Adversarial Learning for Neural Dialogue Generation

Li, Jiwei, Will Monroe, Tianlin Shi, Alan Ritter, and Dan Jurafsky

EMNLP'17

Presented by Yiren Wang (CS546, Spring 2018)

Main Contributions

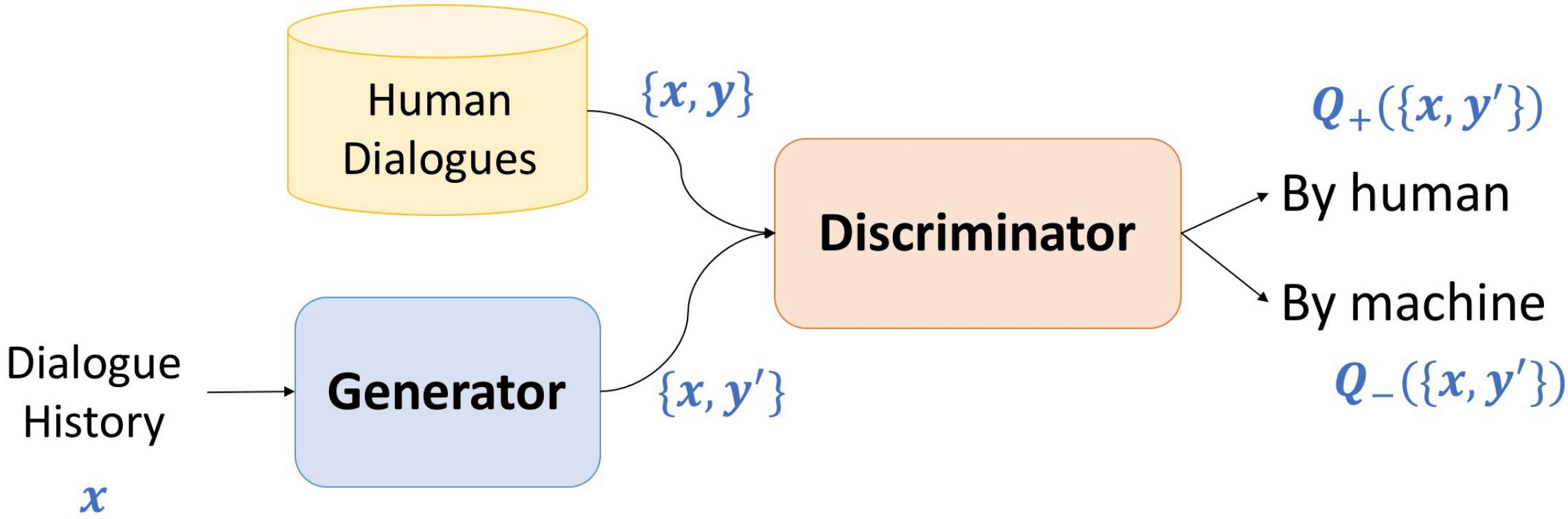
- Goal
  - End-to-end neural dialogue generation system
  - To produce sequences that are **indistinguishable** from human-generated dialogue utterances
- Main Contributions
  - **Adversarial training approach** for response generation
  - Cast the task in a **reinforcement learning** framework.

Outline

- Model Architecture
- Adversarial Reinforce Learning:
  - Adversarial REINFORCE
  - Reward for Every Generation Step (REGS)
  - Teacher Enforcing
  - Overall Algorithm (Pseudocode)
- Experiment Results
- Summary

