Optimization

Yao Zhang

The guy is a populace

Mostly based on Thomas Hofmann's lecture in ETH

https://zhims.github.io/

Dec 20, 2019

Learning as Optimization

1. Machine Learning: uses optimization, but is not equal to optimization
2. First: empirical risk is only a proxy for the expected risk
   - practically: early-stopping, monitoring on validation on validation set
   - we should not overfit the training
3. Second: loss function may only be surrogate
   - for instance: logistic loss instead of (0/1)-classification error
   - we should not overfit the loss function
   - (finally: we should not overfit to the task)

Objectives as Expectations

Structure of learning objective: large finite sums

1. many relevant quantities: sums over all training instances, e.g. empirical risk
2. example: gradient

$$\nabla_\theta \ell(\theta, S_N) = \frac{1}{N} \sum_{i=1}^{N} \nabla_\theta \ell(\theta, x[i], y[i]) \tag{1}$$

3. accuracy-complexity trade-off: subsample terms in sum
4. in practice: use of mini-batches of data

Large Scale Learning

1. Some (not all!) data sets have grown "fast" than compute power. Or let us say: computation and moving data to processors, not data, is the bottleneck.
2. Need trade-off statistical power (more data) with computational power (memory, compute cycles):
   - "super-trooper" algorithm: process few data points in an expensive manner
   - "cheap & easy" algorithm: process many data points in a cheap manner
   - practically: favor cheap over expensive

[Bottou et.al 2008]

Gradient Descent

Compute full gradient (across all parameters) and descent

$$\theta(t+1) = \theta(t) - \eta \nabla_\theta \ell \tag{2}$$

1. $\eta > 0$: step size or learning rate-alternatively: use of line search
2. continuous time dynamics: ordinary differential equation = gradient flow (Euler's method)

$$\dot{\theta} = -\nabla_\theta \ell \tag{3}$$

Definitions

Definition 1 (Convexity)

A function $\ell : \mathcal{X} \subseteq \mathbb{R}^d \to \mathbb{R}$ is convex, if $\mathcal{X}$ is convex and

$$\forall x, x' \in \mathcal{X}, \forall \beta \in [0,1] : \ell\left(\beta x + (1-\beta) x'\right) \leqslant \beta \ell(x) + (1-\beta)\ell(x') \tag{4}$$

Definition 2 (Strong Convexity)

A function ℓ is μ-strongly convex, if $\mathcal{X}$ is convex and $\forall x, x' \in \mathcal{X}, \forall \beta \in [0,1]$:

$$\ell\left(\beta x + (1-\beta) x'\right) \leqslant \beta \ell(x) + (1-\beta)\ell(x') - \frac{\mu}{2}\beta(1-\beta)\|x - x'\| \tag{5}$$

Definition 3 (L-Smooth)

A differentiable function ℓ is L-smooth, if:

$$\forall x, x' : \quad \|\nabla \ell(x) - \nabla \ell(x')\| \leqslant L\|x - x'\|, \quad L > 0 \tag{6}$$

Quadratic Bounds

Proposition 1 (Quadratic upper bound)

If ℓ is L-smooth, then $\forall x, x' \in \mathcal{X}$

$$\ell(x') \leq \ell(x) + \nabla \ell(x)^T (x' - x) + \frac{L}{2} \|x - x'\|^2 \tag{7}$$

Proposition 2 (Quadratic lower bound)

If ℓ is differentiable μ-strongly convex, then $\forall x, x' \in \mathcal{X}$

$$\ell(x') \geq \ell(x) + \nabla \ell(x)^T (x' - x) + \frac{\mu}{2} \|x - x'\|^2 \tag{8}$$

Gradient Descent: Convex Lipschitz Gradients

Theorem 1 (Convergence rate: Convex smooth case)

If ℓ is convex, L-smooth and has a unique minimizer θ^*, then the gradient iterate sequence with (constant) step size $\eta \leq \frac{1}{L}$ fulfills

$$\ell(\theta(t)) - \ell(\theta^*) \leqslant \frac{1}{2\eta t}\|\theta(0) - \theta^*\|^2 \in O\left(t^{-1}\right) \tag{9}$$

