Discrete Sequence Generation with Generative Adversarial Nets

Feng Jie

2017/11/23

Outlines

- **Limits of GAN for discrete output**

- **Two kinds of Possible Solutions**
  - Modify the output: from discrete number to continuous number
  - Work with RL (Reinforcement Learning)

Notations

- One-hot vector
  - Denote a word or character, widely used in NLP (Natural Language Processing)

0	0	0	···	1	···	0	···	0	···

- Discrete
  - **Non Differentiable**
  - Eg. one-hot vector
- Softmax

$$[\text{softmax}(\mathbf{h})]_i = \frac{\exp(\mathbf{h}_i)}{\sum_{j=1}^{K} \exp(\mathbf{h}_j)}$$

Limits of GAN for discrete output

[–] **goodfellow_ian** 11 points 1 year ago

Hi there, this is Ian Goodfellow, inventor of GANs (verification: http://imgur.com/WDnukgP).

GANs have not been applied to NLP because GANs are only defined for real-valued data.

GANs work by training a generator network that outputs synthetic data, then running a discriminator network on the synthetic data. The gradient of the output of the discriminator network with respect to the synthetic data tells you how to slightly change the synthetic data to make it more realistic.

You can make slight changes to the synthetic data only if it is based on continuous numbers. If it is based on discrete numbers, there is no way to make a slight change.

For example, if you output an image with a pixel value of 1.0, you can change that pixel value to 1.0001 on the next step.

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".

Since all NLP is based on discrete values like words, characters, or bytes, no one really knows how to apply GANs to NLP yet.

In principle, you could use the REINFORCE algorithm, but REINFORCE doesn't work very well, and no one has made the effort to try it yet as far as I know.

Gradient cannot be propagated to the Generator
because of the existence of one-hot sampling operation.

Possible Solutions

- **Modify the output (one-hot vector):**
  - Gumbel-Softmax
  - Professor Forcing
  - **WGAN**

  .....
- **Reinforcement learning:**
  - SeqGAN
  - MaliGAN

  .....

Gumbel-Softmax

$$S \rightarrow x \,||\, S+S \,||\, S-S \,||\, S*S \,||\, S/S$$ **Generate grammar**

$x + x - x/x$ and $x - x * x * x * x.$

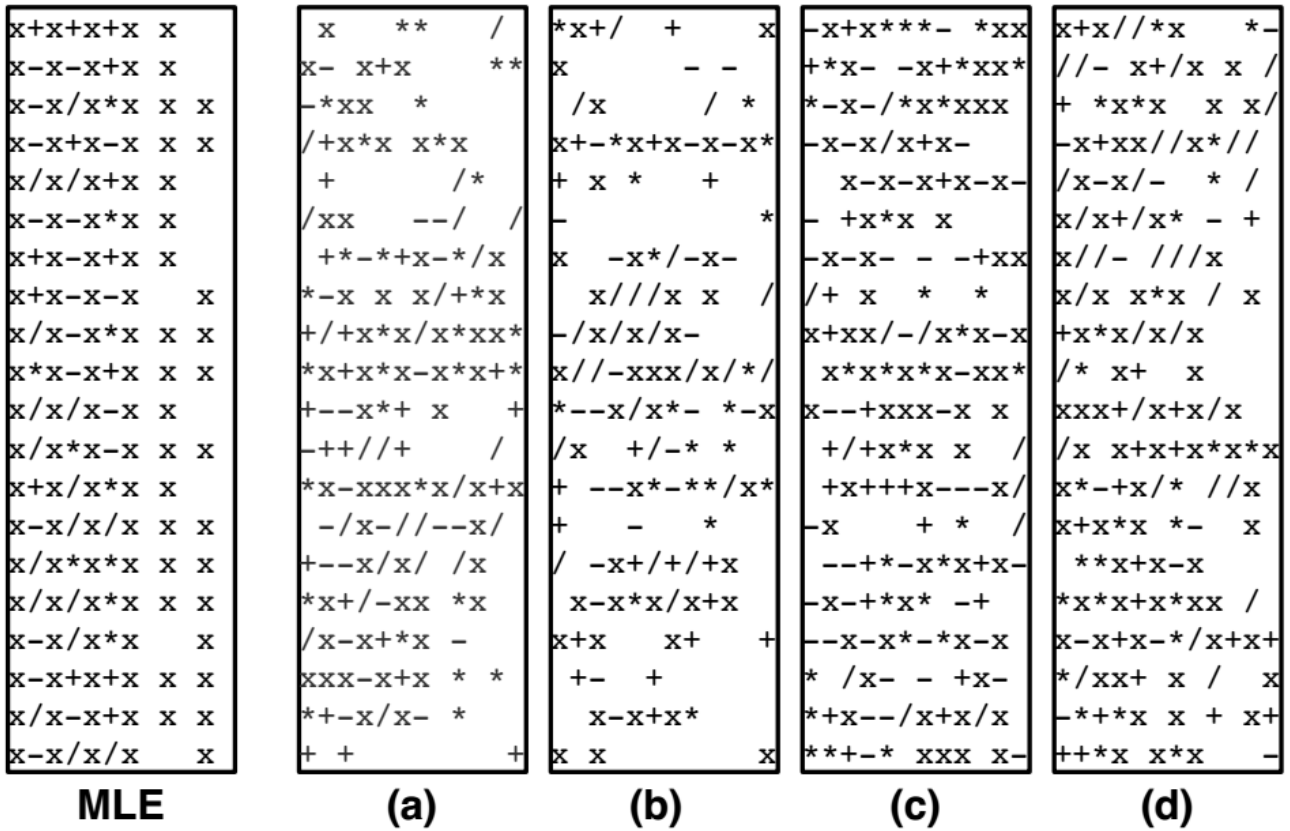
$$\mathbf{y} = \text{one\_hot}(\arg\max_i(h_i + g_i))$$