Adversarial Model

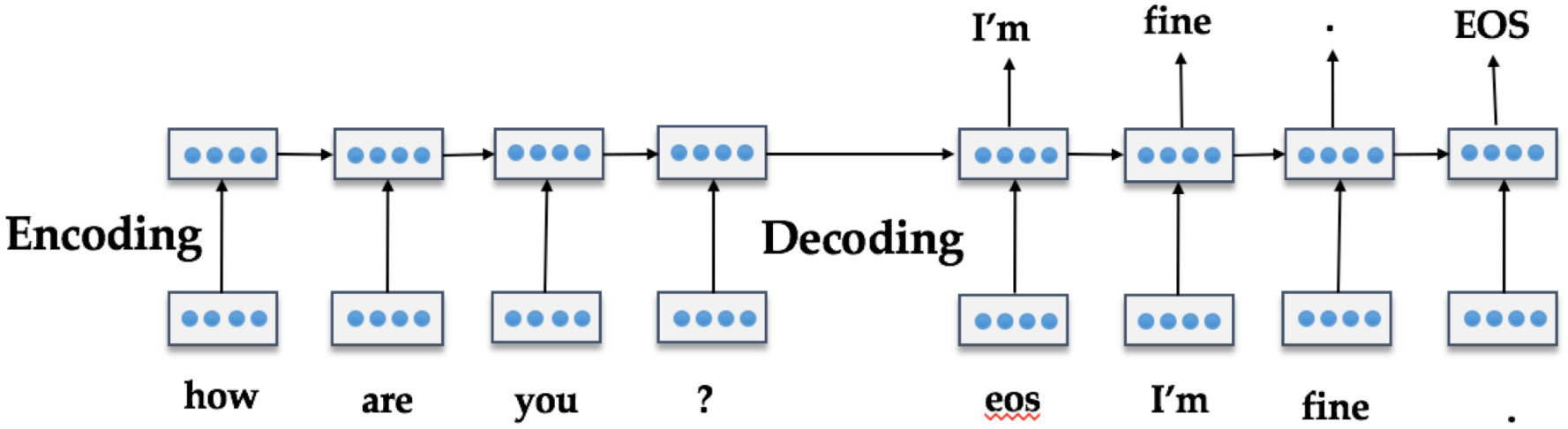
- **Overall Architecture**



Generative Model

- Model: Standard Seq2Seq model with Attention Mechanism
- Input: dialogue history $\boldsymbol{x}$
- Output: response $\boldsymbol{y}$

$$\text{Loss} = -\log p(\text{target}|\text{source})$$



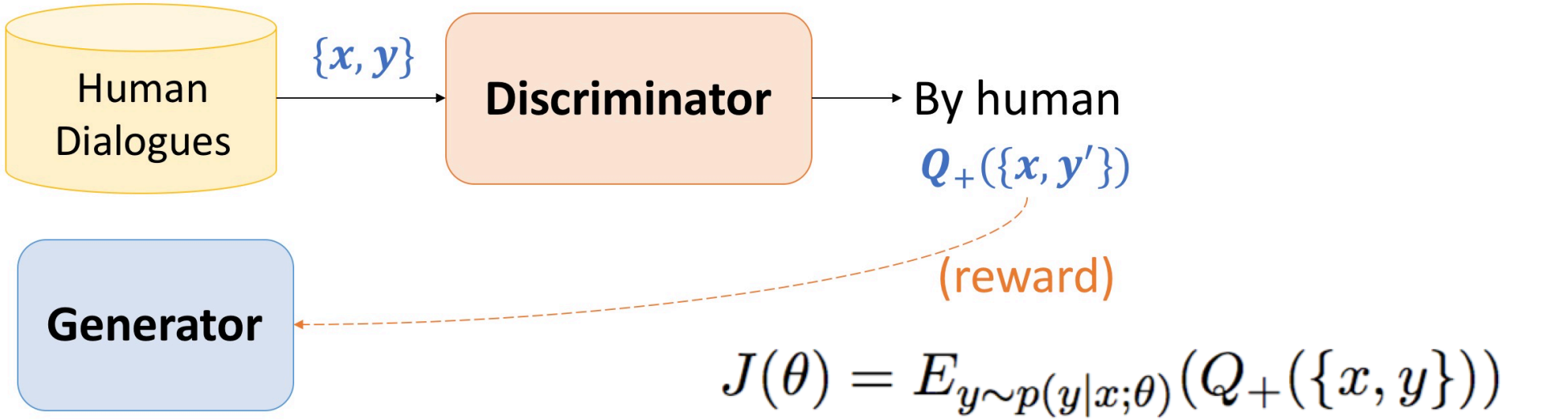
(Sutskever et al., 2014; Jean et al., 2014)