Theorem 2 (Lower Bound, [Nesterov 2004])

If ℓ is convex, L-smooth and has a unique minimizer θ^*, then for any iterate sequence with $\theta(t+1) \in \theta(t) + \text{span}\{\nabla\ell(\theta(0)), \cdots, \nabla\ell(\theta(t))\}$, one has

$$\ell(\theta(t)) - \ell(\theta^*) \geqslant \frac{3L\|\theta(0) - \theta^*\|^2}{32(t+1)^2} \tag{10}$$

Gradient Descent: Strong Convexity

Theorem 3

Let ℓ be ν-strongly convex and L-smooth over $\mathcal{X}$, then the gradient descent iterate sequence fulfills

$$\ell(\theta(t)) - \ell(\theta^*) \leqslant \left(1 - \frac{\mu}{L}\right)^t \Delta\ell, \quad \Delta\ell \triangleq \ell(\theta(0)) - \ell(\theta^*) \tag{11}$$

- exponential convergence ("linear rate")
- rate depends adversely on condition number $\frac{L}{\mu}$

Gradient Descent: Non-Convex Case

Definition 4 (ϵ-stationary point)

θ is a first-order ϵ-stationary point of ℓ, if $\|\nabla \ell(\theta)\| \leqslant \varepsilon$

Theorem 4 (Nesterov 1998?)

Assume ℓ is L-smooth. Then for any $\epsilon > 0$, gradient descent with step size $\eta = \frac{1}{L}$ will visit an ϵ-stationary point at least once in T number of iterations, where

$$T = \frac{L\Delta\mathcal{R}}{\varepsilon^2} \tag{12}$$

- can be generalized to find ϵ-second-order stationary points (essentially avoiding saddle points) by perturbing gradient descent updates with Gaussian noise ([Jin et.al 2019])
- only poly-logarithmic dependency on dimensionality

Challenges: Curvature

Curvature may require to use small step sizes:

$$\mathcal{R}\left(\theta-\eta\nabla\mathcal{R}\right) \overset{Taylor}{\approx} \mathcal{R}\left(\theta\right)-\eta\|\nabla\mathcal{R}\|^2+\frac{\eta^2}{2}\underbrace{\nabla\mathcal{R}^T H\nabla\mathcal{R}}_{=\|\nabla\mathcal{R}\|_H^2} \tag{13}$$

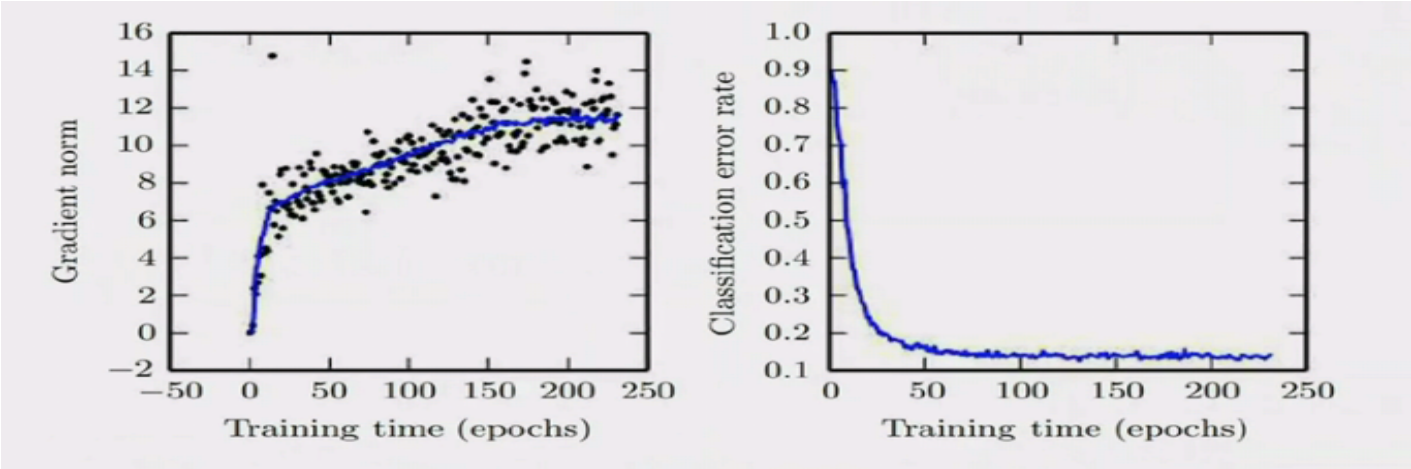
1. Hessian matrix: $H \triangleq \left[\nabla^2\mathcal{R}\right]$
2. problematic ill-conditioning:

