7 Stochastic Gradient Method

일반적으로 GD는 연산량이 많음. (계산어렵고,.. 비용 많이들고...) 우리가 필요한 부분은 전체 GD가 아니라 최적해 일부인데 데이터 일부만 사용할 수 없나? → 근사 gradient를 구해보자. (Stochastic GD)

* SGD는 전체를 확인하는게 아니라 더 noisy하지만 계산이 훨씬 가볍다. But. 수렴을 보장하기 위해 조건 필요 (∌ bias, 2^{nd} Moment)

- Idea

GD: full dataset에 대해 gradient 계산
$x^{k+1} = x^k - \alpha \nabla f(x^k)$

⇒ 조금 noisy하지만 계산이 빠른 추정치 $G(x^k)$를 사용하자.
$x^{k+1} = x^k - \alpha G(x^k)$, where $E[G(x^k)] = \nabla f(x^k)$

simple e.g) $G(x) = \nabla f(x) + \theta$

7.1 Applications

7.1.1. The (randomized) incremental gradient method.

- Problem Setting

$f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x)$ 를 최소화

e.g) MSE Loss

$f_i(x) = \frac{1}{2}(\langle x, a_i\rangle - b_i)^2 \Rightarrow \nabla f_i(x) = (\langle x, a_i\rangle - b_i)\, a_i$

1) 일반 GD

$\nabla f(x) = \frac{1}{n}\sum_{i}^{n} \nabla f_i(x)$

모든 i에 대해 gradient 계산

2) SGD (incremental)

① 각 반복에서 무작위로 index $i_k \in \{1, \cdots, n\}$

② 해당 샘플의 gradient만 사용 $G(x) = \nabla f_{i_k}(x^k)$

③ Update) $x^{k+1} = x^k - \alpha_k \nabla f_{i_k}(x^k)$

+ 수식 성질:
unbiased estimator, stochastic gradient 만족

7.1.2 Binary Classification with soft margin SVMs

< SGD가 실제 classifier(분류기)에 학습되는 예시 > (SVM 최적화 문제는 일반적으로 hinge loss + regularizer 형태.
n개의 Sample에 대한 평균형태 ← SGD 적용)

- 문제: suppose $(x_1, y_1), \cdots, (x_n, y_n)$, $y_i \in \{-1, 1\}$,

$$h(x) = \begin{cases} 1, & \text{if } \langle\theta, x\rangle > 0 \\ -1, & \text{otherwise} \end{cases}$$

여기서의 최적화 문제:

이걸 최소화

$f(\theta) = \frac{1}{n}\sum_{i}^{n} f_i(\theta)$, $f_i(\theta) = \max(1 - y_i\langle\theta, x_i\rangle, 0) + \frac{\lambda}{2}\|\theta\|_2^2$

- random i_k에 대해, hinge loss의 subgradient:

$$g_i(\theta) = \lambda\theta + \begin{cases} -y_i x_i, & \text{if } 1 - y_i\langle\theta, x_i\rangle > 0 \\ 0, & \text{otherwise} \end{cases}$$

→ apply $G(\theta) = g_i(\theta)$

→ update $\theta^{k+1} = \theta^k - \alpha g_i(\theta^k)$

이전까지) empirical risk를 줄였는데, 여기서는 진짜 목표인 population risk를 직접 줄여보자. (이론적, dataset 없이 확률분포에서 직접 샘플링하는 방식)

7.1.3 Minimizing the Population Risk

- Problem Setting:

Suppose unknown joint distribution over (y, x),

Minimize $R(h) = E_{(x,y)}[\text{loss}(h_\theta(x), y)]$

문제점! 이전에는 Loss가 hinge loss였는데
여기선 distribution을 알 수 없음 → true expectation 계산 不.

solution) sample $(x_i, y_i) \sim D$ 하나를 뽑아 사용.

$\theta^{k+1} = \theta^k - \alpha_k \nabla \text{loss}(\theta^k, x_i, y_i)$

→ 즉 SGD는 expectation을 하나로 근사

sample 하나로 계산한 gradient의 expectation ≒ 진 gradient

$E_{(x_i, y_i)\sim D}[G(\theta)] = E[\nabla \text{loss}(\theta, x_i, y_i)] = \nabla R(\theta)$

7.1.4 Stochastic approximation problems → 확률 & 통계 문제에서 SGD가 어떻게 쓰이나? – Robbins-Monroe 방식

- Problem Setting:

  $y = \theta^* + z$, where z gaussian, variance σ^2.

