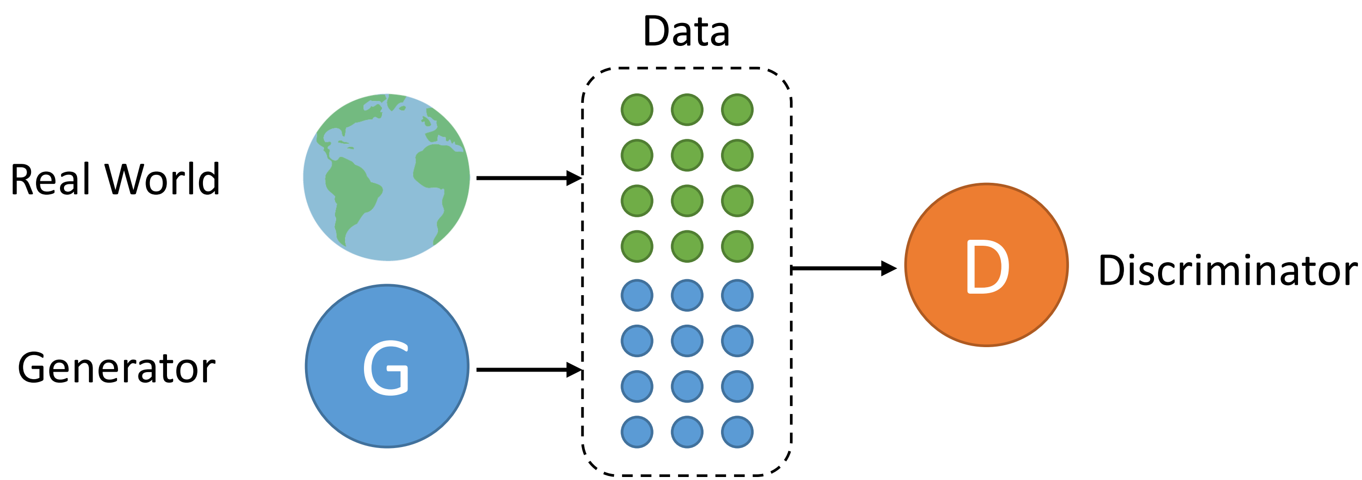
Inference

When generating the next token y_t , sample from:

$$G_\theta(\hat{y}_t | \boxed{\hat{Y}_{1:t-1}})$$

The **guessed** prefix

A promising method: Generative Adversarial Nets (GANs)

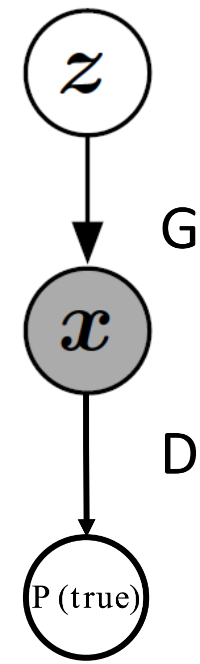


- Discriminator tries to correctly distinguish the true data and the fake model-generated data
- Generator tries to generate high-quality data to fool discriminator
- Ideally, when D cannot distinguish the true and generated data, G nicely fits the true underlying data distribution

[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

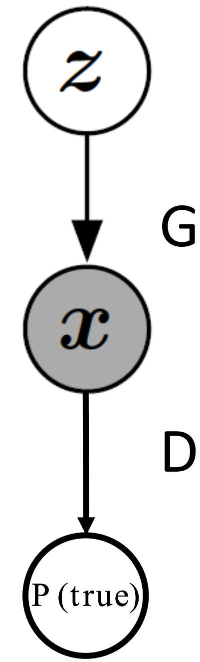
$$\min_G \max_D \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(x)] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