$$\frac{\eta}{2}\|\nabla\mathcal{R}\|_H^2 \gtrsim \|\nabla\mathcal{R}\|^2 \tag{14}$$

3. remedy for first order methods: small step sizes η

Challenges: Curvature

Gradient descent may not arrive at a critical point of any kind.



Can be checked empirically!

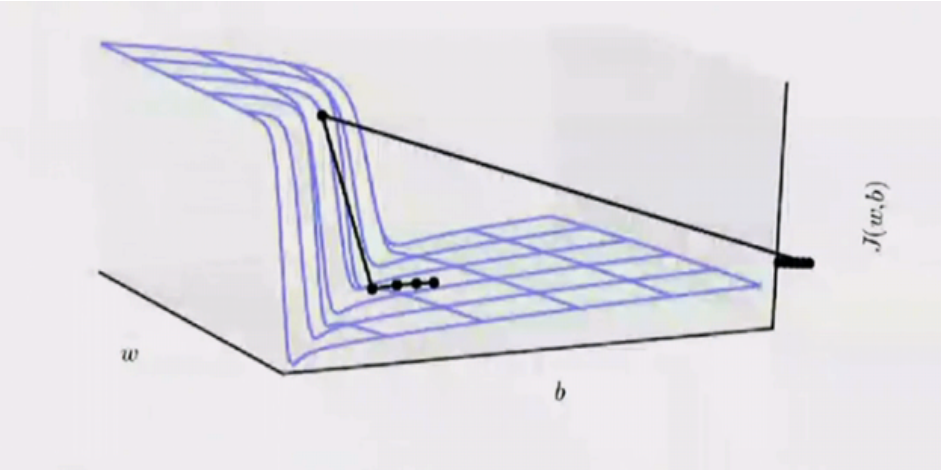
Challenges: Local Minima

Neural network risk functions can have many local minima and/or saddle points - and this is typical. Gradient descent can get stuck.

1. Are local minima a practical issue? Not always: [Gori et.al 1992]
2. Do local minima even exist? Sometimes not (auto-encoder): [Baldi et.al 1989]
3. Are local minima typically worse? Often not (large networks): e.g. [Choromanska et.al 2015]
4. Which local minima generalize well (wide vs. sharp. isolated): ongoing discussion
5. Can we understand the learning dynamics? Deep linear case has similarities with nin-linear case, e.g. [Saxe et.al 2013]

Challenges: Local Minima

Models with multiplication of many weights (depth, recurrence): sharp non-linearities



Motivates gradient clipping heuristics.

Least Squares

1. Study gradient dynamics for simple problem = least squares

$$\ell(A) = \frac{1}{2} E\|y - Ax\|^2 \tag{15}$$

   - single parameter matrix $A \in \mathbb{R}^{m \times n}$
   - expectation with regard to empirical distribution

2. Assumption, Notation, Identities
   - inputs whitened $E\left[xx^T\right] = I$
   - trace identities

$$v^T w = \sum_i Tr\left(vw^T\right), Tr(A+B) = Tr(A) + Tr(B) \tag{16}$$

   - $ETr(X) = TrE[x]$

Least Squares

Proposition 3

For whitened inputs $\left(E\left[xx^T\right] = 0\right)$, *we have that the least squares objective can be written as*

$$\ell(A) = \frac{1}{2}\|A - \Gamma\|_F^2 + \text{ const. } \quad \Gamma \triangleq E\left[xy^T\right]. \tag{17}$$

Proof.

