Apply GANs to NLP generation tasks(1)

邓俊锋

2018.11.29

Related papers

- **Generative Adversarial Nets**
- **SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient**
- Improving Neural Machine Translation with Conditional Sequence Generative Adversarial Nets
- Adversarial Neural Machine Translation
- Adversarial Learning for Neural Dialogue Generation
- Towards Robust Neural Machine Translation

Introduction to GAN

- What is Generation ?
  - Language model (Direct Generation, $P(Y)$)
  - Neural Machine Translation (Conditional Generation, $P(Y|X)$)
- Modules of GAN
  - A generator G learn the real data distribution P_{data}
  - A discriminator D distinguish the real ones from those generated by G
  - G and D paly a minmax game util Nash Equilibrium reached
    - At that time, D always output 0.5, which imply that G has learned the real data distribution P_{data} perfectly (in theory)
  - Formulation
    - $\min_G \max_D V(D,G) = \mathbb{E}_{x\sim P_{data}}[\log D(x)] + \mathbb{E}_{z\sim P_{noise}}\left[\log\left(1 - D\left(G(z)\right)\right)\right]$

The great success of GAN in CV

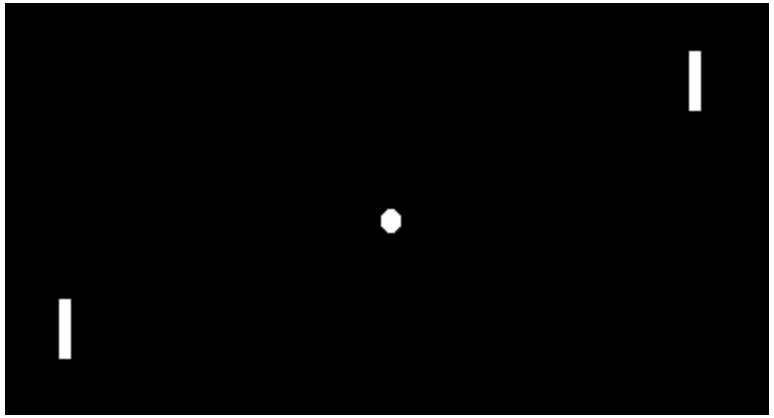
- Lots of GAN Variants
  - α-GAN, β-GAN, γ-GAN……
  - the Greek alphabet is just not enough
- Images generated by GAN variants

Difficults of applying GAN to NLP

- Natural language sentences are sequence of discrete token
  - GAN is designed for generating **real-value, continues** data
  - In a normal setting, utilizing gradient based methods, the signal from D is used to update the G's parameters slightly, but there may be no corresponding token for such '**slight**' change in the limited dictionary space
  - Q1: 以语言模型为例，哪一步导致其不能直接借助GAN来训练？
- D could only provide feedback for a entire sequence
  - Taking NMT as an example, decoder generate tokens step by step, but it's nontrivial for D to score a partially generated sequence, more specifically, when translate "我爱自然语言处理", the partially generated sequence "I love" make no sense for D

SeqGAN

- For differentiation problem
  - Draw Nutrition from RL (reinforcement learning)
  - Not differentiable ≠ Unable to learn
  - Formulate sequence data generation as a sequence decision making process, regard G as a stochastic policy (or the Mario ^_~)
  - Bypass the differentiation problem naturally by utilizing **policy gradient**
    - Technical details about policy gradient will be discussed in next few slides



SeqGAN

- For partially generated sequence scoring problem
  - In most cases, intermediate results in the generation process make no sense, just like the Go game, we could only get the effective feedback when the game is finished
  - Using Monte Carlo search to make it assessable for D
    - Sampling from the partially generated sequence base on G, util a complete sequence is finished
    - Do MC search N times to alleviate the high variance problem
    - Taking “我爱自然语言处理” as an example, when D scores the partially generated sequence “I love”, the MC search will be launched N(say =3) times, and get, for instance, “I love machine learning .”, “I love traveling .”, “I love natural .”, then, the average score of this three sentence will be treated as score of the partially generated sequence “I love”
  - Q2: SeqGAN在解决部分生成序列打分问题时引入了哪些缺点？