$$\mathbf{y} = \text{softmax}(1/\tau(\mathbf{h} + \mathbf{g}))\,,$$

Gumbel distribution

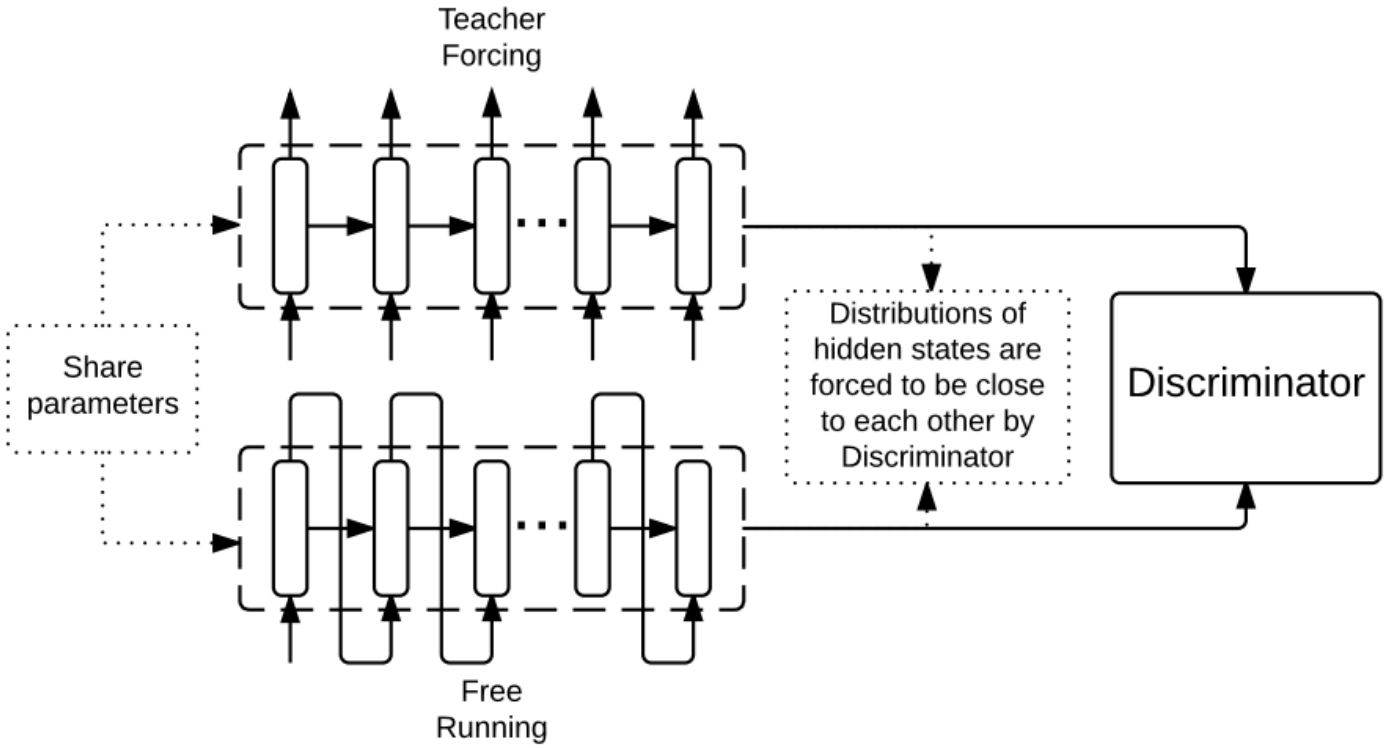
$g = -\log(-\log(u))\,, \qquad u \sim uniform(0,1)$