Rewrite objective as trace functions

$$\begin{aligned}
\ell(A) &\overset{\text{inner to outer}}{=} \frac{1}{2} TrE\left[(y - Ax)(y - Ax)^T\right] \\
&\overset{\text{linearity}}{=} c_1 + \frac{1}{2} Tr\left(AE\left[xx^T\right]A^T\right) - Tr\left(A\Gamma^T\right) \\
&= c_1 + \frac{1}{2} Tr\left(\Gamma\Gamma^T\right) + \frac{1}{2} Tr\left(AA^T\right) - Tr\left(A\Gamma^T\right) \\
&= c_2 + \|A - \Gamma\|_F^2
\end{aligned} \tag{18}$$

□

Least Squares: Two-Layer Linear Network

[Saxe et.al 2013]

1. Two layer linear network $A = QW$
   - with $Q \in \mathbb{R}^{m \times k}, W \in \mathbb{R}^{k \times n}$, k : width of hidden layer
   - if $k \geqslant \min\{n, m\}$, same representational power as single layer network (interest: understanding learning dynamics)
2. Objective (plugging-in $A = QW$)

$$\ell(Q, A) = \frac{1}{2} \|QW - F\|_F^2 \tag{19}$$

3. Computing gradients

$$\frac{1}{2}\nabla_Q \ell = (A - \Gamma) W^T, \quad \frac{1}{2}\nabla_W \ell = Q^T (A - \Gamma) \tag{20}$$

Least Squares: Two-Layer Linear Network

4. Consider SVD of Γ (only data dependence) $\Gamma = U \sum V^T$
5. Perform change of basis $\widetilde{Q} = U^T Q$ and $\widetilde{W} = WV$.

$$A - \Gamma = QW - U \sum V^T = U \left(\widetilde{Q} \widetilde{W} - \sum \right) V^T \tag{21}$$

6. Gradients in new parametrization

$$\begin{aligned} \frac{1}{2} \nabla_{\widetilde{Q}} \ell &= \underbrace{U^T U}_{=I} \left(\widetilde{Q} \widetilde{W} - \sum \right) \underbrace{V^T V}_{=I} \widetilde{W}^T = \left(\widetilde{Q} \widetilde{W} - \sum \right) \widetilde{W}^T \\ \frac{1}{2} \nabla_{\widetilde{W}} \ell &= \widetilde{Q}^T \left(\widetilde{Q} \widetilde{W} - \sum \right) \end{aligned} \tag{22}$$

- diagonal target matrix

Least Squares: Two-Layer Linear Network

7. Define $w_r \triangleq r$-th column of $\widetilde{W}$, $q_r \triangleq r$-th row of $\widetilde{Q}$. We can write the gradients as

$$\begin{aligned} \frac{1}{2}\nabla_{q_r}\ell &= \left(q_r^T w_r - \sigma_r\right) w_r + \sum_{s\neq r}\left(q_r^T w_s\right) w_s \\ \frac{1}{2}\nabla_{w_r}\ell &= \left(q_r^T w_r - \sigma_r\right) q_r + \sum_{s\neq r}\left(q_s^T w_r\right)^2 \end{aligned} \tag{23}$$