Discriminative Model

- Model: binary classifier
  - Hierarchical encoder + 2-class softmax
- Input: dialogue utterances $\{\boldsymbol{x}, \boldsymbol{y}\}$
- Output: label indicating whether generated by human or by machine
  - $\mathbf{Q}_{+}(\{\boldsymbol{x}, \boldsymbol{y}\})$ (by human)
  - $\mathbf{Q}_{-}(\{\boldsymbol{x}, \boldsymbol{y}\})$ (by machine)

Adversarial REINFORCE

- **Policy Gradient Training**
- Discriminator score is used as reward for generator
- Generator is trained to maximize the expected reward



$$J(\theta) = E_{y\sim p(y|x;\theta)}(Q_+(\{x, y\}))$$

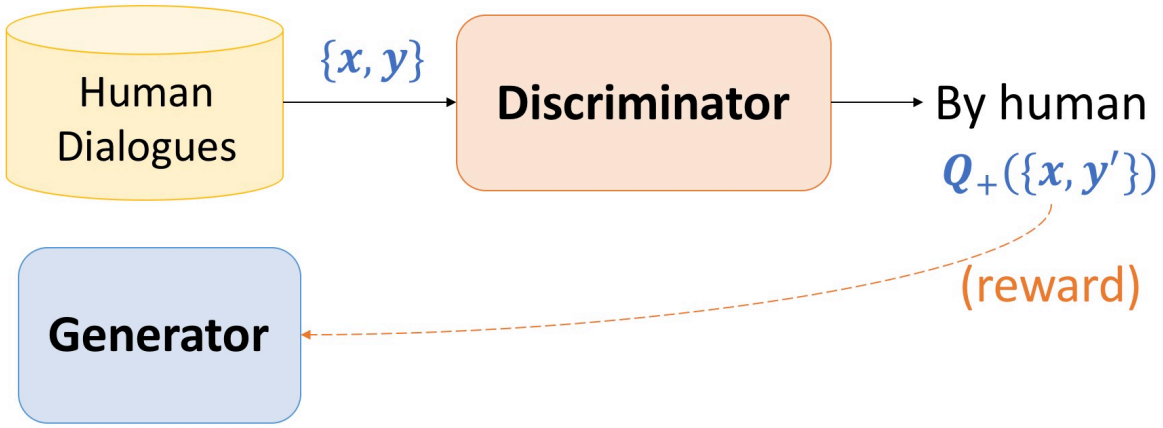
Policy Gradient Training

$$J(\theta) = E_{y \sim p(y|x;\theta)}(Q_+(\{x, y\}))$$

Approximated by likelihood ratio

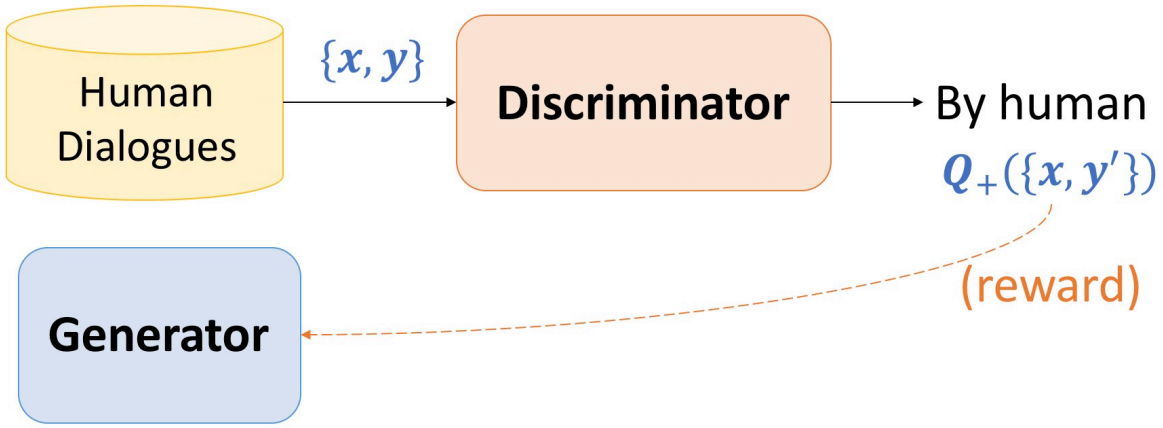
$$\nabla J(\theta) \approx [Q_+(\{x, y\}) - b(\{x, y\})] \nabla \log \pi(y|x)$$

