Introduction to Deep Learning

22. Optimization, Gradient Descent

STAT 157, Spring 2019, UC Berkeley

Alex Smola and Mu Li

courses.d2l.ai/berkeley-stat-157
Optimization
Optimization Problems

• General form:

\[
\text{minimize } f(x) \quad \text{subject to } x \in C
\]

• Cost function \( f : \mathbb{R}^n \rightarrow \mathbb{R} \)

• Constraint set example

\[
C = \{ x \mid h_1(x) = 0, \ldots, h_m(x) = 0, \ g_1(x) \leq 0, \ldots, g_r(x) \leq 0 \}
\]

• Unconstrained if \( C = \mathbb{R}^n \)
Local Minima and Global Minima

- Most optimization problems have no closed-form solution.
- We then aim to find a minima through iterative methods.
- Global minima $x^*$
  
  $$f(x^*) \leq f(x) \quad \forall x \in C$$

- Local minima $x^*$, there exists $\varepsilon$
  
  $$f(x^*) \leq f(x) \quad \forall x : \|x - x^*\| \leq \varepsilon$$
Convex Set

• A subset \( C \) of \( \mathbb{R}^n \) is called convex if

\[
\alpha x + (1 - \alpha)y \in C \quad \forall \alpha \in [0,1] \quad \forall x, y \in C
\]
Convex Function

• $f : C \rightarrow \mathbb{R}$ is called convex if

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y)$$

$\forall \alpha \in [0,1] \; \forall x, y \in C$

• If the inequality is strict whenever $\alpha \in (0,1)$ and $x \neq y$, then $f$ is called strictly convex
First-order condition

• $f$ is convex if and only if

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) \quad \forall x, y \in C$$

• If the inequality is strict, then $f$ is strictly convex
Second-order conditions

• $f$ is convex if and only if

$$\nabla^2 f(x) \geq 0 \quad \forall x \in C$$

• $f$ is strictly convex if and only if

$$\nabla^2 f(x) > 0 \quad \forall x \in C$$
Convex and Non-convex Examples

- Convex
  - Linear regression $f(x) = \|Wx - b\|^2_2$
  - $\nabla f(x) = 2W^T(Wx - b)$, $\nabla^2 f(x) = 2W^TW$
  - Softmax regression

- Non-convex
  - Multi-layer perception
  - Convolution neural networks
  - Recurrent neural networks
Convex Optimization

- If $f$ is a convex function, and $C$ is a convex set, then the problem is called a convex problem.
- Any local minima is a global minima.
- Unique global minima if strictly convex.
Proof

• Assume local minima \( x \), if exists a global minima \( y \)
  • Choose \( \alpha \leq 1 - \varepsilon / |x + y| \) and \( z = \alpha x + (1 - \alpha)y \)
  • Then \( \|x - z\| = (1 - \alpha)\|x + y\| \leq \varepsilon \)
  • Due to \( y \) is a global minima, so \( f(y) < f(x) \)

\[
f(z) \leq \alpha f(x) + (1 - \alpha)f(z) < \alpha f(x) + (1 - \alpha)f(x) = f(x)
\]

• It contradicts \( x \) is a local minima
Gradient Descent
Algorithm

• Choose initial $x_0$
• At time $t = 1, \ldots, T$

$$x_t = x_{t-1} - \eta \nabla f(x_{t-1})$$

• $\eta$ is called learning rate
The Choice of Learning Rate

• Given $\|\Delta\| < \varepsilon$, for any $f$, by the Taylor expansion

\[
f(x + \Delta) \approx f(x) + \Delta^T \nabla f(x)
\]

• Choose small enough learning rate $\eta \leq \varepsilon / \|\nabla f(x)\|$

\[
\| - \eta \nabla f(x)\| \leq \varepsilon
\]

\[
f(x - \eta \nabla f(x)) \approx f(x) - \eta \|\nabla f(x)\|^2 \leq f(x)
\]
Convergence Rate

• Assume \( f \) is convex, and its gradient is Lipschitz continuous with constant \( L \)
  \[ \|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\| \]

• If use learning rate \( \eta \leq 1/L \), after \( T \) steps
  \[ f(x_T) - f(x^*) \leq \frac{\|x_0 - x^*\|^2}{2\eta T} \]

  • Convergence rate \( O(1/T) \)
  • To get \( f(x_T) - f(x^*) \leq \epsilon \), needs \( O(1/\epsilon) \) iterations
Proof

• Gradient L-Lipschitz means

\[ f(y) \leq f(x) + \nabla f(x)^T(y - x) + \frac{L}{2} \|y - x\|^2 \]

• Plug in \( y = x - \eta \nabla f(x) \)

\[ f(y) \leq f(x) - \left( 1 - \frac{L\eta}{2} \right) \eta \|\nabla f(x)\|^2 \]