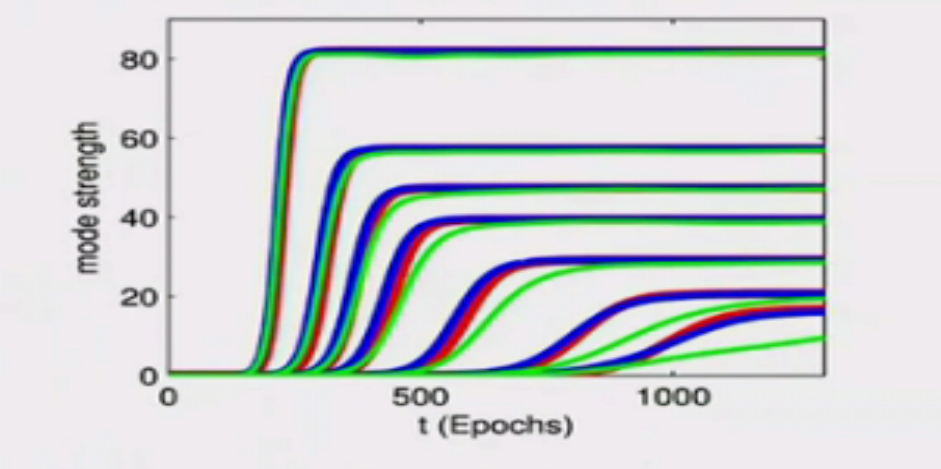
8. Equivalent energy function

$$\widetilde{\ell}\left(\widetilde{Q}, \widetilde{W}\right) = \sum_r \left(q_r^T w_r - \sigma_r\right)^2 + \sum_{s\neq r}\left(q_s^T w_r\right)^2 \tag{24}$$

- cooperation: same input-output mode weight vectors align
- competition: different mode weight vectors are decoupled

Least Squares: Two-Layer Linear Network

9. As learning advances: modes decouple = independent learning dynamics for each mode



red: analytic, blue: linear, green: tanh

For matched weights a, b the dynamics is governed by a loss of the term

$$\ell(a, b) = (\sigma - ab)^2 \tag{25}$$

which can be fully analysed (in the continuous time limit of infinitesimal step sizes)

Least Squares: Deep Linear Network

1. Local minima not a problem.
   Recent work: Deep Linear Networks with Arbitrary Loss: All Local Minima Are Global [Laurent et.al 2018]
2. Convergence rates of gradient descent for deep(er) networks not fully understood.
   Recent work: A convergence analysis of gradient descent for deep linear neural networks [Arora et.al 2018]
   - linear (= fast) convergence rates via imposing careful conditions on initialization.
3. General non-linear case: avoid slow-down around saddle points (e.g. On Nonconvex Optimization for Machine Learning: Gradients, Stochasticity, and Saddle Points [Jin et.al 2019])

Stochastic Gradient Descent

1. Stochastic gradient descent: chose update direction v at random such that $E[v] = -\nabla \ell$.
   - randomization scheme is unbiased
2. SGD via subsampling
   - pick random subset $\mathcal{S}_K \subseteq \mathcal{S}_N, K \leq N$
   - note that $E\ell(S_k) = \ell(S_N) \Rightarrow E\nabla\ell(S_K) = \nabla\ell(S_N)$
   - SGD update step (randomization at each t)

$$\theta(t+1) = \theta(t) - \eta(t)\nabla\ell(t), \ \ell(t) \triangleq \ell(S_K(t)) \tag{26}$$

Stochastic Gradient Descent

3. In practice: permute instances and break-up into mini-batches
   Epoch = one sweep through the data
   - harder to analyse theoretically
   - typically works better in practice
   - no permutation $\Rightarrow$ danger of "unlearning"
4. Mini-batch size
   - "standard SGD": $k = 1$, often most efficient in terms of # backprop steps
   - but: large k better for utilizing concurrency in GPUs or multicore CPUs

Stochastic Gradient Descent: Convergence Rates

1. Under certain conditions SGD convergences to the optimum:
   - convex or strongly convex objective
   - Lipschitz gradients / smoothness
   - decaying learning rate: $\sum_{t=1}^{\infty} \eta^2(t) < \infty$, $\sum_{t=1}^{\infty} \eta(t) = \infty$, typically: $\eta(t) = Ct^{-\alpha}$, $\frac{1}{2} < \alpha \leqslant 1$ (cf. hyperharmonic series)
   - iterate (Polyak) averaging
2. strongly-convex case: can achieve $O\left(\frac{1}{t}\right)$ suboptimality
3. general convex case: $O\left(\frac{1}{\sqrt{t}}\right)$ suboptimality