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



Policy Gradient Training

$$J(\theta) = E_{y \sim p(y|x;\theta)}(Q_+(\{x, y\}))$$



Approximated by likelihood ratio

$$\nabla J(\theta) \approx [Q_+(\{x, y\}) - b(\{x, y\})] \nabla \log \pi(y|x)$$

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

Baseline value to reduce the variance of the estimate while keeping it unbiased

classification score

Policy updates in the parameter space

Problem with vanilla REINFORCE

- Expectation of reward is approximated by only one sample
- Reward associated with the sample is used for all actions

$$[Q_+(\{x, y\}) - b(\{x, y\})]$$

Input : *What's your name*

Human : *I am John*

Machine : *I don't know* (negative reward)

Problem with vanilla REINFORCE

- Expectation of reward is approximated by only one sample
- Reward associated with the sample is used for all actions

Input : *What's your name*

Human : *I am John*

~~Machine : *I don't know* (negative reward)~~

Machine : *I don't know*

(negative reward)

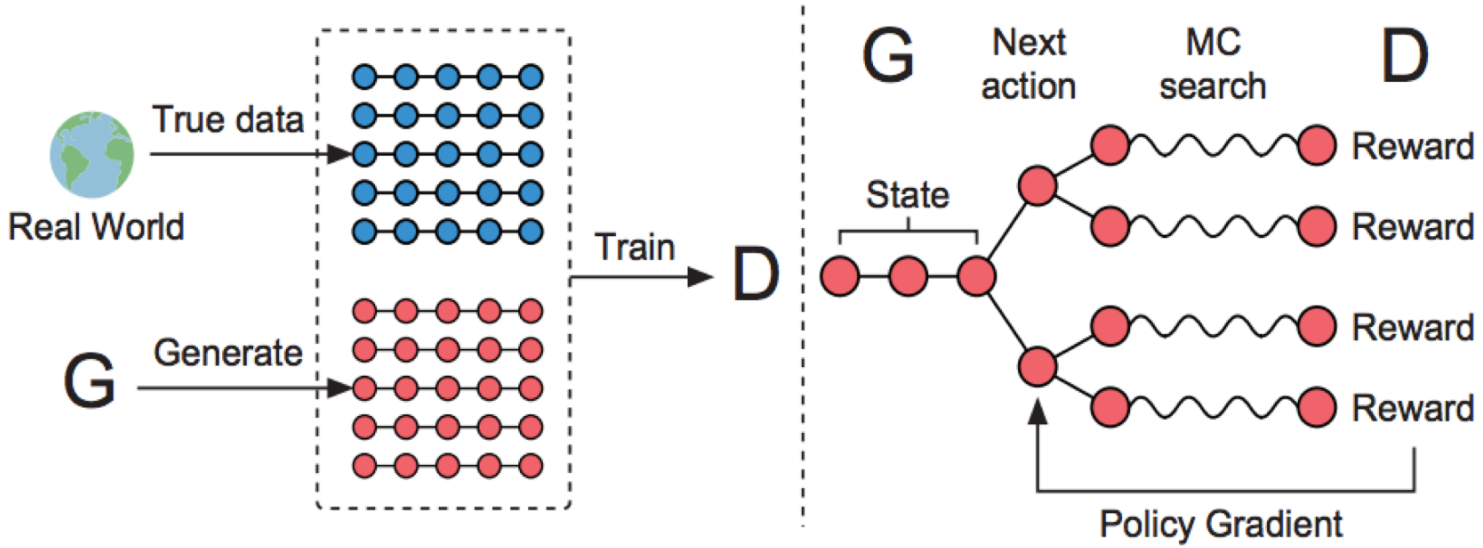
(neutral reward)

