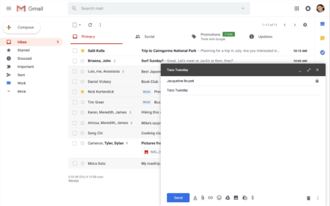
CMPT 409/981: Optimization for Machine Learning

Lecture 1

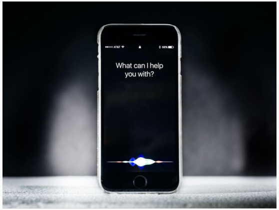
Sharan Vaswani

September 5, 2024

Successes of Machine Learning



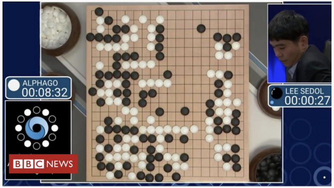
https://www.blog.google/products/gmail/subject-write-emails-faster-smart-compose-gmail/



https://www.cnet.com/news/what-is-siri/

(a) Natural language processing

(b) Speech recognition



https://www.bbc.com/news/technology-35785875

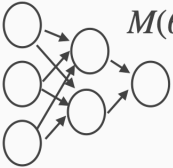
https://www.pbs.org/newshour/science/in-a-crash-should-self-driving-cars-save-passengers-or-pedestrians-2-million-people-weigh-in

(c) Reinforcement learning

(d) Self-driving cars

Machine Learning 101

Dataset {  , cat } ; {  , dog } ...

Training: {  , cat } →  → dog

Model
$M(\theta, .)$

Input-label
{x, y}

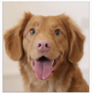
Update parameters using optimization

Prediction
$M(\theta, x)$

Loss
$f(\theta, \{x, y\})$

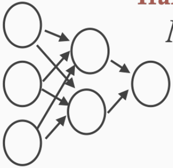
Output: $M(\theta^*, .)$

Machine Learning 101

Validation Dataset: {  , cat }; {  , dog } …

Validation: {  , dog } →

{x, y}



Trained Model

$M(\theta^*, .)$

→ dog

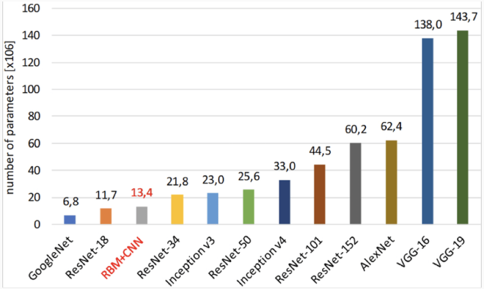
Prediction

$M(\theta^*, x)$

Output: Validation Accuracy

Measures how good the trained model is

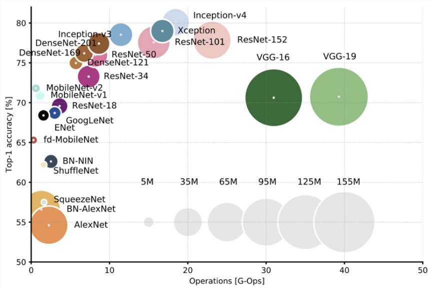
Modern Machine Learning



Sobczak, Szymon, et al. "Restricted Boltzmann machine as an aggregation technique for binary descriptors.", 2019.

Model size

(a)



Canziani et al, "An Analysis of Deep Neural Network Models for Practical Applications", 2016.

Number of operations for computing the loss

(b)

Figure 1: Models for multi-class classification on Image-Net. Number of examples = 1.2 M

Faster optimization methods can have a big practical impact!

Optimization for Machine Learning

- **(Non)-Convex minimization**: Supervised learning (classification/regression), Matrix factorization for recommender systems, Image denoising.
- **Online optimization**: Learning how to play Go/Atari games, Imitating an expert and learning from demonstrations, Regulating control systems like industrial plants.
- **Min-Max optimization**: Generative Adversarial Networks, Adversarial Learning, Multi-agent RL.

Course structure

Objective: Introduce foundational optimization concepts with applications to machine learning.

Syllabus:

- **(Non)-Convex minimization**: Gradient Descent, Momentum/Acceleration, Mirror Descent, Newton/Quasi-Newton methods, Stochastic gradient descent (SGD), Variance reduction
- **Online optimization**: Follow the (regularized) leader, Adaptive methods (AdaGrad, Adam)
- **Min-Max optimization**: (Stochastic) Gradient Descent-Ascent, (Stochastic) Extragradient

What we won't get time to cover in detail: Non-smooth optimization, Convex analysis, Global optimization.

What we won't get time to cover: Constrained optimization, Distributed optimization, Multi-objective optimization.