Stochastic Gradient Descent: Practicalities

1. Almost none of the analysis applies to the non-convex case
2. Choosing a learning rate schedule can be nuisance.
   - fast decay schedules may lead to super-slow convergence
   - in practice: tend to use larger step sizes and level out at a minimal step size (also: periodic schedule ...)
   - justification: SGD with fixed steps size is known to converge to a ball around the optimum (strongly convex case)
3. Common belief: stochasticity of SGD is "feature"
   - escape from regions with small gradients via perturbations
   - narrow vs. wide regions of low error

Heavy Ball Method

1. Modification of gradient descent or SGD with momentum:

$$\theta(t+1) = \theta(t) - \eta \nabla R + \alpha \underbrace{(\theta(t) - \theta(t-1))}_{momentum}, \quad \alpha \in [0,1] \tag{27}$$

   - discretization of time dynamics of motion of a particle with mass in potential field $\mathcal{R}$ (second order ODE)
   - assume gradients are constant (over sufficiently many steps), then update steps are boosted by $\frac{1}{1-\alpha}$

$$\eta \|\nabla J\| \left(1 + \alpha + \alpha^2 + \alpha^3 + \cdots\right) \rightarrow \frac{\eta \|\nabla J\|}{1-\alpha} \tag{28}$$

   - rule of thumb: $\alpha = 0.9 \Rightarrow 10\times$ max acceleration (sometimes initially $\alpha = 0$ and then slowly increased)

Reading List

L. Bottou and O. Bousquet (2008)

The tradeoffs of large scale learning

Proceeding NIPS'07 Proceedings of the 20th International Conference on Neural Information Processing Systems 161 – 168.

Y. Nesterov (2004)

Introductory Lectures on Convex Optimization

Springer Science + Business Media, LLC Vol.87.

M. Gori and A. Tesi (1992)

On The Problem Of Local Minima In Backpropagation

IEEE Transactions on Pattern Analysis and Machine Intelligence Vol.14, 76 – 86

P. Baldi and K. Hornik (1989)

Neural Networks and Principal Component Analysis: Learning from Examples Without Local Minima

Neural Networks Vol.2(1), 53 – 58

Reading List

A. Choromanska, M. Henaff, M. Mathieu, G. Arous and Y. LeCun (2015)

The loss surfaces of multilayer networks

Journal of Machine Learning Researchs

A. Saxe, J. McClelland and S. Ganguli (2013)

Exact solutions to the nonlinear dynamics of learning in deep linear neural networks

arXiv preprint arXiv:1312.6120

T. Laurent and J. Brecht (2018)

Deep Linear Networks with Arbitrary Loss: All Local Minima Are Global

Proceedings of the 35th International Conference on Machine Learning Vol.80, 2902 – 2907

S. Arora, N. Golowich, N. Cohen and W. Hu (2018)

A convergence analysis of gradient descent for deep linear neural networks

7th International Conference on Learning Representations, ICLR 2019

Reading List

C. Jin, P. Netrapalli and M. Jordan(2019)

On Nonconvex Optimization for Machine Learning: Gradients, Stochasticity, and Saddle Points

arXiv preprint arXiv:1902.04811

A. Saxe, J. McClelland and S. Ganguli (2013)

Exact solutions to the nonlinear dynamics of learning in deep linear neural networks

arXiv preprint arXiv:1312.6120

T. Laurent and J. Brecht (2018)

Deep Linear Networks with Arbitrary Loss: All Local Minima Are Global

Proceedings of the 35th International Conference on Machine Learning Vol.80, 2902 – 2907

S. Arora, N. Golowich, N. Cohen and W. Hu (2018)

A convergence analysis of gradient descent for deep linear neural networks

7th International Conference on Learning Representations, ICLR 2019

Last But Not Least

Thank you all of you! –Yao