- Goal: Minimize $f(\theta) = \frac{1}{2} E[(\theta - y)^2]$ 기대값이니까 계산 ↑

- Robbins and Monroe:

  $\theta^{k+1} = \theta^k - \alpha_k \cdot (\theta^k - y_k)$

  - $y_k \sim D$ independent sample
  - $(\theta^k - y_k) \stackrel{!}{=}$ unbiased estimator of descent.
  - 수렴속도) $E[f(\theta^k)] - f(\theta^*) = \mathcal{O}\left(\frac{1}{k}\right)$

if $\text{loss} = \frac{1}{2}(\theta - y)^2$, $\alpha_k = \frac{1}{k+1}$

1) Update: $\theta_1 = \theta_0 - \theta_0 + y_1 = y_1$

$\theta_2 = \theta_1 - \frac{1}{2}(\theta_1 - y_2) = \frac{1}{2}(y_1 + y_2)$

$\theta_3 = \theta_2 - \frac{1}{3}(\theta_2 - y_3) = \frac{1}{3}(y_1 + y_2 + y_3)$

$\theta^k = \frac{1}{k}\sum^k y_i$

→ 결론: 이론적으로 estimate (추정), train (학습), optimize (최적화) 가 동일한 구조로 해결.
≒ 확률 평균 문제를 풀 수 있음.

7.1.5 Stochastic coordinate descent → 전체 gradient 가 아니라 좌표 하나씩만 update 해볼까?

(coordinate)

- when? $f(x)$가 너무 커서 한 번에 계산이 어려운 경우.
- update:
  1) coordinate $i \in \{1, \cdots, d\}$ 선택
  2) 해당좌표: $G(x) = d \cdot [\nabla f(x)]_i \cdot e_i$
  3) $x^{k+1} = x^k - \alpha_k G(x^k)$

- 기대값 $E[G(x)] = \nabla f(x)$ 으로 계산

  → unbiased stochastic gradient 조건 만족

7.2 Epochs and Momentum

1) SGD는 간단 but 수렴 속도↓, local minimum 근처에서 멈춤 (∵ noise), constant step-size는 항상 best가 아님

→ 해결: 1) Epoch-based stepsize schedule 2) Momentum.

- Epoch Schedule

  Epoch t 단위로 stepsize 감소.

  $\alpha_k = \alpha_0 \cdot \gamma^{t-1}$, $\gamma \in [0.8, 0.9]$

  1) 일정 횟수 (Epoch) 마다 learning rate 줄임
  2) 초기에 빠르게 내려가고 후반부 안정적으로 수렴

- Momentum (Polyak, Nesterov ..)

  기존 update: $x^{k+1} = x^k - \alpha \cdot G(x^k)$

  Momentum: $V_{k+1} = \beta V_k + G(x_k)$, $x_{k+1} = x_k - \alpha V_{k+1}$

  1) $\beta \in [0.8, 0.95]$ 이전방향을 얼마나 유지할지 결정
  2) 기울기가 일관된 방향으로 반복되면 가속도가 붙음
  3) noise에 강함, 진동↓

* 정리

7.1.4 Stoch. approx. problem : 확률적 평균/추정 문제에도 SGD가 적용 가능함.

7.1.5 Stoch. coord. descent : 고차원에서 효율적인 SGD 변형 소개.

7.2 Epochs and Momentum : 실전에서 빠르고 안정적으로 만드는 핵심 기법 2가지.

7.3 ~ : 지금까지 소개된 모든 SGD 기법들이 실제 "수렴" 한다는 것 증명.

- Convergence (수렴) = 반복되는 x^k가 어떤 x^*로 가까워진다.
  1) Parameter 수렴: 최적해를 찾자
  2) Function value 수렴: 최적해에 정확히 가지 않아도 비슷해(근사)질 수 있다.
  3) Expectation 수렴: 평균적으로 최적값에 가까워진다

7.3 Analysis of SGD

• Problem Statement :

SGM는 매 반복마다 정확한 gradient $\nabla f(x)$ 대신 확률적 근사치 $G(x)$를 사용해 업데이트 수행한다

이때 알고리즘이 잘 작동하려면? 가정 1) Unbiased (편향 없음) 2) (M,B) - Bounded Second Moment

왜 필요? 1) SGM이 수렴하려면 gradient가 폭주하거나 방향이 이상하면 X 2) 방향 & 변동성을 제한해보자.

• Assumption 1 : Unbiased estimate.

$$E[G(x)] = \nabla f(x).$$

1) 확률 gradient $G(x)$는 평균적으로 정확한 방향

2) 각 step이 noisy 할 수는 있지만, 전체적으로는 descent 방향

• Assumption 2 : (M,B) - Bounded

$$E\left[\|G(x)\|_2^2\right] \le \underbrace{M^2\|x - x^*\|_2^2}_{\min f(x)} + \underbrace{B^2}_{\text{2) 거리와 관계없는 base noise}}$$

1) 거리가 멀수록 noise 大

→ 의미 : gradient $G(x)$ 가 얼마나 unstable 한가.