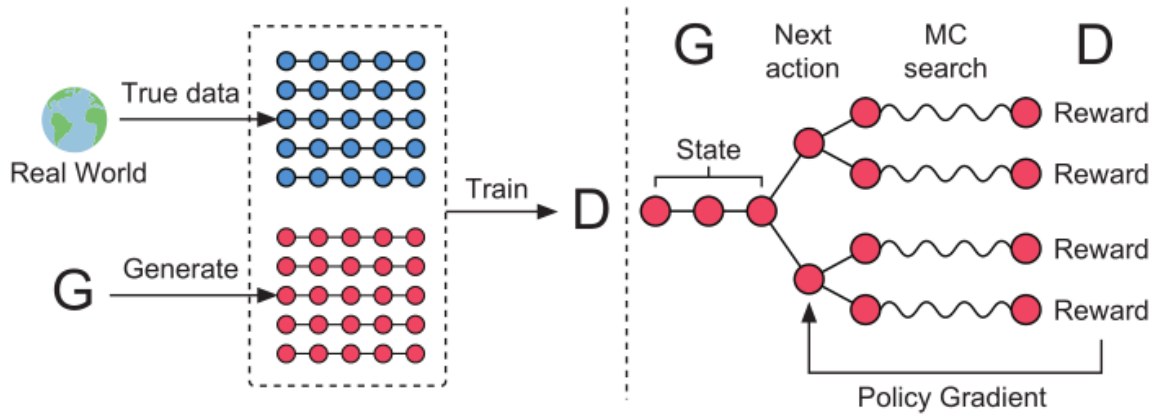


Figure 1: The illustration of SeqGAN. Left: D is trained over the real data and the generated data by G. Right: G is trained by policy gradient where the final reward signal is provided by D and is passed back to the intermediate action value via Monte Carlo search.

SeqGAN

- Formulation
  - Given a dataset of real world sequence
  - Train the generative model G_θ to produce $Y_{1:T} = (y_1, y_2, \dots, y_T), y_t \in \mathcal{Y}$
  - Expressing in terms of reinforcement learning
    - At timestep t, the state s is current generated tokens $(y_1, \dots, y_{t-1})$
    - The action a is the next token y_t to select
    - The objective of agent(i.e. G) is to take action(generate a token) based on current state(already generated $y_1, \dots, y_{t-1}$) to maximize final rewards
- The training of discriminator D is a 2-classification problem
- In adversary stage, G is trained via feedback signal from D using policy gradient, and D is trained using the new data generated by better G

SeqGAN

- The objective of G (or called policy)
  - Max the expected rewards

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

  - s_0 is the initial state, $\mathcal{Y}$ is vocabulary table
  - θ is the parameters of G to be learned
  - $Q_{D_\phi}^{G_\theta}(s, a)$ is action-value function, i.e. the expected reward starting from state s, taking action a, and then follow G_θ

$$Q_{D_\phi}^{G_\theta}(s = Y_{1:t-1}, a = y_t) = \begin{cases} \frac{1}{N} \sum_{n=1}^{N} D_\phi(Y_{1:T}^n), Y_{1:T}^n \in \mathrm{MC}(Y_{1:t}; N), & t < T \\ D_\phi(Y_{1:T}), & t = T \end{cases}$$

$\mathrm{MC}(Y_{1:t}; N) = \{Y_{1:T}^1, \dots, Y_{1:T}^N\}$, where $Y_{1:T}^n = (y_1, \dots, y_t)$ and $Y_{t+1:T}^n$ is sampled from G_θ

SeqGAN

- learn G's parameters using policy gradient update
  - Recall objective is max $J(\theta) = \mathbb{E}[R_T|s_0, \theta] = \sum_{y_1 \in \mathcal{Y}} G_\theta(y_1|s_0) \cdot Q_{D_\phi}^{G_\theta}(s_0, y_1)$
  - The exact derivative of objective w.r.t. G's parameters θ is

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

  - The expection $\mathbb{E}$ can be approximated by sampling methods
  - Using likelihood ratio trick, the derivative could be rewrited as

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

  - Q3*: 采用策略梯度更新参数，是否完全“摆脱”了MLE方法的影响?

Training Algo.

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

Synthetic Data Experiments

- Train a LM using proposed SeqGAN
  - Using a randomly initialized LSTM as the oracle (p_{data}) to produce the "real world" data
  - The objective is to train G to restore the "real world" distribution p_{data}
- Evaluation Metric

$$\mathrm{NLL}_{\mathrm{oracle}} = -\mathbb{E}_{Y_{1:T}\sim G_\theta}[p_{data}(Y_{1:T})]$$

- Training setting
  - Generating 10,000 sequence of length 20 as traning data using the oracle LSTM, which is randomly initialized by normal distribution $\mathcal{N}(0,1)$
  - G is implemented by RNN with GRU unit, and D, text CNN
  - The baseline training approach includes MLE, scheduled sampling, PolicyGradient-BLUE

Experiment results & dicussion

Table 1: Sequence generation performance comparison. The p-value is between SeqGAN and the baseline from T-test.