Problem for Discrete Data

- On continuous data, there is direct gradient

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

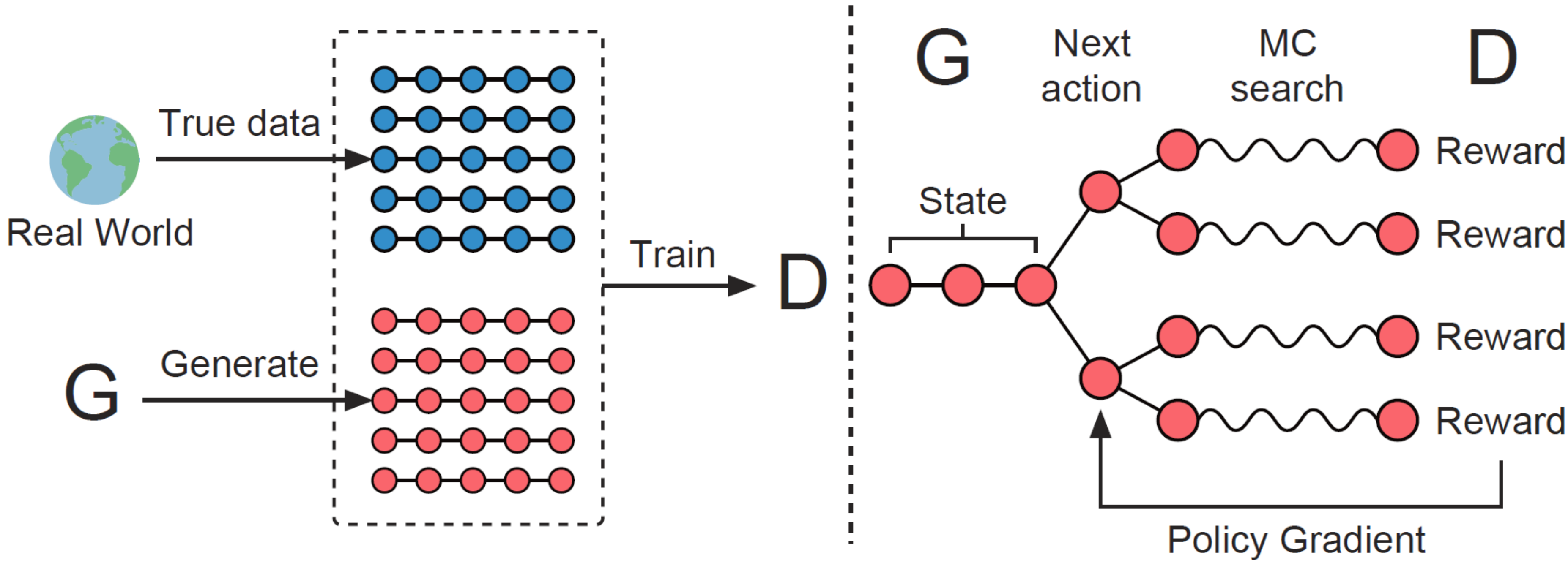
  - Guide the generator to (slightly) modify the output