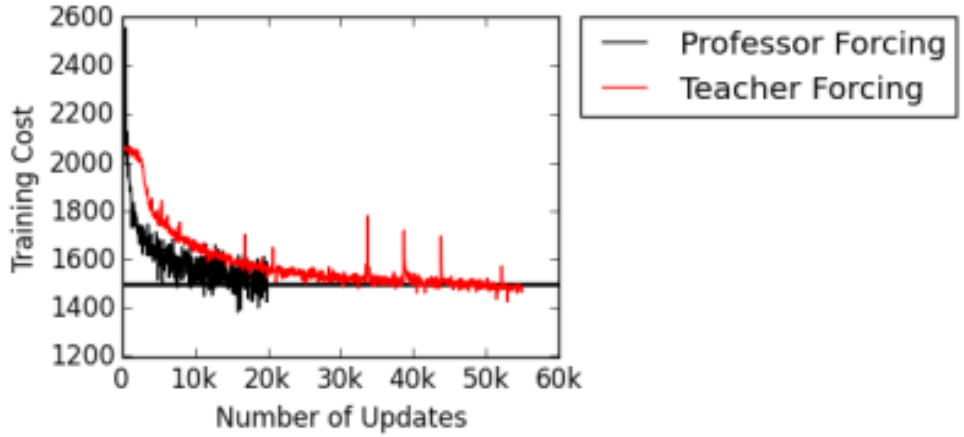
A GAN on discrete data can be trained by using gumbel-softmax, starting with some relatively large τ and then anealing it to zero during training.