Course Logistics

- **Instructor**: Sharan Vaswani (TASC-1 8221) Email: `sharan_vaswani@sfu.ca`
- **Instructor Office Hours**: Thursday, 2.30 pm - 3.30 pm (TASC-1 8221)
- **Teaching Assistant**: Qiushi Lin Email: `qla96@sfu.ca`
- **TA Office Hours**: Monday, 9.30 am - 10.30 am (ASB 9814)
- **Course Webpage**: `https://vaswanis.github.io/409_981-F24.html`
- **Piazza**: `https://piazza.com/sfu.ca/fall2024/cmpt409981/home`
- **Prerequisites**: Linear Algebra, Multivariable calculus, (Undergraduate) Machine Learning

Course Logistics – Grading

Assignments [48%]

- Individual assignments to be submitted online, typed up in Latex with accompanying code submitted as a zip file.
- **Assignment 0** [5%]: Out today. Assignment to recall prerequisite knowledge and get used to notation. Due next week.
- **Assignments 1 & 2** [22%]:
  - Due in 10 days (at 11.59 pm PST).
  - For some flexibility, each student is allowed 1 late-submission and can submit in the next class (no late submissions beyond that).
  - If you use up your late-submission and submit late again, you will lose 50% of the mark.
- **Assignments 3 & 4** [21%]: Released during the semester, but due only at the end of the term (December 10).

Participation [2%]: In class (during lectures, project presentations), on Piazza

Course Logistics – Grading

Final Project [50%]

- Aim is to give you a taste of research in Optimization.
- Projects to be done in groups of 3-4 (more details will be on Piazza)
- Will maintain a list on Piazza on possible project topics. You are free to choose from the list or propose a topic that combines Optimization with your own research area.
- Project Proposal [10%] – Discussion (before October 20) + Report (due October 22)
- Project Milestone [5%] – Update (before November 20)
- Project Presentation [10%] (December 3)
- Project Report [25%] (December 17)

Questions?

Minimizing functions

Consider minimizing a function over the domain $\mathcal{D}$

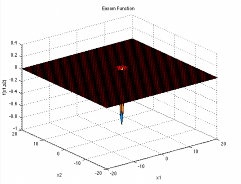
$$\min_{w \in \mathcal{D}} f(w).$$

Setting: Have access to a *zero-order oracle* – querying the oracle at $w \in \mathcal{D}$ returns $f(w)$.

Objective: For a target accuracy of $\epsilon > 0$, if f^* is the minimum value of f in $\mathcal{D}$, return a point $\hat{w} \in \mathcal{D}$ s.t. $f(\hat{w}) - f^* \leq \epsilon$. Characterize the required number of oracle calls in terms of ϵ.

Example 1: Minimize a one-dimensional function s.t. $f(w) = 0$ for all $x \neq w^*$, and $f(w^*) = -\epsilon$.

Example 2: Easom function:
$f(x_1, x_2) = -\cos(x_1)\ -\cos(x_2)\ \exp(-(x_1 - \pi)^2 - (x_2 - \pi)^2)$.



Minimizing generic functions is hard! We need to make assumptions on the structure.

Lipschitz continuous functions

Consider minimizing a function over the domain $\mathcal{D}$:

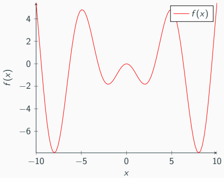
$$\min_{w \in \mathcal{D}} f(w).$$

Assumption: f is *Lipschitz continuous* (in $\mathcal{D}$) meaning that f can not change arbitrarily fast as w changes. Formally, for any $x, y \in \mathcal{D}$,

$$|f(x) - f(y)| \leq G \, \|x - y\|$$

where G is the Lipschitz constant.

Example: $f(x) := -x \sin(x)$ in the $[-10, 10]$ interval.



Lipschitz continuity of the function immediately implies that the gradients are *bounded* i.e. for all $x \in \mathcal{D}$, $\|\nabla f(x)\| \leq G$.

Global Minimization

Consider minimizing a G-Lipschitz continuous function over a unit hyper-cube:

$$\min_{w \in [0,1]^d} f(w).$$

Objective: For a target accuracy of $\epsilon > 0$, if $w^* \in [0,1]^d$ is the minimizer of f, return a point $\hat{w} \in [0,1]^d$ s.t. $f(\hat{w}) - f(w^*) \leq \epsilon$. Characterize the required number of zero-order oracle calls.

Naive algorithm: Divide the hyper-cube into cubes with length of each side equal to $\epsilon' > 0$ (to be determined). Call the zero-order oracle on the centers of these $\frac{1}{(\epsilon')^d}$ cubes and return the point $\hat{w}$ with the minimum function value.

Analysis: The minimizer lies in/at the boundary of one of these cubes. We can guarantee that we have queried a point $\tilde{w}$ that is at most $\frac{\sqrt{d}\epsilon'}{2}$ away from w^*, i.e. $\|\tilde{w} - w^*\| \leq \frac{\sqrt{d}\epsilon'}{2}$. By G-Lipschitz continuity, $f(\tilde{w}) - f(w^*) \leq G \|\tilde{w} - w^*\| \leq G \frac{\sqrt{d}\epsilon'}{2}$. For a target accuracy of ϵ, we can set $\epsilon' = \frac{2\epsilon}{G\sqrt{d}}$, implying that $f(\tilde{w}) - f(w^*) \leq \epsilon$. From the algorithm, we know that $\hat{w}$ is the queried point with the minimum function value. Hence, $f(\hat{w}) \leq f(\tilde{w})$ and consequently, $f(\hat{w}) - f(w^*) \leq \epsilon$. Hence, for this naive algorithm, total number of oracle calls $= \left(\frac{G\sqrt{d}}{2\epsilon}\right)^d$.