- 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

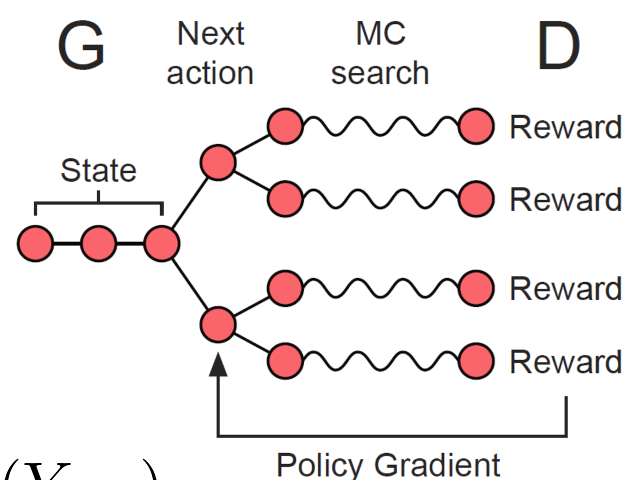
Sequence Generator

- 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

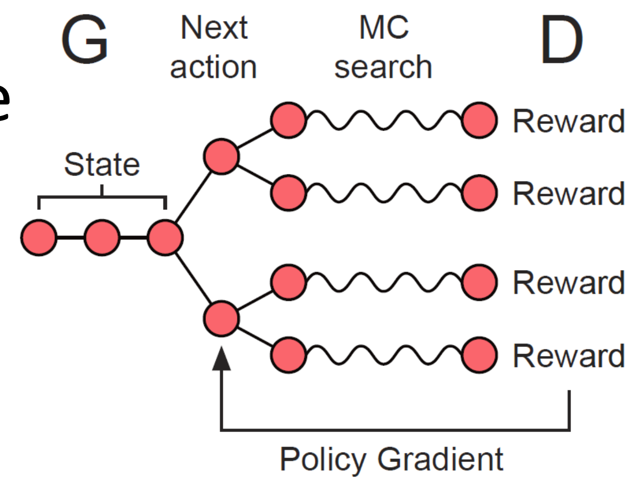
- Reward is only on completed sequence
  - No immediate reward
  - Then the last-step state-action value

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

- 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.]

