Research Report Two

Xinpeng Hong

Number 2019-0002
October 1 2019

同济大学
软件学院
杜庆峰教授实验室
邱娟博士课题组

Research Report Two

Xinpeng Hong

Tongji University, School of Software Engineering, 2019-0002

October 1 2019

1 SeqGAN 论文阅读与理解

1.1 标题

SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient

1.2 动机

最初的 GAN 仅仅定义在实数领域，GAN 通过训练出的生成器来产生合成数据，然后在合成数据上运行判别器，判别器的输出梯度将告诉我们如何通过略微改变合成数据而使其更加现实。
一般来说只有在数据连续的情况下才可以略微改变合成的数据，而如果数据是离散的，则不能简单通过改变合成数据来达到目的。因为 CV 领域中像素是连续的，而 NLP 的基础都是离散值，如“单词”、“字母”或者“音节”，因此 GAN 网络在计算机视觉上得到了很好的应用，NLP 中应用 GAN 却是非常困难的。
在生成 text 的时候，GAN 只能对整个文本序列进行建模打分，对于部分生成的序列，很难判断其在之后生成整个序列时的分数。
在传统的 seq2seq 模型中，通常用极大似然准则来训练模型，但是这个训练方式存在一个严重的问题即 exposure bias。在模型训练阶段用真实的 tokens 作为 decoder input，但在真正预测阶段时只能从上一步产生的分布中以某种方式抽样某 token 作为下一步的 decoder input，也就是说 deocder input 的分布其实可能是不一样的。GAN 的使用大大缓解了这个问题。

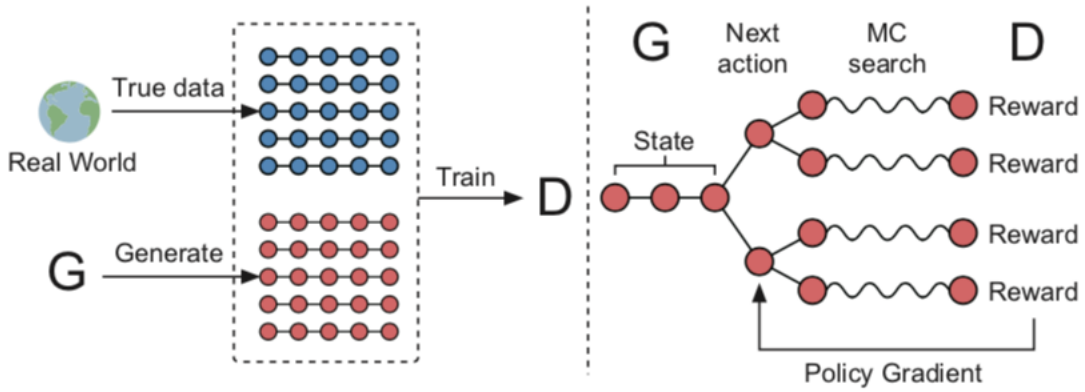
1.3 主体

1.3.1 思路概述

图片左部：有一批 true data，生成器生成一批假数据，利用极大似然方式的方式来预训练生成器，也就是让生成器不断拟合 true data 的分布。

这个过程经过几个回合后，把训练好的生成器生成的数据作为 negtive data，true data 作为 positive data 来预训练判别器。

图片右部：state 为现在已经生成的 token，action 是下一个即将生成的 token，policy 为 GAN 的生成器，reward 为 GAN 的判别器所回传的信息。



生成器是 LSTM 神经网络，判别器是 CNN 网络，true data 是由一个 target LSTM 网络生成的。

1.3.2 预训练

预训练生成器是利用极大似然估计的方法，不需要考虑 reward。

预训练判别器是先由生成器生成一批负样本，然后二分类，训练 CNN 网络。

1.3.3 生成器的目标函数

生成器的目标是 maximize expected end reward，即生成器生成了一句完整的句子后，我们希望尽可能使其所有 tokens 的 reward 之和最大。