Reward for Every Generation Step (REGS)

- **Strategies**
  - Monte Carlo (MC) Search
  - Training Discriminator For Rewarding Partially Decoded Sequences

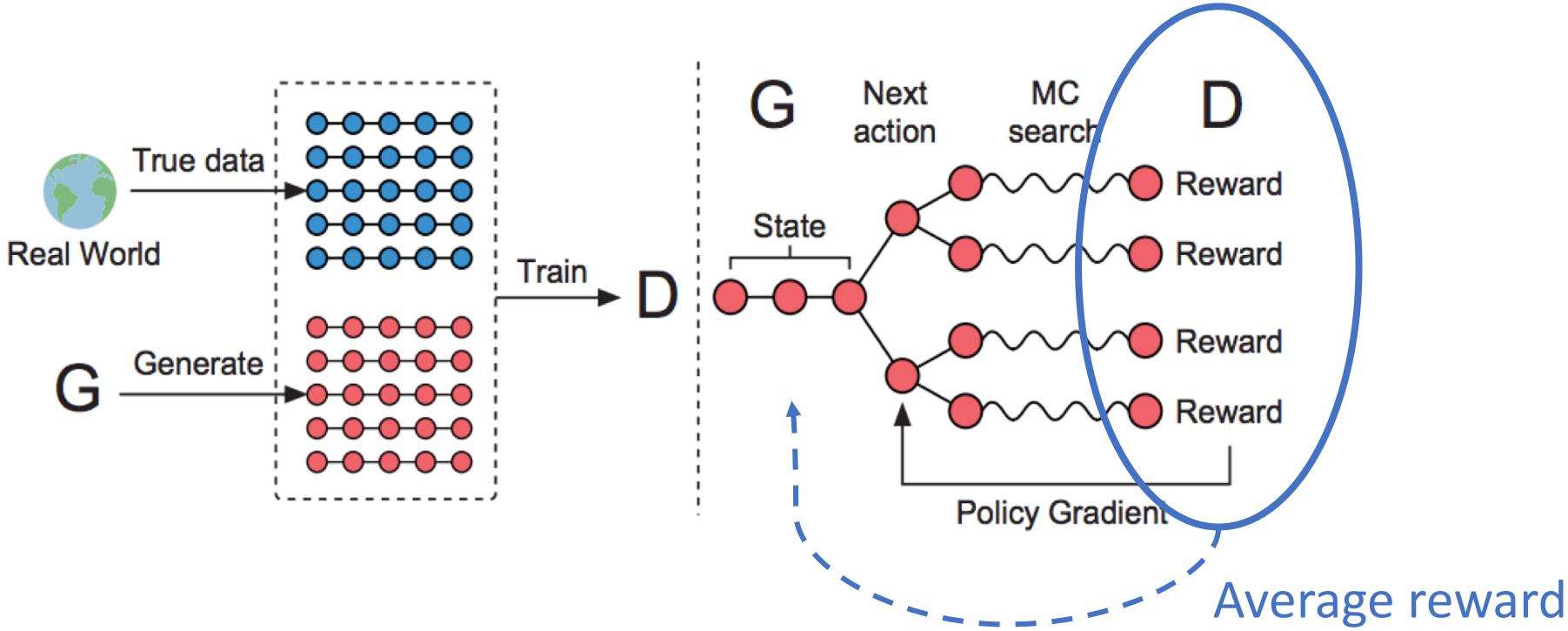
Strategy I: Monte Carlo (MC) Search

- Repeats sampling N times
- Average score is the reward



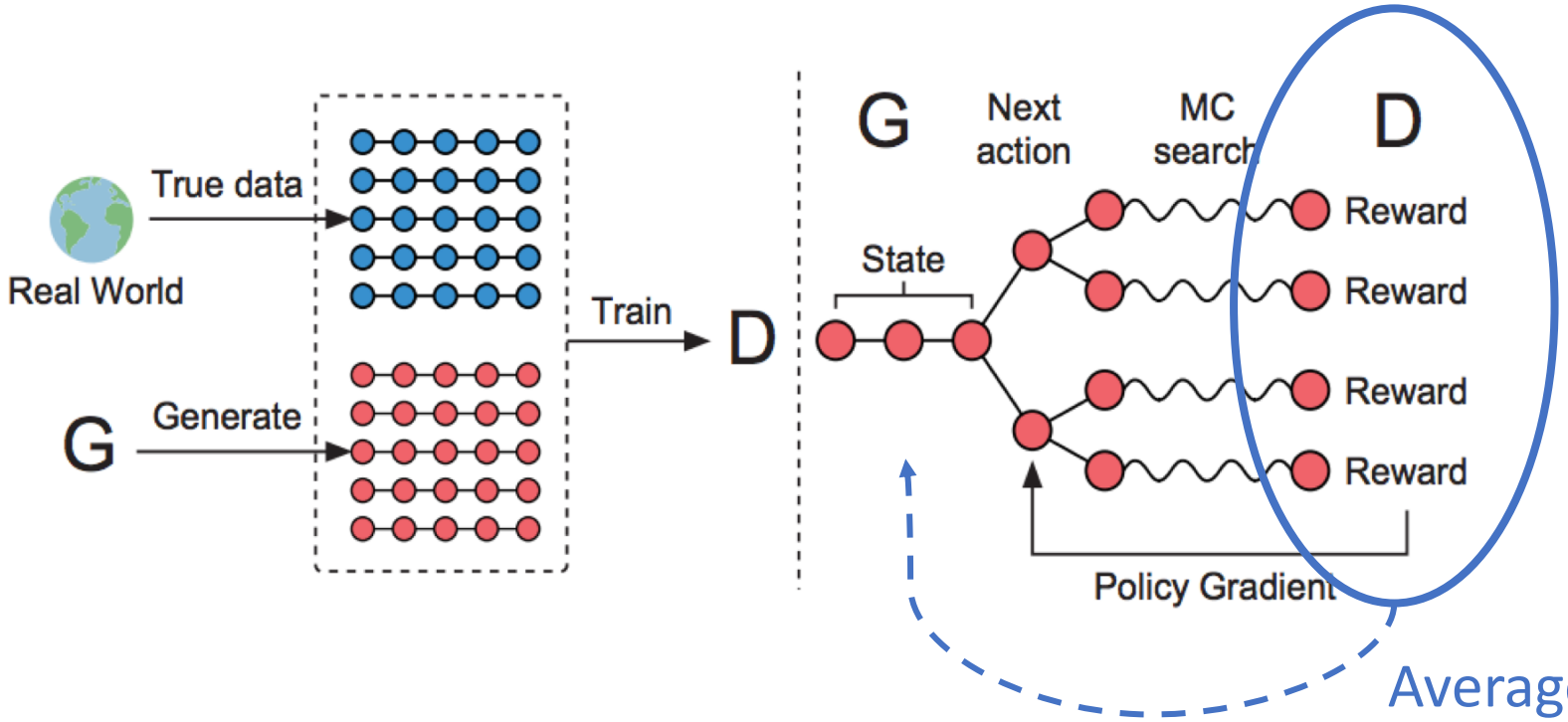
Strategy I: Monte Carlo (MC) Search

- Repeats sampling N times
- Average score is the reward



Strategy I: Monte Carlo (MC) Search

- Repeats sampling N times
- Average score is the reward



√ More accurate

✗ Time consuming

Strategy II: Reward Partially Decoded Seqs

- Break generated sequences into partial subsequences
- Sample one positive and one negative subsequence