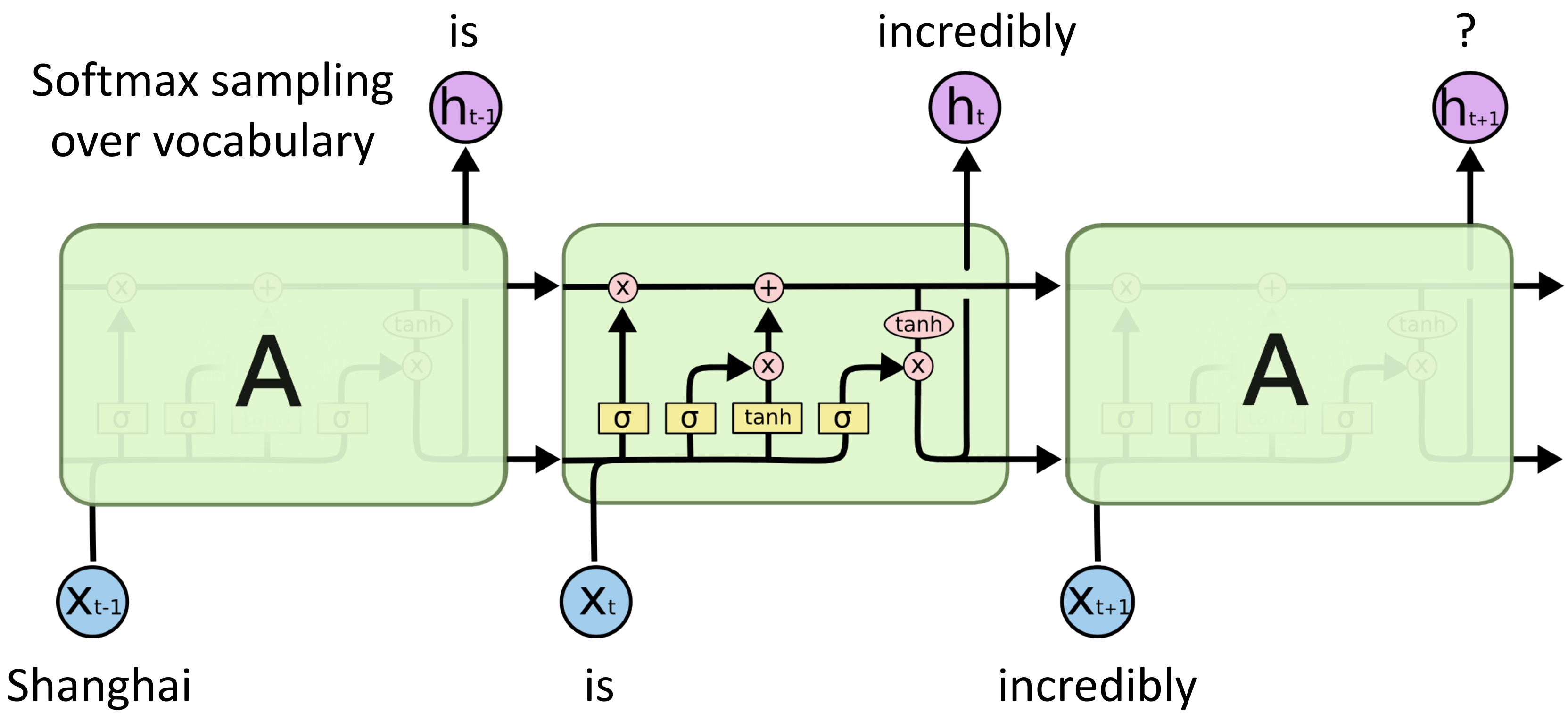
Overall Algorithm

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

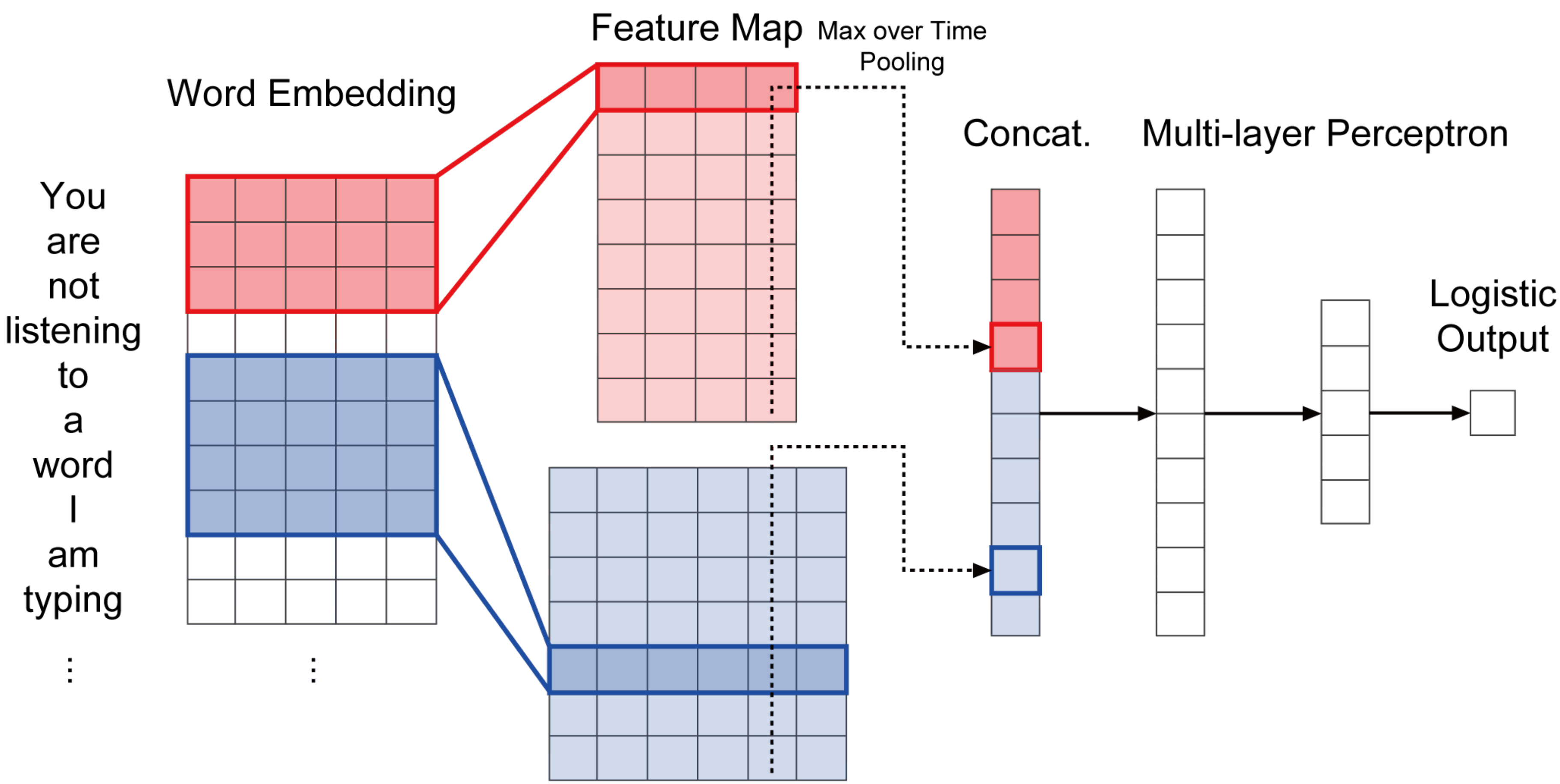
Sequence Generator Model



- 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

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