Global Minimization

Consider minimizing a differentiable, G-Lipschitz continuous function over a unit hyper-cube:

$$\min_{w \in [0,1]^d} f(w).$$

Q: Suppose we do a random search over the cubes – choosing a cube at random (say independently with replacement) and then querying its centre? What is the expected number of function evaluations to find a cube with is at most $\frac{\sqrt{d}\epsilon}{2}$ away from w^*?

Ans: The probability of finding the cube is $p := \epsilon'^d$. If X is the r.v. which corresponds to the number of attempts to find the correct cube, then X follows a Geometric distribution. Hence, expected number of evaluations is $\frac{1}{p} = \frac{1}{(\epsilon')^d} = \left(\frac{G\sqrt{d}}{\epsilon}\right)^d$.

Is our naive algorithm good? Can we do better?

Lower-Bound: For minimizing a G-Lipschitz continuous function over a unit hyper-cube, any algorithm requires $\Omega\left(\left(\frac{G}{\epsilon}\right)^d\right)$ calls to the zero-order oracle.

Questions?

Smooth functions

Recall that Lipschitz continuous functions have bounded gradients i.e. $\|\nabla f(w)\| \leq G$ and can still include *non-smooth* (not differentiable everywhere) functions.

For example, $f(x) = |x|$ is 1-Lipschitz continuous but not differentiable at $x = 0$ and the gradient changes from -1 at 0^- to $+1$ at 0^+.

An alternative assumption that we can make is that f is *smooth* – it is differentiable everywhere and its gradient is Lipschitz-continuous i.e. it can not change arbitrarily fast.

Formally, the gradient ∇f is L-Lipschitz continuous if for all $x, y \in \mathcal{D}$,

$$\|\nabla f(x) - \nabla f(y)\| \leq L \|x - y\|$$

where L is the Lipschitz constant of the gradient (also called the smoothness constant of f).

Q: Does Lipschitz-continuity of the gradient imply Lipschitz-continuity of the function? Ans: No, $\frac{x^2}{2}$ is 1-smooth but its gradient equal to x is unbounded over $\mathbb{R}$.

Smooth functions – Examples

If f is twice-differentiable and smooth, then for all $x \in \mathcal{D}$, $\nabla^2 f(x) \preceq L\, I_d$ i.e. $\sigma_{\max}[\nabla^2 f(x)] \leq L$ where $\sigma_{\max}$ is the maximum singular value.

Q: Does $f(x) = x^3$ have a Lipschitz-continuous gradient over $\mathbb{R}$? Ans: No, $f''(x) = 12x$ which is not bounded as $x \to \infty$

Q: Does $f(x) = x^3$ have a Lipschitz-continuous gradient over $[0, 1]$?

Ans: Yes, because $f''(x) = 12x$ is bounded on $[0, 1]$.

Q: The *negative entropy function* is given by $f(x) = x \log(x)$. Does it have a Lipschitz-continuous gradient over $[0, 1]$? Ans: No, $f''(x) = 1/x \to \infty$ as $x \to 0$.

Smooth functions – Examples

Linear Regression on n points with d features. Feature matrix: $X \in \mathbb{R}^{n\times d}$, vector of measurements: $y \in \mathbb{R}^n$ and parameters $w \in \mathbb{R}^d$.

$$\min_{w\in\mathbb{R}^d} f(w) := \frac{1}{2}\left\|Xw - y\right\|^2$$

$$f(w) = \frac{1}{2}\left[w^\mathsf{T}(X^\mathsf{T}X)w - 2w^\mathsf{T}X^\mathsf{T}y + y^\mathsf{T}y\right]; \nabla f(w) = X^\mathsf{T}Xw - X^\mathsf{T}y; \nabla^2 f(w) = X^\mathsf{T}X$$

(Prove in Assignment 0)

If f is L-smooth, then, $\sigma_{\max}[\nabla^2 f(w)] \leq L$ for all w. Hence, for linear regression $L = \lambda_{\max}[X^\mathsf{T}X]$.

Q: Is the linear regression loss-function Lipschitz continuous? Ans: No. Since $\|\nabla f(w)\| \to \infty$ as $w \to \infty$.

Q: Compute L for *ridge regression* – ℓ_2-regularized linear regression where $f(w) := \frac{1}{2}\left\|Xw - y\right\|^2 + \frac{\lambda}{2}\left\|w\right\|^2$. Ans: $L = \lambda_{\max}[X^\mathsf{T}X] + \lambda$

Questions?