• Take \( 0 < \eta \leq 1/L \)

\[ f(y) \leq f(x) - \frac{\eta}{2} \|\nabla f(x)\|^2 \]

\( f \) decreases every time
Proof II

• By the convexity: \( f(x) \leq f(x^*) + \nabla f(x)^T(x - x^*) \)
• Plug in to \( f(y) \leq f(x) - \frac{\eta}{2} \|\nabla f(x)\|^2 \)

\[
f(y) \leq f(x^*) + \nabla f(x)^T(x - x^*) - \frac{\eta}{2} \|\nabla f(x)\|^2
\]

\[
f(y) - f(x^*) \leq \left(2\eta \nabla f(x)^T(x - x^*) - \eta^2 \|\nabla f(x)\|^2\right)/2\eta
\]

\[
= \left(\|x - x^*\|^2 + 2\eta \nabla f(x)^T(x - x^*) - \eta^2 \|\nabla f(x)\|^2 - \|x - x^*\|^2\right)/2\eta
\]

\[
= \left(\|x - x^*\|^2 - \|x - \eta \nabla f(x) - x^*\|^2\right)/2\eta
\]

\[
= \left(\|x - x^*\|^2 - \|y - x^*\|^2\right)/2\eta
\]
Proof III

• Sum all $T$ steps

$$\sum_{t=1}^{T} f(x_t) - f(x^*) \leq \sum_{t=1}^{T} \left( \|x_{t-1} - x^*\|^2 - \|x_t - x^*\|^2 \right) / 2\eta$$

$$= \left( \|x_0 - x^*\|^2 - \|x_T - x^*\|^2 \right) / 2\eta \leq \|x_0 - x^*\|^2 / 2\eta$$

• $f$ is decreasing every time:

$$f(x_T) - f(x^*) \leq \frac{1}{T} \sum_{t=1}^{T} f(x_t) - f(x^*) \leq \frac{\|x_0 - x^*\|^2}{2\eta T}$$
Apply to Deep Learning

- \( f \) is the sum of loss over all training data, \( \mathbf{x} \) is the learnable parameters

\[
f(\mathbf{x}) = \frac{1}{n} \sum_{i=0}^{n} \ell_i(\mathbf{x})
\]

\( \ell_i(\mathbf{x}) \) the loss for the \( i \)-th example

- \( f \) is often not convex, so the convergence analysis before cannot be applied
Stochastic Gradient Descent

Singapore Dollar (SGD) 1000
~740 USD
Algorithm

- At time $t$, sample example $t_i$

\[ x_t = x_{t-1} - \eta_t \nabla \ell_t(x_{t-1}) \]

- Compare to gradient descent

\[ x_t = x_{t-1} - \eta \nabla f(x_{t-1}) \]

\[ f(x) = \frac{1}{n} \sum_{i=0}^{n} \ell_i(x) \]
Sample Example

- Two rules to sample example it at time $t$
  - Random rule: choose $i_t \in \{1, \ldots, n\}$ uniformly at random
  - Cyclic rule: choose $i_t = 1, 2, \ldots, n, 1, 2, \ldots, n$
    - Often called incremental gradient descent
  - Randomized rule is more common in practice

\[
\mathbb{E} \left[ \nabla \ell_{i_t}(x) \right] = \mathbb{E}[ \nabla f(x) ]
\]

- An unbiased estimate of the gradient
Convergence Rate

• Assume $f$ is convex with a diminishing $\eta_t$, e.g. $\eta_t = O(1/t)$
  $$\mathbb{E}[f(x_T)] - f(x^*) = O(1/\sqrt{T})$$

• Under the same assumption, for gradient descent
  $$f(x_T) - f(x^*) = O(1/\sqrt{T})$$

• Assume gradient L-Lipschitz and fixed $\eta$
  $$f(x_T) - f(x^*) = O(1/T)$$
  - Does not improve for SGD
In Practice

• Does not diminish the learning rate so dramatically
  • We don’t care about optimizing to high accuracy
• Despite converging slower, SGD is way faster on computing the gradient than GD in each iteration
  • Specially for deep learning with complex models and large-scale datasets
Code...
Mini-batch SGD
Algorithm

- At time $t$, sample a random subset $I_t \subset \{1, \ldots, n\}$ with $|I_t| = b$

$$x_t = x_{t-1} - \frac{\eta_t}{b} \sum_{i \in I_t} \nabla \ell_i(x_{t-1})$$

- Again, it’s an unbiased estimate

$$\mathbb{E} \left[ \frac{1}{b} \sum_{i \in I_t} \nabla \ell_i(x) \right] = \nabla f(x)$$

- Reduces variance by a factor of $1/b$ compared to SGD
Code…