Adversarial Learning for Neural Dialogue Generation

Jiwei Li[1], Will Monroe[1], Tianlin Shi[1], Sébastien Jean[2], Alan Ritter[3] and Dan Jurafsky[1]

[1]Stanford University, Stanford, CA, USA
[2]New York University, NY, USA
[3]Ohio State University, OH, USA
jiweil,wmonroe4,tianlins,jurafsky@stanford.edu
sebastien@cs.nyu.edu
ritter.1492@osu.edu

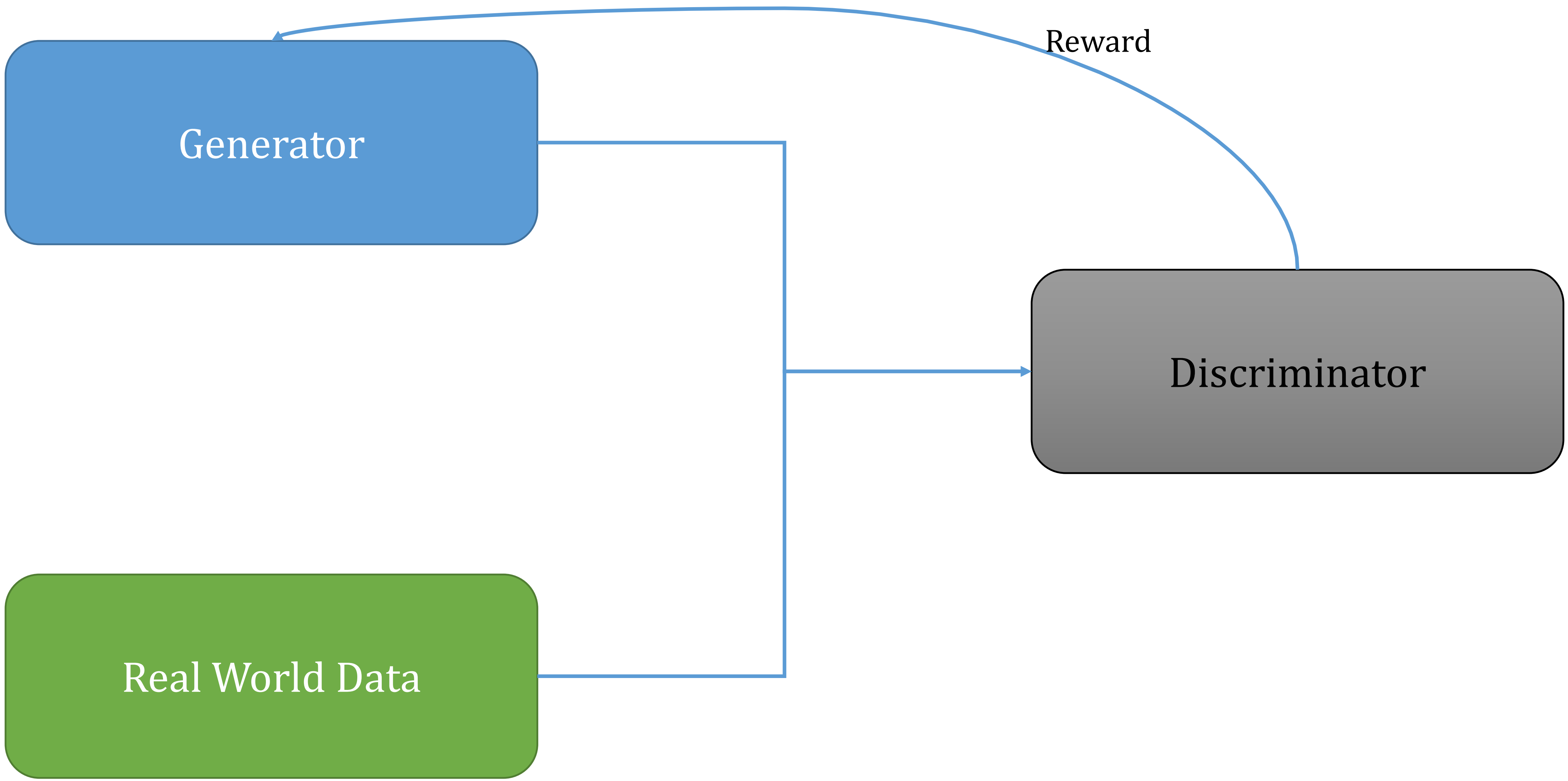
EMNLP2017

Previous Dialogue Generation
1. Use Maximum Likelihood Estimation.
2. Use Seq2Seq-like model.