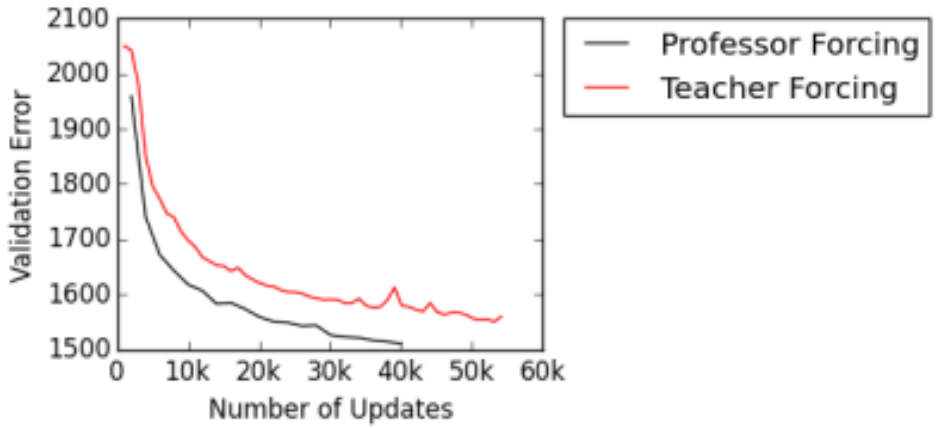


gumbel-softmax

Professor Forcing

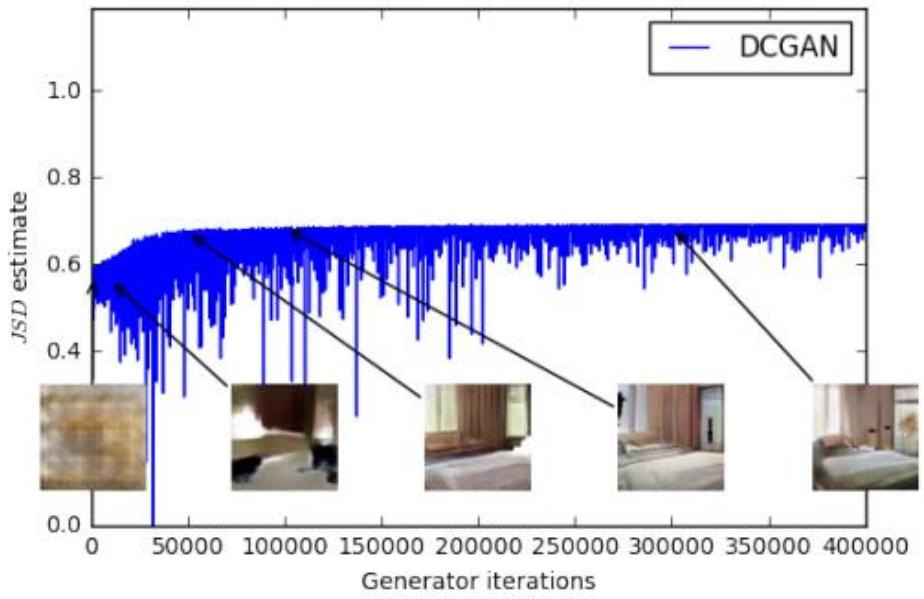






Wasserstein-GAN

- **Significant improvement of GAN**
  - not especially for sequence generation
  - for curing the training problem of GANs
- **Training problem of GANs**
  - the instability of GAN training
  - mode collapse (always produces same outputs)
  - no proper evaluation metric



Wasserstein-GAN

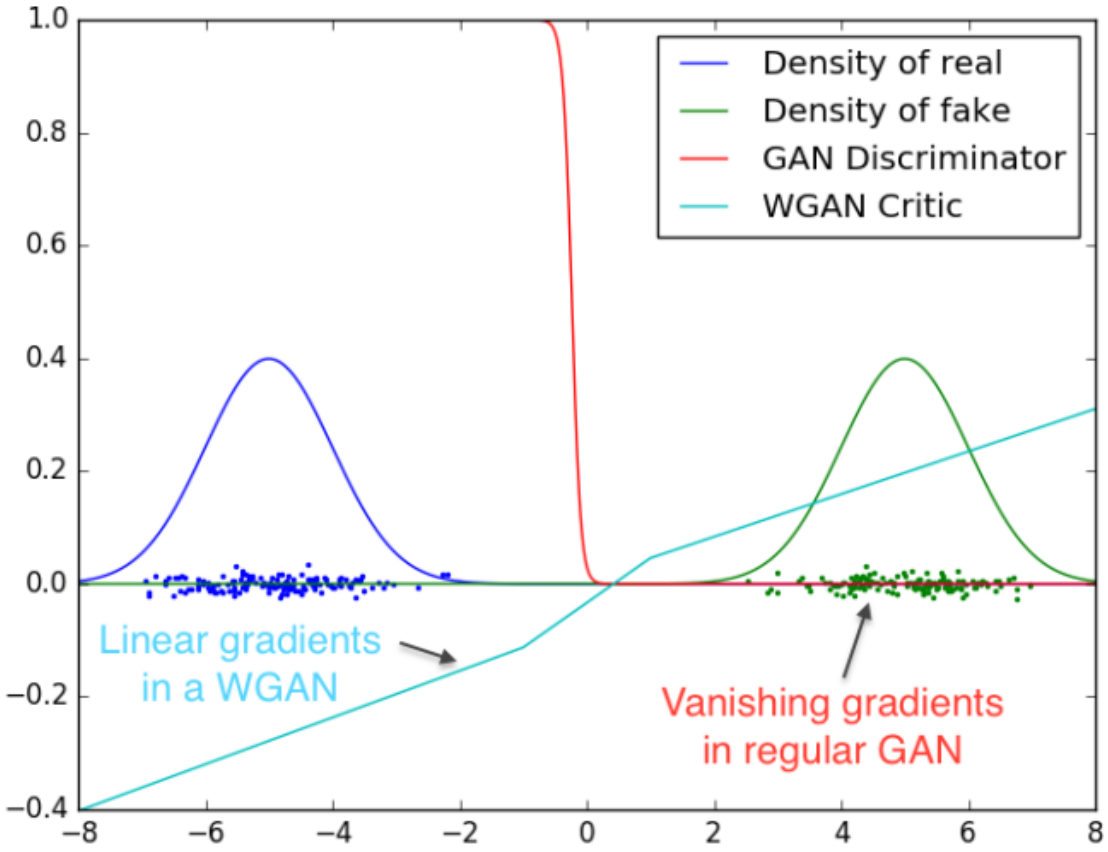
Earth-Mover distance

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}\left[\, \|x - y\| \,\right]$$