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

其中 RT 可理解为一个完整句子的 reward 之和，s0 表示初始状态， 表示生成器的参数。后面的求和过程表示，每生成一个 token，都要计算其生成该 token 的概率与其对应的 reward 值，那么两者相乘即表示生成该 token 的期望 reward 值，求和后即为该整句的期望 reward 值。生成器的目标就是不断 maximize J。

reward 是由后面判别器 D 决定的，下面需要解决的是如何求 reward。

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

利用蒙特卡洛树搜索方法，因为只有是一个完整的句子，其判别器才能对其进行打分。例如在生成第 t 步的 token 时，后面的 tokens 是未知的，我们只能让生成器继续向后面生成 token，直到生成一个完整的句子，然后在给判别器打分，为了让这个打分更有说服力，将这个过程重复 N 次取平均，由于生成器的随机性，每次生成的句子是不同的。

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

式子左边表示 sample 出来的 N 个不同的完整句子。
综上所述，如下式。

$$Q_{D_\phi}^{G_\theta}(s = Y_{1:t-1}, a = y_t) = \tag{4}$$
$$\begin{cases} \frac{1}{N}\sum_{n=1}^{N} D_\phi(Y_{1:T}^n),\ Y_{1:T}^n \in \mathrm{MC}^{G_\beta}(Y_{1:t}; N) & \text{for} \quad t < T \\ D_\phi(Y_{1:t}) & \text{for} \quad t = T, \end{cases}$$

打分过程大致如下：由于生成器是一个 CNN 网络，首先将 input x(二分类的) reshape 成一个四维的张量，然后通过各种卷积池化操作得到一个结果，然后再经过一个线性操作最终得到只有二维的张量，再做一个 softmax 操作得到 ypred 作为 reward 值。总地来说就是将该整句被判别器判别为真样本的概率作为该 token 的 reward，判别为真样本的概率越大，当前步选择该 token 越正向。这里 reward 的方式并不唯一，论文中的对比实验就用了 blue 指标作为 reward 来指导生成器的训练，这里也是可能可以创新的一个点。

1.3.4 判别器的目标函数

判别器是一个 CNN 网络，喂给判别器的样本是一个二分类的样本，即有生成器生成的一批假样本，也有一批真样本，然后直接做个二分类，损失函数就是一个 cross entropy。

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

1.3.5 策略梯度

利用梯度上升法更新生成器参数。

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

对参数 theta 求导结果如下。

$$\nabla_\theta J(\theta) \simeq \sum_{t=1}^{} \sum_{y_t \in \mathcal{Y}} \nabla_\theta G_\theta(y_t|Y_{1:t-1}) \cdot Q_{D_\phi}^{G_\theta}(Y_{1:t-1}, y_t) \qquad (7)$$

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

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

1.3.6 算法概述

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

G-step 中，利用生成器生成一批假样本，生成器每一步都是生成一个在词上的分布，我们以某种方式抽样一个 token 作为本步生成的 token。在 MLE 作为目标函数时，我们需得到 true data 中当前步的 token 在当前生成分布中的概率，在 SeqGAN 中考虑的是当前步得到的 token 在生成分布中的概率以及该 token 的 reward，利用蒙特卡洛树搜索法得到每个 token 的 reward。利用 policy gradient 更新生成器的参数。

D-step 中，利用上面已经训完的生成器生成一批样本作为假样本，加上已有的一批真样本，作为训练数据，来训练一个二分类的判别器。

1.4 个人想法

GAN 的生成器生成的序列需要喂给判别器，然后利用判别器来反向纠正生成器，这个时候梯度的微调不再适用在离散的数据上，并且梯度在回传时可能会有一些困难。SeqGAN 中，生成器每生成一个 token 时，都会计算该 token 在生成分别中的概率，并且利用上一次训完的判别器计算出相应的 reward，经过几轮的训练后，reward 越大的 token 越正向，越容易生成，因此避免了判别器反向传递梯度给生成器。

GAN 只能评估出整个生成序列的 score 和 loss，不能够细化到去评估当前生成 token 的好坏和对后面生成的影响，SeqGAN 模型将强化学习和对抗思想的结合，解决非连续序列生成的问题，产生可用于文本序列生成的模型。

2 SeqGAN 论文复现

下一步计划是基于对论文算法的理解进行复现，届时应该会对论文和算法的理解上一层台阶，同时寻找 GAN+RL 的创新点。