To understand these conditions better,

1) Deterministic Gradient

if $G(x) = \nabla f(x)$:

$$\|G(x)\|^2 = \|\nabla f(x)\|^2 = \|\nabla f(x) - \nabla f(x^*)\|^2 \le M^2\|x - x^*\|^2$$

this condition is called strong smoothness and satisfied for a number of practical loss functions.

2) Additive Gaußian noise

Let $G(x) = \nabla f(x) + z$, where $z \sim \mathcal{N}(0, \sigma^2 I)$:

$$E\left[\|G(x)\|_2^2\right] = \|\nabla f(x)\|^2 + E\left[\|z\|_2^2\right] = \|\nabla f(x)\|^2 + n\sigma^2 \qquad \because E\left[\|z\|_2^2\right] = n\sigma^2$$

If f is M-strongly smooth, then 2^{nd} condition hold with $(M, \sigma\sqrt{n}) \Rightarrow B^2 = n\sigma^2$, M^2는 gradient의 Lipschitz 조건

3) Support Vector Machine

Recall hinge loss $G_i(\theta) = \begin{cases} -y_i x_i & , 1 - y_i\langle\theta, x_i\rangle > 0 \\ 0 & , \text{otherwise} \end{cases}$

Thus, condition holds with $B = \max_i \|x_i\|$ and $M = 0$. 즉 gradient norm은 거리와 무관하게 bounded.

7.3.1 Convergence Analysis

→ 조건을 만족하는 SGM이 convex function f에 대해 얼마나 잘 수렴하나?

- Problem Setting

$f: \mathbb{R}^d \to \mathbb{R}$ convex

$G(x)$, unbiased SG $\Leftrightarrow E[G(x)] = \nabla f(x)$

G $(0,B)$-Bounded $\Leftrightarrow E[\|G(x)\|_2^2] \le B^2$

- **Theorem 1**

f convex, G $(0,B)$-Bounded. Define:

- stepsize : $\bar{\alpha} = \sum_{i=0}^{k-1} \alpha_i$
- Avg. iterate : $\bar{x}_k = \frac{1}{\bar{\alpha}} \sum_{i=0}^{k-1} \alpha_i x^i$.

Then : $E[f(\bar{x}_k) - f(x^*)] \le \frac{1}{2\bar{\alpha}} \left(\|x^0 - x^*\|_2^2 + B^2 \sum_{i=0}^{k-1} \alpha_i^2 \right)$

의미)

1) SGD 수렴률은 Avg. iter. $\bar{x}_k$ 기준으로 분석

2) 수렴 오차는 초기 거리 $\|x_0 - x^*\|^2$ 와 noise $B^2 \sum \alpha_i^2$ 에 의해 결정

3) $\bar{\alpha}$ 가 커질수록 오차 작아짐 ≤ 多 반복 필요.

1) optimal step size : 식을 최소화하는 α_{opt}를 구해보자

$\alpha_i = \alpha_{opt} = \frac{R}{B\sqrt{k}}$, $R = \|x_0 - x^*\|$

$E[f(\bar{x}_k) - f(x^*)] \le \frac{BR}{\sqrt{k}}$ → 기대 오차 $= \mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$

2) step-size가 약간 틀렸을때.

let $\alpha = \alpha_{opt} \cdot \xi$

$E[f(\bar{x}_k) - f(x^*)] \le \frac{BR}{2\sqrt{k}} (\xi + \xi^{-1})$

⇒ 1) $\xi = 1$ 이면 최적

2) ξ 가 커지거나 작아지면 오차⇑

3) decaying stepsize $\alpha_i = \frac{1}{\sqrt{i}}$

we can achieve the same rate, up to a logarithmic factor, at any iteration i by using $\alpha_i = \frac{1}{\sqrt{i}}$

$E[f(\bar{x}_k)] - f(x^*) = \mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$ But, 로그 오차 추가됨

+) Constant stepsize 의 문제점.

$\alpha_i = \alpha$ (constant)

$E[f(\bar{x}_k)] - f(x^*) \le \frac{1}{2\alpha k} \|x_0 - x^*\|_2^2 + \frac{1}{2} B \alpha^2$

1) 오차가 0 수렴 X, $\frac{1}{2} B\alpha^2$ 만큼 남는다

2) 일정 step 유지하려면, 일정 범위내에서 수렴 수 없음

→ Thus, epoch 기반 step 감소가 필요함