If f satisfy $\|f\|_L \leq K$, then the objective:

$$\max_{\|f\|_L \leq 1} \mathbb{E}_{x\sim\mathbb{P}_r}[f(x)] - \mathbb{E}_{x\sim\mathbb{P}_\theta}[f(x)]$$

$$\nabla_\theta W(\mathbb{P}_r, \mathbb{P}_\theta) = -\mathbb{E}_{z\sim p(z)}[\nabla_\theta f(g_\theta(z))]$$



Wasserstein-GAN

- Final Modification:
  - Delete the final sigmoid layer
  - Do not use log loss
  - Weight clipping (for K-Lipschitz constrain)

- WGAN-GP
  - Gradient penalty

$$L = \underbrace{\mathop{\mathbb{E}}_{\tilde{\boldsymbol{x}}\sim\mathbb{P}_g}\left[D(\tilde{\boldsymbol{x}})\right] - \mathop{\mathbb{E}}_{\boldsymbol{x}\sim\mathbb{P}_r}\left[D(\boldsymbol{x})\right]}_{\text{Original critic loss}} + \underbrace{\lambda \mathop{\mathbb{E}}_{\hat{\boldsymbol{x}}\sim\mathbb{P}_{\hat{\boldsymbol{x}}}}\left[\left(\|\nabla_{\hat{\boldsymbol{x}}} D(\hat{\boldsymbol{x}})\|_2 - 1\right)^2\right]}_{\text{Our gradient penalty}}.$$

Wasserstein-GAN

WGAN with gradient penalty (1D CNN)	
Busino game camperate spent odea	Solice Norkedin pring in since
In the bankaway of smarling the	ThiS record (31.) UBS) and Ch
SingersMay , who kill that imvic	It was not the annuas were plogr
Keray Pents of the same Reagun D	This will be us , the ect of DAN
Manging include a tudancs shat "	These leaded as most-worsd p2 a0
His Zuith Dudget , the Denmbern	The time I paidOa South Cubry i
In during the Uitational questio	Dour Fraps higs it was these del
Divos from The ’ noth ronkies of	This year out howneed allowed lo
She like Monday , of macunsuer S	Kaulna Seto consficutes to repor
Standard GAN objective (same architecture)	
ddddddddddddddddddddddddddddddd	ddddddddddddddddddddddddddddddd

Reference

[1] Jang E, Gu S, Poole B. Categorical reparameterization with gumbel-softmax[J]. arXiv preprint arXiv:1611.01144, 2016.

[2] Maddison C J, Mnih A, Teh Y W. The concrete distribution: A continuous relaxation of discrete random variables[J]. arXiv preprint arXiv:1611.00712, 2016.

[3] Kusner M J, Hernández-Lobato J M. GANS for Sequences of Discrete Elements with the Gumbel-softmax Distribution[J]. arXiv preprint arXiv:1611.04051, 2016.

[4] Arjovsky M, Chintala S, Bottou L. Wasserstein gan[J]. arXiv preprint arXiv:1701.07875, 2017.

[5] Gulrajani I, Ahmed F, Arjovsky M, et al. Improved training of wasserstein gans[J]. arXiv preprint arXiv:1704.00028, 2017.

[6] Yu L, Zhang W, Wang J, et al. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient[C]//AAAI. 2017: 2852-2858.

[7] Yang Z, Chen W, Wang F, et al. Improving Neural Machine Translation with Conditional Sequence Generative Adversarial Nets[J]. arXiv preprint arXiv:1703.04887, 2017.

[8] Li J, Monroe W, Shi T, et al. Adversarial learning for neural dialogue generation[J]. arXiv preprint arXiv:1701.06547, 2017.

[9] Lamb A M, GOYAL A G A P, Zhang Y, et al. Professor forcing: A new algorithm for training recurrent networks[C]//Advances In Neural Information Processing Systems. 2016: 4601-4609