Problem:
1. Using MLE has been proved to lead to simple, repetitive, meaningless responses.

Motivation:

1. Use a different evaluation method to guide generate process.
2. Manually defined reward can not cover all crucial aspects of responses.


Reward
Generator
Discriminator
Real World Data

Problems:

1. Text Generation is discrete, makes error hard to backpropagate.
2. Lack of immediate reward.

I	0
am	1
Fine.	2
Thank	3
you,	4
and	5
you?	6

(0, 255)
(0, 255)
(0, 255)
(0, 255)
(0, 255)
(0, 255)
(0, 255)

Reward: 0.01

Problems:

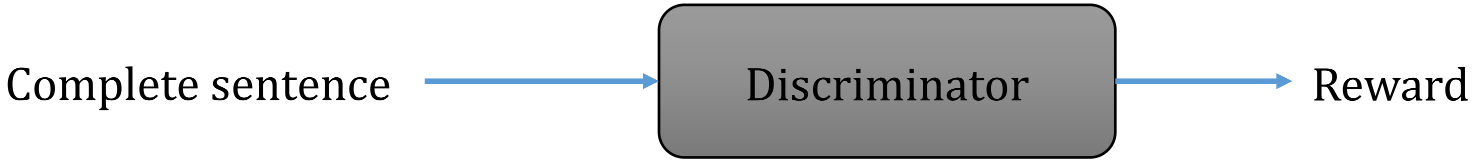
1. Text Generation is discrete, makes error hard to backpropagate.

Solution:

1. **Use Policy Gradient Descent.**
2. Use hidden state rather than generated text.

Problems:

1. Lack of immediate reward.



query

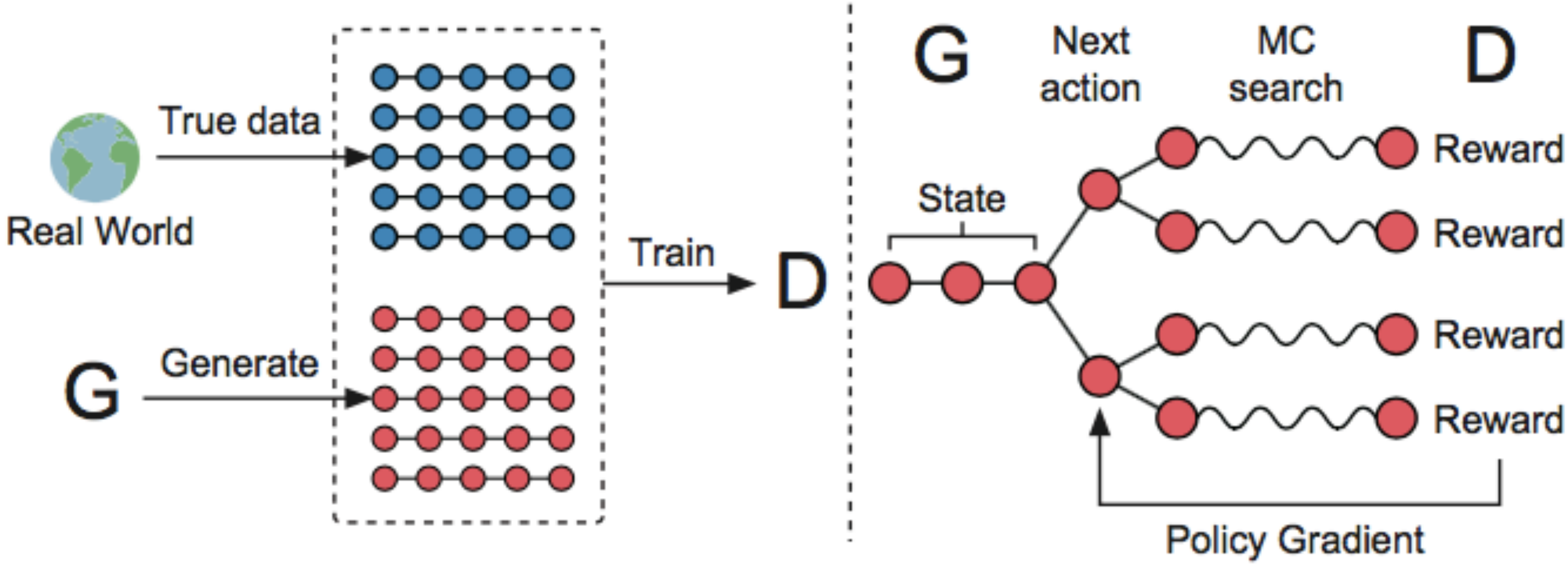
What is your name?