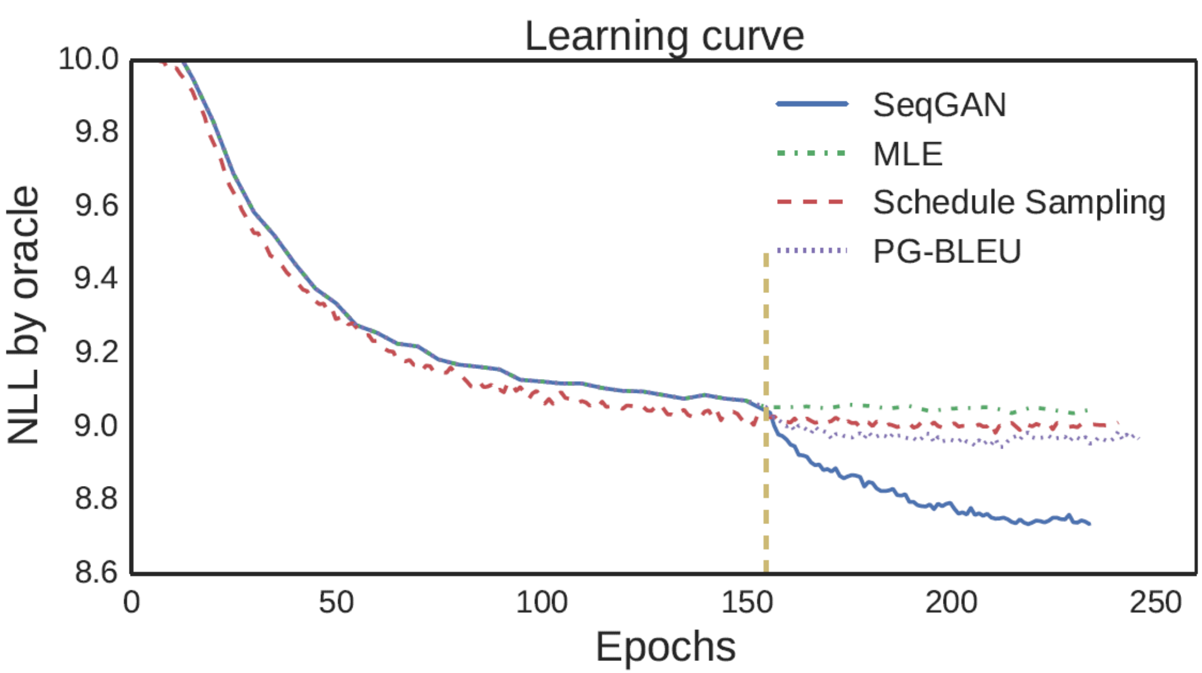
- An oracle model (e.g. the randomly initialized LSTM)
  - Firstly, the oracle model produces some sequences as training data for the generative model
  - Secondly the oracle model can be considered as the human observer to accurately evaluate the perceptual quality of the generative model

