Policy Gradients

Ki Hyun Kim

nlp.with.deep.learning@gmail.com

Before we start,

- Useful Trick:

$$\nabla_\theta \log P(x;\theta) = \frac{\nabla_\theta P(x;\theta)}{P(x;\theta)}$$

$$\begin{aligned}\nabla_\theta P(x;\theta) &= P(x;\theta)\frac{\nabla_\theta P(x;\theta)}{P(x;\theta)} \\ &= P(x;\theta)\nabla_\theta \log P(x;\theta)\end{aligned}$$

Policy Gradients

$$J(\theta) = \mathbb{E}_{\pi_\theta}[r] = v_\theta(s_0)$$
$$= \sum_{s \in \mathcal{S}} d(s) \sum_{a \in \mathcal{A}} \pi_\theta(s, a) \mathcal{R}_{s,a}$$

$$\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmax}}\, J(\theta)$$

$$\theta_{t+1} = \theta_t + \eta \nabla_\theta J(\theta)$$

- By Policy Gradient Theorem:

$$\nabla_\theta J(\theta) \approx \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a|s) Q_{\pi_\theta}(s, a)\right]$$

$$\theta \leftarrow \theta + \eta \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} Q_{\pi_\theta}(s, a) \nabla_\theta \log \pi_\theta(a|s)$$

Policy Gradients

- Q 함수가 미분 될 필요가 없음

$$\nabla_\theta J(\theta) \approx \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a|s) Q_{\pi_\theta}(s,a)\right]$$

$$\theta \leftarrow \theta + \eta \sum_{s \in \mathcal{S}} \sum_{a \in \mathcal{A}} Q_{\pi_\theta}(s,a) \nabla_\theta \log \pi_\theta(a|s)$$

- 따라서, 보상 함수로 미분 불가능한 매우 복잡한 함수를 사용할 수 있음
  - e.g. BLEU

REINFORCE

Given policy $\pi_\theta(a|s)$,

For each episode:

Generate an episode $s_0, a_0, r_1, \cdots, s_{T-1}, a_{T-1}, r_T$ from π_θ.

Loop: update θ for each step of the episode $t = 0, 1, \cdots, T-1$:

$$G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} r_k$$

$$\theta \leftarrow \theta + \eta \gamma^t G \nabla_\theta \log \pi_\theta(a_t|s_t)$$

Intuitive Explanation

Given policy $\pi_\theta(a|s)$,

For each episode:

Generate an episode $s_0, a_0, r_1, \cdots, s_{T-1}, a_{T-1}, r_T$ from π_θ.

Loop: update θ for each step of the episode $t = 0, 1, \cdots, T-1$:

$$G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} r_k$$

$$\theta \leftarrow \theta + \eta \gamma^t G \nabla_\theta \log \pi_\theta(a_t|s_t)$$

Problem?

- 100점 만점 시험에서 1점을 받아왔지만, 여전히 칭찬 받고 있는 것과 같다.
  - 다만, 100점 받은 것보다 칭찬을 받지 못하고 있을 뿐...

- 평균 점수에 대비해서, 평가/보상을 받아야 하는 것 아닐까?

REINFORCE with Baseline

Given policy $\pi_\theta(a|s)$ and value function v_ϕ(s),

For each episode:

Generate an episode $s_0, a_0, r_1, \cdots, s_{T-1}, a_{T-1}, r_T$ from π_θ.

Loop: update θ and ϕ for each step of the episode $t = 0, 1, \cdots, T-1$:

$$G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} r_k$$

$$\delta \leftarrow G - v_\phi(s_t)$$

$$\phi \leftarrow \phi + \eta^\phi \gamma^t \delta \nabla_\phi v_\phi(s_t)$$

$$\theta \leftarrow \theta + \eta^\theta \gamma^t \delta \nabla_\theta \log \pi_\theta(a_t|s_t)$$

MLE vs PG

Maximum Likelihood Estimation

$$\mathcal{D} = \{x^i, y^i\}_{i=1}^N$$

$$\hat{y}^i = \underset{y \in \mathcal{Y}}{\operatorname{argmax}} P(y|x^i;\theta)$$

$$\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmax}} \sum_{i=1}^{N} \log P(y^i|x^i;\theta)$$

$$\mathcal{L}(\theta) = -\sum_{i=1}^{N} \log P(y^i|x^i;\theta)$$

$$\begin{aligned}\theta &\leftarrow \theta - \eta \nabla_\theta \mathcal{L}(\theta) \\ &= \theta + \eta \sum_{i=1}^{N} \nabla_\theta \log P(y^i|x^i;\theta) \\ &= \theta + \eta \sum_{i=1}^{N} \sum_{t=1}^{n} \nabla_\theta \log P(y_t^i|x^i, y_{<t}^i;\theta)\end{aligned}$$

Policy Gradients

$$\nabla_\theta J(\theta) = \mathbb{E}_{a,s\sim\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a|s) \cdot Q_{\pi_\theta}(s,a)\right]$$

$$\begin{aligned}\theta &\leftarrow \theta + \eta \nabla_\theta J(\theta) \\ &\approx \theta + \eta \sum_{s\in\mathcal{S}} \sum_{a\in\mathcal{A}} Q_{\pi_\theta}(s,a) \nabla_\theta \log \pi_\theta(a|s)\end{aligned}$$

RL in NLG

- State
  - 조건으로 주어진 입력 문장(token들): x^i
  - 이전까지 생성된 token들: $\hat{y}^i_{<t}$

- Action
  - 현재 time-step에 생성 할 token을 고르는 것: $\hat{y}^i_t$

- Reward
  - 완성된 생성 문장과 실제 정답과의 유사도: BLEU$(\hat{y}^i_{1:n}, y^i)$

Characteristic of RL in NLG

- 대부분의 RL 문제는 episode가 매우 긴 것이 특징이지만,
  NLG에서는 하나의 문장을 생성하는 것이 episode로 볼 수 있다.
  - 매우 짧은 episode를 갖는 것이 특징 (길어야 100 time-step?)
  - 따라서 Actor Critic과 같은 고급 알고리즘을 구사할 필요성이 매우 낮음

- 단어를 고르는 것이 action이 되므로, action space가 vocabulary로 매우 크다.
  - Exploration 관점에서 매우 비효율적일 수 있음

- 보상(reward)이 문장이 완성된 이후(episode가 끝나면)에 주어진다.
  - Episode 중간에는 reward가 없음
  - Cumulative reward == final reward

Wrap-up: Why We Use RL?

Optimize with BLEU

- PPL은 정확한 생성 품질을 알 수 없음
- PG는 보상 함수의 미분이 필요 없어, BLEU를 통해 최적화 할 수 있음

Remove Teacher Forcing

- NLG는 auto-regressive task이므로 teacher forcing을 통해 학습함
  - 학습과 추론 사이의 괴리가 발생
- RL은 샘플링 기반의 학습이므로, 학습과 추론 방법의 차이가 없음