Real world data

I am liyucheng.

Generated data

I do not know.



Solution:

1. **Monte Carlo Search.**
2. Train Discriminator to give reward to partially decoded sequence.

Monte Carlo Search:

- Generate 5 instance shared the same prefix.
  Use mean reward as the final reward.

Train discriminator with prefixes of responses.

- $\{y_{\{1:1\}}\}\{y_{\{1:2\}}\} \ldots \{y_{\{1:t\}}\}$

Generator: Seq2Seq
Discriminator: Binary classifier.

$$J(\theta) = \mathbb{E}_{y\sim p(y|x)}(Q_+(\{x,y\})|\theta) \qquad (1)$$

Objective function

$$\nabla J(\theta) \approx [Q_+(\{x,y\}) - b(\{x,y\})] \nabla \log \pi(y|x) = [Q_+(\{x,y\}) - b(\{x,y\})] \nabla \sum_t \log p(y_t|x, y_{1:t-1}) \qquad (2)$$

Immediate Reward

$$\nabla J(\theta) \approx \sum_t (Q_+(x, Y_t) - b(x, Y_t))$$
$$\nabla \log p(y_t|x, Y_{1:t-1}) \quad (3)$$

State S; Action a; Reward r

策略：

Policy: $\pi_\theta(s, a) = P(a \mid s)$

Action Seqence:

$\tau = s_1, a_1, s_2, a_2, s_3, \cdots$

$$P(\tau) = P(s_1)P_\theta(a_1 \mid s_1)P(s_2 \mid s_1, a_1)P_\theta(a_2 \mid s_2)\cdots$$

Total reward:

$$R(\tau) = \sum_t r_t$$

$$\bar{R}_\theta = \sum_\tau R(\tau) P_\theta(\tau)$$
$$= E_{\tau \sim P_\theta(\tau)}[R(\tau)]$$

Maximum R:

策略梯度下降 Policy Gradient Descent

$$\nabla \bar{R}_\theta = \sum_\tau R(\tau) \nabla_\theta P(\tau)$$
$$= \sum_\tau R(\tau) P_\theta(\tau) \nabla \log P_\theta(\tau)$$
$$= E_{\tau \sim P_\theta(\tau)}[R(\tau) \nabla \log P_\theta(\tau)]$$
$$\approx \frac{1}{N} \sum_{n=1}^{N} R(\tau^n) \nabla \log P_\theta(\tau^n)$$

Add a bias

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^{N} R(\tau^n) \nabla \log P_\theta(\tau^n)$$

$$= \frac{1}{N} \sum_{n=1}^{N} [R(\tau^n) - b] \nabla \log P_\theta(\tau^n)$$

$$b = E_n[R(\tau^n)]$$

Except adversarial learning:
Use ***Teach Forcing***

1. Update generator both in adversarial learning and in MLE.

For number of training iterations **do**
. **For** i=1,D-steps **do**
. Sample (X,Y) from real data
. Sample $\hat{Y} \sim G(\cdot|X)$
. Update D using (X, Y) as positive examples and $(X, \hat{Y})$ as negative examples.
. **End**
.
. **For** i=1,G-steps **do**
. Sample (X,Y) from real data
. Sample $\hat{Y} \sim G(\cdot|X)$
. Compute Reward r for $(X, \hat{Y})$ using D.
. Update G on $(X, \hat{Y})$ using reward r
. Teacher-Forcing: Update G on (X, Y)
. **End**
End

Evaluation:

1. Use Evaluator to evaluate generation model.
   - Definition of AdverSuc: instance fooled evaluator successfully / total
   - ERE(Evaluator Reliability Error)

Setting	ERE
SVM+Unigram	0.232
Concat Neural	0.209
Hierarchical Neural	0.193
SVM+Neural+multil-features	0.152

Table 2: ERE scores obtained by different models.

ERE:

1. Set human-generated dialogue as both positive and negative examples. Expected AdverSuc=0.5.
2. Set human-generated examples as positive and random sentences as negative. Expected AdverSuc=0
3. ...

Model	*AdverSuc*	*machine-vs-random*
MLE-BS	0.037	0.942
MLE-Greedy	0.049	0.945
MMI+$p(t\|s)$	0.073	0.953
MMI-$p(t)$	0.090	0.880
Sampling	0.372	0.679
Adver-Reinforce	0.080	0.945
Adver-REGS	0.098	0.952

Table 3: *AdverSuc* and *machine-vs-random* scores achieved by different training/decoding strategies.

Machine-vs-random:

the accuracy of distinguishing be- tween machine-generated responses and randomly sampled responses