Experiments on Synthetic Data

- Evaluation measure with Oracle

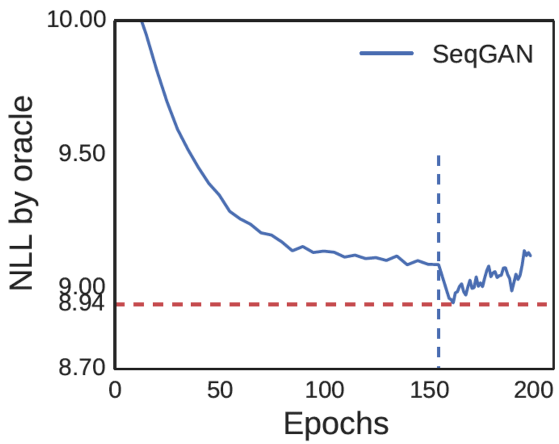
$$\mathrm{NLL}_{\mathrm{oracle}} = -\mathbb{E}_{Y_{1:T}\sim G_\theta}\left[\sum_{t=1}^{T}\log G_{\mathrm{oracle}}(y_t|Y_{1:t-1})\right]$$

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

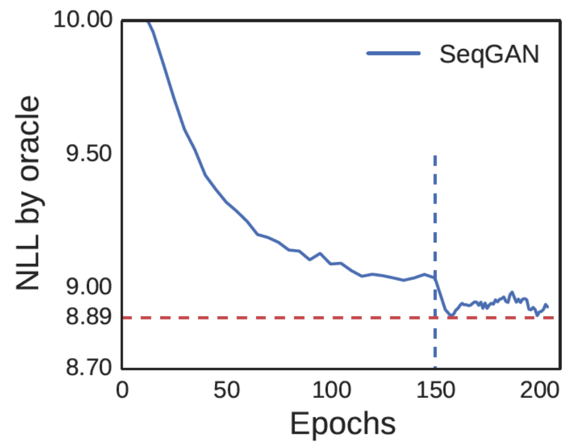


Experiments on Synthetic Data

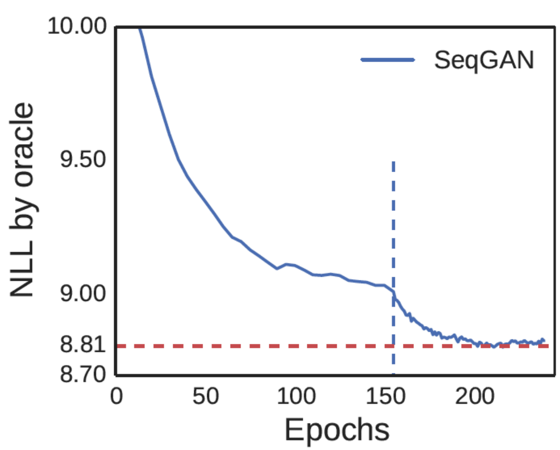
- The training strategy really matters.



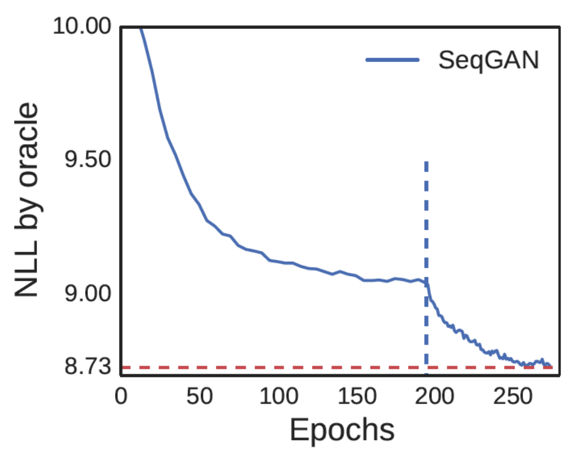
(a) g-steps=100, d-steps=1, k=10



(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

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.

Thank You

Lantao Yu[†], Weinan Zhang[†], Jun Wang[‡], Yong Yu[†]
[†]Shanghai Jiao Tong University, [‡]University College London

Email: yulantao@apex.sjtu.edu.cn

Lantao Yu, Weinan Zhang, Jun Wang, Yong Yu. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. AAAI 2017.