Momentum-based Optimization Techniques in Machine Learning and Deep Learning

Pham Viet Hung

MŰEGYETEM 1782

Content

- Overview
  - Estimation/Learning vs. Pure Optimization
  - Gradient-Based Optimization
- Momentum-based methods
  - Vanila SGD
  - SGD with Polyak Momentum
  - SGD with Nesterov Momentum

But before digging into the topic...

https://www.youtube.com/watch?v=dQ4jUFmp7hQ

Overview

Estimation/Learning vs. Pure Optimization

- **Stats**: Estimation; **CS**: Learning — Meaning: using data to estimate an unknown quantity. (Wasserman, 2004)
- In Machine Learning (ML) / Deep Learning (DL), optimization is a tool to achieve the best possible P (some performance metric) defined w.r.t the test set; hard to track it $\implies$ optimize P indirectly through a cost function $J(\theta)$.
- In pure optimization, we care about only $J(\theta)$.
- Cost Function: $J(\theta) = \mathbb{E}_{(x,y)\sim\hat{p}_{\text{data}}}[L(f(x;\theta),y)]$; $\hat{p}_{\text{data}}$: empirical distro
- Empirical Risk Minimization (ERM):

$$\mathbb{E}_{(x,y)\sim\hat{p}_{\text{data}}}[L(f(x;\theta),y)] = \frac{1}{n}\sum_{i=1}^{n} L(f(x^{(i)};\theta),y^{(i)}) \tag{1}$$

Gradient-based Optimization

- **First-order optimization algorithms**: Using flavours of gradient-only algorithm. Mostly some variants of gradient descent algorithms.
- **Second-order optimization algorithms**: Algorithms that make use of Hessian matrix (a.k.a second derivatives). *Problem*: Poor conditioning of the Hessian matrix.
- The above-mentioned optimization techniques are mostly used in convex optimization. In DL, mostly as a subroutine of some DL algorithms.
- Usually variants of Stochastic Gradient Descent optimization (usually with minibatch) are used in ML and DL.

Momentum-based methods

Vanila SGD (1)

- SGD (with minibatch) and its flavours are the most used optimization methods for ML tasks.
- The step length/learning rate is defined as fixed in SGD, but in practice, we need a gradually decreasing value $\implies$ denote it as ϵ_k.
- The two sufficient conditions to guarantee the convergence of SGD:

$$\sum_{k=1}^{\infty} \epsilon_k = \infty \tag{2}$$

$$\sum_{k=1}^{\infty} \epsilon_k^2 \leq \infty \tag{3}$$

Vanila SGD (2)

Algorithm 8.1 Stochastic gradient descent (SGD) update at training iteration k

Require: Learning rate ϵ_k
Require: Initial parameter $\boldsymbol{\theta}$
  while stopping criterion not met **do**
    Sample a minibatch of m examples from the training set $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ with corresponding targets $\boldsymbol{y}^{(i)}$.
    Compute gradient estimate: $\hat{\boldsymbol{g}} \leftarrow +\frac{1}{m}\nabla_{\boldsymbol{\theta}} \sum_i L(f(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}), \boldsymbol{y}^{(i)})$.
    Apply update: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \epsilon\hat{\boldsymbol{g}}$.
  end while

Figure: Algorithm of the Vanila SGD (Goodfellow et al., 2016)

SGD with Polyak Momentum (1)

- Sometimes learning with SGD is slow.
- Polyak (1964) proposes a method to accelerate learning in a high curvature, small and consistent/noisy gradient environment.
- The key driver is the **velocity** variable v.

SGD with Polyak Momentum (2)

Algorithm 8.2 Stochastic gradient descent (SGD) with momentum

Require: Learning rate ϵ, momentum parameter α
Require: Initial parameter $\boldsymbol{\theta}$, initial velocity $\boldsymbol{v}$
  while stopping criterion not met **do**
    Sample a minibatch of m examples from the training set $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ with corresponding targets $\boldsymbol{y}^{(i)}$.
    Compute gradient estimate: $\boldsymbol{g} \leftarrow \frac{1}{m}\nabla_{\boldsymbol{\theta}} \sum_i L(f(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}), \boldsymbol{y}^{(i)})$.
    Compute velocity update: $\boldsymbol{v} \leftarrow \alpha \boldsymbol{v} - \epsilon \boldsymbol{g}$.
    Apply update: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \boldsymbol{v}$.
  end while

Figure: Algorithm of the SGD w/ Polyak Momentum (Goodfellow et al., 2016)

SGD with Nesterov Momentum (1)

- Based on Nesterov's (1983, 2004) accelerated method, Sutskever (2013) in his PhD thesis proposed a variant of momentum-based SGD optimization technique.
- Difference between Nesterov and Polyak momentum is the evaluation point of the gradient.
- With Nesterov momentum, the evaluation of the the gradient occurs after the current velocity is applied.

SGD with Nesterov Momentum (2)

Algorithm 8.3 Stochastic gradient descent (SGD) with Nesterov momentum

Require: Learning rate ϵ, momentum parameter α
Require: Initial parameter $\boldsymbol{\theta}$, initial velocity $\boldsymbol{v}$
  while stopping criterion not met **do**
    Sample a minibatch of m examples from the training set $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ with corresponding labels $\boldsymbol{y}^{(i)}$.
    Apply interim update: $\tilde{\boldsymbol{\theta}} \leftarrow \boldsymbol{\theta} + \alpha \boldsymbol{v}$.
    Compute gradient (at interim point): $\boldsymbol{g} \leftarrow \frac{1}{m} \nabla_{\tilde{\boldsymbol{\theta}}} \sum_i L(f(\boldsymbol{x}^{(i)}; \tilde{\boldsymbol{\theta}}), \boldsymbol{y}^{(i)})$.
    Compute velocity update: $\boldsymbol{v} \leftarrow \alpha \boldsymbol{v} - \epsilon \boldsymbol{g}$.
    Apply update: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \boldsymbol{v}$.
  end while

Figure: Algorithm of the SGD w/ Nesterov Momentum (Goodfellow et al., 2016)

Thank you very much for your attention!

References 1

- Wasserman, L. (2004). *All of statistics: a concise course in statistical inference (Vol. 26)*. New York: Springer.
- Goodfellow, I., Bengio, Y., Courville, A. (2016). *Deep Learning*. MIT Press.
- Melville, J. (2016). *Nesterov Accelerated Gradient and Momentum*. https://jlmelville.github.io/mize/nesterov.html#Sutskever_Nesterov_Momentum
- Li, F-F. et al. (2022). *CS231n: Deep Learning for Computer Vision*. Stanford University. https://cs231n.github.io/neural-networks-3/

References 2

- Polyak, B. T. (1964). *Some methods of speeding up the convergence of iteration methods*. USSR Computational Mathematics and Mathematical Physics, 4(5), 1–17.

- Nesterov, Y. (1983). *A method of solving a convex programming problem with convergence rate $O(1/k^2)$*. Soviet Mathematics Doklady, 27(2):372–376

- Nesterov, Y. (2004). *Introductory lectures on convex optimization: A basic course*. Kluwer Academic Publishers

- Sutskever, I. (2013). *Training Recurrent Neural Networks*. University of Toronto. https://tspace.library.utoronto.ca/bitstream/1807/36012/6/Ilya_Sutskever_201306_PhD_thesis.pdf