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

score for each partial sequence

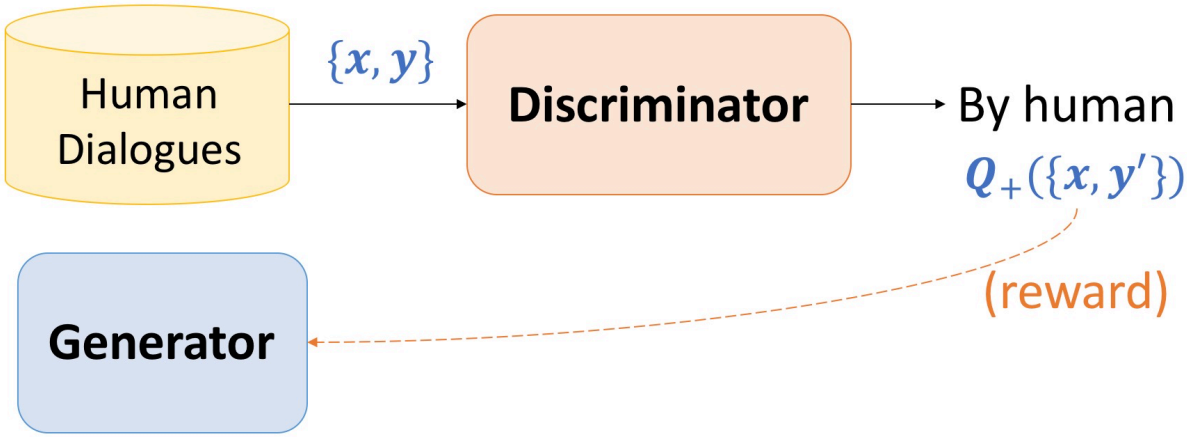
Partially-generated sequence

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

√ Time efficient

✗ Less accurate

Unstable Training



✗ Generator only indirectly exposed to the gold-standard target

- When generator deteriorates:
  - Discriminator does an excellent job distinguishing – from +
  - Generator only knows generated sequences are bad
  - But get lost what are good and how to push itself towards good
  - Loss of reward signals leads to a breakdown in training

Teacher Forcing

- Teacher Forcing:

  ”having a teacher intervene and force it to generate true responses”

- Discriminator:
  - assigns a reward of 1 to the human responses
- Generator:
  - uses this reward to update itself on human generated examples

  √ more direct access to the gold-standard targets

Overall Algorithm

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

Input	tell me ... how long have you had this falling sickness ?
Vanilla-MLE	i 'm not a doctor .
Vanilla-Sample	well everything you did was totally untrue .
REINFORCE	i don 't know how long it 's been .
REGS Monte Carlo	A few months, I guess .
Input	so i had the doctors test sammy 's response to conditioning .
Vanilla-MLE	sammy wrote the test sammy wrote the test .
Vanilla-Sample	objects pick objects up objects objects objects objects objects
REINFORCE	i 'm not sure that 's relevant .
REGS Monte Carlo	so he took the pills .
Input	what are you going to do ?
Vanilla-MLE	i 'm going to the movies .
Vanilla-Sample	get him outta here first !
REINFORCE	i 'm going to get you .
REGS Monte Carlo	i 'm going to get you a new car .
Input	they fear your power your intellect .
Vanilla-MLE	you 're the only one who knows what 's going on .
Vanilla-Sample	when they are conquered and you surrender they will control all of us .
REINFORCE	i 'm afraid i 'm not ready yet .
REGS Monte Carlo	i 'm not afraid of your power .

Table 1: Sampled responses from different models. More in Appendix Tables 5 and 6.

Summary

- Adversarial training for response generation
- Cast the model in the framework of reinforcement learning
  - Discriminator: Turing test
  - Generator: trained to maximize the reward from discriminator

Thanks!