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

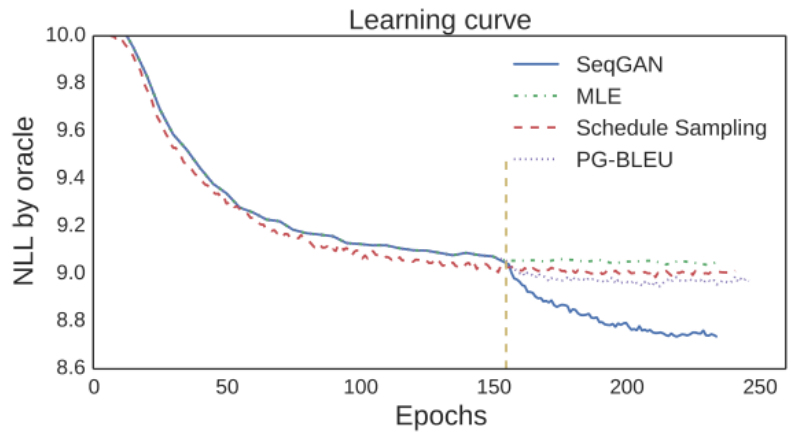


Figure 2: Negative log-likelihood convergence w.r.t. the training epochs. The vertical dashed line represents the end of pre-training for SeqGAN, SS and PG-BLEU.

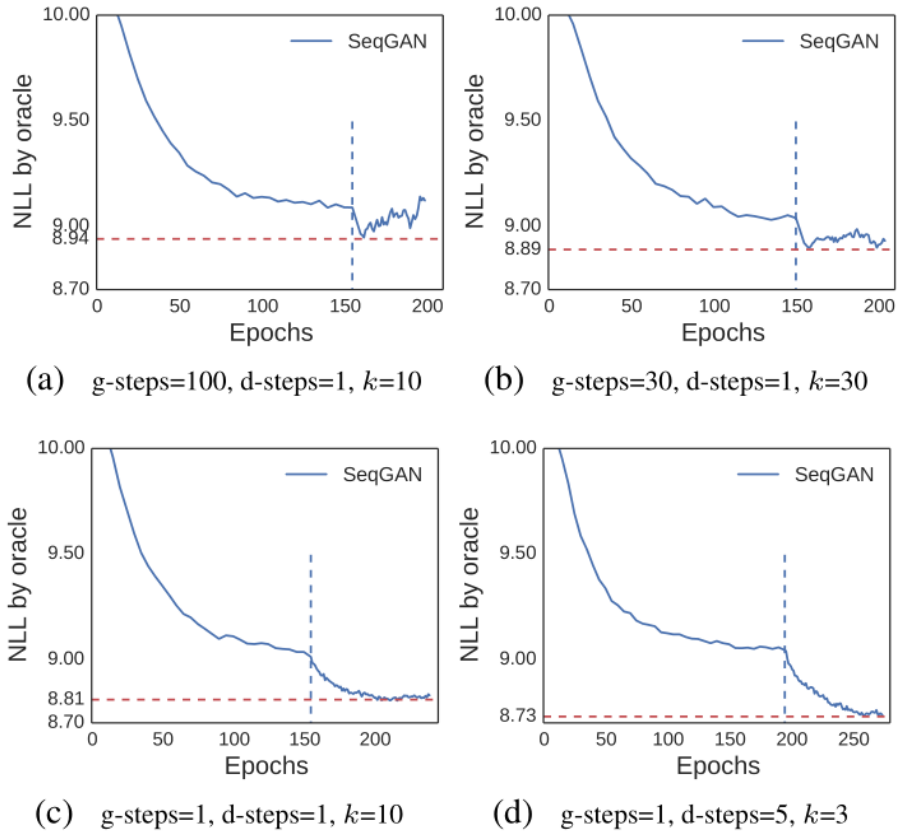


Figure 3: Negative log-likelihood convergence performance of SeqGAN with different training strategies. The vertical dashed line represents the beginning of adversarial training.

Summary

- The first work extending GANs to generate seqence of discrete tokens
- Bypass the differentiation problem via policy gradient
- Q4: 是否存在其他方法来避开采样步骤不可导问题?
- Utilizing MC search to make the partially generated sequence assessable for D
- Next time we'll discuss how to apply SeqGAN to conditional generation task, i.e. adversary NMT

recommend material

- The tensorflow implementation of SeqGAN
  - https://github.com/LantaoYu/SeqGAN
- step-by-step derivation of the objective $J(\theta)$ with respect to G's parameters θ
  - In appendix section of SeqGAN paper